VILNIAUS UNIVERSITETAS
ŠIAULIŲ AKADEMIJA

INFORMACINIŲ TECHNOLOGIJŲ VALDYMO MAGISTRO STUDIJŲ PROGRAMA

ŠARŪNAS KESIARAUSKAS

Magistro studijų baigiamasis darbas

DIRBTINIO INTELEKTO ELEMENTŲ NAUDOJIMAS KOMPIUTERINIAME ŽAIDIME

Darbo vadovas (-ė): Dr. Vaidas Giedrimas

Šiauliai, 2024

Studijuojančiojo, teikiančio baigiamąjį darbą, GARANTIJA

WARRANTY of Final Thesis

Vardas, pavardė *Name, Surname*	**Šarūnas Kesiarauskas**
Padalinys *Faculty*	**Šiaulių akademija** ***Šiauliai Academy***
Studijų programa *Study Programme*	Informacinių technologijų valdymas studijų programa Valstybinis kodas 6211BX001 Information technology management study program State Code 6211BX001
Darbo pavadinimas *Thesis topic*	**Dirbtinio intelekto elementų naudojimas kompiuteriniame žaidime** ***Using elements of artificial intelligence in a computer game***
Darbo tipas *Thesis type*	**Baigiamasis darbas** ***Final Thesis***

Garantuoju, kad mano baigiamasis darbas yra parengtas sąžiningai ir savarankiškai, kitų asmenų indėlio į parengtą darbą nėra. Jokių neteisėtų mokėjimų už šį darbą niekam nesu mokėjęs.
Šiame darbe tiesiogiai ar netiesiogiai panaudotos kitų šaltinių citatos yra pažymėtos literatūros nuorodose.

I guarantee that my thesis is prepared in good faith and independently, there is no contribution to this work from other individuals. I have not made any illegal payments related to this work.
Quotes from other sources directly or indirectly used in this thesis, are indicated in literature references.

Aš, Šarūnas Kesiarausaks, pateikdamas (-a) šį darbą, patvirtinu (*pažymėti*) ☒

Embargo laikotarpis
Embargo Period

Prašau nustatyti šiam baigiamajam darbui toliau nurodytos trukmės embargo laikotarpį:
I am requesting an embargo of this thesis for the period indicated below:

☐ _____ mėnesių / *months*
(embargo laikotarpis negali viršyti 60 mėn. / *an embargo period shall not exceed 60 months*).

☒ Embargo laikotarpis nereikalingas / *no embargo requested.*

Embargo laikotarpio nustatymo priežastis / *Reason for embargo period*:

TURINYS

PAVEIKSLĖLIŲ SARAŠAS

LENTELIŲ SARAŠAS

SANTRUMPŲ SARAŠAS

UN – žaidimo variklis „Unity“

GMS2 - žaidimo variklis „GameMaker Studio 2“

UE4 - žaidimo variklis „Unreal Engine 4“

DI – dirbtinis intelektas (angl. Artificial Intelligence)

UE5 - žaidimo variklis „Unreal Engine 5“

PC - stacionarus kompiuteris (angl. Stationary computer)

UML – Vieninga modeliavimo kalba (angl. Unified Modeling Language)

VFX – vizualus efektai (angl. Visual Effects)

SDK – programinės įrangos kūrimo įrankis (angl. Software Development Kit)

NPC - personažai žaidime (angl. Non-Player Character)

DB – duomenų bazė (angl. Data Base)

FP – žaidimas yra žaidžiamas iš pirmo asmens perspektyvos (angl. First Person)

TP – žaidimas yra žaidžiamas iš trečio asmens perspektyvos (angl. Third Person)

VR – virtuali realybė (angl. Virtual Reality)

AR – augmentuotos realybės aplikacija (angl. Augmented Reality Application)

TD – žaidimas yra žaidžiamas iš antros perspektyvos (angl. Top Down)

SANTRAUKA

Magistro darbo tikslas yra empiriniu būdu pasinaudojus UE5 žaidimo varikliu sukurti mašinėlę, turinčią dirbtinio intelekto elementų. Palyginti sukurtos DI mašinėlės santykį, kokybę ir žmogiškąjį resursą su kitų asmenų, įmonių sukurtais DI.

