



Kauno technologijos universitetas

Informatikos fakultetas

**Virtualiosios realybės daugelio žaidėjų kovos žaidimas su
specializuotu karkasu virtualiosios realybės žaidimams tinkle**

Baigiamasis bakalauro studijų projektas

Eligijus Kiudys

Projekto autorius

lekt. Andrius Paulauskas

Vadovas

Kaunas, 2021



Kauno technologijos universitetas

Informatikos fakultetas

Virtualiosios realybės daugelio žaidėjų kovos žaidimas su specializuotu karkasu virtualiosios realybės žaidimams tinkle

Baigiamasis bakalauro studijų projektas

Programų sistemos (612I30002)

Eligijus Kiudys

Projekto autorius

lekt. Andrius Paulauskas

Vadovas

lekt. Tomas Valatkevičius

Recenzentas

Kaunas, 2021

Eligijus, Kiudys. Virtualiosios realybės daugelio žaidėjų kovos žaidimas su specializuotu karkasu virtualiosios realybės žaidimams tinkle. Bakalauro studijų baigiamasis projektas / vadovas lekt. Andrius Paulauskas; Kauno technologijos universitetas, Informatikos fakultetas.

Studijų kryptis ir sritis (studijų krypčių grupė): Informatikos mokslai, Programų sistemos.

Reikšminiai žodžiai: daugelio žaidėjų karkasas, kovų žaidimas, žaidimas, karkasas.

Kaunas, 2021. 64 p.

Santrauka

Darbe pristatomas virtualios realybės kovų žaidimas ir daugelio žaidėjų karkasas. Žaidimų poreikiui nuolatos augant reikia ir naujų žaidimų. Kuriamas kovų žaidimas ir daugelio žaidėjų karkasas yra aktualus ne vien žaidėjų kūrėjams, bet ir žaidėjams, kurie žaidžia virtualios realybės žaidimus. Daugelio žaidėjų karkasas yra naudojamas virtualios realybės daugelio žaidėjų kūrimu. Kovų žaidimas yra skirtas pademonstruoti daugelio žaidėjų karkasui. Pagrindinis darbo tikslas yra sukurti daugelio žaidėjų karkasą, kuris leidžia sukurti daugelio žaidėjų žaidimus ir pademonstruoti jo veikimą sukuriant virtualios realybės kovų žaidimą.

Analizės metu buvo palyginami virtualios realybės žaidimo kūrimo įrankiai ir daugelio žaidėjų žaidimų kūrimo įskiepiai „Unity“ žaidimų varikliui. Pastebėta, kad virtualios realybės žaidimams kurti žaidimų varikliai neturi didelės įtakos, kadangi analizuoti žaidimų įrankiai palaiko virtualios realybės platformą. Išanalizavus daugelio žaidėjų žaidimo kūrimo įrankius pamatyta, kad visi daugelio žaidėjų įrankiai yra skirti įvairių tipų žaidimams kurti. Kovų žaidimas skirtas žaidėjams, kurie žaidžia virtualios realybės žaidimu, o daugelio žaidėjų karkasas yra skirtas kūrėjams, kurie kuria šiuos žaidimus.

Sistemos projektavimo metu yra suprojektuoti kovų žaidimo funkciniai reikalavimai, kurie demonstruoja daugelio žaidėjų karkaso veikimą. Projektavimo metu taip pat buvo iškelti nefunkciniai reikalavimai ir sukurti naudotojo sąsajos eskizai. Rengiant kovų žaidimo projektą sistema buvo suskaidyta į tokius komponentus – žaidimo pagrindinių funkcijų komponentus, virtualios realybės įvesčių komponentus ir serverio komponentus.

Daugelio žaidėjų karkasas ir kovų žaidimas buvo įvykdyti sėkmingai. Karkaso kūrimui buvo naudojama „Net“ ir „Socket“ bibliotekos. Taip pat, karkaso ir žaidimo kūrimui buvo naudojama C# programavimo kalba. Žaidimo realizacijai buvo naudojamas „Unity“ žaidimų variklis ir įvairūs įskiepiai virtualios realybės platformai palaikyti. Žaidimas ir daugelio žaidėjų karkasas buvo testuojamas naudojant „Visual Studio“ statinio kodo analizės įrankį, komponentų testavimą ir testavimą scenarijais. Išttestavus sistemą buvo parengta detali naudotojo dokumentacija.

Eligijus, Kiudys. Virtual Reality Multiplayer Fighting Game with Specialized Networking Framework for Virtual Reality Games. Bachelor's Final Degree Project / supervisor lect. Andrius Paulauskas; Informatics Faculty, Kaunas University of Technology.

Study field and area (study field group): Computer Sciences, Software Systems.

Keywords: virtual reality, multiplayer framework, battle, game, framework, fighting game.

Kaunas, 2021. 64.

Summary

The paper presents a virtual reality combat game and a multi-player framework. As the demand for games continues to grow, so do new games. The battle game and the multiplayer framework are relevant not only for developers, but also to players who play virtual reality games. The fighting game is designed to demonstrate the multi-player framework which is being developed. The main goal of the work is to create a multiplayer framework that allows you to create multiplayer games and demonstrate its operation by creating a virtual reality fighting game.

The analysis compared virtual reality game development tools and multiplayer game development plugins for the “Unity” game engine. It has been observed that game engines for developing virtual reality games do not have a significant impact as the analyzed game tools support the virtual reality. An analysis of multiplayer game development tools has shown that all multiplayer tools are designed to create different types of games. The fighting game is for players who play a virtual reality game, and the multiplayer framework is for developers who create these games.

During the design step, the functional requirements of the fighting game were designed, which demonstrates the performance of the multi-player framework. Non-functional requirements were raised during this step and user interface sketches were created. During the preparation of the combat game project, the system was divided into the following components - the components of the main functions of the game, the components of virtual reality inputs and the components of the server.

The multiplayer frame and fighting game have been completed successfully. The “.Net” and “Socket” libraries were used to create the framework. Also, the C # programming language was used to create the framework and game. The game was implemented using the “Unity” game engine and various plugins to support the virtual reality platform. The game and multiplayer framework were tested using the “Visual Studio” static code analysis tool, component testing, and scripting testing. After testing the system, detailed user documentation was prepared.

Turinys

Lentelių sąrašas.....	6
Paveikslų sąrašas	7
Santrumpū ir terminų sąrašas	9
Ivadas.....	10
1. Analizė.....	11
1.1. Techninis pasiūlymas	11
1.1.1. Sistemos apibrėžimas	11
1.1.2. Bendras veiklos tikslas	11
1.1.3. Sistemos pagrįstumas	11
1.1.4. Virtualios realybės žaidimų ir įrankių analizė.....	11
1.1.5. Konkurencija rinkoje	14
1.1.6. Prototipai ir pagalbinė informacija.....	17
1.1.7. Ištekliai, reikalingi sistemai sukurti.....	17
1.2. Galimybių analizė.....	17
1.2.1. Techninės galimybės	17
1.2.2. Vartotojų pasiruošimo analizė	18
2. Projektas	19
2.1. Reikalavimų specifikacija	19
2.1.1. Komercinė specifikacija	19
2.1.2. Sistemos funkcijos.....	19
2.1.3. Vartotojo sąsajos specifikacija	29
2.1.4. Realizacijai keliami reikalavimai	33
2.1.5. Techninė specifikacija	33
2.2. Projektavimo metodai.....	34
2.2.1. Projektavimo valdymas ir eiga	34
2.2.2. Projektavimo technologija.....	34
2.2.3. Programavimo kalbos, derinimo, automatizavimo priemonės, operacinė sistemos.....	35
2.3. Sistemos projektas	35
2.3.1. Statinis sistemos vaizdas	35
2.3.2. Dinaminis sistemos vaizdas.....	38
3. Testavimas	42
3.1. Testavimo planas	42
3.2. Testavimo kriterijai	42
3.3. Statinė kodo analizė.....	42
3.4. Komponentų testavimas	42
3.5. Testavimas scenarijais	43
4. Dokumentacija naudotojui.....	47
4.1. Apibendrintas sistemos galimybių aprašymas.....	47
4.2. Vartotojo vadovas.....	47
4.2.1. Kovos žaidimo vadovas.....	47
4.2.2. Daugelio žaidėjų karkaso vadovas	53
4.3. Diegimo vadovas	61
Rezultatai ir išvados.....	62
Literatūros sąrašas	63

Lentelių sąrašas

1.1 lentelė. Žaidimo variklių palyginimo lentelė.....	13
1.2 lentelė. Konkurentų apžvalga	15
3.1 lentelė. „Prisijungti“ panaudojimo atvejo testavimo lentelė	43
3.2 lentelė. „Atsijungti“ panaudojimo atvejo testavimo lentelė	43
3.3 lentelė. „Valdyti nustatymus“ panaudojimo atvejo testavimo lentelė.....	44
3.4 lentelė. „Žaisti“ panaudojimo atvejo testavimo lentelė	44
3.5 lentelė. „Pradėti žaidimą“ panaudojimo atvejo testavimo lentelė	44
3.6 lentelė. „Sužaloti priešininką“ panaudojimo atvejo testavimo lentelė	45
3.7 lentelė. „Valdyti interaktyvų daiktą“ panaudojimo atvejo testavimo lentelė	45
3.8 lentelė. „Valdyti diržo daiktus“ panaudojimo atvejo testavimo lentelė	45
3.9 lentelė. „Atidaryti/uždaryti duris“ panaudojimo atvejo testavimo lentelė	45

Paveikslų sąrašas

1.1 pav. „Unity“ žaidimų variklis	12
1.2 pav. „Unreal Engine“ žaidimų variklis	12
1.3 pav. „Photon PUN“ daugelio žaidėjų karkasas [1].....	14
1.4 pav. „Unet“ Unity žaidimų variklio daugelio žaidėjų karkasas.....	15
1.5 pav. Žaidimas „Half-Life: Alyx“.....	16
2.1 pav. Pavyzdinio kovos žaidimo panaudojimo atvejų diagrama.	19
2.2 pav. „Sinchronizuoti duomenis“ panaudojimo atvejo veiklos diagrama.....	20
2.3 pav. „Prisijungti“ panaudojimo atvejo veiklos diagrama	21
2.4 pav. „Atsijungti“ panaudojimo atvejo veiklos diagrama.....	22
2.5 pav. „Valdyti nustatymus“ panaudojimo atvejo veiklos diagrama.....	23
2.6 pav. „Pradėti žaidimą“ panaudojimo atvejo veiklos diagrama.....	24
2.7 pav. „Automatiškai atrakinti duris“ panaudojimo atvejo veiklos diagrama.....	25
2.8 pav. „Žaisti“ panaudojimo atvejo veiklos diagrama.....	25
2.9 pav. „Valdyti interaktyvų daiktą“ panaudojimo atvejo veiklos diagrama.....	26
2.10 pav. „Sužaloti priešininką“ panaudojimo atvejo veiklos diagrama.....	27
2.11 pav. „Valdyti diržo daiktus“ panaudojimo atvejo veiklos diagrama	28
2.12 pav. „Atidaryti/uždaryti duris“ panaudojimo atvejo veiklos diagrama	29
2.13 pav. Pradžios lango eskizas	30
2.14 pav. Žaidimo nustatymu eskizas pradžios lange	31
2.15 pav. Žaidėjo laikrodžio eskizas	31
2.16 pav. Žaidėjo laikrodžio eskizas	32
2.17 pav. Žaidimo meniu eskizas	32
2.18 pav. Žaidimo nustatymu eskizas laikrodžio lange	33
2.19 pav. „Trello“ projekto valdymo įrankio dalis	35
2.20 pav. Sistemos diegimo diagrama	36
2.21 pav. Sistemos komponentų diagrama	36
2.22 pav. Sistemos paketu diagrama (atnaujinti klasiu pavadinimus).....	37
2.23 pav. Sistemos UML klasių diagrama.....	38
2.24 pav. „Prisijungti“ panaudojimo atvejo sekų diagrama	38
2.25 pav. „Sužaloti priešininką“ panaudojimo atvejo sekų diagrama (turetu būti broadcast ir zaidejas i save atlika veiksma ir tada siuncia data)	39
2.26 pav. „Žaisti“ panaudojimo atvejo sekų diagrama.....	40
3.1 pav. Sékmingai atliki testai.....	42
3.2 pav. Sistemos testų padengimas	43
4.1 pav. Kovos žaidimo pradžios langas	47
4.2 pav. Kovos žaidimo nustatymų langas	47
4.3 pav. Kovos žaidimo garso nustatymu langas.....	48
4.4 pav. Kovos žaidimo vaizdo nustatymu langas.....	48
4.5 pav. Kovų žaidimo laukimo žemėlapis.....	48
4.6 pav. Kovų žaidimo laikomas interaktyvus objektas	49
4.7 pav. Kovų žaidimo interaktyvus diržas	49
4.8 pav. Priešo užpuolimas kovų žaidime	49
4.9 pav. Užkrautas žaidimo žemėlapis	50
4.10 pav. Atnaujintas išmanus laikrodis su likusiui laiku.	50
4.11 pav. Išmanaus laikrodžio gyvybių langas.....	50
4.12 pav. Kovų žaidimo meniu langas	51
4.13 pav. Kovų žaidimo meniu pasirinkimai.....	51
4.14 pav. Kovų žaidimo išmanaus laikrodžio nustatymai	51
4.15 pav. Kovų žaidimo atsijungimo langas.....	52
4.16 pav. Atrakintų durų atidarymas	52
4.17 pav. Žaidėjo mirtis	52

4.18 pav. Laimėjimo langas.....	53
4.19 pav. „Boolean“ tipo paketo siuntimo ir gavimo komponentai	53
4.20 pav. „Transform“ tipo paketo siuntimo ir gavimo komponentai.....	54
4.21 pav. „Rotation“ tipo paketo siuntimo ir gavimo komponentai.....	54
4.22 pav. „Boolean“ tipo paketo siuntimo ir gavimo komponentai.	55
4.23 pav. „Int“ tipo paketo siuntimo ir gavimo komponentai.	55
4.24 pav. „Grab“ tipo paketo siuntimo ir gavimo komponentai.....	56
4.25 pav. „Message List Send“ paketų siuntimo komponentas.....	56
4.26 pav. „Boolean“ tipo gavėjas ir valdytojas.	57
4.27 pav. Objekto būsenos valdymas.	57
4.28 pav. Privalomas žaidėjo komponentas.....	58
4.29 pav. Privalomi komunikacijos su serveriu komponentai.....	58
4.30 pav. „Disconnect“ komponentas.....	59
4.31 pav. Žaidėjo komponentas.....	59
4.32 pav. „UDP Connection Manager“ komponentas.....	60
4.33 pav. „UDP Player Manager“ komponentas	60
4.34 pav. „Worker“ komponentai.....	60
4.35 pav. Failo priedai kuriuos reikia pridėti į „manifest.json“ failą.....	61
4.36 pav. Paleisto daugelio žaidėjų serverio langas	61

Santrumpų ir terminų sąrašas

Santrumpos:

UML (angl. Unified Modeling Language) – modeliavimo ir specifikacijų kūrimo kalba, skirta atvaizduoti bei dokumentuoti objektines programų sistemas

GPU (angl. graphics processing unit) – grafinis procesorius skirtas greitai manipuliuoti ir keisti atmintyje saugomus įrašus taip paspartinant vaizdų kūrimą kadro buferyje.

TCP/IP (angl. Transmission control protocol/Internet Protocol) – standartinis duomenų perdavimo protokolų rinkinys, kurio pagrindas – TCP ir IP protokolai.

UDP (angl. User Datagram Protocol) – TCP/IP naudojamas perdavimo protokolas, kuris neturi klaidų tikrinimo mechanizmo.

Terminai:

Bitų glaudimas – duomenų dydžio sumažinimas.

Įskiepis – įvairūs įrankiai integruojami į sistemą.

Sinchronizacija – reikšmių sinchronizacija žaidimo metu.

Įvadas

Žaidimai yra populiarusia pramogos šaka, kadangi kiekvienas namuose turi kompiuterius ir su jais gali žaisti žaidimus. Žaidimų rinkai sparčiai augant atsiranda didelis poreikis naujų ir įdomių žaidimų. Virtualios realybės žaidimai suteikia galimybę pasinerti į virtualų pasaulį (1). Su specialiais virtualios realybės akiniais žaidėjas gali išsitraukti į virtualią realybę, kuri suteikia naujas galimybes, tyrinėti ir sąveikauti su virtualiais pasauliais. Daugelio žaidėjų virtualios realybės žaidimai pagerina pagrindinį virtualios realybės konceptą, kadangi žaidėjai gali tyrinėti ir sąveikauti virtualiose pasauliuose kartu su kitais žaidėjais. Kuriamas daugelio žaidėjų karkasas supaprastina virtualios realybės daugelio žaidėjų žaidimų kūrimą – pateikiant pradinus sinchronizavimo ir serverio komponentus, kurie gali būti lengvai praplečiami arba sukurti nauji. Naujo karkaso pademonstravimui buvo sukurtas pavyzdinis virtualios realybės žaidimas, kuriame yra pademonstruojamas pagrindinis virtualios realybės sąveikavimas su aplinka. Žaidimas padeda suprasti kaip turėti veikti daugelio žaidėjų kovos žaidimas virtualioje realybėje.

Darbo tikslas – supaprastinti daugelio žaidėjų virtualios realybės žaidimų kūrimą – pateikiant daugelio žaidėjų karkasą kuris yra lengvai konfigūruojamas ir kuris yra lengvai plečiamas. Karkaso demonstracijai pateikti daugelio žaidėjų kovų žaidimą. Šio tikslo įgyvendinimui yra iškelti šie uždaviniai:

1. Išanalizuoti žaidimų variklius ir virtualios realybės įskiepius žaidimų kūrimu.
2. Išanalizuoti panašius daugelio žaidėjų karkasus ir žaidimus.
3. Sudaryti virtualios realybės žaidimo ir daugelio žaidėjų karkaso projektą
4. Realizuoti bei ištenuoti virtualios realybės žaidimą ir daugelio žaidėjų karkasą
5. Parengti daugelio žaidėjų karkaso ir VR žaidimo dokumentaciją.

Darbą sudaro keturi pagrindiniai skyriai - analizė, projekto, testavimo ir dokumentacijos. Analizės dalyje yra apibrėžiamos sistemos bei atliekamo tyrimo pagrįstumas, susipažištama su žaidimų varikliais bei virtualios realybės žaidimų kūrimų karkasais. Apžvelgti rinkoje naudojamus daugelio žaidėjų kūrimo karkasus bei panašius virtualios realybės žaidimus. Projekto skyriuje yra pateikiami kovos žaidimo funkcioniniai bei nefunkcioniniai reikalavimai, naudotos technologijos, programavimo kalbos bei žaidimo sistemos projektas, kuriame analizuojamas statinis bei dinaminis sistemos vaizdas. Testavimo skyriuje pateikiamas testavimo planas, apžvelgiami taikomi testavimų tipai bei gauti testavimų rezultatai. Dokumentacijos skyriuje yra pateikiami vartotojo sistemos, bei kūrėjo naudojimo vadovai.

Karkaso ir žaidimo kūrimui buvo skirta 400 valandų ir sukurta apie 8000 eilučių kodo. Sistemą sudaro dvi posistemės virtualios realybės daugelio žaidėjų karkasas ir virtualios realybės daugelio žaidėjų kovų žaidimas.

1. Analizė

1.1. Techninis pasiūlymas

1.1.1. Sistemos apibrėžimas

Virtualios realybės daugelio žaidėjų karkasas – tai karkasas, kuris yra skirtas žaidimui kūrėjams, kurie nori kurti virtualios realybės daugelio žaidėjų žaidimus. Sistema suteikia komponentus, kurie yra naudojami siusti ir gauti sinchronizacijos žinutes. Karkasas yra pritaikytas virtualios realybės žaidimui kūrimui, bet gali būti naudojamas ir kituose žaidimuose.

Kovų žaidimas – tai virtualios realybės žaidimas, kuris yra skirtas smagiai praleisti laiką su draugais virtualioje realybėje. Kovų Žaidime galima susijungti iki 10 žaidėjų, jie atsiras name kuriame susiras ginklą su kuriuo eis į kovą kur išgyvens tik vienas stipriausias.

1.1.2. Bendras veiklos tikslas

Bendras veiklos tikslas – palengvinti daugelio žaidėjų virtualios realybės žaidimui kūrimą sukuriant daugelio žaidėjų karkasą. Karkasas suteikia galimybę sinchronizuoti daugelio žaidėjų žaidimus priimant ir siunčiant žinutes su duomenimis. Panaudojant karkasą sukurti virtualios realybės kovų žaidimą kuris padėtų kūrėjams kurti naujus žaidimus naudojant naują karkasą.

1.1.3. Sistemos pagrįstumas

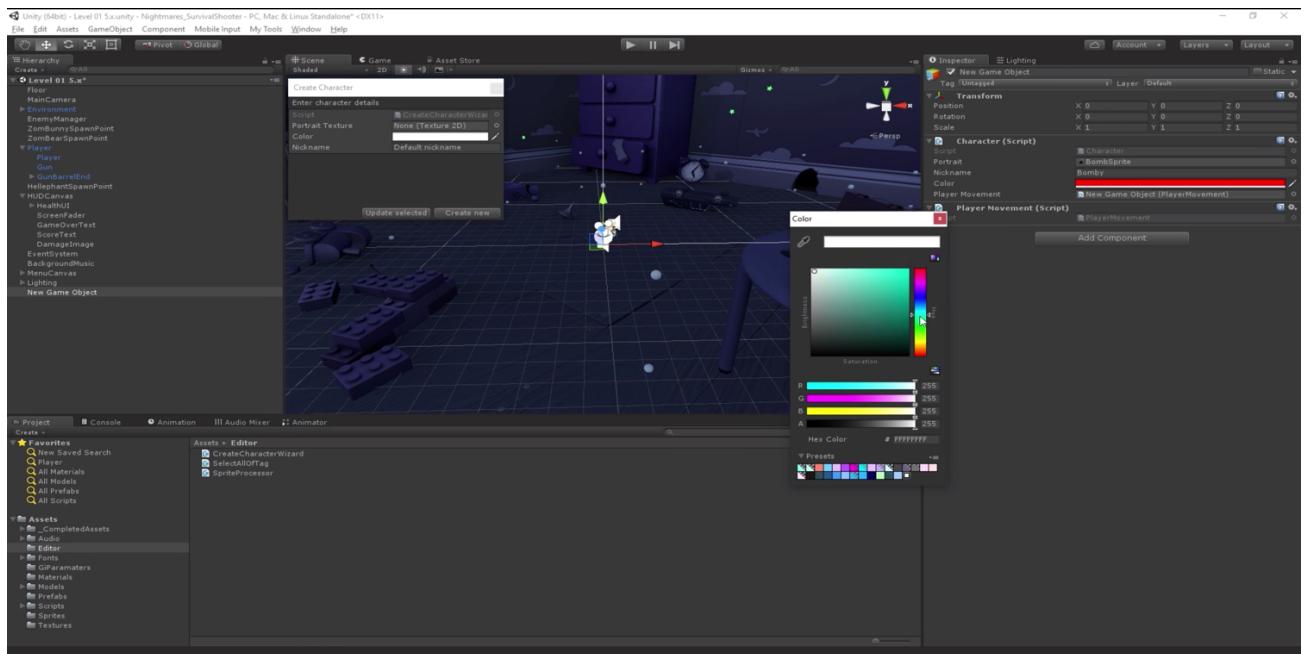
Žaidimų rinka plečiantis didėja poreikis naujų inovaciinių žaidimų kurie padėtų atsiplėsti nuo kasdieninių sunkumų (2). Žaidėjų kūrėjai ieško paprastesnių būdų kaip sukurti įvairaus žanro žaidimus naudojantis paprastus įrankius kuriuos gali praplėsti. Dabartiniai daugelio žaidėjų žaidimų karkasai yra labai dideli. Kūrėjai naudojant karkasus turi spręsti specifikuotas žaidimo problemas kaip sinchronizacija, autorizaciją tarp žaidėjų ir kitokias problemas kurios priklauso nuo kuriamo žaidimo. Dideli karkasai pateikia įvairias funkcijas kurias gali būtų galima naudoti įvairiems žaidimams nepriklausant nuo žaidimo specifikos todėl tokie karkasai nesuteikia specifinių funkcijų kurių reikia kūrėjui. Naujas daugelio žaidėjų karkasas yra kuriamas norint suteikti kūrėjams laisvę duodant pagrindines funkcijas kurias gali plėsti. Kūrėjai gali naudotis suteiktomis funkcijomis ir plėsti karkaso funkcionalumą pagal kuriamo žaidimo poreikius.

Esant virtualios realybės žaidimų trūkumui specializuoti karkasai turi didelę įtaką tokių žaidimų kūrimui. Kovos žaidimas naudoja minėtą karkasą. Kovų žaidimas padės išsiaiškinti daugelio žaidėjų karkaso naudojimą pateikiant žaidimą ir jo komponentus kaip pavyzdį karkaso naudojimui. Žaidimo pavyzdys padeda kurti naujus žaidimus kurie užpildo rinką naujais žaidimais.

1.1.4. Virtualios realybės žaidimų ir įrankių analizė

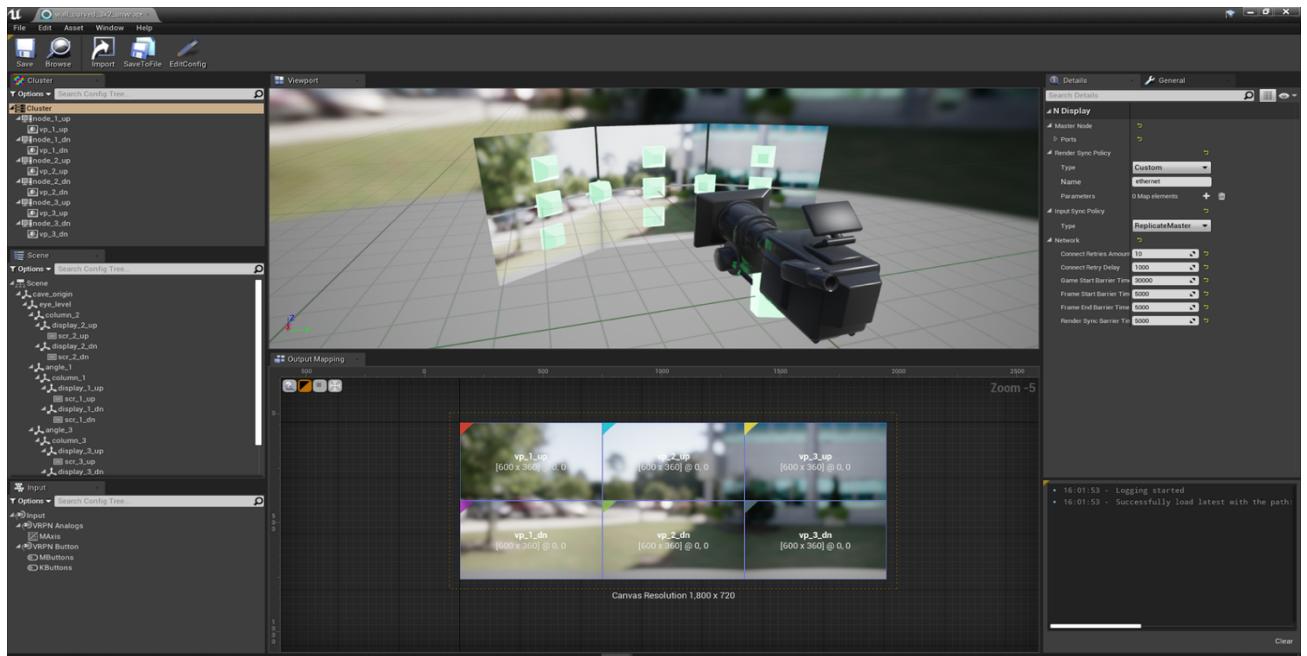
Žaidimų kūrimui egzistuoja daug įrankių ir įskiepių kurie yra naudojami žaidimui kūrėjų. Analizuoti varikliai ir įskiepiai skirti kurti žaidimus yra plačiai naudojami - pradedant nuo 2D žaidimo baigiant virtualios realybės žaidimais.

Vienas iš populiariausių žaidimų variklių yra „Unity“ (3) žaidimų variklis. Šis variklis kūrėjams suteikia galimybę kurti įvairaus tipo žaidimus ir interaktyvias aplikacijas. Kūrėjai gali kurti gali pasirinkti kokio tipo žaidimus jie nori kurti – 2D arba 3D tipo žaidimus. „Unity“ žaidimų variklis valdomas naudojant „Unity Editor“ vartotojo sąsają, kuri suteikia lengvesnę įskiepių integraciją bei lengvą komponentų valdymą. „Unity“ žaidimų variklis naudoja C# programavimo kalbą naujų komponentų kūrimui.



1.1 pav. „Unity“ žaidimų variklis

Kitas šioje srityje egzistuojanti variklis yra „Unreal Engine“ (4). Minėtas variklis suteikia kūrėjams įvairius įrankius su kuriais galima kurti įvairaus tipo 3D žaidimus. „Unreal Engine“ žaidimų variklis yra valdomas per pateiktą vartotojo sąsają, kuri padeda valdyti įvairius įskiepius ir komponentus. „Unreal Engine“ turi plačią bendruomenę ir didelę dokumentaciją skirtą padėti kūrėjams. Žaidimų variklio komponentams kurti yra naudojama C++ programavimo kalba arba vizualinė programavimo kalba, kuri yra specifiškai naudojama šiam žaidimų kūrimo varikliui.



1.2 pav. „Unreal Engine“ žaidimų variklis.

1.1 lentelė yra pateikiamas žaidimų variklių apibendrinimas.

1.1 lentelė. Žaidimo variklių palyginimo lentelė.

Lyginimo kriterijai	Unity	Unreal Engine
Virtualios realybės žaidimų palaikymas	+	+
Papildyto realybės žaidimų palaikymas	+	+
XR karkasas	+	-
Įskiepių palaikymas	+	+
Programavimo kalba	C#	C++
Vizualines programavimo kalba	+	+

Išanalizavus žaidimų variklius jie yra labai panašūs. Pasirinkus vieną ar kitą žaidimų variklį yra galima kurti įvairius žaidimus. Yra viena išimtis – „Unreal Engine“ oficialiai nepalaiko 2D žaidimų kūrimo. Išanalizavus abu žaidimų variklius buvo pasirinktas „Unity“ žaidimų variklis. Platformos pasirinkimą nulémė programavimo kalba.

Pasirinktas žaidimų variklis turi įvairius įskiepius virtualios žaidimų kūrimui. Išanalizuoti įvairius įskiepius pagal jų galimybes. Pasirinkti įskiepių analizės metu bus naudojami karkaso ir pavyzdinio žaidimo kūrimo etape.

„Steam VR“ (5) – populiarus įskiepis kuris suteikia įrankius su kuriais galima kurti virtualios realybės žaidimus. Šia platformą palaiko populariausiai virtualios realybės įrenginiai kurie yra skirti naudoti kompiuterinę įrangą. „Steam VR“ įskiepis palaiko tik „Steam VR“ (6) platformą.

„Oculus“ (7) – yra kitas populiarus įskiepis kuris suteikia įrankius su kuriais galima kurti virtualios realybės žaidimus. Šis įrankis yra skirtas „Oculus“ virtualios realybės įrenginių palaikymui.