Šiam tikslui pasiekti pirmiausia turėjome išsiaiškinti, kokie žaidimo varikliai buvo viešai prieinami, ko tie žaidimo varikliai skiriasi vienas nuo kito. Kai nusprendėme kokį žaidimo variklį naudosime, turėjome tiksliai išsiaiškinti kas yra DI, kas DI sudaro ir kaip jį mes panaudosime. Atlikus pilną ir detalią DI analizę, buvo reikalinga peržiūrėti kelis mokslinius straipsnius, kuriuose būtų aprašyta DI naudojimas ir tobulėjimas. Išanalizavome kelias gerai žinomas įmones ir jų sukurtus produktus naudojančius DI. Šio etapo metu buvo kuriamos UML diagramos, kurios buvo panaudotos palengvinti mūsų kuriamą DI. Kuriamas dirbtinis intelektas buvo detaliai sekamas, visi žingsniai, iššūkiai, problemos, nesklandumai ir trūkumai buvo aprašomi ir sekami. Sukurtas DI ir veikimo aprašymo duomenys buvo naudojami atlikti jo ir įmonės *Creative Assembly* sukurto DI palyginimui. Peržiūrėjus visą turimą informaciją, duomenys ir gauti rezultatai panaudoti atsakymui į tyrimo klausimą.

ĮVADAS

Temos aktualumas: Išmaniųjų technologijų naudojimas yra kasdienė žmogaus gyvenimo dalis. Daugeliui žmonių sunku įsivaizduoti gyvenimą be išmaniųjų telefonų, automobilių su savarankiško vairavimo funkcijomis, nešiojamų ir stacionarių kompiuterių, radijo, ausinių su Bluetooth prisijungimu, išmaniųjų laikrodžių, planšečių, lengvai ir visur pasiekiamo interneto su 4G ar net 5G kartos internetu. Visi šie ir kiti panašūs įrankiai per paskutinius penkiasdešimt metų buvo sukurti, tobulinti, gerinami ir testuojami taip, kad jų naudingumas ir naudojimas kiltų, kad žmonės iš skirtingų amžių grupių ir kartų, turėtų prieinamus įrankius, padedančius jiems susidoroti su kasdieniais gyvenimo iššūkiais. Tie iššūkiai gali būti bet kas: nuo grindų valymo, namų darbų ruošimo, dokumentų tvarkymo, prezentacijos paruošimo, brėžinių pertvarkymo, muzikos kūrimo, senų nuotraukų albumų archyvavimo ir daug daugiau. Visiems šiems dalykams buvo reikalinga žmogaus darbo jėga, tai yra rankų darbas. Visa tai reikalaudavo daug darbo ir laiko. Tačiau su tobulėjančiomis technologijomis šie sunkūs darbai, kurie užtrukdavo nuo kelių savaičių iki kelių mėnesių, dabar visa tai galima atlikti per kelias dienas arba net kelias valandas.

Tačiau ši informacija jau visiems asmenims yra gerai žinoma ir girdėta. Penkiasdešimt metų nėra trumpas laikas ir tai nieko nenustebins. Tačiau kaip ir laikas nestovi vietoje, taip technologijos, inovacijos ir naujovės nestovi vietoje. Laikui einant viskas tobulėja ir gerėja, ypač kai pradedame šnekėti apie technologijas ir inovacijas, kurios vyksta šioje srityje. Dabar, kai jau artėjam prie 2024 metų, dauguma žmonių per paskutinius kelis metus, žiniose, reklamose, naujienose arba TV laidose yra girdėję kalbas apie dirbtinį intelektą, dar kitaip žinomą kaip DI (angl. AI). Dauguma tuos žodžius girdėjo, tačiau atsakyti, kas tai ir kuo jis yra ypatingas, turbūt galėtų atsakyti tik nedaugelis. Tiems, kurie nežino, paprasčiausias paaiškinimas būtų, tai, kad dirbtinis intelektas yra kompiuterinė sistema, galinti atlikti užduotis, kurias šiaip atlikti reikėtų žmogaus lygio intelekto. Tai apima algoritmus ir sistemas, kurios gali mokytis, samprotauti, suvokti, spręsti problemas, panašiai kaip žmogus. Jei reikėtų dar paprasčiau pasakyti, tai dirbtinis intelektas yra daiktas, kuris yra sukurtas imituoti žmogaus sugebėjimus, kurie gali būti fizinio tipo kaip automobilio vairavimas arba protinio tipo, kaip labirinto sprendimas, modelių, paveikslėlių atpažinimas ir gautos informacijos pritaikymas prie naujos situacijos.