„Steam VR“ ir „Oculus“ platformos yra skirtos virtualios realybės žaidimams kurti. Tiek viena tiek kita platforma suteikia panašų įrankių kieki. Apsvarsčius buvo pasirinkta „Steam VR“ įskiepis, kadangi šis įskiepis leidžia kurti virtualios realybės žaidimus nepriklausant nuo virtualios realybės įrenginio.

Analizuojame dar du papildomus įskiepių, kurie leidžia kurti virtualios realybės žaidimus nepriklausant nuo platformos.

„Unity XR“ (8) - įskiepis yra virtualios realybės įrankių abstrakcija. Įskiepis palaiko visus virtualios įrenginius ir platformas. Kadangi įskiepis yra vis dar kuriamas, įrankių kiekis yra labai ribotas.

„VRTK“ (9) - įskiepis yra virtualios realybės įrankių abstrakcija. Naudojant šitą įskiepių galima naudoti „Steam VR“ ir „Oculus“ platformas. Įskiepis suteikia ne vien reikalingus įrankius virtualios realybės žaidimams kurti, bet suteikia papildomų įrankių kurie palengvina virtualios realybės žaidimų kūrimą.

Išanalizavus likusius įskiepius buvo pasirinktas „VRTK“ įskiepis. Pasirinktas įskiepis suteikia papildomų įrankių, su kuriais galima kurti virtualios realybės žaidimus. „Unity XR“ įskiepis nebuvo pasirinktas, kadangi jis yra ankstyvoje kūrimo stadijoje, įrankis suteikia tik būtinus įrankius virtualios realybės žaidimu.

1.1.5. Konkurencija rinkoje

1.1.5.1. Žaidimų varikliai

Šiuo metu egzistuoja keletas panašių karkasų skirtų „Unity“ žaidimų kūrimo varikliui, su kuriais būtų galima kurti daugelio žaidėjų žaidimus virtualioje realybėje. Pasirinkti konkurentų karkasai yra plačiai naudojami daugelio žaidėjų žaidimams kurti. Kiekvienas karkasas gali būti naudojamas kurti skirtingo tipo daugelio žaidėjų žaidimus, kadangi karkasai néra vienodi. Daugelio žaidėjų kovos žaidimo pavyzdžiui buvo pasirinktas vienas virtualios realybės žaidimas kuris naudojamas žaidimų funkcionalumui palyginimui.

Vienas iš populiariausių daugelio žaidėjų kūrimo karkasų skirtų „Unity“ žaidimų varikliui yra „Photon PUN“ (10). Su šiuo karkasu galima kurti įvairaus tipo daugelio žaidėjų žaidimus naudojant Unity žaidimų variklį. Karkasas suteikia įvairius įrankius serverio plėtimui ir žaidimo kūrimui.

„Photon PUN“ suteikia galimybę naudotis sukurtu ir patalpintu serveriu nemokamai iki 100 žaidėjų prisijungusiu vienu metu, norint naudoti patalpintą serverį reikės mokėti už mėnesinę prenumeratą.

„Photon PUN“ suteikia galimybę naudotis privačiu serveriu kurį galima valdyti pačiam. Karkaso serviso naudojamas yra mokamas. Karkasas suteikia ne vien žinučių siuntimą tarp serverio ir kitų žaidėjų, įvairių sinchronizavimo funkcijų - komunikacijos protokolų pasirinkimas, skirtinių prisijungimo būdai prie žaidimo.



1.3 pav. „Photon PUN“ daugelio žaidėjų karkasas [1]

Unity žaidimų variklis turi tris karkasus skirtus kurti daugelio žaidėjų žaidimus. Visi karkasai nebus aprašyti, nes du karkasai yra ankstyvoje kūrimo stadioje. Bus analizuojami „Unet“ (11) ir „MLAPI“ (12) karkasai, „MLAPI“ karkasas pakeis „Unet“ karkasą, kadangi „Unet“ karkasas nebėra atnaujinamas.

„Unet“ karkasas yra nebeatnaujinamas, bet vis dar plačiai naudojamas kuriant daugelio žaidėjų žaidimus. Karkasas suteikia visas reikalingas funkcijas kaip – prisijungimą prie žaidimo, atsijungimą, komandų siuntimą į serverį, komunikaciją naudojant „UDP“ komunikacijos protokolą. „Unet“ daugelio žaidėjų karkasas yra naudojamas žaidimams kurie nereikalauja labai optimizuoto žaidimo.



1.4 pav. „Unet“ Unity žaidimų varikio daugelio žaidėjų karkasas

„MLAPI“ karkasas yra naujas Unity žaidimų varikio daugelio žaidėjų kūrimo karkasas. Šis naujas karkasas yra kūrimo ankstyvoje stadijoje. Karkasas neturi didelės dokumentacijos ir nėra labai paplitęs. Nepilna dokumentacija sunkina daugelio žaidimo kūrimą, kadangi nėra parašyta visų funkcijų veikimas. „MLAPI“ karkasas suteikia pagrindines funkcijas kaip – prisijungimas prie žaidimo, atsijungimas, duomenų sinchronizacija.

Virtualios realybės daugelio žaidėjų karkasas turi keletą pranašumų ir keletą trūkumų. Skirtingai nei „Unet“ ir „MLAPI“ karkasai virtualios realybės daugelio žaidėjų karkasas turi bitų glaudimą kurs sumazina tinklo apkrovą ir pagreitina duomenų persiuntimą. Karkasas gali būti lengvai praplėstas, kadangi karkaso kodas gali būti keičiamas, ko neleidžia kiti karkasai. Virtualios realybės daugelio žaidėjų karkasas neturi sudėtingų funkcijų kurios padėtu lengviau sinchronizuoti žaidimą.

1.2 lentelė. Konkurentų apžvalga yra pateikiamas visų anksčiau paminėtų sistemų ir virtualios realybės daugelio žaidėjų karkaso sistemos palyginimas pagal kriterijus.

1.2 lentelė. Konkurentų apžvalga

Lyginimo kriterijai	Photon PUN	Unity Unet	Unity MLAPI	Virtualios realybės daugelio žaidėjų karkasas
Žaidimo sesijos sukūrimas	+	+	+	-
Skirtingų platformų palaikymas	-	+	+	+
Kaina	0.19 €/Žaidėjas/mėn.	Nemokamai	Nemokamai	Nemokamai
Lengvas išplėtimas	-	-	+	+
Karkaso palaikymas	+	-	+	+
Bitų glaudinimas	-	-	-	+
Dedikuotas Serveris	+	-	+	+
Kliento-pusės spėjimas (angl. guess)	+	-	-	-

Lyginimo kriterijai	Photon PUN	Unity Unet	Unity MLAPI	Virtualios realybės daugelio žaidėjų karkasas
Žaidėjo autentifikacija	+	+	+	-
TCP komunikacija	+	-	+	-
UDP komunikacija	+	+	+	+

Galime teigti, kad artimiausias konkurentas yra „Unity MLAPI“ karkasas. Kiti karkasai turi savų privalumų ir trūkumų. Virtualios realybės daugelio žaidėjų karkasas nėra idealus sprendimas visiems daugelio žaidimams, kadangi šitas karkasas yra labai specifikuotas. Mažas funkcijų kiekis gali lemti kūrėją tarp karkasų. Mažas karkasas leidžia plėsti jį kaip kūrėjas nori - pridedant naujų funkcijų arba perdaryti visą karkasą vienam specifiniam sprendimui. Didelis karkasas leidžia kurti daugelio žaidėjų žaidimą per daug nesirūpinant karkaso plėtimu.

Išanalizavus galimus karkaso konkurentus buvo pasirinktas vienas virtualios realybės žaidimas, kuris skirtas išanalizuoti galimas žaidimų funkcijas ir palyginti su virtualios realybės daugelio žaidėjų kovos žaidime. Konkurento žaidimo analizė padės išsiaiškinti kokios funkcijos galėtu būti sukurtos ir sinchronizuojamos pavyzdiniame žaidime.

1.1.5.2. Žaidimas

Populiariausias virtualios realybės žaidimas „Half-Life: Alyx“ (13) yra puikus pavyzdys kurį galima išanalizuoti. Žaidėjas yra perkeliamas į nematytą virtualios realybės pasaulį, kuriame pagrindinis tikslas yra kovoti su ateiviu rase naudojant ginklus ir sprendžiant įvairias dėliones. Žaidimas pasižymi detalumu, kiekvienas daiktas gali būti paimtas ir panaudotas. Žaidime yra naudojamos dėlionės kurios gali būti naudojamos tik virtualios realybės žaidimuose – durų atidarymas naudojant virtualias rankas, vėliavėlių išvengimas.



1.5 pav. Žaidimas „Half-Life: Alyx“

Virtualios realybės kovų žaidimas yra kuriamas virtualios realybės daugelio žaidėjų karkasui pademonstruoti, todėl žaidimo negalime lyginti su išbaigtais produktais kurie yra rinkoje. Žaidimas pateikia paprastą funkcionalumą, kaip daiktų paėmimą, durų atitarimą ir sinchronizaciją, skirtą naudojamam karkasui demonstruoti.

Virtualios realybės žaidimas „Half-Life: Alyx“ padėjo atskleisti, kad pavyzdinis kovos žaidimas turi labai ribotą funkcionalumą. Žaidimui būtų galima pridėti įvairių funkcijų kurios pagyvintu žaidimą - ginklai, įvairus daiktai, žaidėjo gydymas naudojant švirkštus, paimti daiktus iš atstumo ir kitokios funkcijos. Kadangi žaidimas yra skirtas karkaso demonstracijai, žaidimui užtenka pradinių funkcijų kurios padės atvaizduoti virtualios realybės daugelio žaidėjų karkaso funkcionalumą.

1.1.6. Prototipai ir pagalbinė informacija

Kuriant daugelio žaidėjų karkasą buvo naudojamas „UDP“ (14) komunikacijos protokolas kuris realizuotas naudojant „.Net Sockets“ (15) biblioteką. Buvo naudojama „.Net“ (16) dokumentacija. „Sockets“ biblioteka buvo naudojama duomenų siuntimui į serveri ir atgal.

Kuriant virtualios realybės žaidimą nebuvo naudojamos jokiomis panašiomis realizacijomis ar prototipais, bet buvo naudojami kiti papildomi šaltiniai – „Unity“ variklio dokumentacija, „.Net“ dokumentacija, nemokami 3D erdvės modeliai bei garso takelis.

1.1.7. Ištekliai, reikalingi sistemai sukurti

Virtualios realybės žaidimo kūrimui reikalinga techninė kompiuterinė įranga bei virtualios realybės įrenginys. Kompiuterinė įranga privalo turėti grafikos procesorių (GPU) kuris paliko virtualios realybės įrenginius. Grafikos procesorius turi būti naujesnis arba lygiavertis NVIDIA GTX 970 / AMD R9 290 grafikos procesoriams.

Įgyvendinti daugelio žaidėjų karkasą testavimo metu yra reikalingas lokalaus tinko per kurį būtų galima atlikti karkaso testavimą.

Serverio daliai įgyvendinti buvo pasirinkta „.NET“ platforma, C# programavimo kalba ir „UDP“ komunikacijos protokolas. „.NET“ platforma turi išsamią dokumentaciją bei nemažą atviro kodo bendruomenę. „UDP“ komunikacijos protokolas turi nemažai dokumentacijos bei įvairiausių naudojimo pavyzdžių.

Virtualios realybės žaidimo kūrimui buvo pasirinktas „Unity“ žaidimų kūrimo variklis. „Unity“ žaidimų variklis suteikia įvairius įrankius kurie padeda kurti žaidimus ir interaktyvias aplikacijas. „Unity“ žaidimų variklis palaiko tik C# programavimo kalbą, dėl šios priežasties žaidimo komponentai yra programuojami naudojant C# programavimo kalbą. Projekto paruošimui virtualios realybės įrenginių palaikymui reikėjo įdiegti papildomų įskiepių – „Steam VR“ ir „VRTK“ įskiepius. Daugelio žaidėjų palaikymui yra naudojamas mano sukurtas karkasas.

„Steam VR“ - suteikia pagrindines virtualios realybės funkcijas, įrenginių įvestis ir išvestis.

„VRTK“ - suteikia pagrindines virtualios realybės funkcijas nepriklausant nuo pasirinktos platformos. Įskiepis naudojamas kurti virtualios realybės sąveikas su virtualios realybės pasaulyu nepriklastant nuo platformos pasirinkimo. Įskiepis sukuria abstrakcijos sluoksnį valdant platformos pateiktas įvestis ir išvesti.

Sistemos kodo bazei saugoti yra reikalingos dvi atskiros „GitHub“ (17) kodo talpyklos, skirtos saugoti sistemos serverio kodui ir kovų žaidimo kodui. Sistemos įgyvendinimui yra skiriama apie 400 valandų. Sistemos apimtis gavosi apie - 8000 kodo eilučių. Sistemai įgyvendinti reikia vieno programuotojo.

1.2. Galimybių analizė

1.2.1. Techninės galimybės

Karkaso realizacijai techninių kliūčių nėra. Pasirinkti ištekliai yra plačiai paplitę ir labai lengva rasti informaciją kuri yra reikalinga. Karkaso realizacijoje naudojamas „UDP“ protokolas yra plačiai

naudojamas tiesioginėms transliacijoms perduoti ir daugelio žaidėjų kūrimui. C# programavimo kalba yra naudojama įvairiuose projektuose.

Kovų žaidimo realizacijai techninių kliūčių nekyla. „Unity“ žaidimų variklis yra plačiai naudojamas, turi didelę dokumentaciją bei palaiko įvairius kūrimo įskiepius. Kovų žaidimo realizacijai buvo pasirinktas „VRTK“ įskiepis, kuris suteikia įvairių virtualios realybės įrenginių palaikymą.

1.2.2. Vartotojų pasiruošimo analizė

Norint naudotis daugelio žaidėjų karkasu reikia turėti papildomų žinių kaip reikia naudotis „Unity“ žaidimų varikliu ir „.NET“ platforma. Norint praplėsti karkaso funkcionalumą reikia mokėti programuoti su C# programavimo kalba.

Pavyzdinio kovų žaidimo naudojimui reikia mokėti naudoti „Unity“ žaidimų varikliu, kadangi žaidimas yra sukurtas naudojant minėtą žaidimų variklį.

2. Projektas

2.1. Reikalavimų specifikacija

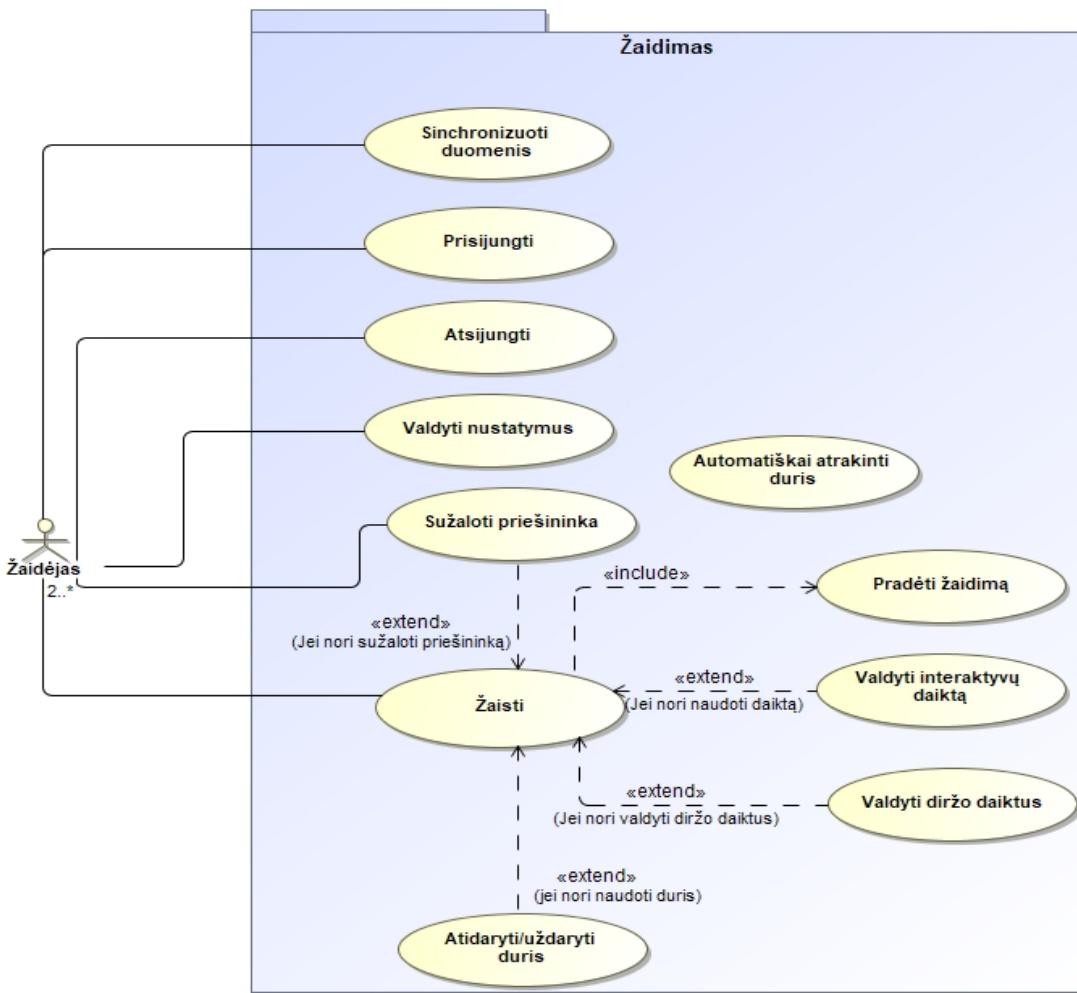
2.1.1. Komercinė specifikacija

Projektas užsakovo neturi. Tai yra universitetinis projektas, kuriamas bakalauro baigiamajam darbui. Projektą sugalvojo bei vykdo – universiteto studentas Eligijus Kiudys. Projektinio darbo naudotojai – virtualios realybės žaidimų kūrėjai kurie nori sukurti daugelio žaidėjų žaidimus.

Projekto biudžetas nėra nustatytas, kadangi tai yra asmeninis projektas. Numatyta projekto baigimo data – 2021 m. birželio 5d. Iki numatytos datos projektas turi būti suprogramuotas, ištestuotas ir paruoštas naudojimui.

2.1.2. Sistemos funkcijos

Sistemos funkciniai reikalavimai yra pateikiami UML diagramų pavidalu. (2.1 pav.) yra pateikiama daugelio žaidėjų karkaso pavyzdinio kovos žaidimo panaudojimo atvejų diagrama.



2.1 pav. Pavyzdinio kovos žaidimo panaudojimo atvejų diagrama.

Pavyzdinis kovos žaidimas ir naudojamas karkasas yra viena didelė posistemė. Žaidimo sistema yra atsakinga už karkaso funkcionalumo atvaizdavimą. Ši sistema suteikia galimybę žaidėjams žaisti daugelio žaidėjų žaidimą – žaidėjai gali prisijungti prie žaidimo, atsijungti iš žaidimo, valdyti pateiktus žaidimo daiktus ir kovoti išlikimo kovoje. Kūrėjui pavyzdinis žaidimas suteikia sistemą

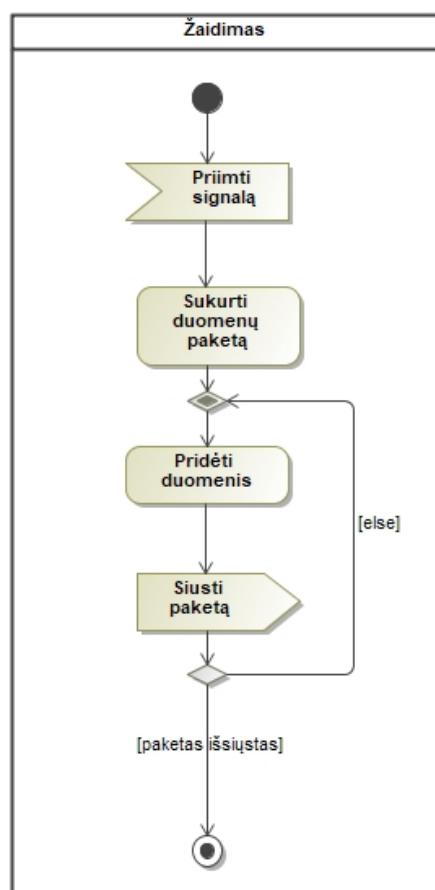
kuria gali remtis naudojant daugelio žaidėjų karkasą - synchronizuojant funkcijas kaip interaktyvių daiktų valdymas.

Panaudojimų atvejų aktorius – žaidėjas žaidžia pavyzdinį žaidimą naudojant pavaizduotas žaidimo funkcijas.

Kūrėjas – žmogus kuris kuria žaidimus, jis diagramoje nebuvo pavaizduotas, kadangi jis kuria kitą žaidimą remiantis pavyzdinio žaidimo funkcionalumu.

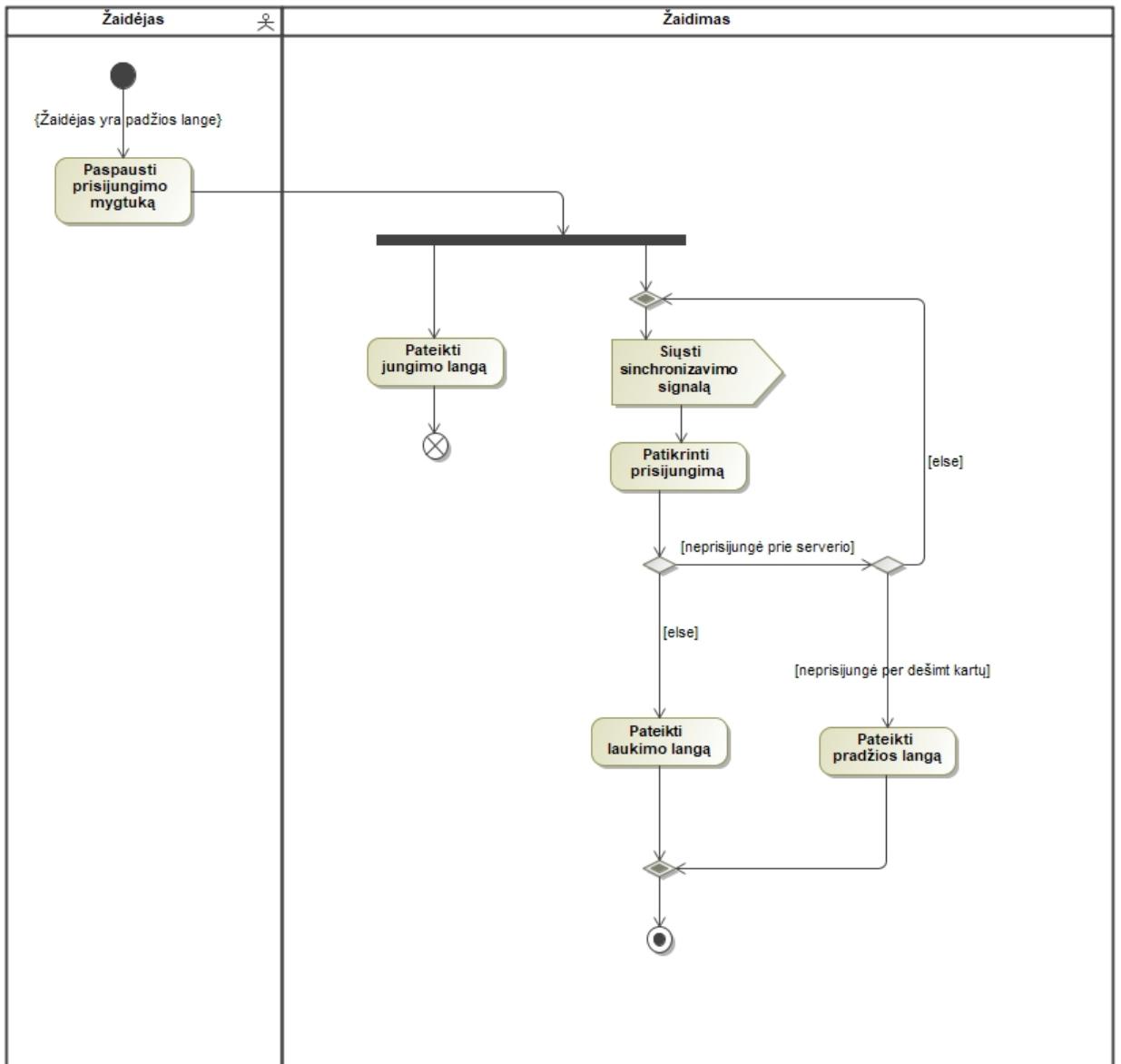
Posistemui panaudojimo atvejai yra pateikiami UML veiklos diagramomis, kurios nurodo supaprastintus panaudojimo atvejų vykdymo scenarijus, nurodant, kokie veiksmai sistemoje yra atliekami aktoriaus ir kaip sistema reaguoja į jo atliktus veiksmus. 2.2 pav. – 2.17 pav. paveikslėliuose yra pateikiami administracinių posistemės panaudojimo atvejų veiklos diagramos.

Sistemos panaudojimo atvejai yra pateikiami UML veiklos diagramomis, kurios parodo paprastą panaudojimo atvejų scenarijų. Nurodant kaip sistema turėtų reaguoti žaidėjui atliktus veiksmus. Paveikslėliuose 2.2 pav. - 2.12 pav. yra pavaizduojama žaidimo panaudojimo atvejus naudojant veiklos diagramas.



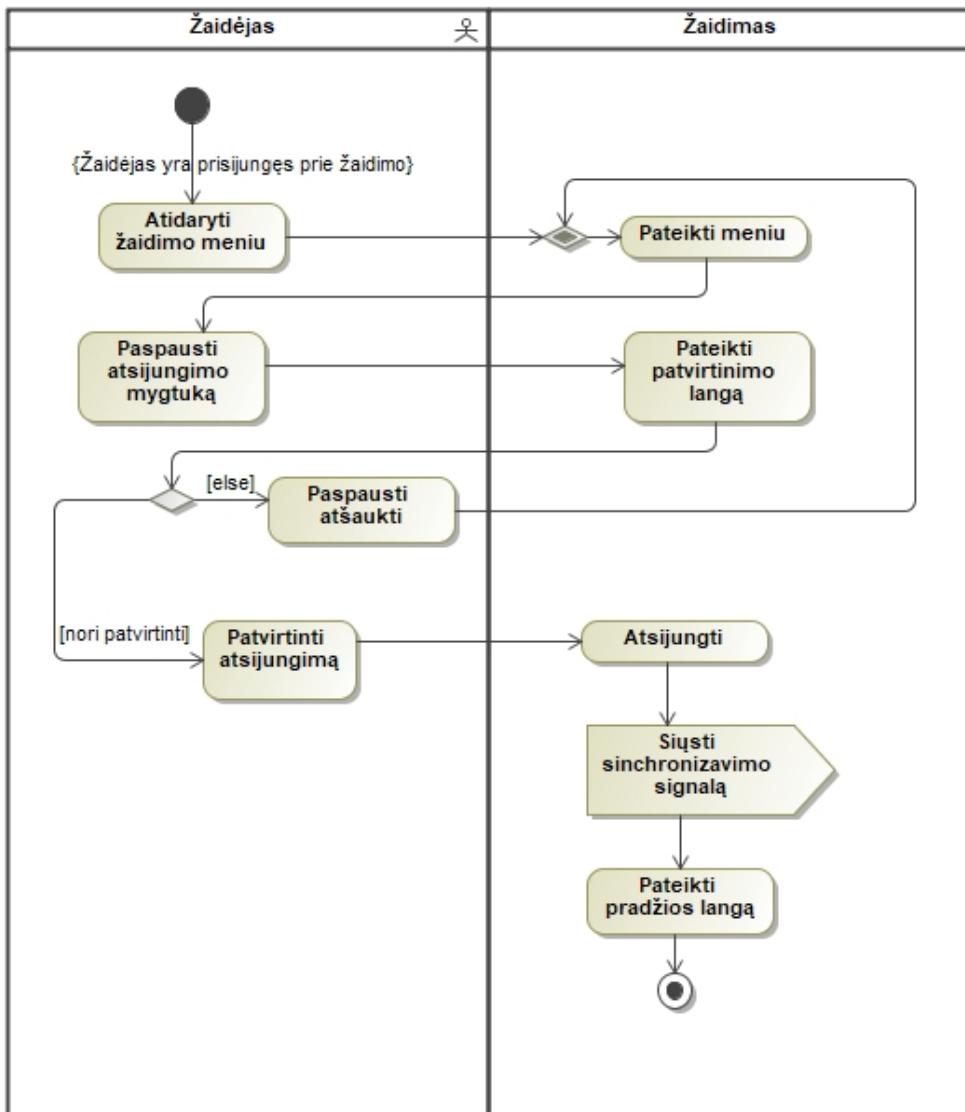
2.2 pav. „Synchronizuoti duomenis“ panaudojimo atvejo veiklos diagrama

Synchronizavimo signalas yra naudojamas synchronizuoti žaidimą su serverio (2.2 pav.). Funkcijai gavus signalą su duomenis yra sukuriamas naujas duomenų paketas. Gauti duomenys signalo metu yra pridėti į paketą po jo sukūrimo. Paketas su pridėtais duomenimis yra siunčiamas į serverį kuriamo paketas yra apdorotas serveryje. Apdorotas paketas yra siunčiamas atgal arba yra siunčiamas visiems žaidėjams.



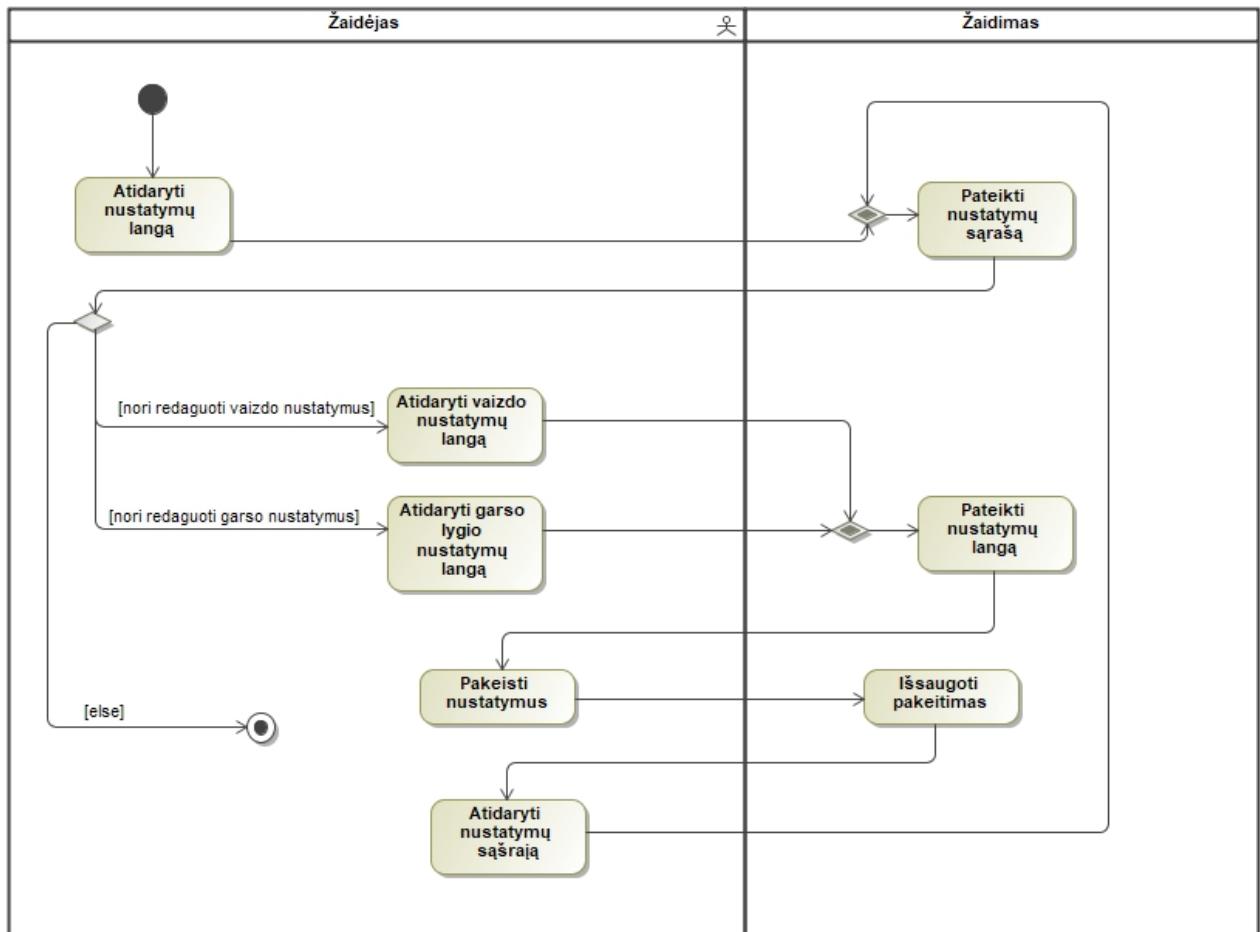
2.3 pav. „Prisijungti“ panaudojimo atvejo veiklos diagrama

Žaidimo pradžioje arba atsijungus nuo serverio yra pateikiamas pradžios langas per kurį galima prisijungti prie žaidimo (2.3 pav.). Žaidėjui paspaudus prisijungimo mygtuką, sistema parodo jungimosi langą ir bando jungtis prie serverio. Jungimosi metu yra siunčiamas synchronizavimo signalas (2.2 pav.). Gautas serverio rezultatas yra patikrinamas. Negavus rezultato iš serverio per 10 kartų žaidimas stabdo jungimosi procesą ir pateikia pradžios. Žaidėjui prisijungus sėkmingai prie serverio yra pateikiama pradžios scena. Kai serveri yra pilnas - žaidėjui yra pateikiamas pradžios langas kur gali bandyti prisijungti iš naujo.



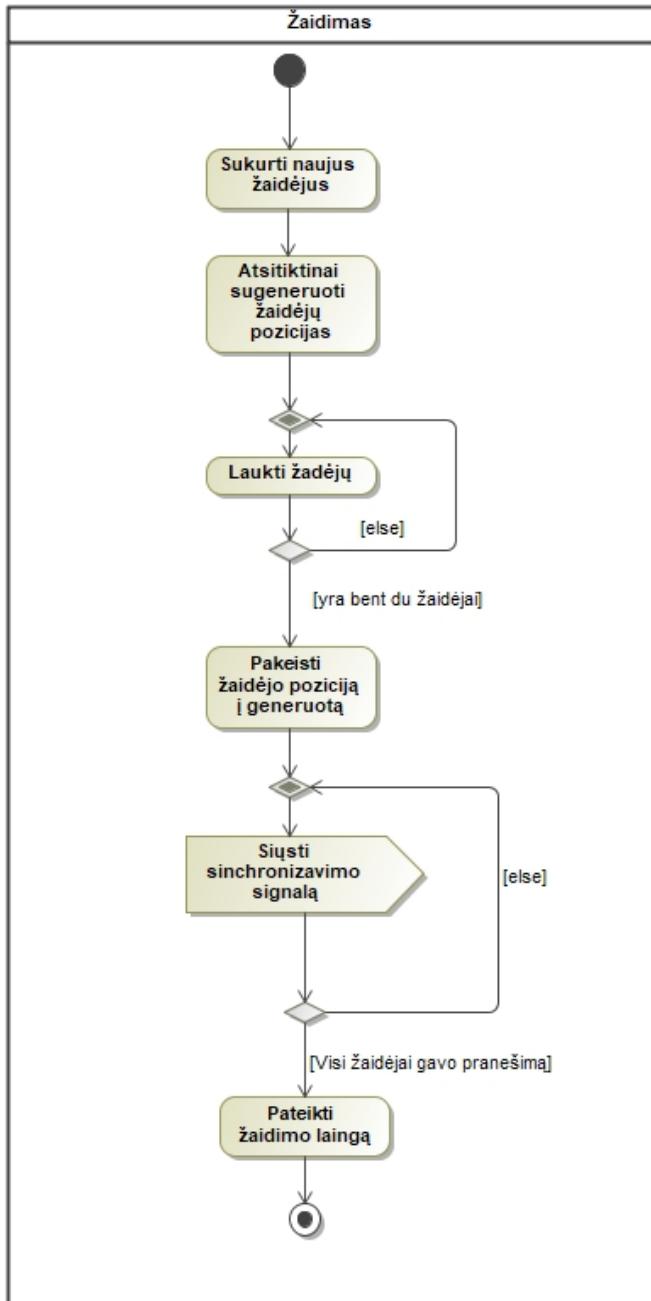
2.4 pav. „Atsijungti“ panaudojimo atvejo veiklos diagramma

Jei prisijungęs naudotojas nori atsijungti iš žaidimo, žaidėjas atidaro žaidimo meniu kur paspaudžia ant atsijungimo mygtuko. Paspaudus ant mygtuko yra pateikiamas patvirtinimo langas kur žaidėjas gali pasirinkti ar jis tikrai nori atsijungti, žaidėjui paspaudus atšaukimo mygtuką patvirtinimo langas yra pakeičiamas į meniu langą. Žaidėjui paspaudus patvirtinimo mygtuką pirmiausia žaidėjas atsijungia nuo žaidimo. Žaidėjui atsijungus nuo žaidimo yra siunčiamas synchronizavimo signalas į serverį (2.2 pav.). Išsiuntus synchronizavimo signalą sistema pateikia pradžios langą.



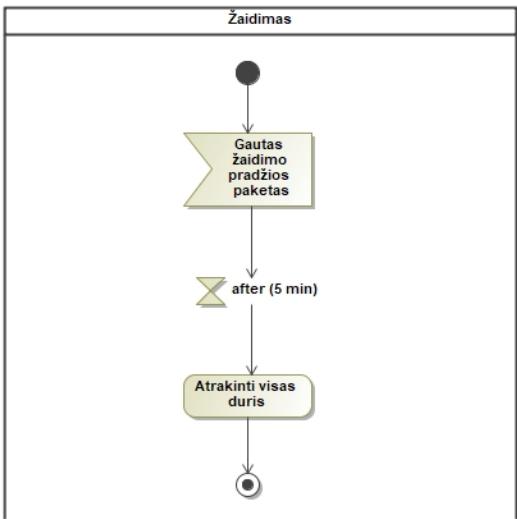
2.5 pav. „Valdyti nustatymus“ panaudojimo atvejo veiklos diagrama

Žaidimo pradžioje arba atsivertus žaidimo meniu galima atidaryti nustatymų langą. Paspaudus ant nustatymo mygtuko yra pateikiamas nustatymų sąrašas. Jame galimą pasirinkti nustatymus kurie bus keičiami. Naudotojui pasirinkus nustatymus, kuriuos jis keis, yra spausdžiama ant pasirinkto mygtuko. Parsirinkus vaizdo nustatymus žaidėjas gali pasirinkti kokią vaizdo kokybę nori pakeisti. Pasirinkus garso nustatymus naudotojas gali valdyti pagrindinį garsą. Pakeitus nustatymus naudotojas gali grįžti į nustatymų langą.



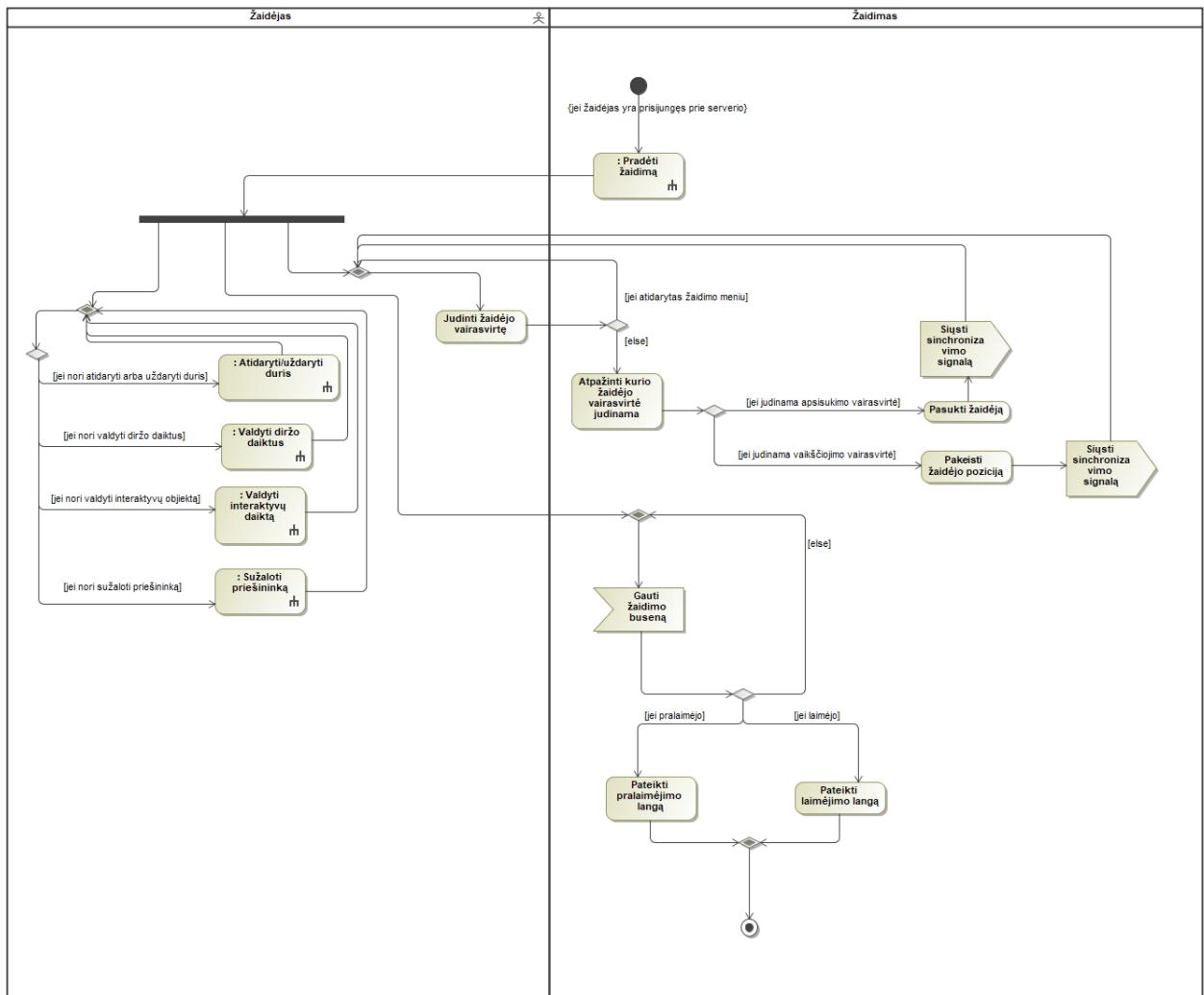
2.6 pav. „Pradėti žaidimą“ panaudojimo atvejo veiklos diagrama

Žaidėjui prisijungus prie serverio yra pradedamas žaidimas (2.8 pav.). Pradžios metu yra sukuriami nauji žaidėjai, kurie yra skirti prie žaidimo prisijungusius žaidėjus. Sistema atsitiktinai sugeneruoja žaidėjo poziciją. Žaidėjai laukia žaidimo laukimo lange su kitais žaidėjais kol prisijungia bent du žaidėjai. Prisijungus reikiama žaidėjų kiekui yra siunčiamas synchronizavimo signalas (2.2 pav.). Visus žaidėjus susinchronizavus yra paleidžiamas žaidimas.



2.7 pav. „Automatiškai atrakinti duris“ panaudojimo atvejo veiklos diagrama

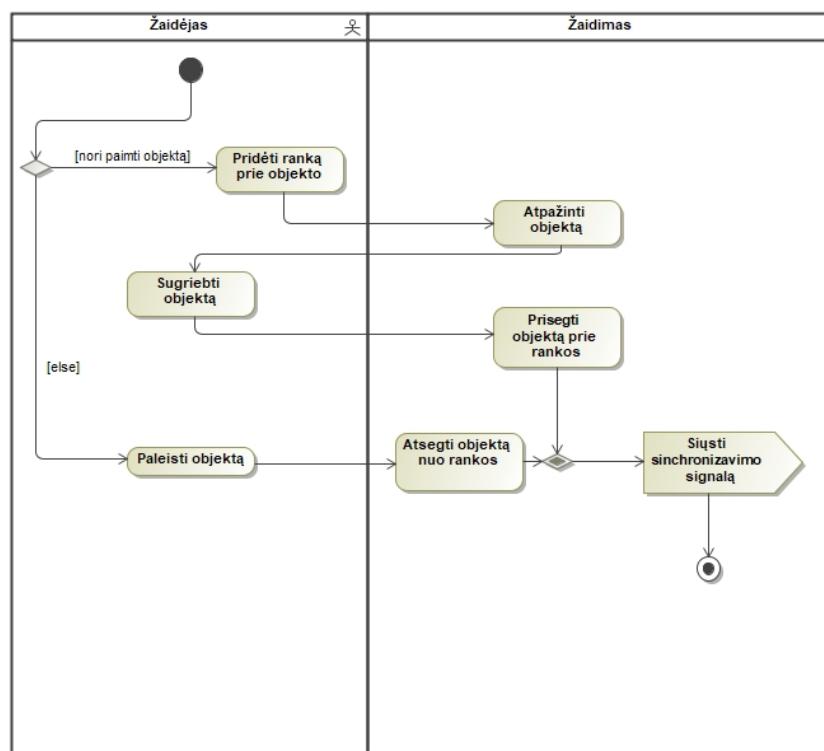
Žaidimo pradžioje yra duotos penkios minutės. Šio momentu žaidėjas yra patalpintas į namą kuriame turi susirasti įrankį su kuriuo eis į kovą. Praėjus penkioms minutėms visiems žaidėjams durys yra atrakinamos, tuomet jie gali eiti iš namo į kovą prieš kitus žaidėjus.



2.8 pav. „Žaisti“ panaudojimo atvejo veiklos diagrama

Prisijungus prie serverio yra paleidžiamas pagrindinis žaidimo panaudojimo atvejis (2.8 pav.). Paleidus panaudojimo atvejį yra pirmiausiai vykdomas žaidimo pradžios panaudojimo atvejis (2.6 pav.). Sulaukus žaidėjų ir juos susinchronizavus pasileidžia pagrindinis žaidimas. Žaidimas susideda iš trijų lygiagrečių dalių – žaidėjo vaikščiojimo, žaidėjo pasirinkto veiksmo ir žaidimo būsenos atnaujinimo.

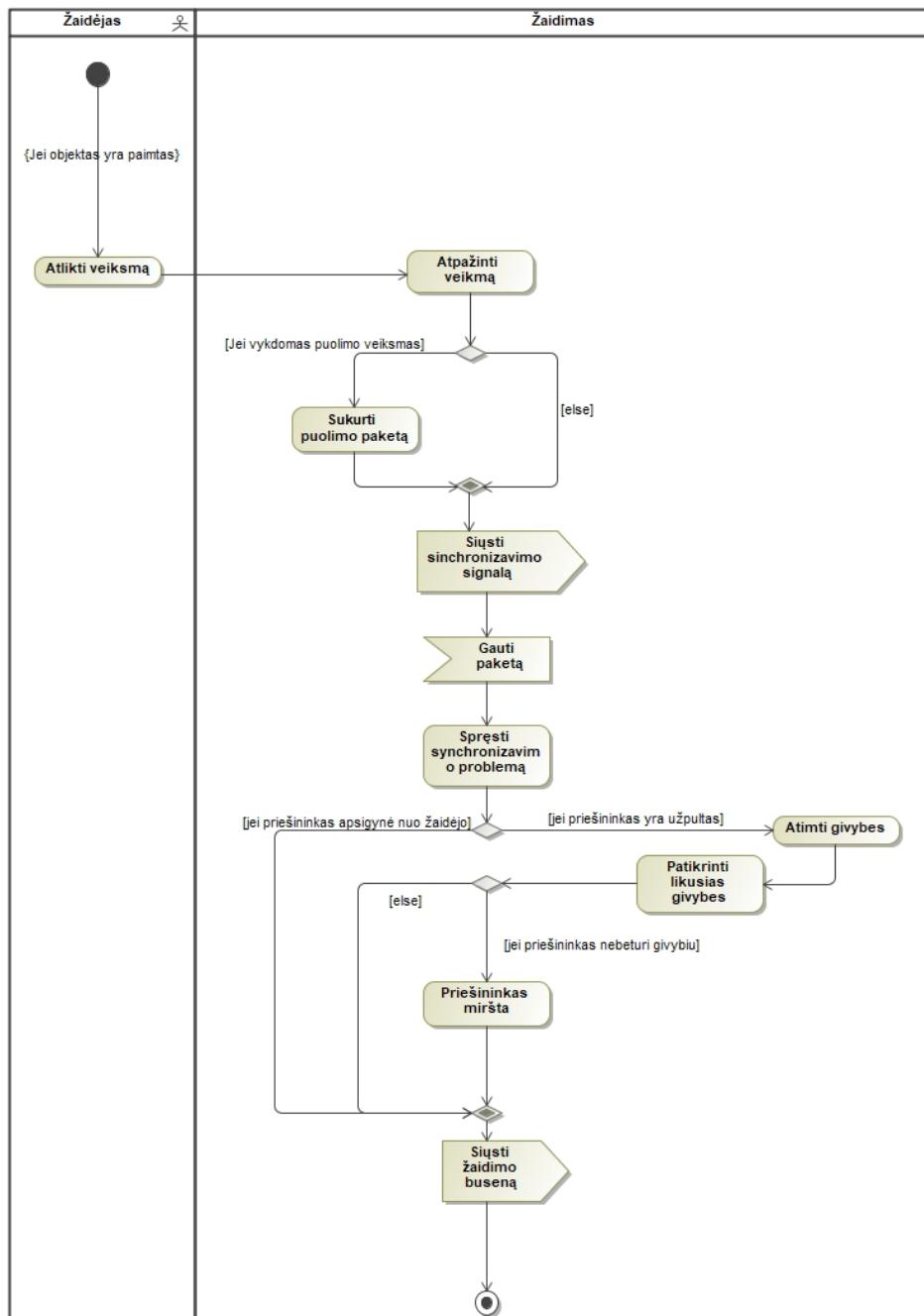
1. Žaidėjas gali atliskti įvairius veiksmus žaidimo metu. Veiksmai gali būti - durų atidarinėjimas ir uždarinėjimas jei jos yra atrakintos (2.12 pav.), diržo inventoriaus valdymas – įdėti arba išimti daiktus iš jo (2.11 pav.), valdyti interaktyvų daiktą – daiktų pakėlimas ir jų pametimas (2.10 pav.), sužaloti priešininką – žaidėjas su daiktu gali užpulti priešininką (2.9 pav.) ir gali atsijungti iš žaidimo (2.4 pav.) visi paminėti veiksmai yra kaip atskiri panaudojimo atvejai.
2. Žaidėjas lygiagrečiai gali vaikščioti ir atliskti įvairius veiksmus. Žaidėjas vaikšto naudojant vairasvirtę. Žaidėjui įsijungus žaidimo meniu jis nebegali vaikščioti. Žaidėjui galint vaikščioti yra tikrinama kuri vairasvirtė yra judinama. Judinant apsisukimo vairasvirtę žaidėjas yra pasukamas į vieną arba kitą pusę. Pakreipus vairasvirtę į dešinę pusę žaidėjas pasisuka į dešinę pusę. Pakreipus vairasvirtę į kairę pusę žaidėjas pasisuka į kairę pusę. Žaidėjui pasisukus į vieną arba kitą pusę yra siunčiamas sinchronizacijos signalas (2.2 pav.). Judinant vaikščiojimo vairasvirtę žaidėjas eina į tą kurią į kurią yra pakreipta žaidėjo vairasvirtė. Žaidėjui judant yra siunčiamas sinchronizacijos signalas (2.2 pav.). Žaidėjas gali vaikščioti kol žaidimas nėra baigtas.
3. Žaidimo Būsena atsinaujina lygiagrečiai žaidžiant žaidėjui. Gautos žaidimo būsenos gali būti dvi – laimėjimas ir pralaimėjimas. Žaidėjui laimėjus žaidimą yra pateikiamas laimėjimo langas. Žaidėjui pralaimėjus yra pateikiamas pralaimėjimo langas. Žaidimui laimėjus arba pralaimėjus žaidėjas gali atsijungti iš žaidimo.



2.9 pav. „Valdyti interaktyvų daiktą“ panaudojimo atvejo veiklos diagrama

Naudotojas gali pasisiimti arba paleisti daiktą žaidimo metu. Naudotojui norint paleisti laikomą daiktą reikia atleisti laikymo mygtuką. Paleidus daiktą jis yra atsegamas nuo rankos ir yra išsiunčiamas daikto sinchronizacijos signalas (2.2 pav.) kadangi daiktas buvo pamestas iš rankos. Naudotojui pasirinkus paimti daiktą jis turi pridėti ranką prie daikto kurį nori paimti. Prisilietus

rankai prie norimo daikto jis yra atpažįstamas sistemos. Žaidėjas paspaudžia sugriebimo mygtuko kai ranką yra prie objekto. Sugriebus objektą jis yra prisegamas prie rankos ir yra siunčiamas synchronizavimo signalas (2.2 pav.) objekto synchronizacijai.



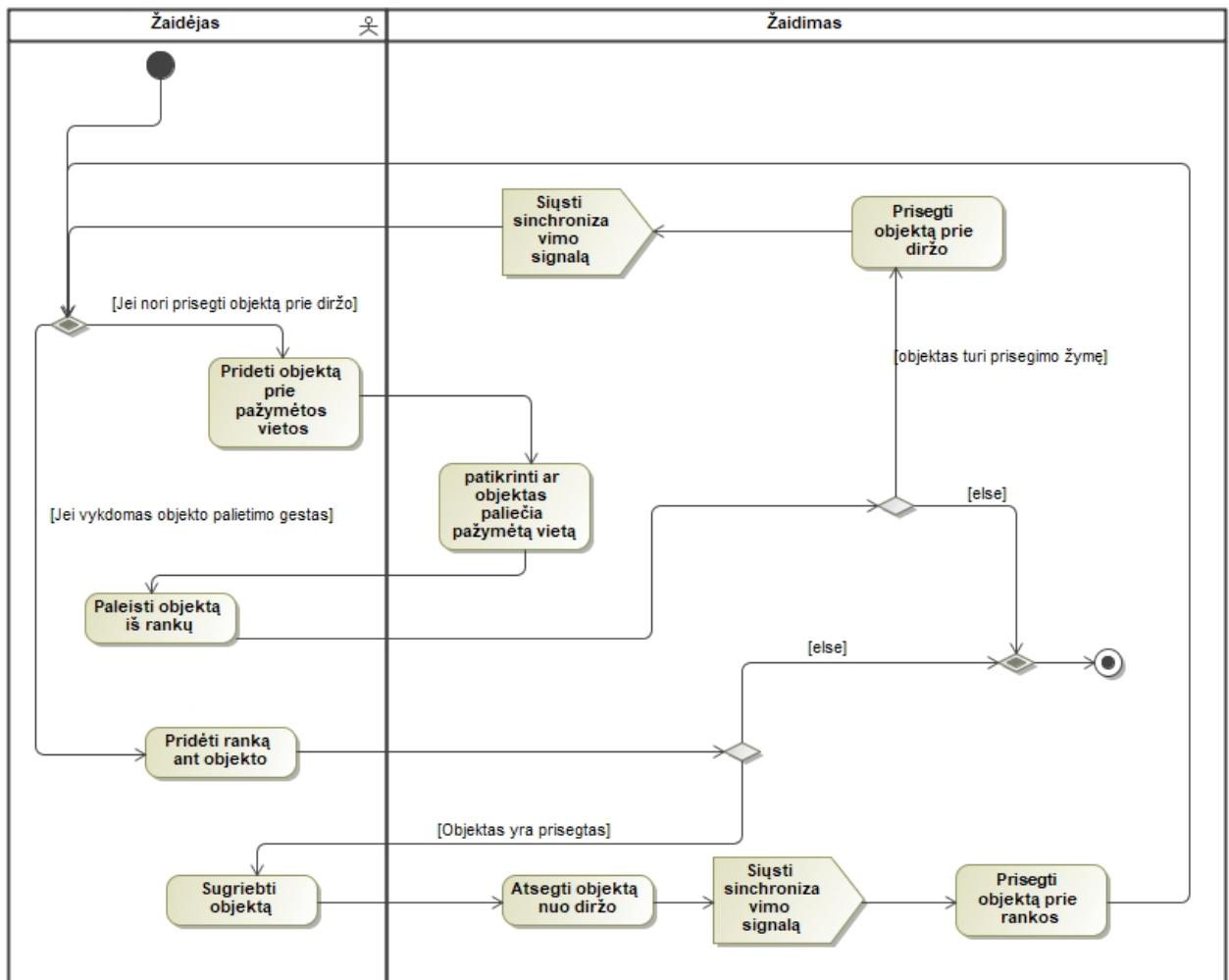
2.10 pav. „Sužaloti priešininką“ panaudojimo atvejo veiklos diagrama

Durims atsirakinus žaidėjai išeiti į lauką ir gali užpulti vienas kitą. Žaidėjai gali gintis arba pulti kitus žaidėjus.

1. Žaidėjui pasirinkus kitų žaidėjų puolimą. Žaidėjas trenkia su daiktu į kitą žaidėja. Kitam žaidėjui nepavykus apsiginti ataka įvyksta sėkmingai ir yra sukuriamas puolimo paketas, kuris yra siunčiamas į serverį. Serveriui gavus paketą yra spendžiama synchronizavimo problema – yra tikrinama ar iš abiejų žaidėjų yra gaunamas vienodas rezultatas. Jei įvyksta klaida ataka būna parodoma, kad žaidėjas apsigynė. Jei atakai įvykus žaidėjas buvo užpultas tada yra sumažinamos žaidėjo gyvybės. Žaidėjo gyvybėms pasiekus nulį žaidėjas miršta. Po žaidėjo

mirties arba atakos atlikimo yra siunčiama žaidimo būsena žaidėjams. Sėkmingai įvykus atakai ir žaidėjui apsigynus yra siunčiama žaidimo būsena.

2. Žaidėjui pasirinkus apsiginti nuo kitų žaidėjų. Žaidėjas ginasi su turimu ginklu apsiginti nuo kitų žaidėjų kurie bando jį užpilti. Žaidėjui sėkmingai apsigynus yra siunčiamas synchronizavimo signalas serveriui. Serveriui gavus atakos synchronizavimo paketą yra spendžiama synchronizacijos problema gavus atakos paketą ir apsigynimo paketa problema yra sprendžiama palyginus laiką. Paketas kuris buvo atsiųstas ankšciau nustato veiksmą kuris būna vykdomas. Žaidėjui apsigynus nuo kitų yra siunčiama žaidimo būsena. Žaidėjui nespėjus apsiginti jam yra atimamos gyvybės. Gyvybėms nukritus iki nulio žaidėjas miršta. Atėmus gyvybes arba žaidėjui mirus yra siunčiama žaidimo būsena.

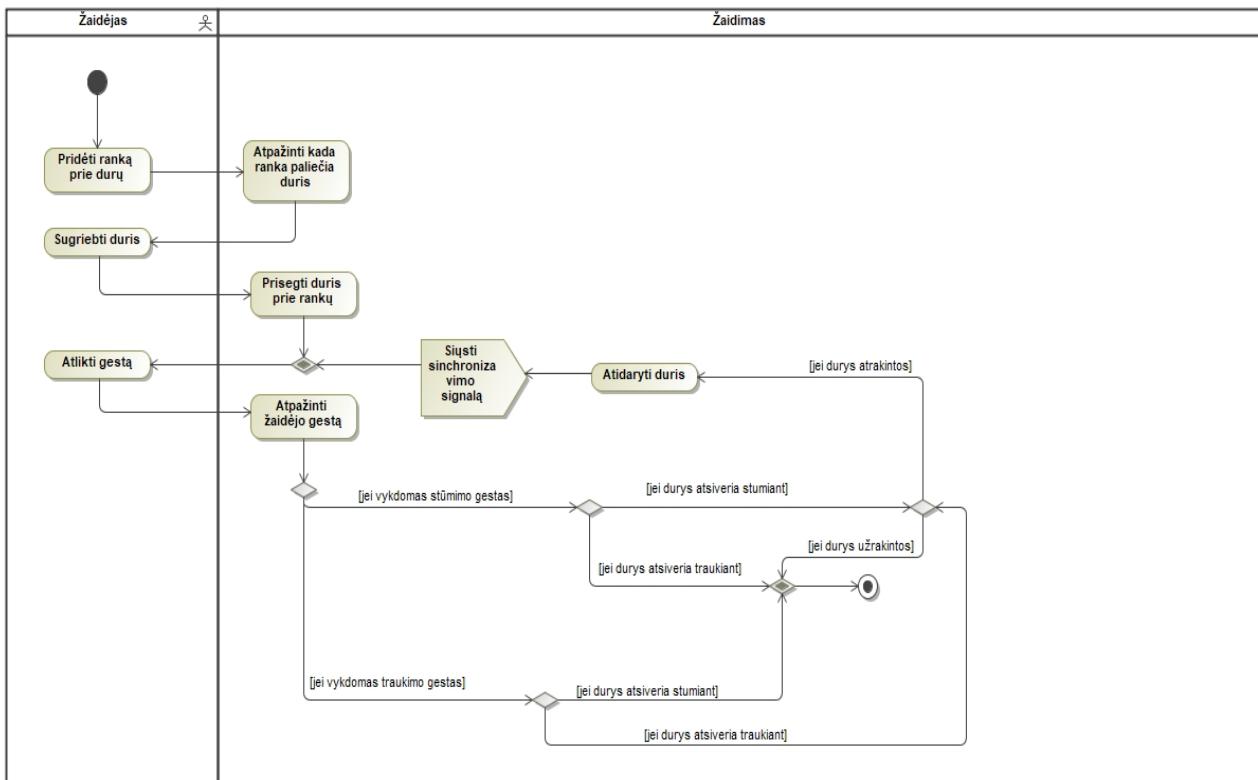


2.11 pav. „Valdyti diržo daiktus“ panaudojimo atvejo veiklos diagrama

Naudotojas žaidimo metu gali valdyti savo diržo inventoriją. Žaidėjas gali išimti daiktą arba pridėti daiktą į inventoriją.

1. Žaidėjui norint pridėti daiktą į inventoriją jis turi laikyti daiktą, kurį nori pridėti į inventoriją. Daiktas turi būti priliestas prie pažymėtos prisegimo vienos. Pridėjus daiktą prie pažymėtos vienos yra patikrinama ar objekto liečia pažymėtą vietą. Žaidėjui pametus daiktą iš rankos kai daiktas liečia prisegimo vietą yra patikrinama ar daiktas turi prisegimo žymę. Objektui turint prisegimo žymę jis yra prisegamas prie diržo ir išsiunčiamas synchronizacijos signalas (2.2 pav.). Kitu atveju daiktas nėra prisegamas prie diržo daiktų.