Taigi dabar, kai žinome, kas yra tas dirbtinis intelektas, turbūt nėra sunku įsivaizduoti, ką tokio tipo technologija gali reikšti žmonijai ir pasaulio ateičiai. Dėl tos priežasties per paskutinius 5-10 metų atsirado įmonių kaip *GoogleAI, Facebook AI Research, NVidia, Samsung,* kurios pradėjo stumti DI intelekto kūrimą ir tobulinimą. Viena iš geriau žinomų įmonių yra *OpenAI*, šita įmonė yra

atsakinga už *ChatGPT* sukūrimą. *ChatGPT* yra sukurtas DI, kuris sugeba remdamasis gautais duomenimis dalyvauti pokalbiuose ir sugeneruoti tikslius, žmogiškus atsakymus.

Dabar turbūt yra sunku įsivaizduoti ir nuspėti, kas bus ateityje. Tačiau nesunku nuspėti, kad ši DI banga nesustos, o tik didės, laikui bėgant prisijungs ir daugiau įmonių, kurios pradės dirbti link savo tipo DI kūrimo ir naudojimo. Bėgant laikui, visa mūsų turima įranga pradės naudoti vieno arba kelių tipo DI, kurių dėka, mūsų turimi darbai taps dar lengvesni. Tačiau iškyla klausimas dėl paprasto žmogaus ir jo prisidėjimo prie DI kūrimo. Ar yra galimybė vienam asmeniui sukurti bent kažkokio tipo DI, kuris galėtų atlikti kažkokio tipo funkciją? Ar asmuo, kuris nėra dirbęs su DI galėtų per tam tikrą laiko limitą, išmokti reikalingą informaciją ir sukurti savo DI? Kokie būtų iššūkiai ir skirtumai tarp DI sukurto įmonės ir vieno asmens?

Tiriamoji problema: DI kūrimas ir vystymas labai priklauso nuo didžiųjų technologinių kompanijų.

Darbo objektas: Dirbtinio intelekto kūrimo procesas.

Darbo tikslas: Empiriniu būdu bandyti nustatyti žmogiškųjų resursų, reikalingų DI elementų kūrimui, santykį su gauto produkto kokybe.

Darbo uždaviniai:

1. Aprašyti DI sąvoką, reikalavimus jam ir jo kūrimo procesui;
2. Apžvelgti DI naudojančius žaidimo kūrimo variklius ir išanalizuoti vieną iš jų;
3. Parengti kuriamo DI planą, projektą naudojant pasirinkta žaidimo variklį;
4. Pasirinkti vieną įmonę, sukūrusią DI. Išanalizuoti reikalingą resursų kiekį, įvertinti jų ir įmonės dydžio santykį;
5. Palyginti abiejų projektų rezultatus.

Tyrimo metodai: Literatūros analizė, eksperimentas.