2. Žaidėjui norint imti daiktą kuris yra prisegtas prie diržo, reikia paimti daiktą kuris yra prisegtas prie diržo. Naudotojas turi pridėti ranką prie objekto kurį nori paimti. Jei objektas yra prisegtas prie diržo jis gali būti atsegtas nuo diržo. Žaidėjui paėmus daiktą jis yra atsegamas nuo diržo. Atsegus objektą nuo diržo yra siunčiamas synchronizacijos signalas (2.2 pav.). Galiausiai daiktas yra prisegamas prie rankos.



2.12 pav. „Atidaryti/uždaryti duris“ panaudojimo atvejo veiklos diagrama

Žaidimo metu žaidėjas gali atidarinėti ir uždarinėti duris jei jos yra atrakintos. Žaidėjui norint atidaryti ar uždaryti duris reikia pridėti ranką prie durų ir jas sugriebti. Žaidėjui neprilietus rankos prie durų griebiant niekas neįvyks. Žaidėjui sugriebus duris jos yra prisegamos prie rankos. Žaidėjui atliekant gestą durų atidarymo gestą, durys varstosi priklausant nuo jų konfigūracijos. Žaidėjui stumiant duris kai jas reikia traukti jos neatsidarys. Žaidėjui traukiant duris kai jos sukonfigūruotos, kai jas reikia stumti jos neatsidarys. Žaidėjui bandant atverti užrakintas duris jos neatsivers. Žaidėjui reikia atligli gestą pagal konfigūraciją, kad durys atsivertu. Durims atsivérus yra siunčiamas synchronizavimo signalas (2.2 pav.).

2.1.3. Vartotojo sasajos specifikacija

Daugelio žaidėjų karkasui vartotojo sasajos specifikavimai nebuvo iškelti.

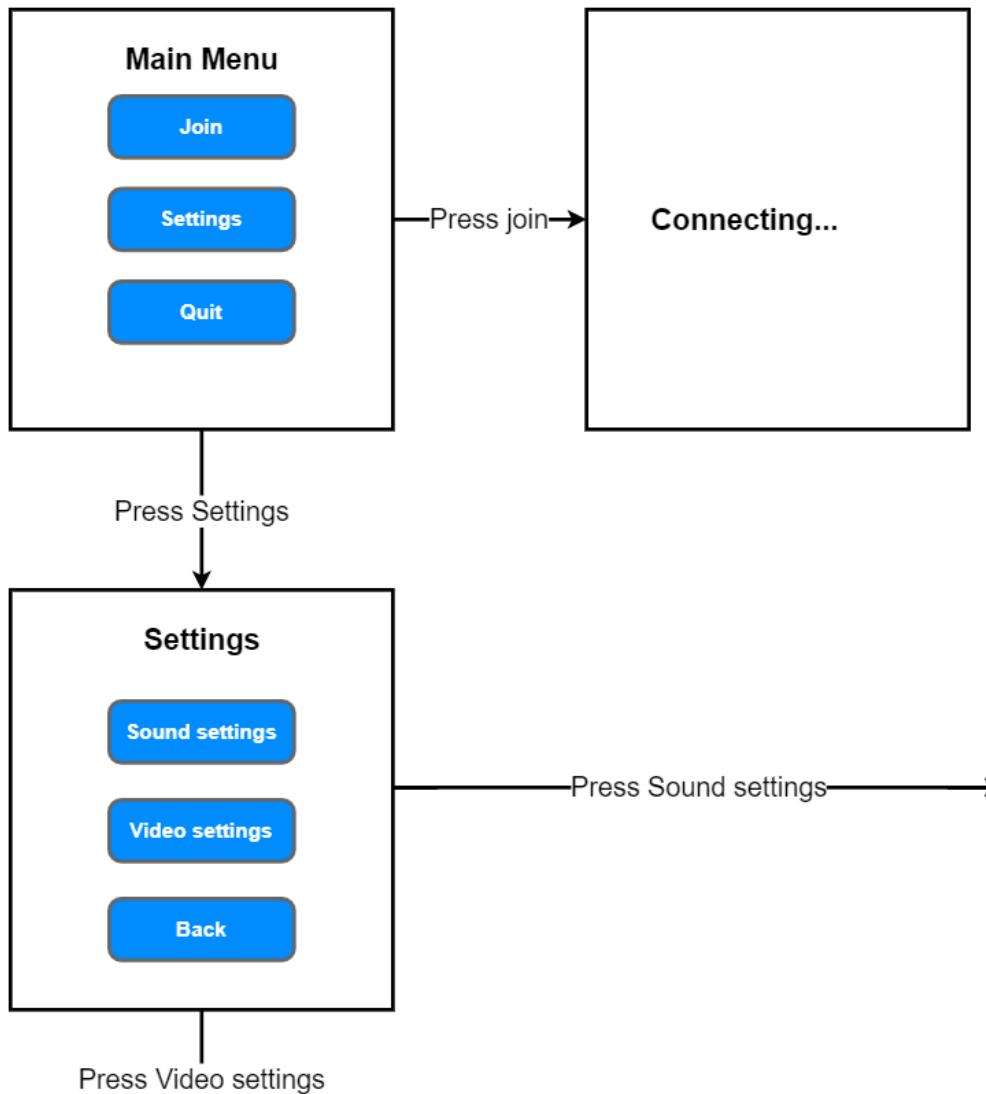
Virtualios realybės žaidimų sasajos kūrimui yra naudojamos įvairios naudotojo sasajos gaires.

Remiantis „Oculus“ naudotojo sasajos gaires (18) buvo sudarytos naudotojo sasajos gairių sąrašas.

Žemiau pateiktas naudotojo sasajai rekomenduojamas gairių sąrašas (gairių išdėstymo tvarka nėra svarbi).

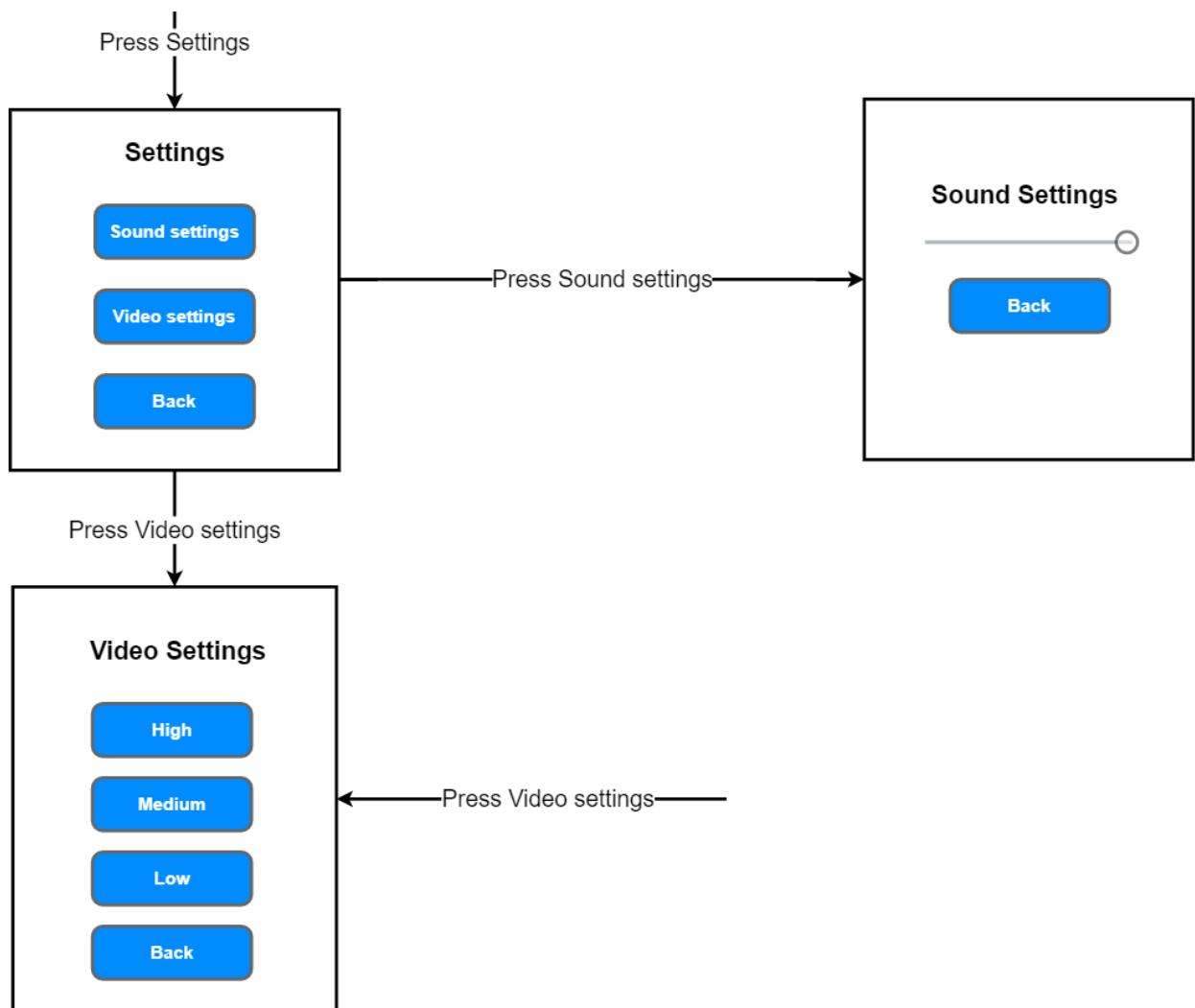
- Judantys naudotojo sasajos langai turi šiek tiek lėčiau judėti nei juda naudotojo vaizdas.
- Negalima pateikti statinio naudotojo lango kuris bus rodomas per visą vaizdą.
- Naudotojo sasajos langas turėtų būti išlenktas.
- Nenaudoti tokį sasajos langų kurie yra labai arti arba prilipę prie naudotojo vaizdo.
- Naudotojo sasajos langai negali būti atvaizduoti jei sasajos langas yra užstojamas kito daikto.

Sistemos eskizu kūrimui buvo naudojamas „Draw.io“ įrankis skirtas įvairių eskių ir diagramų braižymui. Eskizai naudojami naudotojo sąsajos kūrimui. Eskizų kiekis nėra didelis, kadangi žaidimuose naudotojo sąsaja yra naudojama žaidimo valdymui.



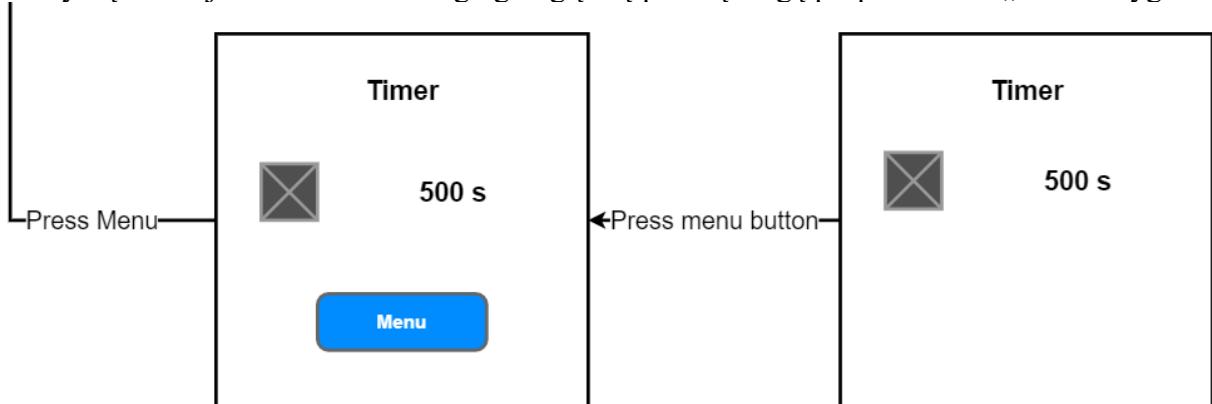
2.13 pav. Pradžios langų eskizas

Pirmiausia naudotojui paleidus žaidimą yra pateikimas žaidimo pradžios langas (2.13 pav.). Žaidėjui paspaudus „Join“ mygtuką jis yra nukreipiamas į prisijungimo langą. Norint keisti žaidimo nustatymus, naudotojas turi paspausti „Settings“ mygtuką. Naudotojas gali grįžti iš kiekvieno lango į pagrindinį langą paspaudus ant „Back“ mygtuko.



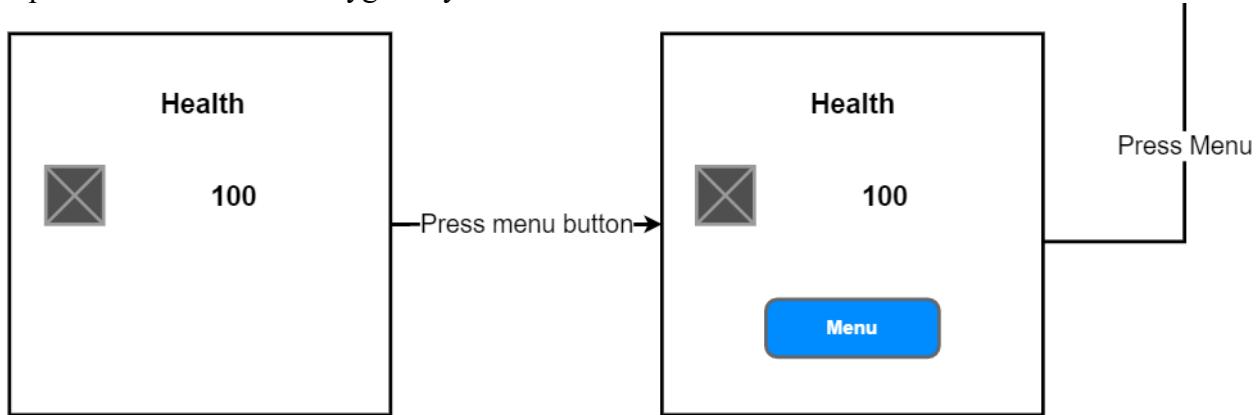
2.14 pav. Žaidimo nustatymų eskizas pradžios lange

Žaidėjui atidarius žaidimo nustatymų langą (2.14 pav.) yra pateikiami galimi žaidimo nustatymai. Naudotojui paspaudus ant „Sound settings“ jis gali keisti pagrindinį žaidimo garsą valdant. Naudotojui pasirinkus keisti vaizdo nustatymus jis turi paspausti ant „Video settings“ mygtuko. Naudotojas yra peradresuojamas į vaizdo nustatymų langą kur gali pasirinkti vieną iš vaizdo nustatymų. Žaidėjas iš kiekvieno lango gali grįžti į praetą langą paspaudus ant „Back“ mygtuko.



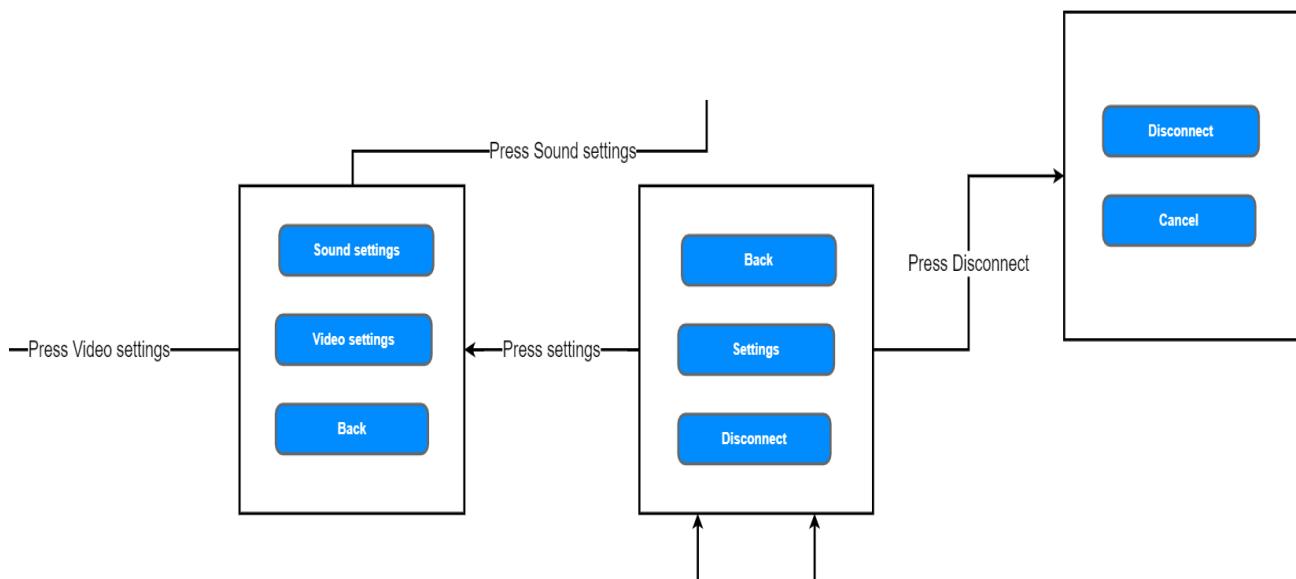
2.15 pav. Žaidėjo laikrodžio eskizas

Žaidėjui prisijungus prie žaidimo yra įjungimas laikrodis (2.15 pav.) kuris parodo likusį laiką iki durų atrakinimo. Naudotojui paspaudus meniu mygtuką yra pateikiamas mygtukas apačioje. Paspaudus ant atsiradusio mygtuko yra atidaromas žaidimo meniu.



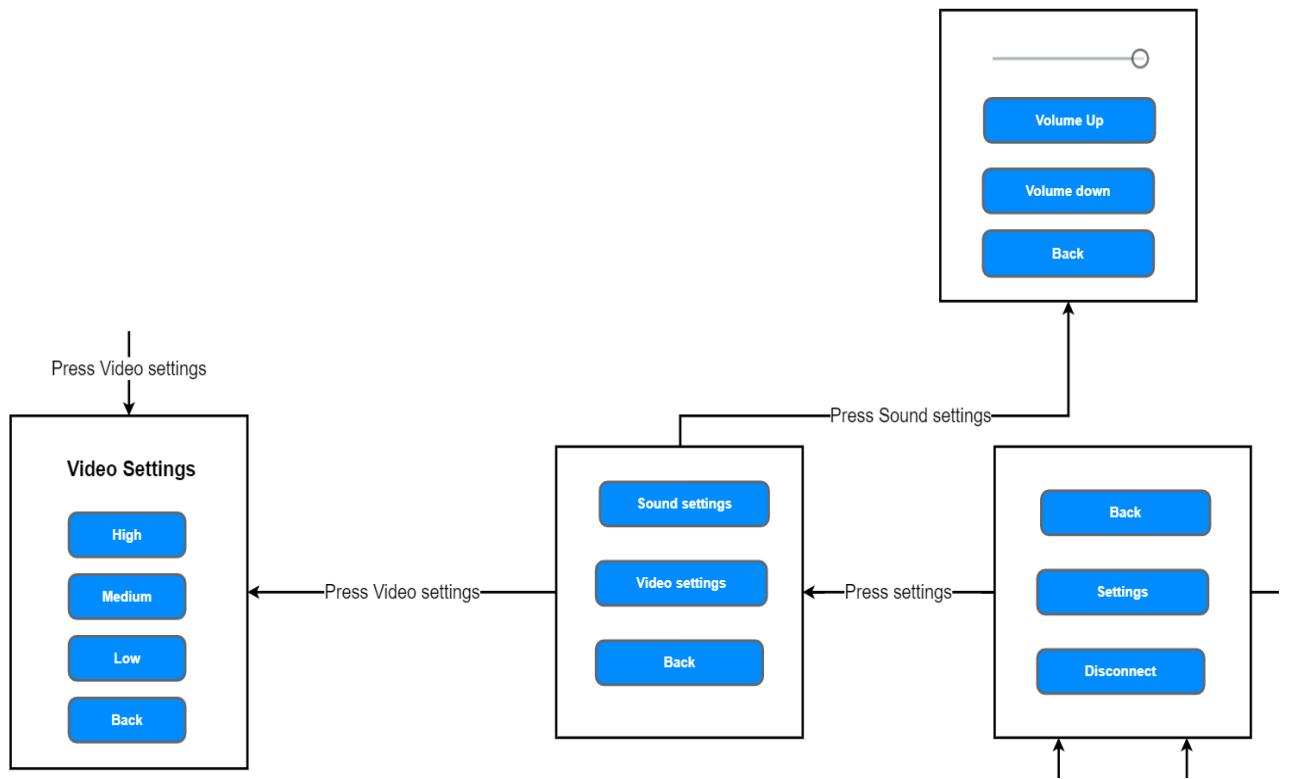
2.16 pav. Žaidėjo laikrodžio eskizas

Žaidimui atrakinus duris laikmačio langas yra pakeičiamas į gyvybių langą (2.16 pav.). Naudotojui paspaudus meniu mygtuką yra pateikiamas mygtukas apačioje. Paspaudus ant atsiradusio mygtuko yra atidaromas žaidimo meniu.



2.17 pav. Žaidimo meniu eskizas

Atidarius žaidimo meniu langą (2.13 pav.). Norint keisti žaidimo nustatymus, naudotojas turi paspausti „Settings“ mygtuką. Žaidėjui norint atsijungti iš žaidimo yra spaudžiamas „Disconnect“ mygtukas kuris pateikia patvirtinimo langą skirtą atsijungimui patvirtinti. Žaidėjui pasirinkus patvirtini atsijungimą iš žaidimo yra paspaudžiamas „Disconnect“ mygtukas. Naudotojas gali grįžti iš kiekvieno lango į praėjusį langą paspaudus ant „Back“ mygtuko.



2.18 pav. Žaidimo nustatymų eskizas laikrodžio lange

Žaidėjui pasirinkus atidaryti nustatymų langą (2.18 pav.). Žaidėjas gali keisti garso arba vaizdo nustatymus. Vaizdo nustatymai nesiskiria nuo (2.14 pav.) vaizdo nustatymų. Žaidėjui spaudus ant „Sound settings“ mygtuko gali valdyti pagrindinį žaidimo garso lygi. Žaidimo garsumas yra keičiamas naudojant „Volume up“ garso prigarsinimo mygtuką ir „Volume down“ garso sumažinimo mygtuką. Žaidėjas iš kiekvieno lango gali grąžti atgal spaudus ant „Back“ mygtuko.

2.1.4. Realizacijai keliami reikalavimai

Daugelio žaidėjų karkasui ir kovų žaidimui keliami realizacijos reikalavimai:

1. Sistemos kodas turi būti saugomas „GitHub“ kodo talpyklos paskyroje
2. Kovų žaidimo naudotojo sąsaja turi palaikyti anglų kalbą
3. Kovų žaidimas turi palaikyti virtualios realybės įrenginius.
4. Daugelio žaidėjų karkaso serveris turi veikti ant Windows ir Linux platformų
5. Žaidimas turi pasileisti Windows aplinkoje
6. Kovų žaidimas turi veikti stabiliai naudojant virtualios realybės įrenginius
7. Daugelio žaidėjų Karkasas turi palaikyti paketų dydžio glaudimas.
8. Karkaso ir jo kodas turi būti lengvai prieinamas.
9. Karkasas turi palaikyti „UDP“ komunikacijos protokolą su serveriu.

2.1.5. Techninė specifikacija

Techninė kompiuterinė įranga, kuri palaiko virtualios realybės akinius ir virtualios realybės įrenginius. Kompiuterinė įranga privalo turėti grafikos procesorių (GPU), kuris paliko virtualios realybės įrenginius. Grafikos procesorius turi būti naujesnis arba lygiavertis NVIDIA GTX 970 / AMD R9 290 grafikos procesoriams. Kompiuteris privalo turėti „Windows“ operacinę sistemą. Kompiuteryje turi būti suinstaliuota „Steam“ (19) internetinė žaidimų parduotuvė ir suinstaliuotą „Steam VR“ programa skirta virtualios realybės įrenginių palaikymui.

Daugelio žaidėjų virtualios realybės karkaso sistemai reikia įrenginio, kuriame bus patalpintas karkaso serveris. Kadangi karkaso ir jo serverio panaudojimas yra numatytas tik asmeniniam naudojimui. Karkaso serveris bus diegiamas lokaliamje kompiuteryje. Lokalios mašinos parametrai: AMD Ryzen™ 5 2600 ir 16GB darbinės atminties.

2.2. Projektavimo metodai

2.2.1. Projektavimo valdymas ir eiga

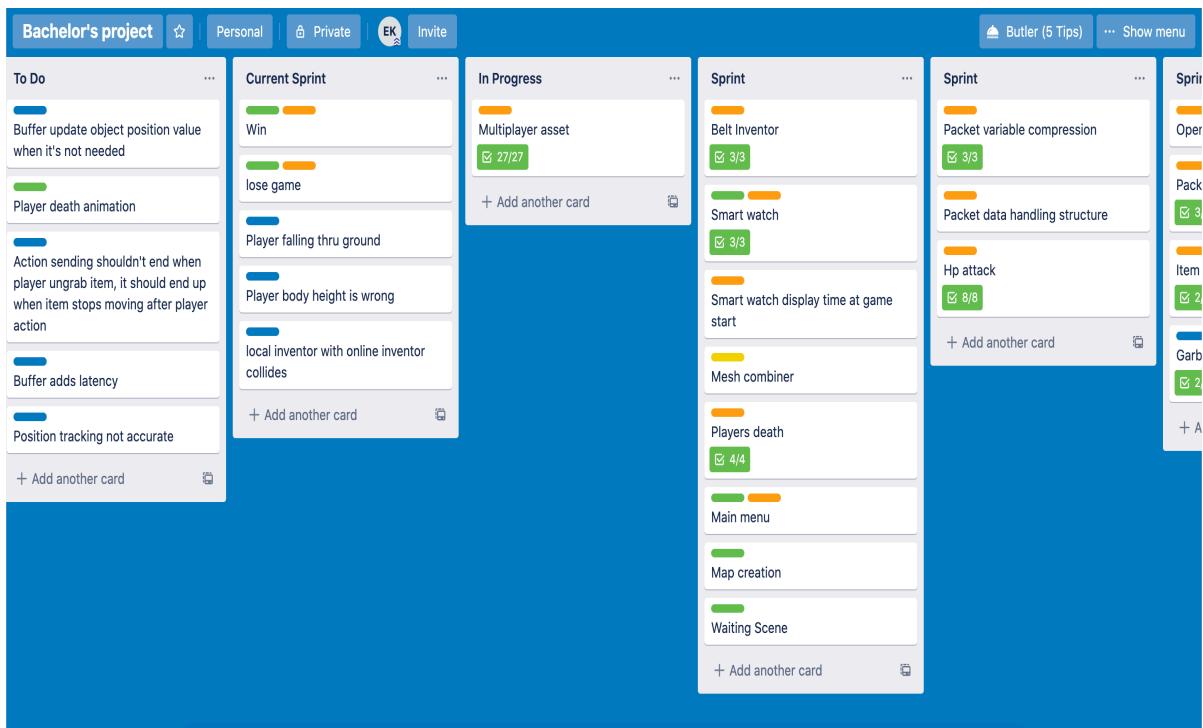
Sistemos kūrimo metu buvo naudojamas iteracinis programos kūrimo modelis, kuris remiasi pastovaus ir lankstaus sistemos keitimu ir reikalavimų atnaujinimui. Vienos iteracijos trukmė yra 2 svaitės. Atlirkę iteracijų kiekis yra 6 iteracijos.

1. Primoje iteracijoje buvo sukurtos „GitHub“ kodo saugyklos projektui, išanalizuoti virtualios realybės įrankiai ir žaidimų varikliai, išanalizuota daugelio žaidėjų karkasai skirti „Unity“ žaidimų varikliui ir pasirinkta technologija su kuria buvo kuriamas karkasas.
2. Antroje iteracijoje Buvo sukurtas naujas „Unity projektas“, sukonfigūruotas virtualios realybės palaikymas naujame projekte, sukurtas pirmasis daiktas kuris gali būti paimtas, sukurtas serverio projektas, sukurtas paprastas komunikavimo ryšys su žaidimu, serverio žinučių gavimas iš kliento ir persiuntimas klientui.
3. Trečioje iteracijoje buvo padarytos durys kurios gali būti atidaromos ir uždaromos, sukurta žinutės struktūra žaidimo ir serverio komunikavimui, sukurtas pirmasis daiktas, kuris yra sinchronizuojamas daugelio žaidėjų kovų žaidimo ir serverio
4. Ketvirtoje iteracijoje įvykdytas karkaso paketų dydžio glaudimas, paketų struktūros valdymas, žaidėjo užpuolimas,
5. Penktoje iteracijoje buvo padarytas diržas skirtas saugoti daiktus, išmanusis laikrodis, praplėstas išmanaus laikrodžio funkcionalumas, įrankis padedantis optimizuoti žaidimą, žaidėjo mirtis, pradžios meniu, žemėlapio sukūrimas, laukimo scenos sukūrimas.
6. Šeštojoje iteracijoje buvo padaryta žaidimo laimėjimas, žaidimo pralaimėjimas, sutvarkyta keletas klaidų.

Kiekvienos iteracijos metu buvo tvarkomos sistemos klaidos bei plečiamas karkasas.

2.2.2. Projektavimo technologija

Sistemos daugelio žaidėjų karkaso ir kovų žaidimo projektavimui naudotas UML 2.5.1 diagramų standartas. Diagramų projektavimui buvo naudojamas „MagicDraw“ (20) įrankis. Projektas yra saugomas „MagicDraw Teamwork“ serveryje. Kovų žaidimas ir daugelio žaidėjų karkasas yra saugomi „GitHub“ kodo talpykloje. Projektų versijų kontrolei yra naudojamas „Git“ įrankis. Projekto darbų planavimui yra naudojama „Trello“ (21) - „Kanban“ stiliums sarašų sudarymo programa.



2.19 pav. „Trello“ projekto valdymo įrankio dalis

2.2.3. Programavimo kalbos, derinimo, automatizavimo priemonės, operacinė sistemos

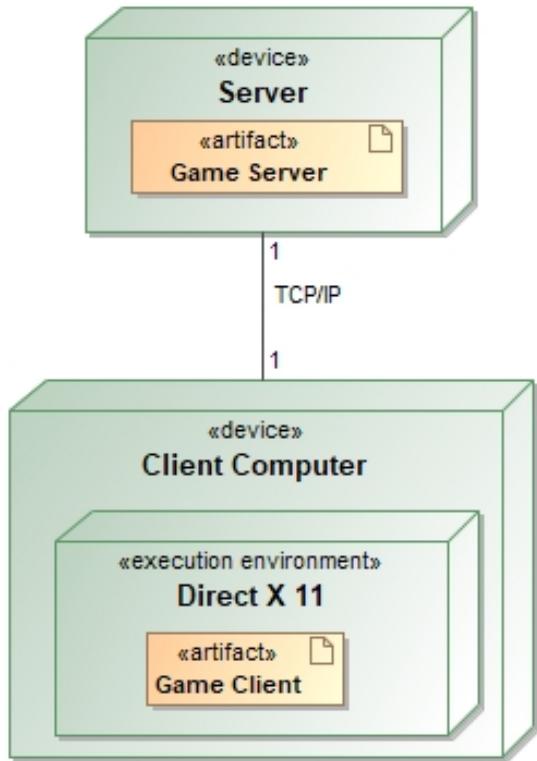
Daugelio žaidėjų kovų žaidimas buvo sukurtas naudojant „Unity“ žaidimų variklį. Žaidimas buvo programuojamas naudojant C# objektine programavimo kalba. „Unity“ žaidimų varikliui buvo suinstaliuoti papildom „VRTK“ ir „Steam VR“ įskiepiai virtualios realybės palaikymui. Programavimui buvo pasirinkta „Visual Studio“ (22) integruota programavimo aplinka. Žaidimo kūrimo metu buvo naudojama „Windows 10“ operacinė sistema.