Darbe taikyti ir programinės realizacijos kūrimo metodai, naudotos priemonės ir įrankiai:

Pagrindinės galimos įrangos, kurios yra prieinamos mums, bent šiuo metu yra UE4 ir UE5 žaidimų varikliai. Abu žaidimo varikliai yra sukurti kompanijos *Epic Game Launcher*. UE4 buvo išleistas 2005 metais, o UE5 išleistas 2022 metų balandžio 5 dieną. Didžiausias skirtumas tarp UE5 ir UE4 yra tai, kad *Unreal Engine 5* turi pačius naujausius ir moderniausius 3D virtualios funkcijos įrankius, leidžiančius kūrėjams sukurti realistiškai tikrus žemėlapius, veikėjus, atmosferas ir pasaulius. Visi prieinami įrankiai ir technologija padeda sukurti žaidimus ir filmus, kurie su kiekviena diena „laužo realybės vartus‘‘. Negalime pamiršti, kad dėl *Epic Game Launcher* įmonės pastoviai išleidžiamų atnaujinimų ir tvarkymų, esami įrankiai ir technologijos tobulėja ir tampa vis geresni.

Įmonės *Epic Game Launcher* sukurti žaidimo varikliai, UE4 ir UE5, naudoja C++ programavimo kalbą, kaip savo pagrindinę, tačiau ir vienas ir kitas žaidimo variklis turi papildomą sukurtą funkciją, suteikiančią galimybę vartotojui kurti savo norimą produktą be C++ programavimo kalbos mokėjimo ir naudojimo. Tos funkcijos pavadinimas yra *blueprint*, dar kitaip žinomos kaip *NODES*. Tie *blueprints* yra vizualinės scenarijų kūrimo įrankis, kuris leidžia sujungti skirtingų tipų dokumentus, failus, kodus, funkcijas į vieną bendrą failą. *Blueprints funkcija yra* skirta kūrėjams, programuotojams, dizaineriams ir menininkams, kurie neturi didelių programavimo įgūdžių, sugebėjimų ir patirties, kuriant žaidimus ir filmus su jau iš anksto suprogramuotomis mazgų sistemomis.

Blueprints suteikia naujiems kūrėjams draugišką būdą suprogramuoti įvairius naujus projektus, kuriuos bus galima lengvai sutvarkyti, koreguoti, modifikuoti, perdaryti ir sujungti be papildomų žingsnių. Asmenis, kurie naudoja šią sistemą, gali kurti ir jungti mazgus, atvaizduojančius įvykius, logiką, kintamuosius ir funkcijas taip, kad sukurti dalykai susisietų tvarkingai be jokių sąveikos pažeidimų ir gedimų.

Visi šie dalykai suteikia galimybę vartotojams kurti skirtingų tipų sistemas, kurios atstovauja skirtingoms funkcijų sąlygoms ir veiksmams. Visas šis lengvas priėmimas ir modifikavimas suteikia naują požiūrį į programų, žaidimų ir filmų kūrimą bei prototipų testavimą ir eksperimentavimą.

Gautų rezultatų apžvalga ir reikšmingumas: Naudojantis atlikta DI, įmonių ir žaidimų variklių analize, bus pasirinktas vienas labiausiai tinkamas žaidimo variklis, kuris atitiks visus mūsų DI kūrimo reikalavimus. Pasirinkus žaidimo variklį bus sukurtas žaidimo laukas, kuriame pradėsime DI kūrimo testavimą. DI kūrimo ir testavimo metu bus naudojamasi UML diagramomis, kurios buvo sukurtos tuo pačiu metu, kai buvo daroma DI analizė. Vienas iš pagrindinių kuriamų DI bus lenktyninio tipo automobilis su sukurta DI autopiloto funkcija. Kūrimo ir testavimo metu visos patirtos problemos ir klaidos bus fiksuojamos ir užrašomos. Sukurtas DI bus išsaugotas faile, kurį bus galima parsisiųsti ir ištestuoti. Taip bus suteikta galimybė visiems besidomintiems asmenims ištirti, išbandyti sukurtą DI ir peržiūrėti jo kodą. Kadangi čia yra mano pirmasis bandymas kurti DI, galima skaityti, kad sukurtas DI bus *pre-alpha* versijos. Turimas sukurtas DI bus vėliau panaudotas gautų rezultatų palyginimui su pasirinktos įmonės sukurtu DI. Skirtumai tarp sukurtų DI bus pateikti išvadose. Ateityje yra galimybė praplėsti sukurtą DI, bus galima pridėti naujų funkcijų, idėjų ir ištaisyti rastas problemas, tačiau tai bus palikta ateičiai.