Daugelio žaidėjų karkaso kūrimui buvo pasirinkta „.NET“ platforma. Karkasas buvo programuojamas naudojant C# objektine programavimo kalbą. Serveriui komunikuoti su žaidimu yra naudojama „Socket“ biblioteka per UDP komunikacijos protokolą. Kuriant daugelio žaidėjų karkasą buvo naudojama „Windows 10“ ir „Linux“ operacinės sistemos, „Windows 10“ operacinė sistema buvo skirta projektuoti ir programuoti karkasą, „Linux“ operacinė sistema naudojama serverio testavimui.

Kodo saugojimui buvo pasirinkta „GitHub“ kodo talpykla ir „Git“ versijų kontrolės įrankis.

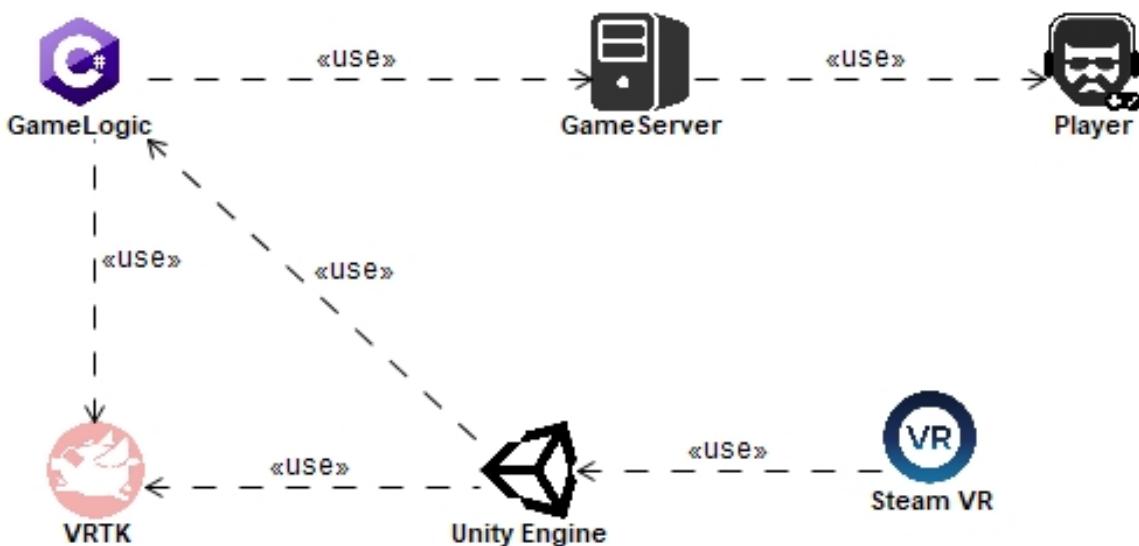
2.3. Sistemos projektas

2.3.1. Statinis sistemos vaizdas



2.20 pav. Sistemos diegimo diagramma

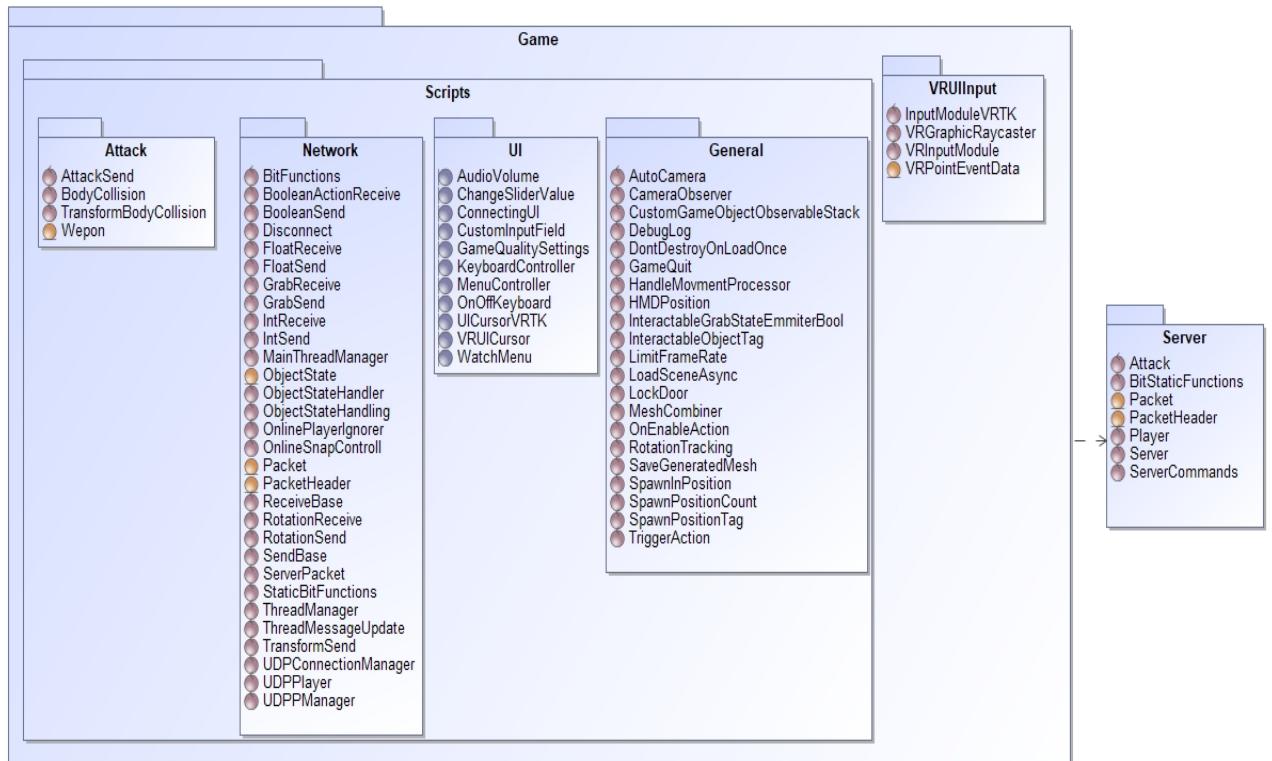
Sistemą sudaro dvi dalys – serverio ir kliento dalis. Sistemos dalys buvo įdiegtos tiesiogiai, įdiegimui nebuvo naudojama virtualizacija. Serverio dalis suinstaliuota ir vykdoma „Linux“ sistemoje kurioje yra internetas ir yra įdiegta „.NET“ įranga. Serveris su žaidimo klientu komunikuoja per TCP/IP protokolą. Žaidimas yra įdiegtas kompiuteryje kuris turi „Windows 10“ operacine sistemą, „DirectX 11“ (23) įrankius ir „Steam VR“ virtualios realybės įrankį.



2.21 pav. Sistemos komponentų diagramma

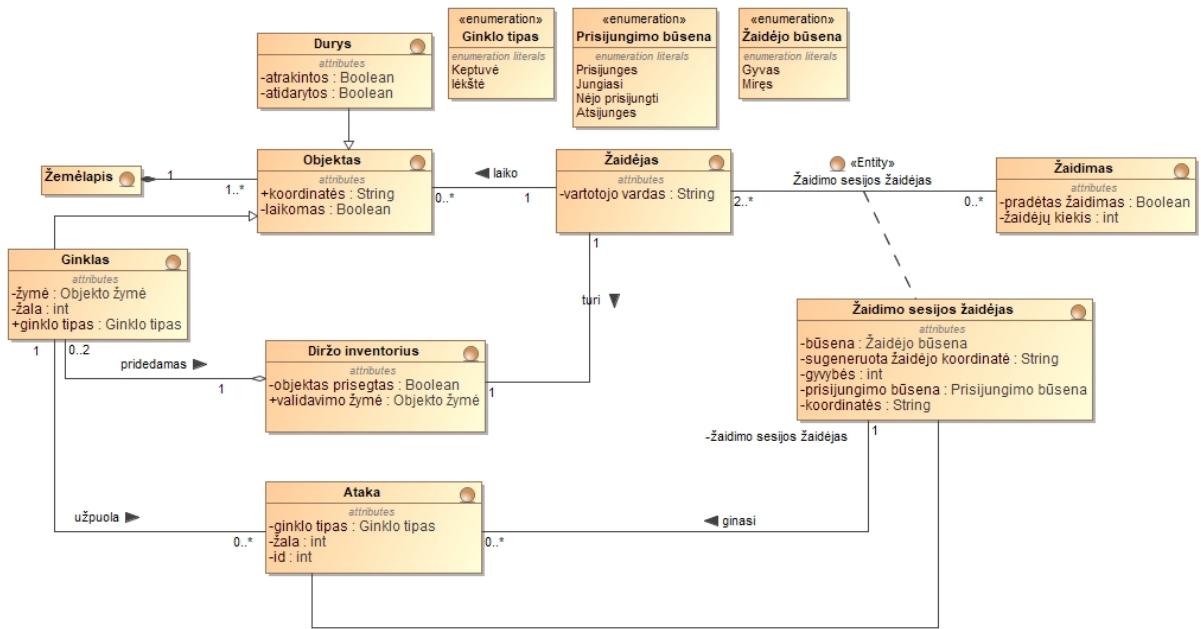
Daugelio žaidėjų karkasas ir kovų žaidimas susideda iš šešių komponentų. Komponentai atvaizduoja logines sistemos dalis. Sistema susidaro iš tokių komponentų – „Steam VR“ virtualios realybės įskiepio, „Unity Engine“ - „Unity“ žaidimo variklio, „VRTK“ virtualios realybės įskiepio,

„GameLogic“ suprogramuotos žaidimo ir karkaso loginės pusės, „GameServer“ žaidimo serverio, „Player“ serverio žaidėjo valdymo. Virtualios realybė žaidėjo veiksmams sekti ir virtualios realybės vaizdui atvaizduoti yra naudojamas „Steam VR“ komponentas, jis kreipiasi į „Unity Engine“ komponentą su žaidėjo įvesties duomenimis žaidimo apdorojimui. „Unity Engine“ komponentas apdoroja visas įvesti, „VRTK“ komponentas apdoroja „Unity Engine“ apdorotas įvestis, šis komponentas padeda valdyti virtualios realybės sąveikavimą su virtualiu pasauliu. Logikos komponentas „GameLogic“ yra naudojamas žaidimo logikai aprašyti, naudojamas karkasas ir kiti elementai, „UnityEngine“ kreipiasi į „GameLogic“ komponentą viso žaidimo metu. Žaidimo logikos komponentas kreipiasi į „GameServer“ komponentą žaidimo sinchronizacijai. „GameServer“ komponentas turi komponentą „Player“ skirtą žaidėjams valdyti šitas komponentas yra viena kasė, kuri valdo žaidėjus žaidimo serveryje.



2.22 pav. Sistemos paketu diagramma (atnaujinti klasiu pavadinimus)

Sistemos struktūra buvo sudėliota žaidimą ir daugelio žaidėjų karkasą išskirstant į logines dalis. Diagramoje matome išskirstytą žaidimo paketą į žaidimo logines dalis. Žaidime yra paketai susiję su žaidimu ir komunikacija su serveriu. „Attack“ paketas yra skirtas prieš užpuolimui ir jo valdymui, „Network“ paketas yra skirtas žaidimo ir serverio komunikacijai išskirti, „UI“ paketas talpina virtualios realybės naudotojo sėsajos funkcionalumą, „General“ pakete yra sutalpintas visas žaidimo funkcionalumas, kuris neįeina nei į vieną iš paminėtų paketų, „VRUIInput“ pakete yra sutalpinta virtualios realybės sąveikavimo su naudotojo sėsajos langais realizacija, „Server“ paketas yra serveris, kuris sinchronizuojia žaidimus tarpusavyje.

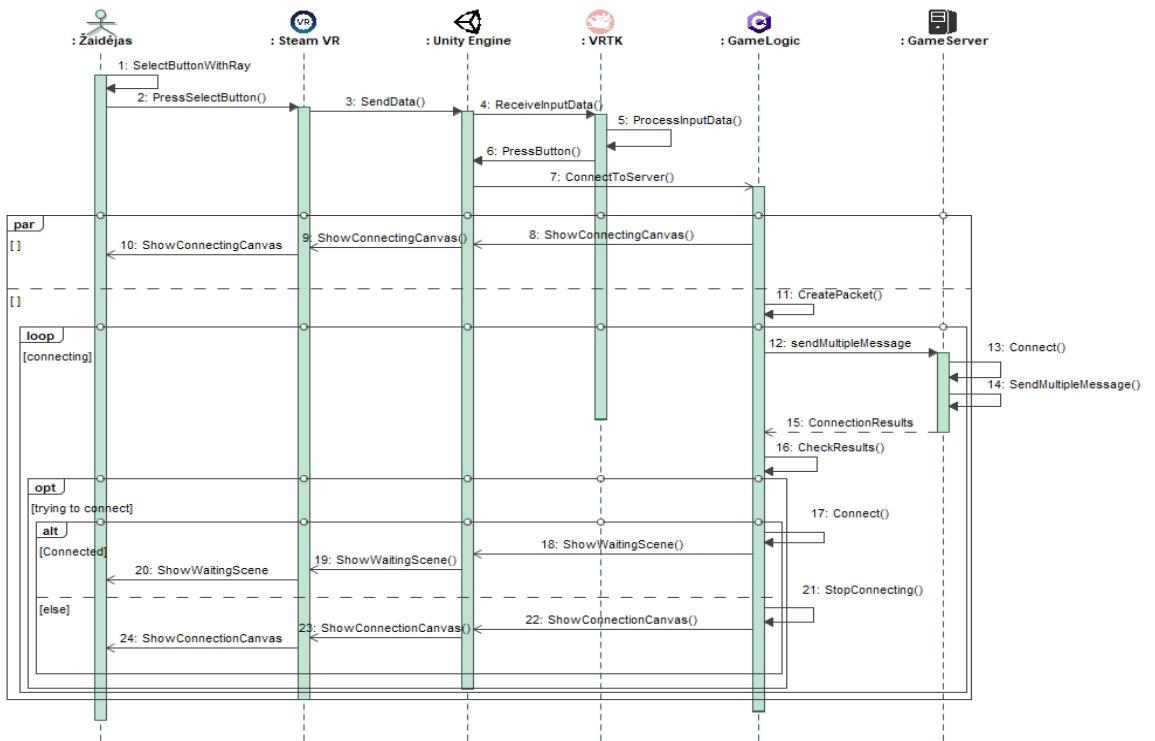


2.23 pav. Sistemos UML klasų diagrama

Žaidimui ir serveriui nebuvo naudojama duomenų bazė. Diagramoje yra parodyta žaidimo ir serverio komponentų saugomi ir sinchronizuojami duomenys. Kiekvienas žaidimo komponentas turi savo duomenis, kurie gali keisti žaidėjui atlikus veiksmą. Komponentų duomenys yra reikalingi žaidimo valdymui ir jo sinchronizavimui.

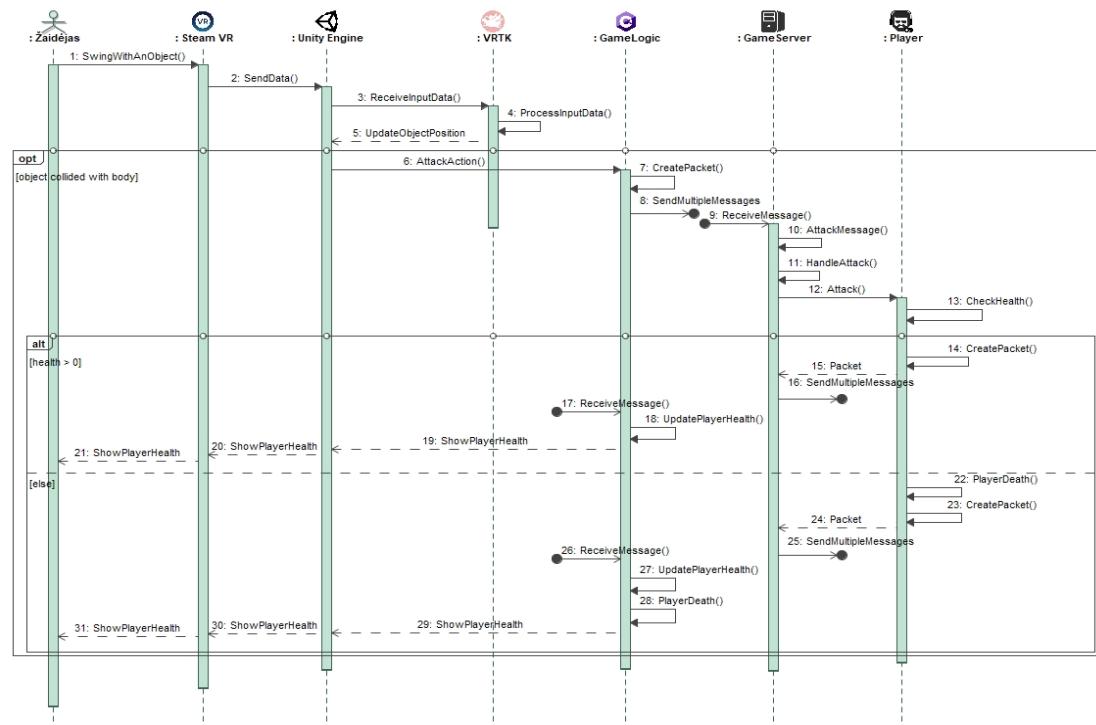
2.3.2. Dinaminis sistemos vaizdas

Daugelio žaidėjų karkaso ir kovų žaidimo sisteminis vaizdas yra vaizduojamas naudojant sekų diagramas. Jos leidžia atskleisti detalų sistemos komponentų sąveikavimą ir panaudojimo atvejo detalų veikimą.



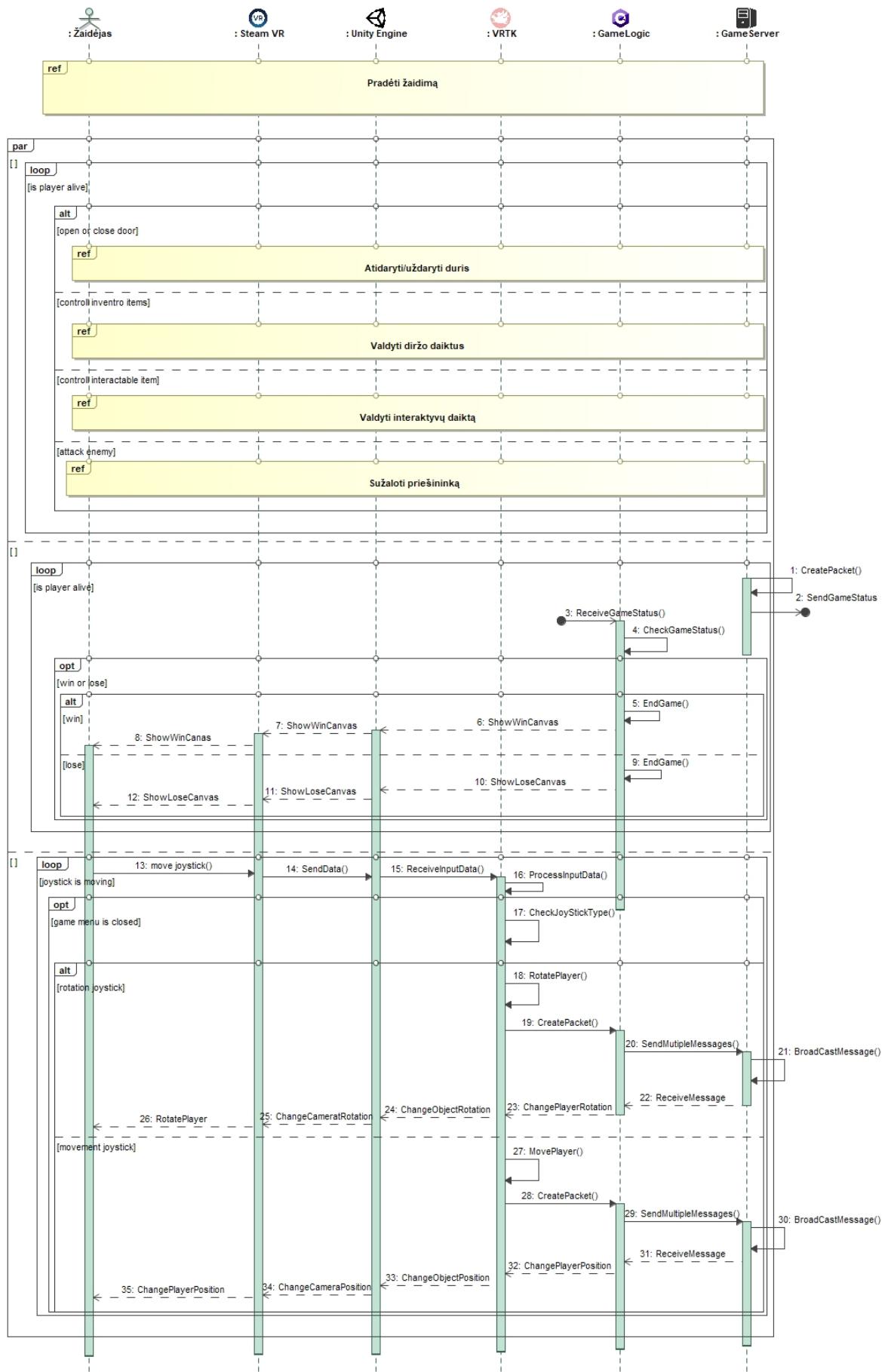
2.24 pav. „Prisijungti“ panaudojimo atvejo sekų diagrama

Panaudojimo atvejo metu (2.24 pav.) žaidėjas prisijungia prie žaidimo pasirikus prisijungimo mygtuką. Naudotojui paspaudus ant mygtuko yra perduodamas mygtuko paspaudimas „Unity Engine“ komponentui, kuris perduoda duomenis „VRTK“, šis komponentas paspaudžia mygtuką pagal pateiktus duomenis. Paspaudus mygtuką žaidėjui yra parodos jungimosi langas, kol žaidimas bando prisijungti prie serverio. Žaidėjui rodant prisijungimo langą žaidimas siunčia sukurtą paketą į serverį su prisijungimo užklausa, kol yra gaunamas rezultatas iš serverio. Negavus pranešimo iš serverio per 10 kartų žaidimas nebesijungia į serverį ir žaidėjui yra pateikiamas praeitas langas. Žaidimui gavus paketą su duomenimis, jie yra panaudojami prisijungimo patikrai. Žaidimui prisijungus prie serverio yra pateikiama žaidimo laukimo žemėlapis. Gavus iš serverio duomenis jog jis yra pilnas žaidime yra pateikiamas prieš tai buvęs langas.



2.25 pav. „Sužaloti priešininkų“ panaudojimo atvejo sekų diagrama (turetu būti broadcast ir zaidejas i save atlika veiksma ir tada siuncia data)

Sužalojant kitą žaidėją (2.25 pav.), žaidėjas turi laikyti daiktą su kuriuo jis galėtu ji užpulti. Žaidėjo veiksmai yra sekami „Steam VR“ komponento, jis perduoda žaidėjo duomenis „Unity Engine“ komponentui, šis komponentas persiunčia gautus duomenis „VRTK“ komponentui, kuris atnaujina laikomo daikto koordinates. Žaidėjui bandant užpulti su virtualiu daiktu kitą žaidėją yra patikrinama ar virtualus objektas susiduria su užpulto žaidėjo kūnu. Nesusidūrus virtualiam daiktui su žaidėju, kuris turėjo būti užpultas nieko neivyksta. Virtualiam daiktui susidūrus su užpultu priešu yra sukuriamas užpuolimo paketas, kuris yra siunčiamas keletą kartų į serverį. Pranešimų gavimo metu serveris naudoja vieną gautą paketą, o kitus vienodus paketus pašalina. Serveryje yra valdoma ataka, kurios metu yra sprendžiama ar ataka valdi. Serveriui patvirtinus atakos valdumą ataka yra perduodama „Player“ komponentui. Šitas komponentas yra skirtas serverio žaidėjui valdyti. Komponentą sudaro viena logikos klasė, kuri valdo žaidėjų serveryje. Užpuolus žaidėjų serveryje yra atimami žaidėjo gyvybės taškai. Atėmus gyvybes žaidėjas yra patikrinamas ar jos dar gyvas. Jei žaidėjas yra gyvas yra persiunčiami žaidėjams atnaujintos gyvybės. Atnaujintos gyvybės yra parodos ant išmanaus laikrodžio. Žaidėjui nebeturint gyvybės taškų jis miršta, serveryje žaidėjas yra nužudomas. Atakos rezultatai yra persiunčiami žaidėjams. Žaidėjui mirus žaidime yra atnaujinamos žaidėjo gyvybės ir žaidėjas yra nužudomas. Nužudytam žaidėjui yra parodoma, kad jis mirė.



2.26 pav. „Žaisti“ panaudojimo atvejo sekų diagrama

Panaudojimo atvejo sekos diagrama (2.26 pav.) yra suskaidyta į tris pagrindines dalis – sąveikavimas su virtualiu pasauliu, žaidėjo vaikščiojimas ir žaidimo būsenos atnaujinimas. Pagrindinės dalims yra pavizituotas lygiagretumas, kadangi žaidėjas žaidžiant gali vaikščioti ir atlikti pasirinktą veiksmą vienu metu. Žaidėjui atliekant veiksmus žaidimo būsena yra atnaujinama. Šitoje dalyje lygiagretumas reiškia, kad žaidėjas gali atlikti ne vieną veiksmą vienu metu, bet kelis veiksmus. Būsenos atnaujinimas vyksta žaidimo metu kai žaidėjas žaidžia. Žaidimo pradžioje yra įvykdomas „Pradėti žaidimą“ panaudojimo atvejis (2.6 pav.).

Pirmoje lygiagretumo gijoje yra parodomas žaidėjo galimas veiksmas virtualios realybės pasaulyje. Žaidėjas gali atlikti keletą veiksmų – „Atidaryti/uždaryti duris“ žaidėjas gali atidaryti arba uždaryti duris jei jos yra atrakintos (2.12 pav.), „Valdyti diržo daiktus“ – žaidėjas gali prisegti daiktą prie diržo arba ji atsegti nuo diržo (2.11 pav.), „Valdyti interaktyvų daiktą“ – žaidėjas gali paimti ir neštis daiktą arba ji mesti ant žemės (2.9 pav.), „Sužaloti priešininką“ – žaidėjas gali sužaloti savo priešininką su paimtu daiktu (2.25 pav.).

Antroje lygiagretumo gijoje yra parodomas žaidimo būsenos atnaujinimas. Serveris sukuria paketą, kuriamo yra žaidimo būsena, sukurtas paketas yra išsiunčiamas į žaidimą. Paketas yra naudojamas žaidimo būsenai atnaujinti. Žaidimui gavus paketą yra patikrinama kokia būsena buvo gauta, būsenos gali būti dvi – laimėjimas arba pralaimėjimas. Žaidimui gavus laimėjimo būseną yra baigiamas žaidimas ir parodomas laimėjimo langas. Kitu atveju, žaidimui gavus pralaimėjimo būseną žaidimas yra baigiamas ir žaidėjui yra pateikiamas pralaimėjimo langas.

Trečioje lygiagretumo gijoje žaidėjas gali judėti jeigu jis nori. Žaidėjui judinant vairasvirtę, jos duomenis yra perduodami per „Steam VR“ komponentą, jis seka virtualios realybės įvesti ir perduoda sekamas įvesti „UnityEngine“ komponentui. „UnityEngine“ komponentas perduoda įvestis „VRTK“ komponentui, kuris juos apdoroja. Prieš pradedant keisti kameros pasiskimo kampą arba žaidėjo poziciją yra patikrinama ar žaidėjas yra uždaręs žaidimo meniu. Tik patikrinus galima keisti žaidėjo poziciją arba kameros pasiskimo kampą. Vairasvirtės tipai yra du - žaidėjo vaikščiojimo ir žaidėjo pasiskimo vairasvirtės. „VRTK“ komponentas patikrina vairasvirtės tipą galima pradėti veiksmą. Jei patikrintas vairasvirtės tipas yra pasiskimo vairasvirtė, žaidėjo kamera būna pasukta. Žaidėjui passukus yra sukuriamas paketas su kameros pasiskimo kampu ir paketas yra išsiunčiamas į serverį sinchronizacija. Gautas paketas būna persiustas visiems žaidėjams. Žaidimui gavus paketą žaidėjo kameros pasiskimo kampas yra atnaujintas. Jei patikrintas vairasvirtės tipas yra vaikščiojimo vairasvirtė, žaidėjas eina tą pusę į kurią yra nukreipta vairasvirtė. Pajudinus žaidėjų yra sukuriamas paketas su žaidėjo koordinatėmis, paketas yra siunčiamas į serverį. Išsiustas paketas serveryje yra persiunčiamas visiems žaidėjams. Žaidimui gavus koordinacijų paketą žaidėjo pozicija yra atnaujinama.

3. Testavimas

3.1. Testavimo planas

1. Statine kodo analize
 2. Automatinis vienetų testavimas
 3. Vartotojo sąsajos testavimas

Sistemine kodo analizė buvo atliekama projekto gale. Atlikus analizei kodas buvo tvarkomas pagal iškeltus kriterijus. Karkaso ir žaidimo kūrimo metu buvo sukurti tik keletas automatinių vienetų testų. Sistema buvo testuojama po kiekvienos funkcijos sukūrimo naudojant vartotojo sąsajos testavimą.

3.2. Testavimo kriterijai

Vykstant Statine kodo analizę laikytasi šiuų kriterijų.

1. Kode neturi būti nenaudojamų kintamųjų
 2. Kode neturi būti bibliotekų, kurios yra nenaudojamos
 3. Visos klasės ir jų funkcijos turi būti panaudotos
 4. Kodas negali turėti klaidų, kurios neleis sukompiliuoti projekto

Vykstant komponentų testavimą iškelti kriterijai:

1. Testas privalo testuoti tik vieną scenarijų
 2. Visi komponentų testai turi įvykti be klaidų

Vartotojo sąsajos testavimo iškelti kriterijai:

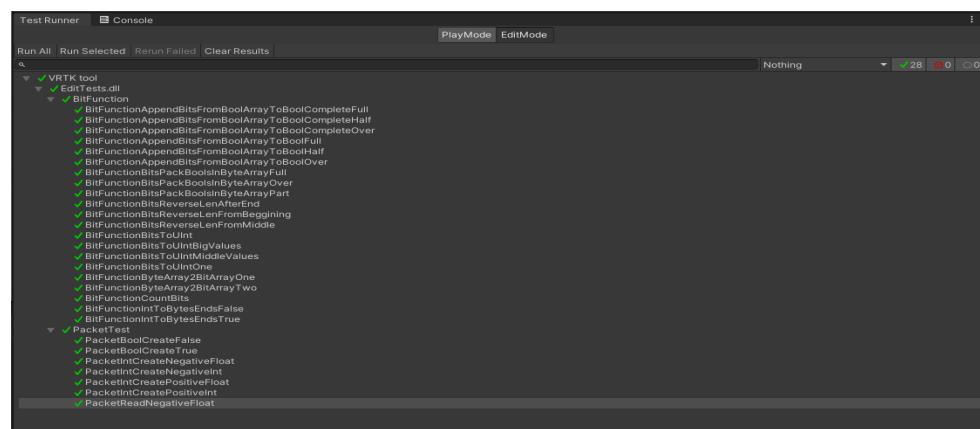
1. Visi testai turi būti įvykdysti.
 2. Po kiekvienos naujos funkcijos sukūrimo visi testai turi būti įvykdysti iš naujo

3.3. Statiné kodo analizē

Satinai kodo analizei buvo naudojamas „Visual Studio“ įrankis, kuriame yra įdiegtas „ErrorProne.Net.CoreAnalyzer“ paketas. Naudojant „Visual Studio“ įrankį buvo atlikta statinė kodo analizė. Atlikus kodo analizę buvo išvalyti nenaudojami kintamieji ir nenaudojamos bibliotekos.