Darbo sunkumai ir apribojimai: Pagrindinė problema, kuri buvo patirta darbo metu yra tai, kad vienas asmuo buvo atsakingas už daugelį skirtingų užduočių. Užduotys, kurios galėjo būti nuo DI medžiagos rinkimo, naudojamo žaidimo variklio mokymąsi, programos rašymą, testavimą ir

tvarkymą, taip pat reikia nepamiršti ir pasirinktos įmonės analizės ir jos sukurtos DI struktūros darbo eigos. Visi šie ir kiti dalykai sudaro darbo krūvį, kuris vienam asmeniui tampa labai sunkiai atliekamas. Čia yra viena iš priežasčių, dėl kurios daugumą susigalvotų planų, užduočių ir norų nebuvo galima įvykdyti ir prievartos būdu reikėjo arba šalinti, arba keisti taip, kad atitiktų laiko apribojimus.

Darbo loginės struktūros ir paaiškinimai: Darbo projektą sudaro keturi skyriai - analizės dalis, kurioje bus analizuojama DI tipai, įmonės ir tų įmonių sukurti DI, publikai prieinami žaidimo varikliai; techniniai testavimai, kur bus bandoma testuoti ir išbandomi galimi žaidimo varikliai; projekto dalis, kurioje bus aprašoma, kuriamo DI nustatyti tikslai ir jo kūrimo aprašymas; DI realizacijos dalis, kurioje bus pateikiama, koks žaidimo variklis buvo naudotas, kaip buvo atliktas DI kūrimas ir su kokiais iššūkiais susidūrėme. Gautus rezultatus ir išvadas palyginsime su pasirinktos įmonės sukurtais DI ir peržiūrėsime pagrindinius skirtumus, trūkumus, iššūkius, problemas ir rezultatus tarp vieno asmens ir visos įmonės sukurtos DI.

Svarbiausia naudota literatūra: Darbe buvo naudojama daug skirtingų informacijos šaltinių. Naudojami literatūros šaltiniai: moksliniai straipsniai, *Youtube* vaizdo įrašai, *Reddit* skelbimai, *Github* skelbimai, *Discord* grupės, UE5 mokymosi kursai ir pagalbos grupės, kurios buvo sudarytos socialiniuose tinkluose. Tačiau pačios svarbiausios ir labiausiai naudojamos medžiagos yra vaizdo įrašai iš *Youtube* svetainės ir moksliniai straipsniai, kaip „Failure of AI projects: understanding the critical factors, Procedia Computer Science“, „AI-enchanced Crowdsourcing as an Element of Information Systems Development“, „Dirbtinis Intelektas Edukacijoje: integravimo galimybių teorinė analizė“, „Dirbtinis intelektas ir autorių teisė“ ir „Deep Reinforcement learning for general Video Game AI“. Mokslinių straipsnių buvo ieškoma naudojantis *Google Scholar* naršyklės pagalba.

Duomenys apie sukurtos sistemos diegimą: Sukurtas DI šiuo metu yra skirtas eksperimentavimo tikslams, tai yra, kad sukurtas DI nėra visiškai užbaigtas. DI buvo kuriamas naudojantis UE5 5.2 versiją, tai reiškia, kad laikui einant, sukurtas DI gali nustoti veikti taip, kaip buvo numatyta. Dėl tos priežasties yra suteikta galimybė visiems norintiems ir besidomintiems asmenims, parsisiųsti sukurtus DI failus ir juos patiems ištestuoti ir išbandyti. Visa informacija apie failų parsiuntimą, instaliaciją ir paleidimą yra pateikta dokumento apačioje prie priedų.