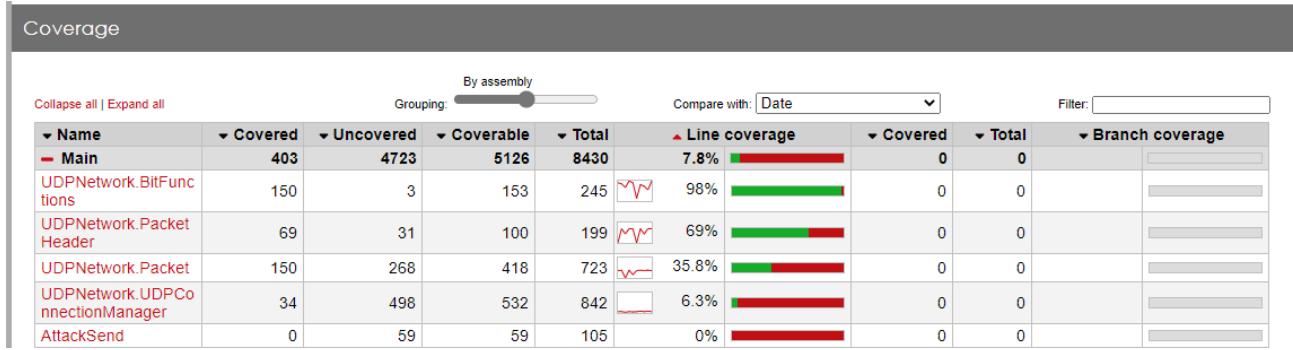
3.4. Komponentų testavimas

Komponentų testai buvo atliekami tik žinučių kūrimui. Testavimui buvo pasirinkta naudoti „Unity Test Runner“ įrankį. Šis įrankis yra pritaikyta žaidimų testavimui. Naudojamas įrankis taip pat leidžia peržiūrėti žaidimo kodo padengimą (3.2 pav.).



3.1 pav. Sekmingai atlikti testai.

Visi sukurti testai buvo atlikti sėkmingai (3.1 pav.). Testavimo metu buvo ištaisyto aptiktos klaidos.



3.2 pav. Sistemos testų padengimas.

Atlikus žaidimo testavimą buvo pasiektais 7,8% sistemos padengimas. Kodo padengimui buvo parašyti 28 testai. Nedidelis testų kiekis leido sėkmingai ištestuoti tik dalį sistemos.

3.5. Testavimas scenarijais

Vartotojo sąsajos testavimas buvo atliekamas žaidžiant žaidimą. Kiekvienas sąsajos testas yra atliekamas žaidėjo, jam atlikus testus rezultatas yra išanalizuojamas naudojant pateiktas testavimo lenteles (3.1 lentelė. - 3.9 lentelė.).

3.1 lentelė. „Prisijungti“ panaudojimo atvejo testavimo lentelė

Pagrindinė sąlyga		
Naudotojas nėra prisijungęs prie žaidimo		
Atliekamas veiksmas	Rezultatas	Tinkamas rezultatas
Naudotojas spausdžia ant „Join“ mygtuko	Nepavykus prisijungti pateikiamas pradžios langas	+
Naudotojas spausdžia ant „Join“ mygtuko	Prisijungus yra pateikiamas laukimo langas	+

3.2 lentelė. „Atsijungti“ panaudojimo atvejo testavimo lentelė

Pagrindinė sąlyga		
Naudotojas yra prisijungęs prie žaidimo		
Atliekamas veiksmas	Rezultatas	Tinkamas rezultatas
Paspaudžiamas ant valdiklio žaidimo meniu mygtukas	Pateikiamas meniu mygtukas	+
Paspaudžiamas žaidimo meniu mygtukas	Pateikiamas meniu langas	+
Paspaudžiamas „Disconnect“ mygtukas	Pateikiamas atsijungimo patvirtinimo langas	+
Naudotinas paspaudžia „Cancel“ mygtuką	Naudotojui yra pateikiamas meniu langas	+
Naudotojas paspaudžia ant „Disconnect“ mygtuko	Naudotojas grįžta į pradžios langą	+

3.3 lentelė. „Valdyti nustatymus“ panaudojimo atvejo testavimo lentelė

Atliekamas veiksmas	Rezultatas	Tinkamas rezultatas
Paspaudžiamas „Settings“ mygtukas	Pateikiamas nustatymų pasirinkimai	+
Naudotojas pasirenka pakeisti vaizdo nustatymus spaudžiant „Video Settings“ mygtuką	Pateikiamas vaizdo nustatymų langas	+
Naudotojas spaudžia „High“ mygtuką	Žaidimo vaizdo nustatymai pasikeičiami į aukšto lygio vaizdo nustatymus	+
Naudotojas spaudžia „Medium“ mygtuką	Žaidimo vaizdo nustatymai pasikeičiami į vidutinio lygio vaizdo nustatymus	+
Naudotojas spaudžia „Low“ mygtuką	Žaidimo vaizdo nustatymai pasikeičiami į žemo lygio vaizdo nustatymus	+
Žaidėjas spaudžia „Back“ mygtuką	Pateikiamas nustatymu langas	+
Žadėjas pasirenka valdyti garso nustatymus spaudžiant „Sound Settings“ mygtuką	Pateikiamas garso nustatymų langas	+
Keičiamas garso lygis keičiant slankiklio mygtuko vietą.	Pakeičiamas garso lygis	+
Paspaudžiamas „Back“ mygtukas	Pateikiamas nustatymu langas	+
Paspaudžiamas „Back“ mygtukas	Pateikiamas pradžios langas	+

3.4 lentelė. „Žaisti“ panaudojimo atvejo testavimo lentelė

Pagrindinė sąlyga		
Naudotojas yra prisijungęs prie žaidimo		
Atliekamas veiksmas	Rezultatas	Tinkamas rezultatas
Nužudo visus priešininkus	Pateikiamas laimėjimo langas	+
Žaidėjas nužudomas	Pateikiamas pralaimėjimo langas	+
Žaidėjas judina vaikščiojimo vairavirte	Žaidėjas pastumiamas į vairavirčės nukreiptą pusę	+
Žaidėjas pakreipia pasukimo vairavirę į dešinę	Žaidimo kamera pasukama į dešinę pusę	+
Žaidėjas pakreipia pasukimo vairavirę į kairę pusę	Žaidimo kamera pasukama į kairę pusę	+

3.5 lentelė. „Pradėti žaidimą“ panaudojimo atvejo testavimo lentelė

Pagrindinė sąlyga		
Naudotojas yra prisijungęs prie žaidimo		
Atliekamas veiksmas	Rezultatas	Tinkamas rezultatas
Sulaukiama bent du žaidėjai	Pateikiamas žaidimo langas	+

3.6 lentelė. „Sužaloti priešininką“ panaudojimo atvejo testavimo lentelė

Pagrindinė sąlyga		
Naudotojas yra prisijungęs prie žaidimo		
Atliekamas veiksmas	Rezultatas	Tinkamas rezultatas
Žaidėjas užpuola priešininką su laikomu ginklu	Žaidėjas yra užpultas	+
Žaidėjas užpuola priešininką su laikomu ginklu	Žaidėjas apsigynė	+

3.7 lentelė. „Valdyti interaktyvų daiktą“ panaudojimo atvejo testavimo lentelė

Pagrindinė sąlyga		
Naudotojas yra prisijungęs prie žaidimo		
Atliekamas veiksmas	Rezultatas	Tinkamas rezultatas
Žaidėjas prideda ranką prie daikto ir paspaudžia paémimo mygtuką	Daiktas yra paimtas	+
Žaidėjas paleidžia laikymo mygtuką	Daiktas nukrenta ant žemės	+

3.8 lentelė. „Valdyti diržo daiktus“ panaudojimo atvejo testavimo lentelė

Pagrindinė sąlyga		
Naudotojas yra prisijungęs prie žaidimo		
Atliekamas veiksmas	Rezultatas	Tinkamas rezultatas
Žaidėjas prideda daiktą prie vienos iš diržo vietų ir jį paleidžia daiktui turint prisegimo žymą	Daiktas yra prisegamas prie diržo	+
Žaidėjas prideda daiktą prie vienos iš diržo vietų ir jį paleidžia daiktui neturin prisegimo žymą	Daiktas nukrenta ant žemės	+

3.9 lentelė. „Atidaryti/uždaryti duris“ panaudojimo atvejo testavimo lentelė

Pagrindinė sąlyga		
Naudotojas yra prisijungęs prie žaidimo		
Atliekamas veiksmas	Rezultatas	Tinkamas rezultatas
Jei durys yra stumiamos, nors jos yra užrakintos	Durys nejuda	+
Jei durys yra traukiamas, nors jos yra užrakintos	Durys nejuda	+
Jei durys yra traukiamas, nors jos turi būti stumiamos	Durys nejuda	+
Jei durys yra stumiamos, nors jos turi būti traukiamas	Durys nejuda	+
Durys yra uždarytos ir žaidėjui jas stumiant durys pilnai atvertų durų kampą	Durys atsiveria ir pasiekia pilnai atvertų durų kampą	+

Durys yra atidarytos ir žaidėjui jas stumiant durys pilnai užsidaro	Durys užsiveria ir pasiekia pilnai užvertų durų kampą	+
Durys yra uždarytos ir žaidėjas jas atidaro nepilnai	Durys atsiveria iki žaidėjo rankos atverto kampo	+
Durys yra pradarytos ir žaidėjas jas atidaro nepilnai	Durys atsiveria iki žaidėjo rankos atverto kampo	+
Durys yra pradarytos ir žaidėjas jas atidaro iki galo	Durys atsiveria ir pasiekia pilnai atvertų durų kampą	+
Durys yra pradarytos ir žaidėjas bando atidaryti daugiau negu yra leidžiama	Durys sustoja pasiekus pilnai atvertų durų kampą	+
Durys yra uždarytos ir žaidėjas bando atidaryti daugiau negu yra leidžiama	Durys sustoja pasiekus pilnai atvertų durų kampą	+
Durys yra atidarytos ir žaidėjas bando atidaryti daugiau negu yra leidžiama	Durys sustoja pasiekus pilnai atvertų durų kampą	+
Durys yra uždarytos ir žaidėjui jas traukiant durys pilnai atsidaro	Durys atsiveria ir pasiekia pilnai atvertų durų kampą	+
Durys yra atidarytos ir žaidėjui jas traukiant durys pilnai užsidaro	Durys užsiveria ir pasiekia pilnai užvertų durų kampą	+
Durys yra atidarytos ir žaidėjas jas uždaro nepilnai	Durys užsiveria iki žaidėjo rankos atverto kampo	+
Durys yra pradarytos ir žaidėjas jas uždaro nepilnai	Durys užsiveria iki žaidėjo rankos atverto kampo	+
Durys yra pradarytos ir žaidėjas jas uždaro iki galo	Durys užsiveria ir pasiekia pilnai užvertų durų kampą	+
Durys yra pradarytos ir žaidėjas bando atidaryti daugiau negu yra leidžiama	Durys sustoja pasiekus pilnai užverstų durų kampą	+
Durys yra uždarytos ir žaidėjas bando atidaryti daugiau negu yra leidžiama	Durys sustoja pasiekus pilnai užverstų durų kampą	+
Durys yra atidarytos ir žaidėjas bando atidaryti daugiau negu yra leidžiama	Durys sustoja pasiekus pilnai atvertų durų kampą	+

4. Dokumentacija naudotojui

4.1. Apibendrintas sistemos galimybių aprašymas

Daugelio žaidėjų karkasas ir kovų žaidimas yra aprašoma skirtingomis dalimis. Daugelio žaidėjų karkasas yra skirtas žaidimui kūrėjams. Kovų žaidimas yra skirtas Žaidėjams ir kūrėjams. Kūrėjai naudojantys daugelio žaidėjų karkasą gali sukurti virtualios realybės daugelio žaidėjų žaidimus, sinchronizuoti žaidimo veikimą siunčiant ir priimant synchronizacijos paketus. Kovų žaidimo posistemėje žaidėjas gali kovoti naudojant ginklus su kitais žaidėjais iki išlikimo kovų arenaje.

4.2. Vartotojo vadovas

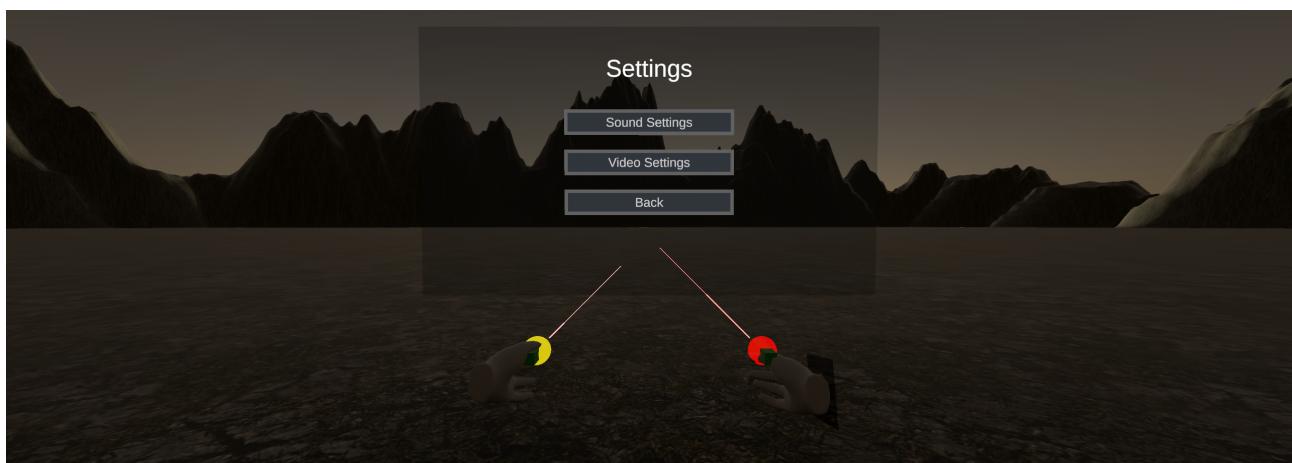
4.2.1. Kovos žaidimo vadovas

Kiekvienas žaidėjas, kuris nori išbandyti žaidimą turi jį pasileisti. Pasileidus žaidimui yra pateiktiamas pradžios langas Jame žaidėjas gali pasirinkti vieną iš duotų pasirinkimų (4.1 pav.).



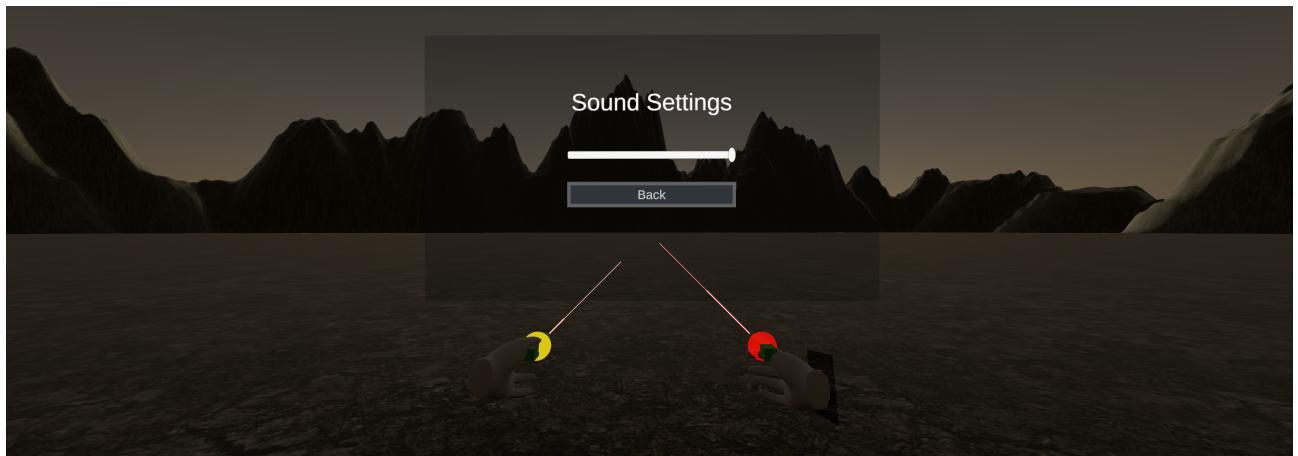
4.1 pav. Kovos žaidimo pradžios langas

Pasirinkus keisti nustatymus yra pateiktiamas nustatymu langas (4.2 pav.). Žaidėjas gali pasirinkti vieną iš nustatymų arba grįžti atgal į pagrindinį meniu.



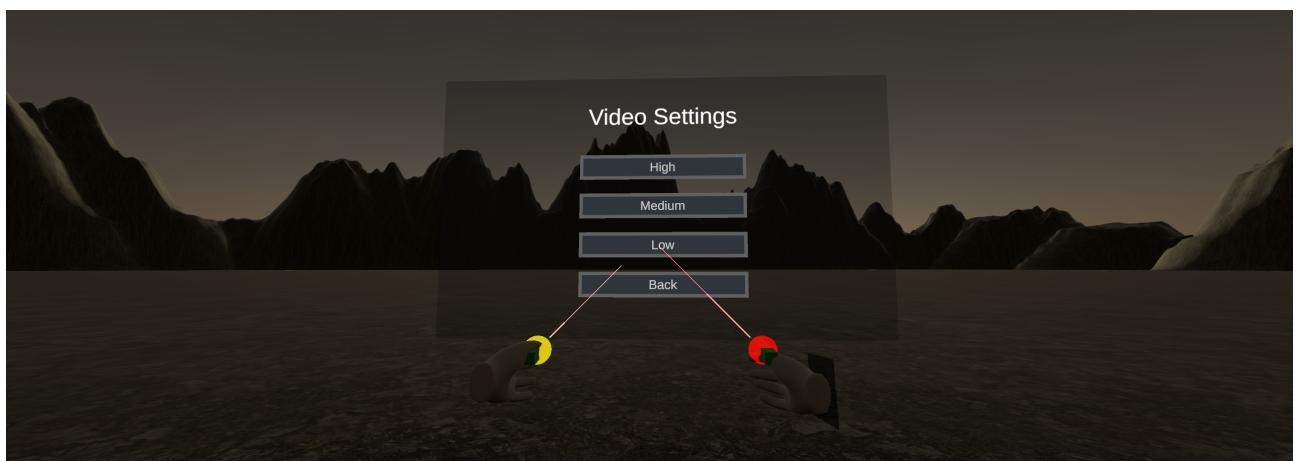
4.2 pav. Kovos žaidimo nustatymų langas

Pakeičiamas langą į garso nustatymų langą (4.3 pav.). Žaidėjas gali valdyti pagrindinį garsą arba grįžti į nustatymų langą (4.2 pav.).



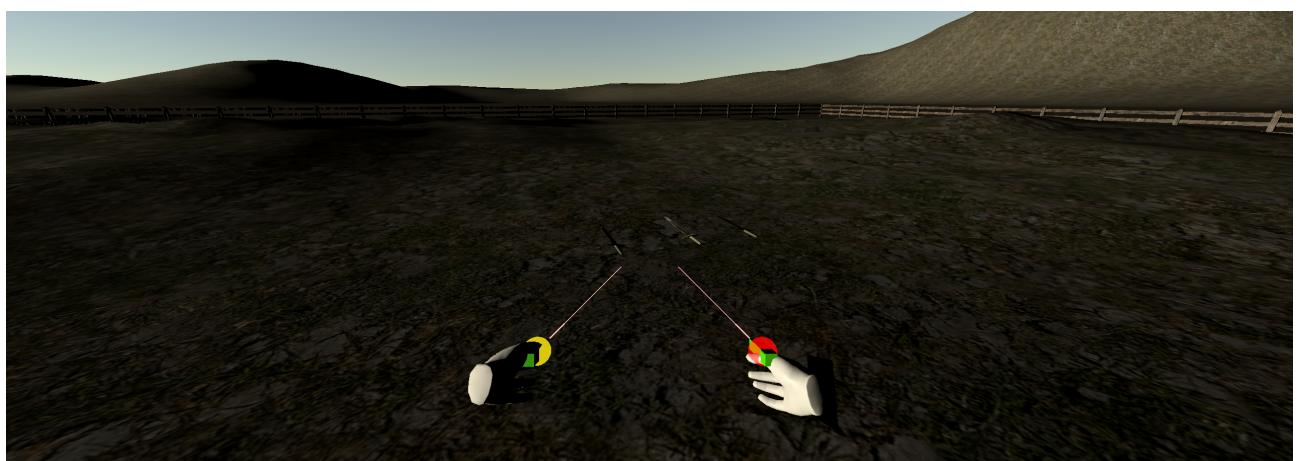
4.3 pav. Kovos žaidimo goso nustatymu langas

Pasirinkus vaizdo nustatymų langą (4.4 pav.) naudotojas gali pasirinkti vieną iš vaizdo kokybės nustatymų. Žaidėjui pasirinkus vaizdo nustatymo lygi jis yra atnaujinamas. Žaidėjas gali grįžti į nustatymų langą spaudus ant „Back“ mygtuko (4.2 pav.).



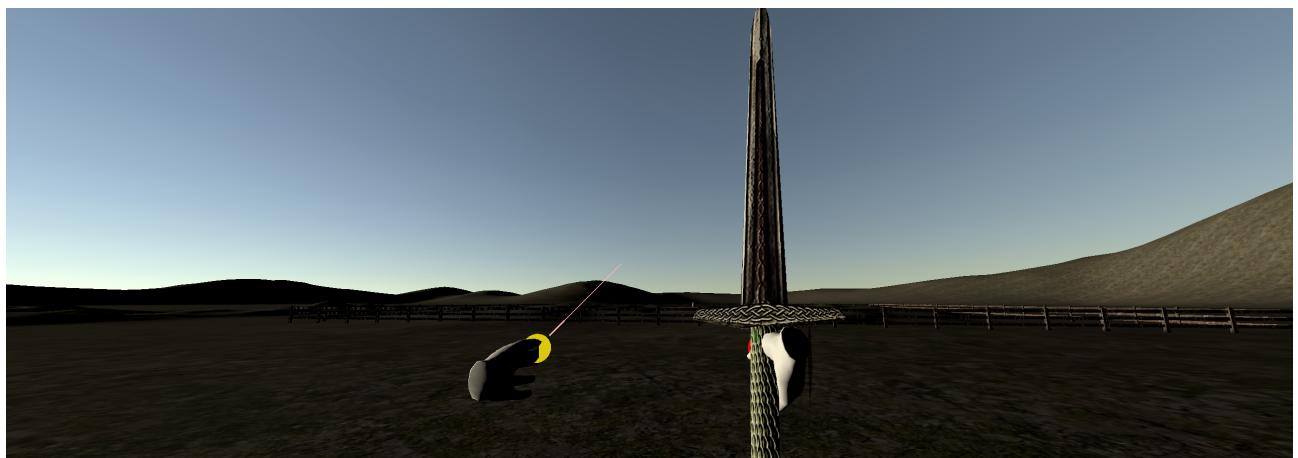
4.4 pav. Kovos žaidimo vaizdo nustatymu langas

Žaidėjui prisijungus prie žaidimo jis yra nukreipiamas į laukimo žemėlapį (4.5 pav.).



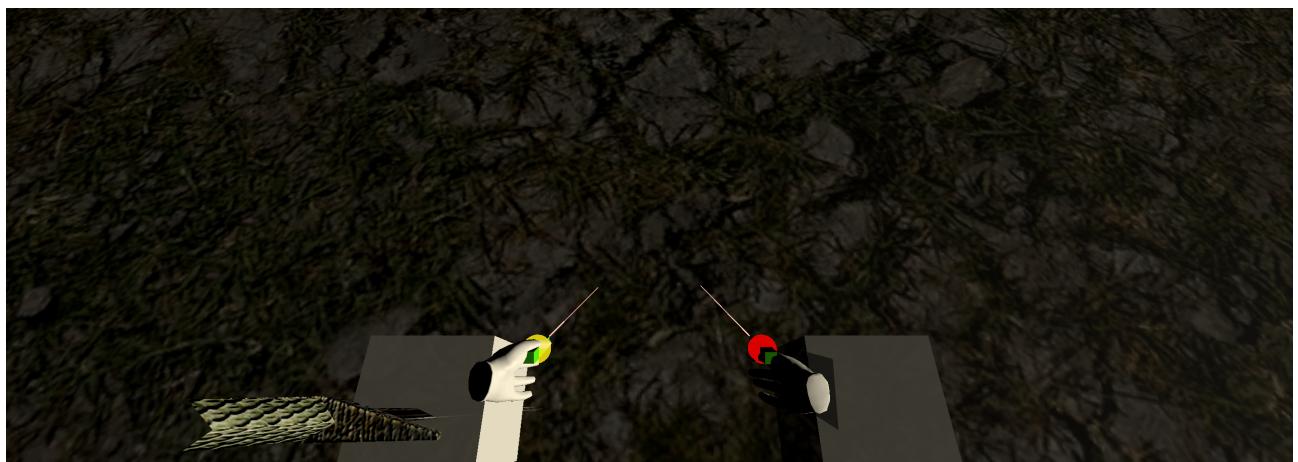
4.5 pav. Kovų žaidimo laukimo žemėlapis.

Laukimo ir žaidimo žemėlapiuose žaidėjas gali nešiotis rankose daiktus (4.6 pav.).



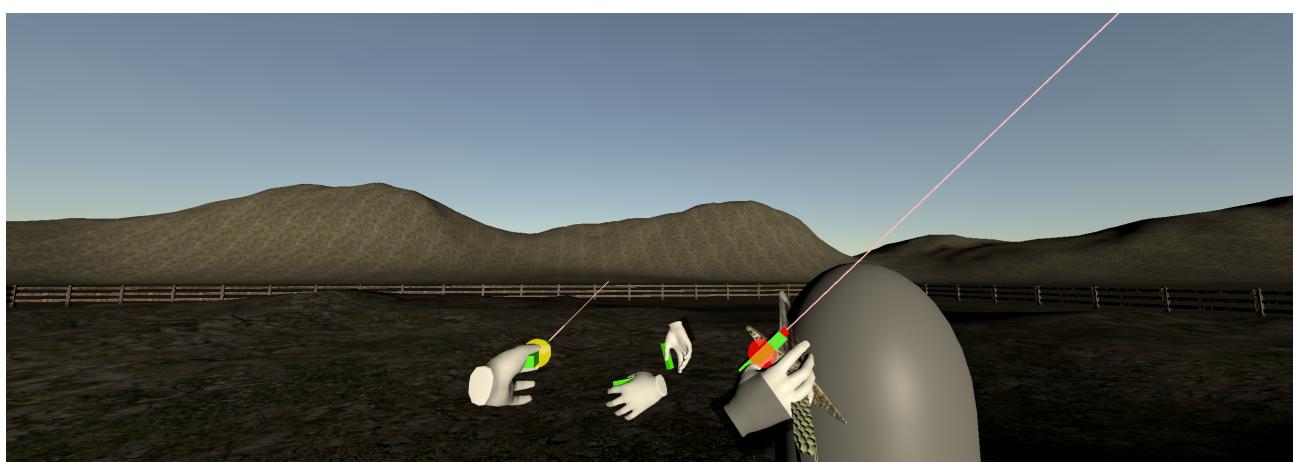
4.6 pav. Kovų žaidimo laikomas interaktyvus objektas.

Interaktyvus laikomas daiktas gali būti prisegtas prie diržo, kuris seka žaidėją viso žaidimo metu (4.7 pav.).



4.7 pav. Kovų žaidimo interaktyvus diržas.

Žaidėjui norint užpulti priešą jis turi turėti ginklą rankoje ir su juo užsimoti priešininkui (4.8 pav.).



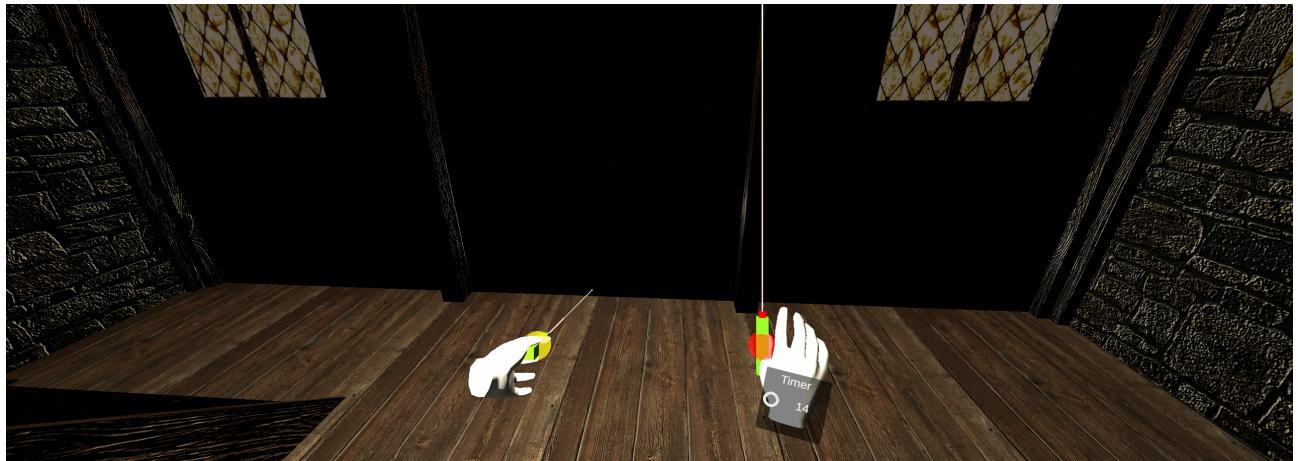
4.8 pav. Priešo užpuolimas kovų žaidime

Prie žaidimo prisijungus bent dviem žaidėjams yra užkraunamas žaidimo žemėlapis (4.9 pav.). Žaidimo pradžioje žaidėjas atsiranda name, kuriame turi susirasti ginklą su kuriuo eis į kovą.



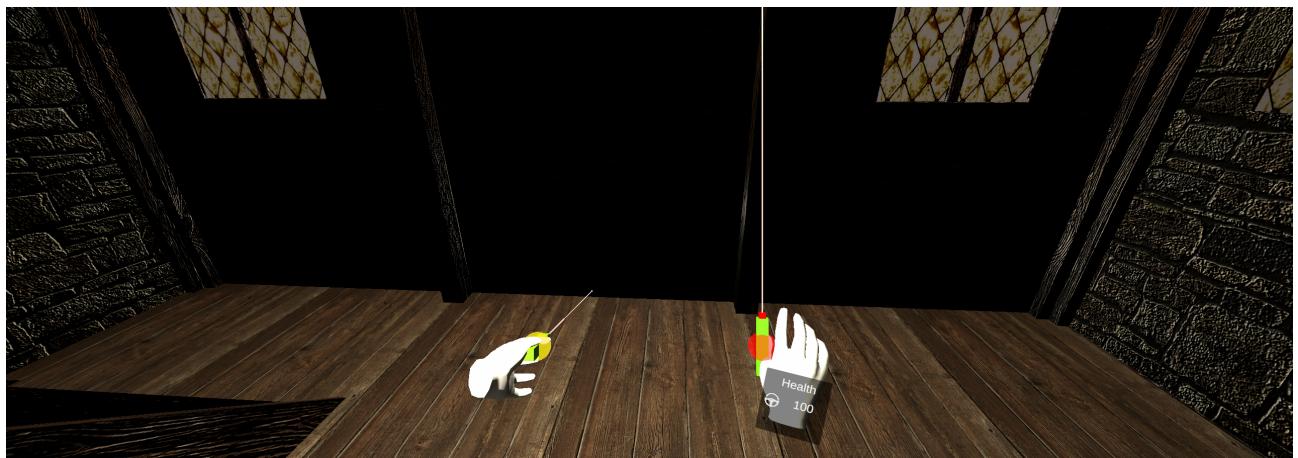
4.9 pav. Užkrautas žaidimo žemėlapis

Žaidimo pradžioje yra atnaujinamas išmanus laikrodis (4.10 pav.), kuriame yra pateiktas likęs laikas iki durų atrakinimo.



4.10 pav. Atnaujintas išmanus laikrodis su likusiu laiku.

Atsirakinus durims laikrodžio langas yra pakeičiamas į likusių gyvybių langą (4.11 pav.).



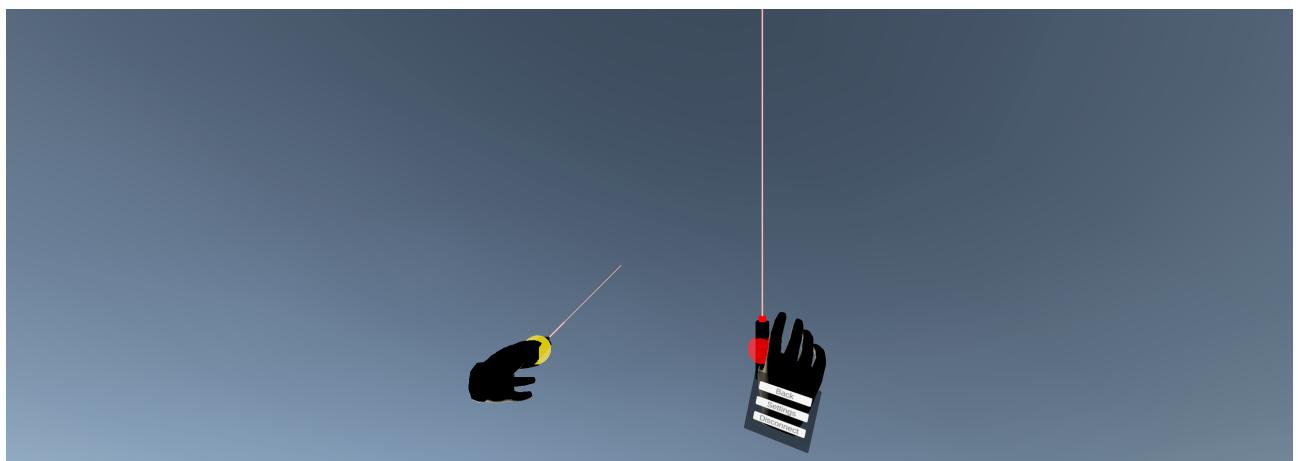
4.11 pav. Išmanaus laikrodžio gyvybių langas

Žaidėjas gali atsidaryti žaidimo meniu paspaudus meniu mygtuką (4.12 pav.). Meniu lange yra pateikiamas meniu mygtukas, kuris nukreipia į langą su visais pasirinkamais (4.13 pav.).



4.12 pav. Kovų žaidimo meniu langas.

Paspaudus ant meniu mygtuko yra pateikiamos visos meniu pasirinktys (4.13 pav.). Žaidėjas gali pasirinkti keisti nustatymus, arba atsijungti iš žaidimo. Žaidėjui nenorint atlikti nei vieno pasirinkimo jis gali grįžti į prieš tai buvusį langą (4.12 pav.).



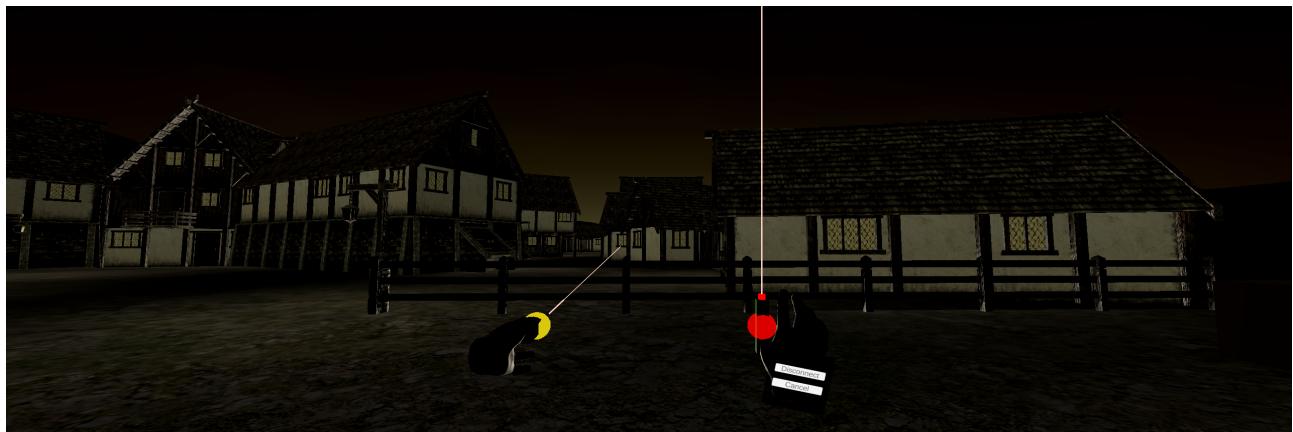
4.13 pav. Kovų žaidimo meniu pasirinkimai

Pasirinkus žaidimo nustatymus (4.14 pav.) yra pateikiami du nustatymų pasirinkimai – vaizdo ir garso nustatymai. Naudotojui nenorint keisti nustatymų jis gali grįžti į meniu pasirinkimų langą (4.13 pav.).



4.14 pav. Kovų žaidimo išmanaus laikrodžio nustatymai

Pasirinkus atsijungti iš žaidimo (4.15 pav.) yra pateikiamas atsijungimo langas kur žaidėjas gali atsijungti arba grįžti į meniu pasirinkimų langą (4.13 pav.). Žaidėjui pasirinkus atsijungti jis yra nukreipiamas į žaidimo pradžią (4.1 pav.).



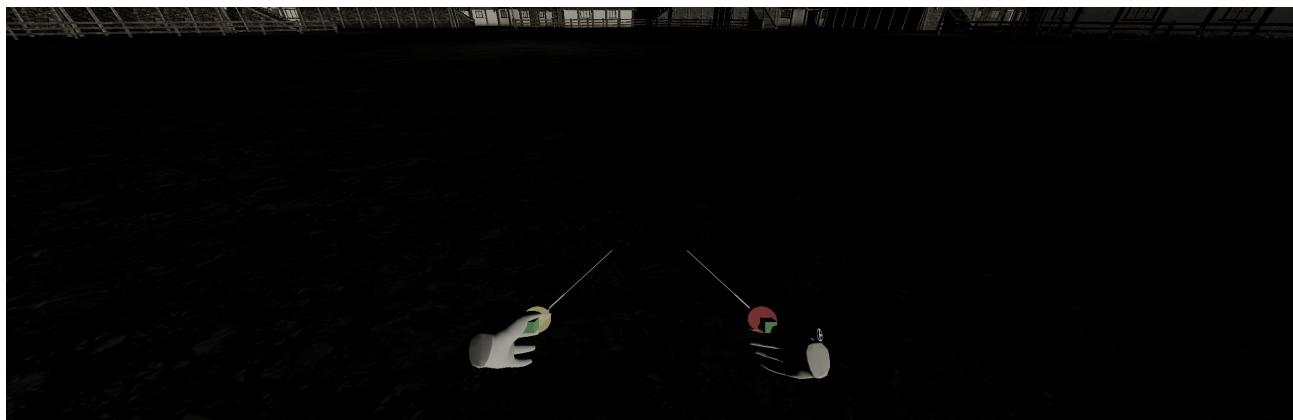
4.15 pav. Kovų žaidimo atsijungimo langas

Žaidimo metu žaidėjas gali atidaryti arba uždaryti duris (4.16 pav.). Žaidėjas duris gali atidarinėti ir uždarinėti duris jei durys yra atrakintos. Duris atidaryti arba uždaryti galima jas laikant ir traukiant arba stumiant.



4.16 pav. Atrakintų durų atidarymas

Žaidėjui mirus (4.17 pav.) yra pakeičiama žaidimo spalva, kad žaidėjas suprastu jog jis yra mireš. Žaidėjui mirus žaidėjas gali atsijungti iš žaidimo per žaidimo meniu (4.12 pav.)



4.17 pav. Žaidėjo mirtis

Žaidėjui laimėjus žaidimą yra pateikiamas laimėjimo langas (4.18 pav.). Žaidėjas iš žaidimo gali atsisiungti naudojant žaidimo meniu (4.12 pav.).



4.18 pav. Laimėjimo langas

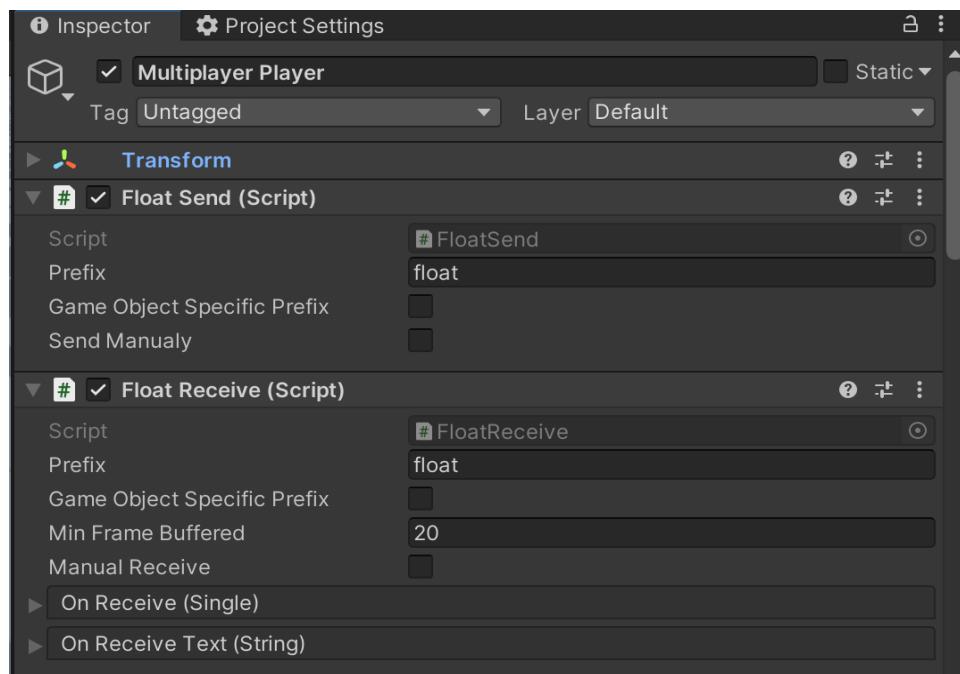
4.2.2. Daugelio žaidėjų karkaso vadovas

Žaidimo sinchronizacijai yra naudojami žinučių siuntimo ir gavimo komponentai (4.19 pav. - 4.25 pav.), jie yra skirti gauti žinutes. Identifikacijos žodis „Prefix“ turi sutapti tarp komponentų.

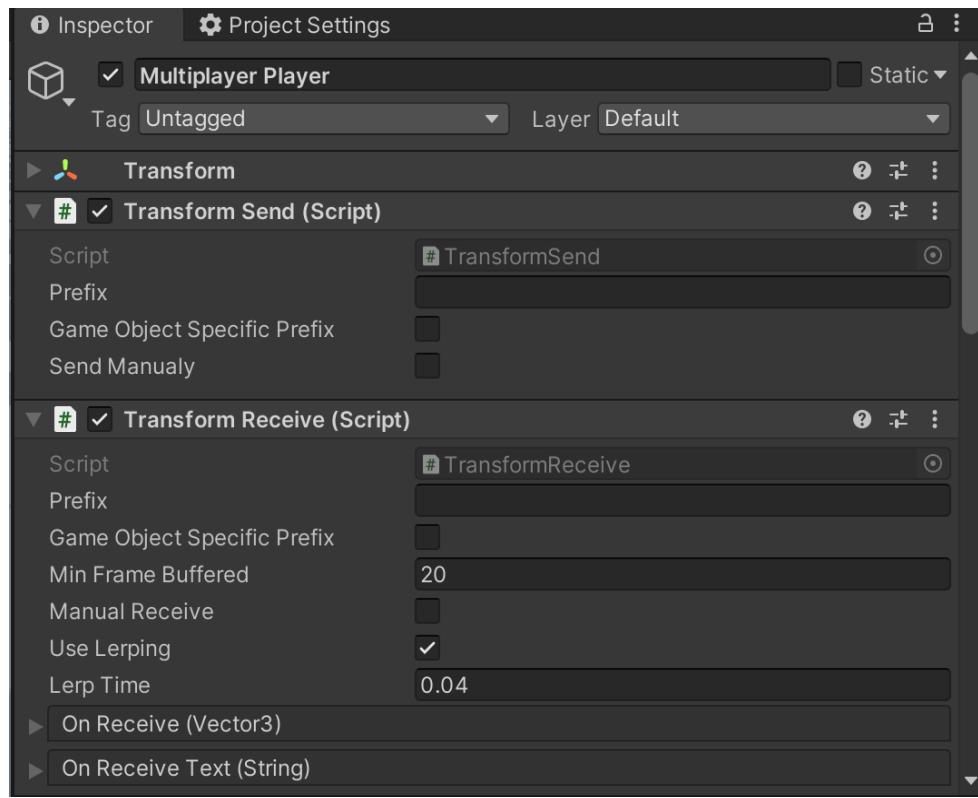
Kiekvienas paketų gavimo komponentas turi laukus skirtus perduoti gautas reikšmes per „Unity Editor“. Gavimo komponente yra buferis, kurio pagalba yra užtikrinama žinučių gavimo darna, buferio dydis gali būti didinamas arba mažinamas. Siuntimo komponentas gali būti valdomas per funkciją pažymėjus „Send Manually“ laukelį siuntimo komponente. Gavimo komponentas gali būti valdomas per funkciją kaip ir siuntimo komponentas pažymėjus „Manual Receive“ laukelį, kuris yra gavimo komponente. Iš jų galima sudaryti įvairias siuntimo ir gavimo struktūras. Galimos struktūros išvardintos pateiktame sąraše.

Galimų struktūrų sąrašas:

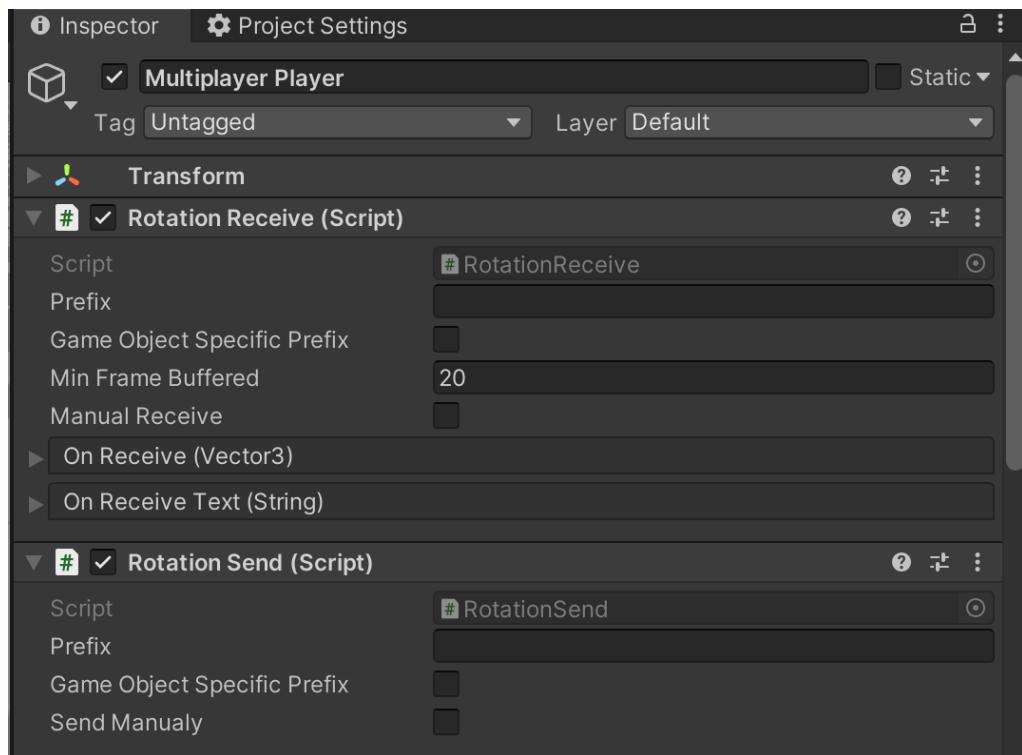
- Vienas žinučių siuntimo komponentas ir keli žinučių gavimo komponentai.
- Vienas žinučių siuntimo komponentas ir vienas žinučių gavimo komponentas.
- Daug žinučių siuntimo komponentų ir vienas žinučių gavimo komponentas.
- Daug žinučių siuntimo komponentų ir daug žinučių gavimo komponentų.



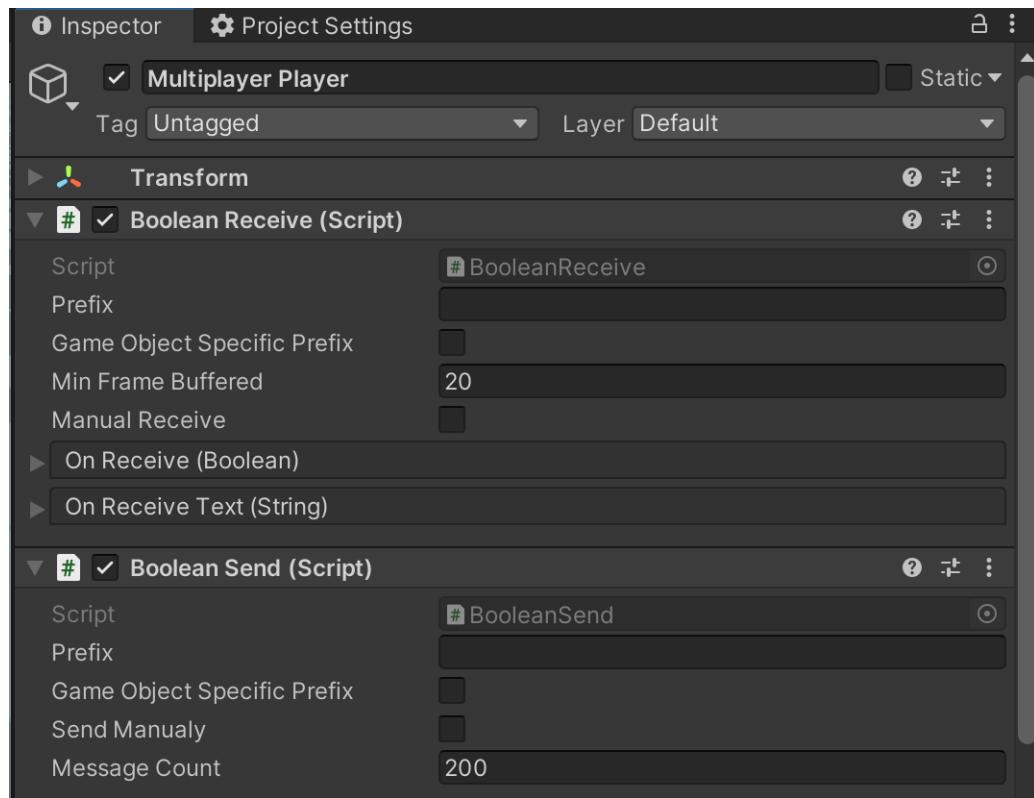
4.19 pav. „Boolean“ tipo paketo siuntimo ir gavimo komponentai



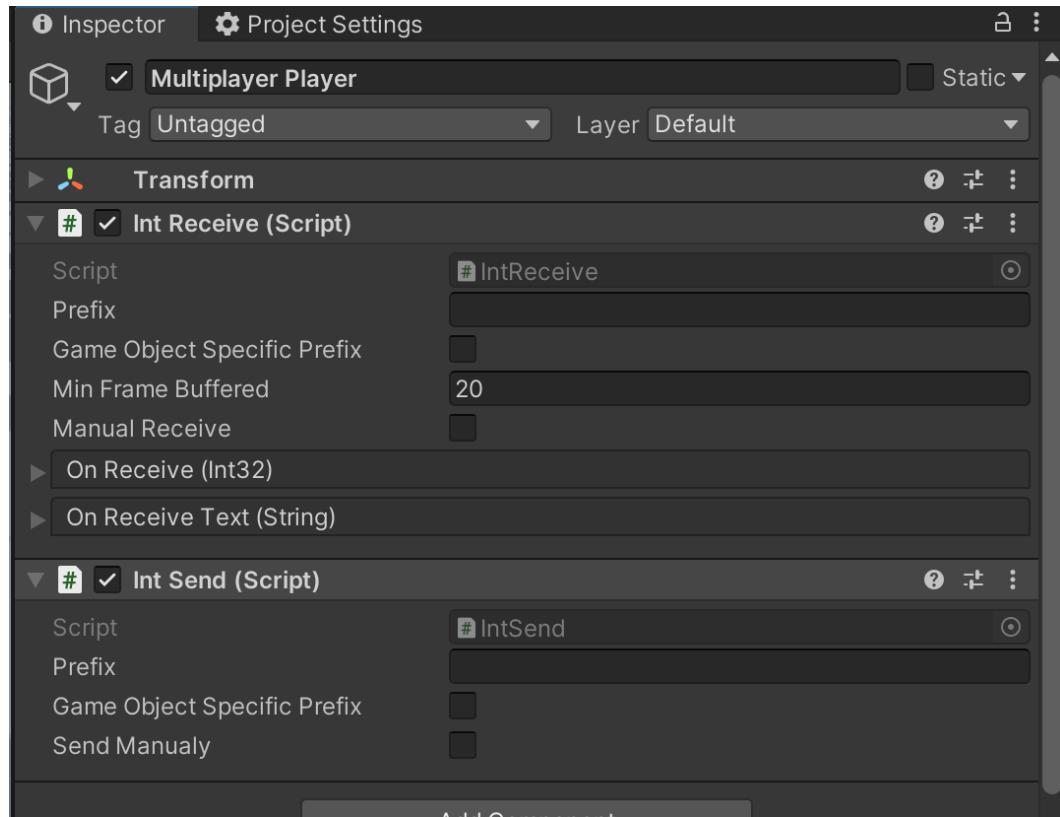
4.20 pav. „Transform“ tipo paketo siuntimo ir gavimo komponentai



4.21 pav. „Rotation“ tipo paketo siuntimo ir gavimo komponentai

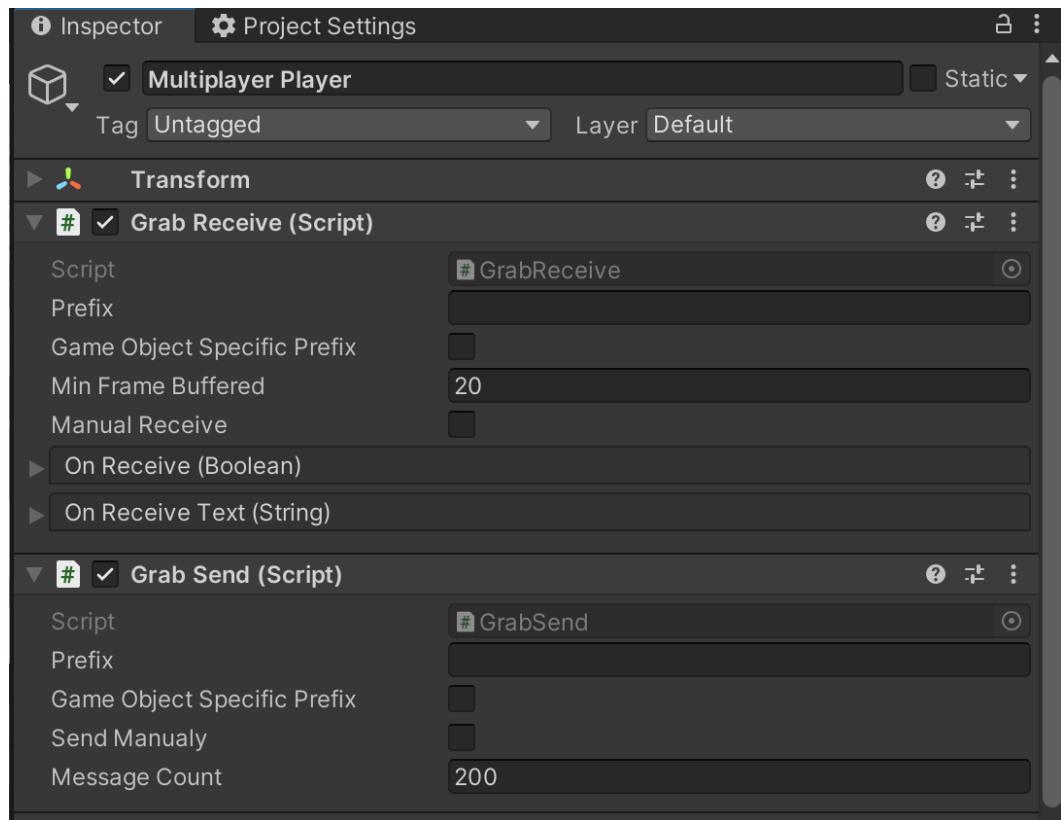


4.22 pav. „Boolean“ tipo paketo siuntimo ir gavimo komponentai.



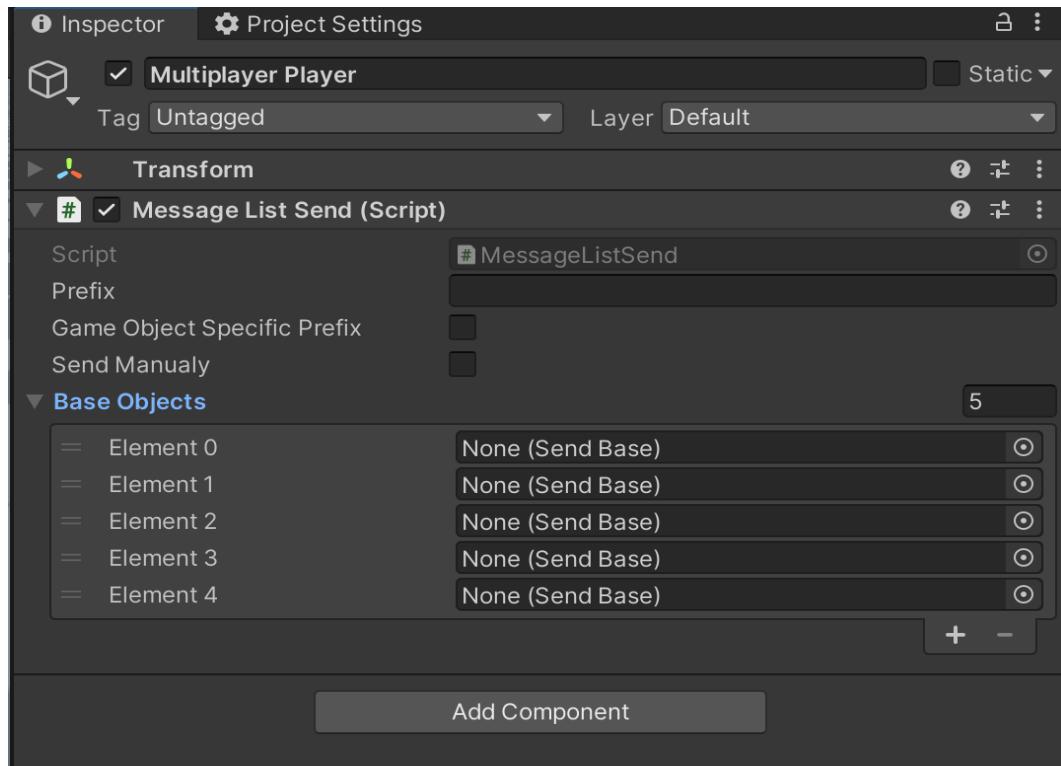
4.23 pav. „Int“ tipo paketo siuntimo ir gavimo komponentai.

„Grab“ tipo komponentai (4.24 pav.) nesiskiria nuo „Boolean“ komponentų (4.22 pav.)



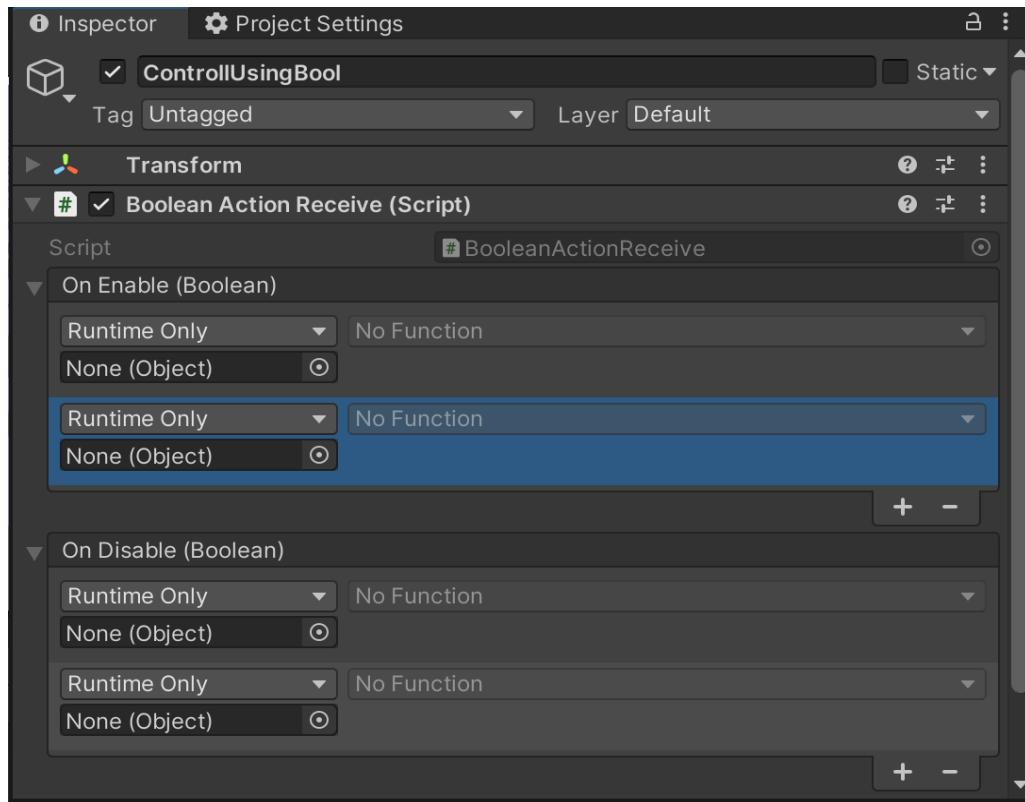
4.24 pav. „Grab“ tipo paketo siuntimo ir gavimo komponentai.