Darbo struktūra ir apimtis: Tai darbą sudaro 56 puslapiai, 21 paveikslėliai, 3 lentelės, testavimas ir 3 priedai.

1. ANALIZĖS DALIS

Šiame skyriuje buvo atliekami keli veiksmai. Vienas iš jų yra DI analizė, turėjome tiksliai išsiaiškinti, kas yra DI, kas jį sudaro, aprašyti kelis žinomus DI tipo variantus, jų reikalavimus bei kūrimo procesus. Toliau atliekama kompiuterinių žaidimų variklių analizės dalis, peržiūrimas variklis, atitinkantis mūsų reikalavimus ir lūkesčius. Buvo pasirinkta viena arba kelios įmonės ir jų sukurti DI, rinkimasis buvo atliekamas pagal tris parametrus: Vienas iš jų buvo pagal tai, kaip lengva yra gauti leidimą ir prieigą naudotis sukurtą DI. Antras vertinimo parametras buvo populiarumas, tai yra, kiek gerai yra žinomas ir girdėtas sukurtas DI. Paskutinis parametras, pagal kurį buvo pasirinktos įmonės ir jų sukurti DI, yra pagal tai, kiek atitinka mūsų kuriamo projekto modelį. Šis kriterijus buvo vienas iš svarbiausių, nes pagal šį kriterijų buvo nusprendžiama, kokio tipo DI bus bandoma kurti. Tuo pačiu metu buvo kuriamos diagramos, kurios vėliau buvo panaudotos palengvinti mūsų kuriamą DI.

1.1. Temos aktualumas

Klausimas, koks yra optimalus darbuotojų skaičius programinės įrangos projektuose ir kaip jį apskaičiuoti pagal numatomos PĮ savybes yra aktualus [3,8,7]. Taikomi optimizavimo [3], daugiakriterinės analizės [8] ir kt. metodai. Tačiau DI plitimo kontekste net šių tyrimų rezultatai tolsta nuo tikrovės.

Iš vienos pusės, DI priemonės paspartina PĮ kūrimo procesą, pagerina jo kokybę [12]. Kita vertus – dirbtinio intelekto įrankiai patys savaime yra programinės įrangos projektų objektas, t.y. kurdami DI, mes kuriame programinę įrangą arba bent jos dalį. Dirbtinio intelekto (DI) naudojimas pastaraisiais metais labai išaugo, tačiau daugeliui įmonių nepavyksta pasiekti laukiamos naudos arba projektai nutraukiami net jų nepabaigus. Nors yra nemažai ankstesnių tyrimų, kuriuose pabrėžiami AI projektų iššūkiai, svarbiausi veiksniai, lemiantys projekto nesėkmę, dažniausiai nežinomi.

Darbe [18] panagrinėtos pagrindinės DI projektų klaidos. Tyrimo [18] metu buvo organizuoti interviu su įvairių pramonės šakų AI srities ekspertais, o rezultatai analizuojami naudojant kokybinės analizės metodus. Rezultatai rodo, kad projekto nesėkmę gali sukelti tiek organizacinės, tiek technologinės priežastys. Kaip rizikos faktoriai minimi pernelyg aukšti reikalavimai, srities, kurioje bus taikoma AI sudėtingumas, kūrimui skirto biudžeto ar kt. resursų trūkumas, techninės problemos (tokios kaip modelio nestabilumas, galimybė manipuliuoti rezultatais ir t.t).

Darbe [8] atkreipiamas dėmesys, kad evoliucionuojant DI gali būti vis labiau aktualu individams (atskiriems asmenims ar mažoms kompanijoms) būti ne tik DI įrankių naudotojais, bet ir

jų kūrėjais. Panašiai, kaip šiuo metu naudodami socialinius tinklus esame *web* turinio kūrėjai. Tačiau literatūroje AI projektų sėkmės priklausomybė nuo įmonės dydžio yra panagrinėta silpnai [18]. Maža to, šio magistro darbo rašymo metu nepavyko rasti nei vieno šaltinio kuris teoriniu ar praktiniu aspektu nagrinėtų individualių DI projektų specifiką.