Yra vienas išskirtinis siuntimo komponentas (4.25 pav.), kuris yra skirtas išsiusti pasirinktą kiekį pranešimų įkeliant siuntimo komponentus į „Base Objects“ sąrašą. Šiam komponentui įrašyti „Prefix“ identifikavimo kodo nereikia, kadangi jis naudojamas pasirinktų komponentų paketu siuntimui per tuos komponentus.



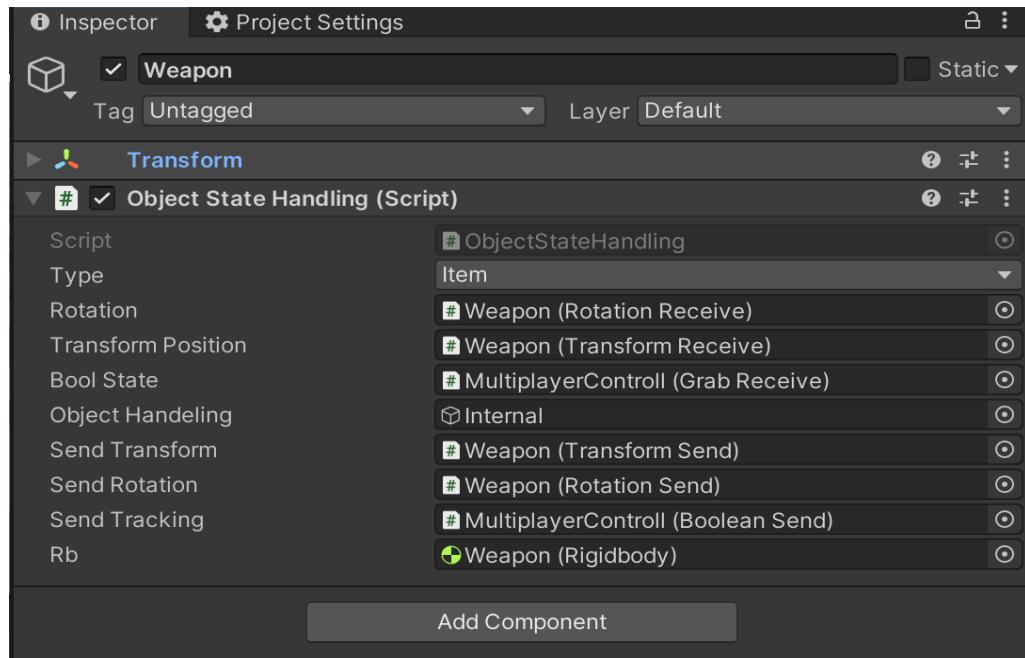
4.25 pav. „Message List Send“ paketu siuntimo komponentas.

„Boolean Action Receiver“ komponentas yra „Unity Editor“ komponentas, kuris iškviečia pasirinktas funkcijas gavus tiesos arba netiesos įvestį.



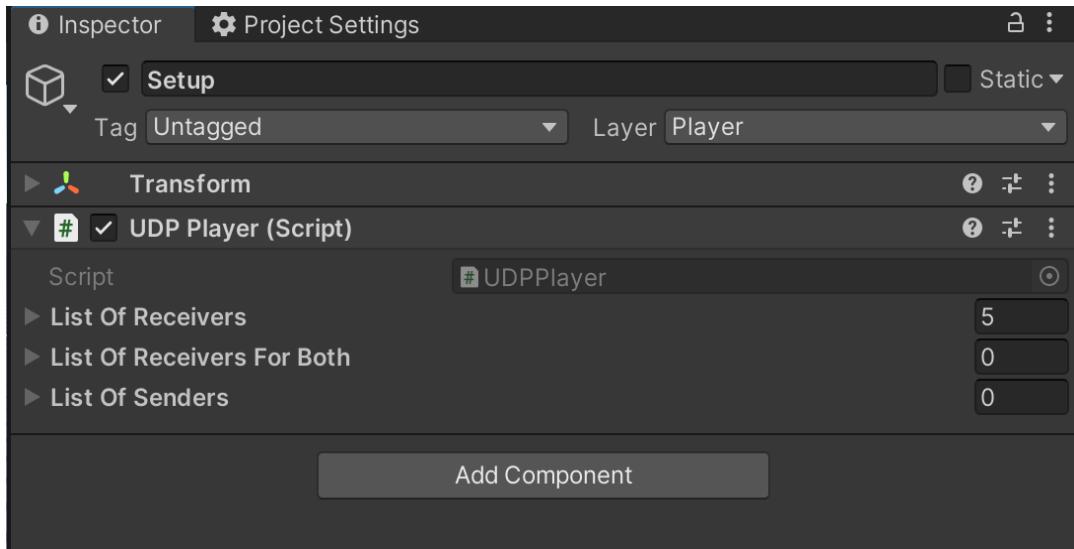
4.26 pav. „Boolean“ tipo gavėjas ir valdytojas.

Komponentas skirtas siuntimo ir gavimo komponentus valdyti (4.27 pav.). Komponentas valdo įdėtus komponentus. Komponento valdymas vyksta per pateiktas funkcijas. Valdomas komponento tipas gali būti pasirenkamas komponento viduje „Type“ langelyje. Komponentas palaiko tik du tipus – „Item“ interaktyvus objektas ir „Door“ valdomos durys. Valdomoms durims nereikia įkelti „Transform“ siuntimo ir gavimo komponentų, kadangi durys sukačia aplink savo ašį.

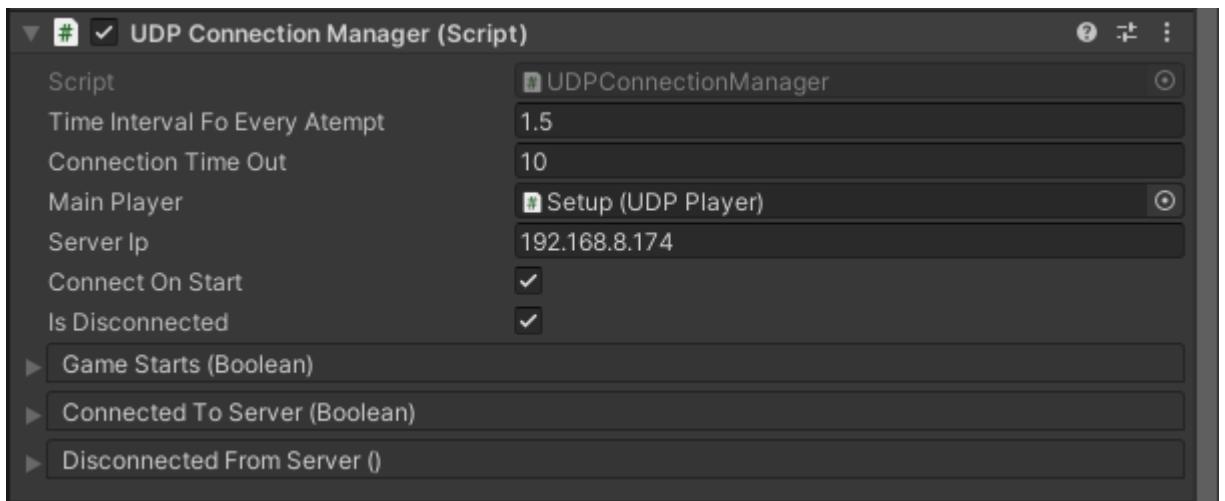


4.27 pav. Objekto būsenos valdymas.

Privalomi daugelio žaidėjų komponentai (4.28 pav. - 4.29 pav.). Komponentai privalo būti žaidime ir vienoje vietoje kaip parodyta nuotraukose. Komponentu išdėliojimui skiriantis daugelio žaidėjų karkasas gali neveikti. Komponentai pateikti paveikslėliuose yra parašyti vėliau.

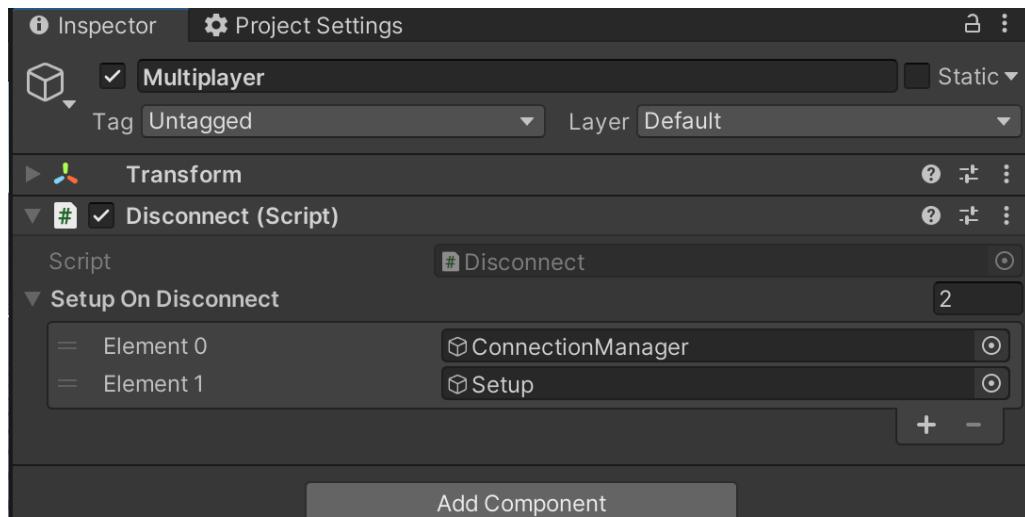


4.28 pav. Privalomas žaidėjo komponentas



4.29 pav. Privalomi komunikacijos su serveriu komponentai.

„Disconnect“ komponentas (4.30 pav.). Atjungimo nuo serverio metu yra ieškomi komponentai, kurie gali būti atkurti į pradinę būseną. Atkuriami komponentai yra ieškomi naudojant „Setup On Disconnect“ sąrašą. „Disconnect“ ieško visų komponentų, kuriuos gali atkurti sąrašo elementuose ir kiekviename pateikto elemento viduje.



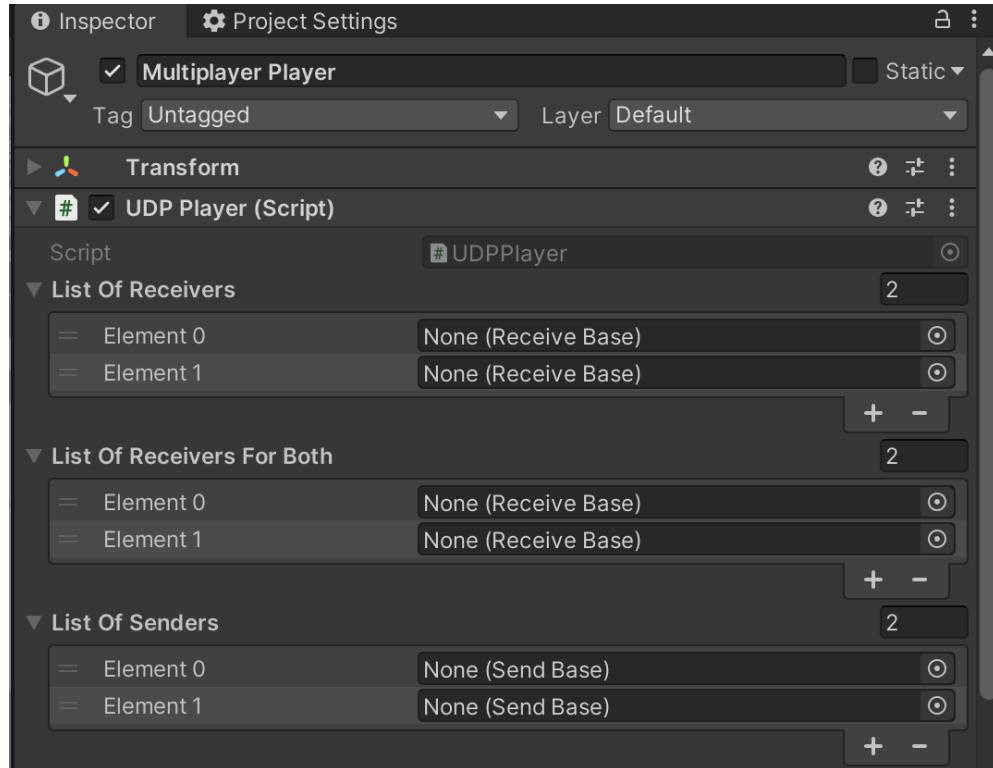
4.30 pav. „Disconnect“ komponentas

Naujo žaidėjo kūrimui yra naudojamas „UDP Player“ komponentas. Šitas komponentas turi būti pridėtas prie žaidėjo objekto žaidime. Komponente yra pateikti trys sąrašai sinchronizacijai. Kiekvienas žaidėjas turi savo žaidimo sesijos identifikacijos kodą, pagal šį kodą yra skirstomi paketai žaidėjams.

„List Of Receivers“ sąraše yra pridedami paketų gavimo komponentai, kurie turi gauti paketus tik iš žaidėjo su tokiu pačiu identifikacijos kodu.

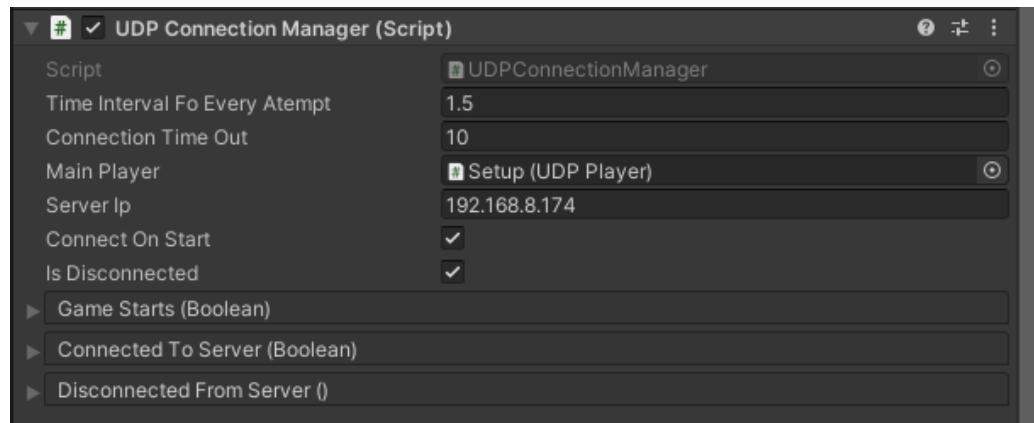
„List Of Receivers For Both“ sąrašas yra skirtas paketų gavimo komponentams sudėti, kuriuos paketus turi gauti žaidėjo komponentas ir jo interneto atvaizdas.

„List Of Senders“ sąrašas yra skirtas aktyvuoti paketų komponentus, kurie būna išjungti arba néra aktyvuoti, paketai yra aktyvuojami žaidėjo aktyvacijos metu (prisijungus žaidėjui prie žaidimo).



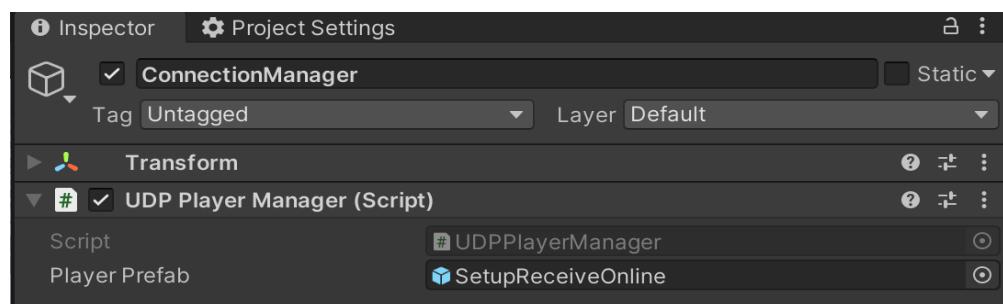
4.31 pav. Žaidėjo komponentas

Daugelio žaidėjų karkaso pagrindinis komponentas yra „UDP Connection Manager“ (4.32 pav.), šis komponentas valdo žaidimo komunikaciją su serveriu. Komponentas suteikia tokias galimybes – prisijungti prie serverio, atsijungti nuo serverio, paketu valdymas, paketu siuntimas.



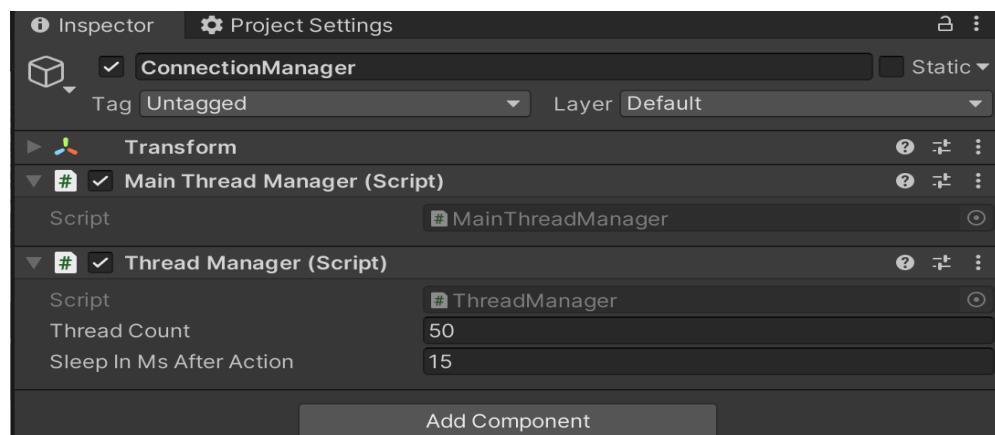
4.32 pav. „UDP Connection Manager“ komponentas

Žaidėjų valdymui yra naudojamas „UDP Player Manager“ komponentas (4.33 pav.), šis komponentas yra neišskiriamas karkaso komponentas žaidėjų valdymui. Komponentas sukuria naujas žaidėjus ir persiunčia paketus skirtus žaidėjų gavimo komponentams. Naujas žaidėjas yra sukuriamas pagal įdėtą objekto. Žaidėjo objektas privalo turėti žaidėjo komponentą (4.31 pav.).



4.33 pav. „UDP Player Manager“ komponentas

Paketų gavimui ir siuntimui yra naudojami gijų darbininkai. Gijų darbininkai įvykd़o visas funkcijas, kurios yra jų sąraše kol yra darbo. Gija po kiekvieno darbo ilsisi nustatyta laiką, kadangi jos ištisai negali veikti kitaip nukentės pagrindinės gijos darbas.



4.34 pav. „Worker“ komponentai.

4.3. Diegimo vadovas

Žaidimo paleidimui naudotojas turi turėti virtualios realybės įrenginį, kuris palaiko „Steam VR“ platformą. Naudotojui užtikrinus, kad virtualios realybės įrangą palaiko „Steam VR“ platformą jis turi įdiegti „Steam“ žaidimų ir kitų aplikacijų parduotuvę. Naudojant „Steam“ praduotuvę naudotojas turi įdiegti „Steam VR“ virtualios realybės programą skirtą virtualios realybės palaikymui. Žaidimas yra palaikomas tik ant „Windows 10“ operacinės sistemos.

Kovos žaidimo minimalūs kompiuterio reikalavimai:

- Vaizdo plokštė - NVIDIA GTX 970 / AMD R9 290
- Processor - Intel i3-6100/AMD Ryzen 3 1200, FX4350
- Operatyvioji atmintis – 8GB

Įdiegus visas reikiamas aplikacijos žaidimo paleidimas vyks tokia nustatyta tvarka:

1. „Steam“ parduotuvė
2. „Steam VR“ virtualios realybės aplikacija
3. Kovos žaidimas

Karkaso naudojimui reikia turėti „Unity“ projektą, kuriame bus suinstaliuotas daugelio žaidėjų karkasas.

Įdiegiant karkasą reikia nueiti į „Unity“ žaidimo projekto aplankalą, nuėjus į projekto aplankalą reikia surasti „Packages“ aplankalą. Atidarius minėtą aplankalą reikia susirasi „manifest.json“ failą ir jį atidaryti naudojant failų redaktorių (4.35 pav.).

```
{
  "scopedRegistries": [
    {
      "name": "npmjs",
      "url": "https://registry.npmjs.org/",
      "scopes": [
        "com.moonleaf"
      ]
    }
  ],
  "dependencies": {
    "com.moonleaf.udpnetwork.unity": "0.1.1",
    ...
  }
}
```

4.35 pav. Failo piedai kuriuos reikia pridėti į „manifest.json“ failą.

Serverio įdiegimui yra reikalinga „.NET“ programinė įranga. Serveris gali būti paleistas ant bet kurios operacinės sistemos, kuri palaiko „.NET“ platformą. Jo paleidimui reikia turėti „Visual Studio“ ar bet kokią kitą programą, kuri gali paleisti „.NET“ projektą.

Naudojant terminalą reikia nueiti į serverio aplankalą ir paleisti serverį paleidus "dotnet run" komandą.

```
/root/UnityGameServer/UnityGameServerUDP/Player.cs(73,21): warning CS0219: The variable 'index' is assigned but its value is never used [/root/UnityGameServer/UnityGameServerUDP/UnityGameServerUDP.csproj]
/root/UnityGameServer/UnityGameServerUDP/Server.cs(113,29): warning CS0219: The variable 'index' is assigned but its value is never used [/root/UnityGameServer/UnityGameServerUDP/UnityGameServerUDP.csproj]
/root/UnityGameServer/UnityGameServerUDP/Player.cs(28,16): warning CS0169: The field 'Player.random' is never used [/root/UnityGameServer/UnityGameServerUDP/UnityGameServerUDP.csproj]
ip address: 141.136.44.126:5002
```

4.36 pav. Paleisto daugelio žaidėjų serverio langas

Rezultatai ir išvados

- Išanalizavus kuriamą sistemą buvo pasirinkti sistemos realizavimo įrankiai ir technologijos. Žaidimo kūrimui buvo pasirinktas „Unity“ žaidimų variklis, kuris turi įvairius įskiepius virtualios realybės žaidimų kūrimui. Buvo pasirinkti du įskiepiai „Steam VR“ ir „VRTK“, šitie įskiepiai leido fokusuotis į karkaso ir žaidimo kūrimą, suteikiant sąveikavimo komponentus, kurie padeda realizuoti žaidėjo sąveiką su virtualiu pasauly.
- Atlikus konkurentų analizę pastebėta, kad daugelio žaidėjų žaidimams yra naudojami skirtini karkasai, skirtiniems žaidimams kurti. Realizavimo metu buvo atkreiptas dėmėsis į konkurentų trūkumus ir privalumus. Kadangi konkurentų įrankiai orientuoti į įvairaus žanro daugelio žaidėjų žaidimų kūrimą, buvo pasirinka kruti daugelio žaidėjų karkasą skirtą specifinei platformai. Sukūrus įrankį buvo pastebėti trūkumai - limituotas žinutės ilgis, nepastovus žinučių siuntimas ir gavimas. Karkaso privalumai yra žinučių dydžio gaudimas, karkaso kodas gali būti kaičiamas pagal kūrėjo poreikius.
- Išanalizavus pasirinktą virtualios realybės žaidimą buvo pasirinktos funkcijos kuriamam kovų žaidimui. Pasirinktos funkcijos padėjo realizuoti ir suprojektuoti daugelio žaidėjų kovų žaidimo funkcijas, kaip - daiktų paėmimas, durų atidarinėjimą, kurios bus sukurtos ir sinchronizuojamos kovų žaidime.
- Darbo metu buvo parengtas daugelio žaidėjų karkaso ir kovų žaidimo sistemos projektas, jo sudarymo metu buvo suprojektuoti funkcioniniai reikalavimai, sistemai iškelti nefunkcioniniai reikalavimai ir sudaryta naudotojo sasajos maketas. Suprojektuotas projektas buvo sėkmingai panaudotas kovų žaidimo ir daugelio žaidėjų kūrimui.
- Sistemos testavimą buvo identifikuojamos daugelio žaidėjų karkaso ir žaidimo klaidos. Statinės analizės metu buvo aptikta nenaudojamų bibliotekų ir kintamuju, kurie buvo pašalinti. Komponentų testavimo metu buvo pasiekta 7,6% kodo padengimas. Sukūrus naują funkciją buvo atliekamas testavimas scenarijais. Kiekvienas scenarijus buvo atliekamas ir patikrinamas ar įvykdytas be klaidų.
- Kovų žaidimas ir daugelio žaidėjų karkasas buvo atlikti remiantis sistemos projektą. Visi funkcioniniai reikalavimai ir nefunkcioniniai reikalavimai buvo išpildyti. Sistemos realizacija veikia kaip buvo suprojektuota.
- Karkasui naudojant „UDP“ komunikacijos protokolą buvo pastebėta problema darant prisijungimą prie serverio. „UDP“ protokolas negarantuoją, kad gavėjas gaus žinutę. Prisijungimui prie serverio prisijungimo žinutės gavimas yra būtinė. Šiai problemai išspręsti buvo siunciami pranešimai į serverį kol jis gaus prisijungimo pranešimą ir prisijungus prie žaidimo serveris siūs atsaką kol klientas sustos siusti prisijungimo pranešimus. Prisijungimui prie serverio būtų galima naudoti „TCP“ protokolą, kuris suteikia prisijungimo galimybes, o žaidimo sinchronizacijoms siuntinėti būtu galima toliau naudoti „UDP“ protokolą.

Žaidimas ir daugelio žaidėjų karkasas nėra pilnai paruoštas išleidimui. Norint karkasą paruošti išleidimui, reikia atlikti sistemos testavimą, sutvarkyti karkas klaidas, bei papildyti jo galimybes. Žaidimo paruošimui, reikia pridėti naują funkcionalamą, kuris pagyvintu žaidimą, pagerinti žaidėjų sinchronizaciją bei atlikti žaidimo testavimą ir ištisysti rastas klaidas. Atnaujintu žaidimui reikėtu atlikti rinkos analizę ir duoti žaidėjams jį pažaisti, surinkus reikiamus duomenis žaidimas galėtu būti išleistas.

Literatūros sąrašas

1. Vyšniauskas, E., & Nemuraitė, L. (2006). Transforming Ontology Representation from OWL to Relational Database. *Information Technology and Control*, 35A(3), 333–343.
2. Masiulis, K., & Krupavičius, A. (2007). *Valstybės tarnyba Lietuvoje: praeitis ir dabartis: kolektyvinė monografija*. Vilnius: Praction.
3. Biržiška, V. (1929). Spaudos draudimo klausimai. *Kultūra*(5), 249-235.
4. Apie LITNET. (2012 m. birželio 05 d.). Paimta 2013 m. balandžio 04 d. iš Litnet: <http://www.litnet.lt/index.php/apie-litnet>
5. Valiulytė, I. (2000 m. vasaris). *Išlaidos krašto apsaugai, jų pagrįstumas ir tikslėliai*. Paimta 2001 m. gruodžio 12 d. iš Sociumas: <http://www.sociumas.lt>
6. Library, D. U. (2009). *IEEE Citation style guide*. Paimta 2013 m. 04 11 d. iš http://libraries.dal.ca/content/dam/dalhousie/pdf/library/Style_Guides/IEEE_Citation_Style_Guide.pdf
7. Gradauskas, R. (2000). Hibridinis velomobilis. *Transporto priemonės - 99*, (p. 81-83). Kaunas.
8. Smith, N. (2021 m. vasario m. 5 d.). *The Washington Post*. Nuskaityta iš Virtual reality is starting to see actual gains in gaming: <https://www.washingtonpost.com/video-games/2021/02/04/virtual-reality-future-games/>
9. Champion, M. K., & Bekele, E. (2021 m. rugsėjo m. 24 d.). *A Comparison of Immersive Realities and Interaction Methods: Cultural Learning in Virtual Heritage*. (Frontiers in Robotics and AI) Nuskaityta iš Frontiers in Robotics and AI: <https://www.frontiersin.org/articles/10.3389/frobt.2019.00091/full>
10. Unity. (be datos). (Unity technologies) Nuskaityta iš Unity: <https://unity.com/>
11. Unreal Engine. (be datos). Nuskaityta iš Unreal Engine: <https://www.unrealengine.com/>
12. SteamVR Unity Plugin. (be datos). Nuskaityta iš Steam VR: https://valvesoftware.github.io/steamvr_unity_plugin/
13. Oculus Integration SDK. (2021 m. balandžio mėn. 29 d.). (Facebook Technologies) Nuskaityta iš Oculus: <https://developer.oculus.com/downloads/package/unity-integration/>
14. XR. (be datos). (Unity Technologies) Nuskaityta iš Unity: <https://docs.unity3d.com/Manual/XR.html>
15. SteamVR. (be datos). (Valve Corporation) Nuskaityta iš Steam.
16. Virtual Reality Toolkit. (be datos). Nuskaityta iš VRTK: <https://www.vrtk.io/>
17. Pun. (be datos). (Exit Games) Nuskaityta iš Photon: <https://www.photonengine.com/pun>
18. Multiplayer and Networking. (2021 m. balandžio mėn. 30 d.). (Unity Technologies) Nuskaityta iš Unity: <https://docs.unity3d.com/Manual/UNet.html>
19. Unity Multiplayer Networking. (be datos). (Unity Technologies) Nuskaityta iš Unity Multiplayer Networking: <https://docs-multiplayer.unity3d.com/>
20. Half-Life: Alyx. (be datos). (Valve Corporation) Nuskaityta iš Steam: https://store.steampowered.com/app/546560/HalfLife_Alyx/
21. .NET. (be datos). (Microsoft) Nuskaityta iš Microsoft: <https://dotnet.microsoft.com/>
22. .NET. (be datos). (Microsoft) Nuskaityta iš Microsoft: <https://dotnet.microsoft.com/>
23. System.Net.Sockets Namespace. (be datos). (Microsoft) Nuskaityta iš Microsoft: <https://docs.microsoft.com/en-us/dotnet/api/system.net.sockets?view=net-5.0>
24. UDP – USER DATAGRAM PROTOCOL. (be datos). (IPV6) Nuskaityta iš IPV6: <https://www.ipv6.com/general/udp-user-datagram-protocol/>
25. GitHub. (be datos). (GitHub, Inc) Nuskaityta iš GitHub: <https://github.com/>

26. *See our new notification and interest settings! Sign-in to review and receive the most relevant information to you.* (be datos). (Facebook Technologies) Nuskaityta iš Oculus: <https://developer.oculus.com/learn/>
27. *Steam.* (be datos). (Valve Corporation) Nuskaityta iš Steam: <https://store.steampowered.com/about/>
28. *MagicDraw.* (be datos). (No Magic) Nuskaityta iš No Magic: <https://www.nomagic.com/products/magicdraw>
29. *Trello.* (be datos). (Atlassian) Nuskaityta iš Trello: <https://trello.com/home>
30. Willetts, S. (be datos). *How DirectX defined PC gaming... with help from a shotgun-toting Bill Gates.* (PC Gamer) Nuskaityta iš PCGAMER: <https://www.pcgamer.com/history-directx-windows-microsoft/>
31. *Visual Studio.* (be datos). (Microsoft) Nuskaityta iš Microsoft: <https://visualstudio.microsoft.com/>