1.2. Dirbtinis intelektas

Dirbtinis intelektas nėra visiškai naujas terminas ir visa jo idėja neatsirado per paskutinius 10 metų. Paties dirbtinio intelekto istoriją galima atsekti beveik prie šimtą metų, iki pat tų laikų, kai XX amžiuje žmonės bandė sukurti mašinas, imituojančias protingą elgesį. Tačiau pirmasis oficialiai terminuotas sritis naudojanti DI terminą buvo datuojama XX amžiaus viduryje, tai Alano Turingo darbas kompiuterijos srityje Dartmuno konferencijoje, kuri vyko 1956 metais.

Pats dirbtinis intelektas, dar kitaip žinomas kaip AI (angl. *Artificial Intelligence*), reiškia žmogaus intelekto simuliaciją mašinose, kuri yra suprogramuota galvoti, mokytis ir atlikti užduotis, kurios dažniausiai reikalauja žmogaus intelekto ir sugebėjimų. Ši technologija leidžia mašinoms priimti sprendimus, atpažinti modelius, analizuoti duomenis ir spręsti problemas, panašiai, kaip spręstų žmogus. Tačiau skirtumas tarp žmogaus ir mašinos naudojančios DI yra tai, kad mašina gali atlikti visų tipų darbus, šimtus, net ir tūkstančius kartų greičiau už žmogų. Visi žino, kad dabar turimi kompiuteriai ir sukurti DI gali atlikti nerealius ir neįtikėtinus dalykus, tokius kaip meno kūrimas, kodo tvarkymas, dokumentų surašymas, teksto apibendrinimas, objektų ir planų testavimas. Tačiau kaip ir žmonės, laikas nestovi vietoje taip ir DI panaudojimas, ir galimybės nestovi vietoje. Jau dabar matote DI sukeltą revoliuciją pramonės šakose ir kasdieniame gyvenime. Jau dabar sveikatos priežiūros srityje DI padeda diagnozuoti ligas, atrasti vaistus ir sudaryti individualizuotus gydymo planus. Finansų srityje DI padeda optimizuoti prekybos strategijos ir rizikos valdymus. Automobilių sektorius tiria autonomines transporto priemones, varomas dirbtinio intelekto, kad būtų saugesnis ir efektyvesnis. Natūralios kalbos apdorojimas (NLP) palengvina kalbos vertimą, virtualius asistentus ir jausmų analizę.

Tačiau nepaisant visos padarytos pažangos, DI susiduria su savo tipo iššūkiais. Kaip etinės dilemos, apimančias algoritmų šališkumą, privatumo pažeidimą, darbo vietų praradimą dėl automatizavimo, problemos dėl DI panaudojimo karo ir stebėjimo priežastims. Tai yra tik keli iš daugumos turimų ir numatomų problemų, kuriuos pristabdo DI naudojimą ir tobulėjimą. Dėl tos priežasties yra kuriama ir pridėta papildomų reguliacijų, taisyklių ir struktūrų, kurių pagrindinė paskirtis yra suvaldyti, pristabdyti ir kontroliuoti DI.

1.3. Žinomi dirbtinio intelekto tipai

Pats svarbiausias žingsnis visame šitame procese yra DI tipų analizės dalis. Atliekant šį žingsnį, privalome labai tiksliai įsigilinti į [1,7,9,13,14] ir skirtumus tarp žinomų DI tipų. Priklausant nuo šios analizės bus sprendžiama, kokio tipo DI bandysime kurti savo žaidime. Taip pat reikia atsižvelgti į mūsų turimus apribojimus, kaip limituotus finansus, laiką ir įrangą. Visi šie 3 aspektai bus labai svarbūs ir labai reikšmingi kuriant mūsų pasirinktą DI.

DI tipai:

- ***Reactive machines*** – dirbtinio intelekto tipas, kuris veikia pagal jau iš anksto nustatytas taisykles ir algoritmus. Viskas veikia pagal tam nustatytas įvestis ir dirgiklius, kurie sukuria iš anksto nustatytus atsakymus. Šios sistemos veikia tiesiogiai reaguodamos į artimiausią aplinką, nesugebėdamos formuoti prisiminimų ar mokytis iš praeities patirties. Jie analizuoja esamą įvestį ir generuoja iš anksto nustatytus atsakymus, laikui bėgant nesikeičia ir neprisitaiko. Pagrindinės reaktyviųjų mašinų savybės:
    - Šios mašinos laikosi iš anksto nustatytų taisyklių arba algoritmų, kad nustatytų jų reakciją į tam tikras įvestis ar situacijas.
    - Negali saugoti informacijos arba mokytis iš ankstesnės sąveikos. Kiekvienas atsakymas generuojamas nepriklausomai, remiantis dabartine įvestimi, neatsimenant praeities įvykių.
    - Sistemos priima sprendimus arba atlieka veiksmus remdamosi tik dabartine įvestimi arba kontekstu, neatsižvelgdamos į ankstesnius duomenis ar būsimas pasekmes.

  Reaktyvių mašinų pavyzdžiai apima kai kurias pagrindines dirbtinio intelekto formas, kurios reaguoja į konkrečius paleidiklius arba įvestis su fiksuotais atsakymais, pvz., tam tikri algoritmai arba sistemos, sukurtos paprastoms užduotims, kaip: automatiniams pranešimų atsakymams arba pagrindinių sprendimų priėmimui.
- ***Limited Memory AI*** – dirbtinio intelekto tipas, galintis išlaikyti ir panaudoti tam tikrą praeities patirtį ar duomenis, kad priimtų geresnio tipo sprendimus, veiksmus. Nors ir ribotai, palyginti su žmogaus atmintimi ar mokymosi galimybėmis, pagrindinės savybės būtų:
    - Mokymasis iš istorinių duomenų: šios DI sistemos turi galimybę mokytis iš istorinių duomenų ar patirties ir juos įtraukti į sprendimų priėmimo procesus.
    - Talpos apribojimai: skirtingai nei žmogaus atmintis, ribotos atminties DI atminties arba saugojimo talpa yra apribota, todėl galima išsaugoti ir naudoti būsimiems sprendimams tik ribotą praeities duomenų ar patirties kiekį.

- Patobulintas sprendimų priėmimas: naudodamas ankstesnius duomenis ar patirtį, ribotos atminties DI gali priimti labiau pagrįstus sprendimus arba laikui bėgant pagerinti savo veiksmus.

Ribotos atminties DI pavyzdžių galima rasti įvairiose programose, sistemose internetinėse platformose arba tam tikrų tipų autonominėse transporto priemonėse. Šios sistemos analizuoja istorinius naudotojų duomenis, kad suasmenintų rekomendacijas arba panaudotų ankstesnę patirtį, kad pagerintų vairavimo sprendimus, taip parodydamos ribotą gebėjimą mokytis ir prisitaikyti, remiantis sukaupta informacija ar patirtimi.

- ***Theory of Mind AI*** – čia dirbtinio intelekto lygis, kuris turėtų suprasti ir interpretuoti žmogaus emocijas, įsitikinimus, ketinimus ir psichines būsenas. Ši koncepcija apima DI sistemas, turinčias galimybę suprasti ir numatyti žmogaus elgesį, perspektyvas, emocijas ir ketinimus. Pagrindinės savybės būtų:
    - Žmogaus psichinių būsenų supratimas: šios DI sistemos turėtų suprasti ir interpretuoti žmonių emocijas, įsitikinimus, troškimus ir ketinimus, taip suprasdamos psichines būsenas, kurios nulemia žmonių elgesį.
    - Žmogaus elgesio numatymas: proto teorija DI leistų mašinoms numatyti ir numatyti žmogaus veiksmus ir reakcijas, remiantis jų supratimu apie žmogaus psichines būsenas ir emocijas.
    - Empatinė sąveika: Tokios DI sistemos galėtų parodyti empatiją ir tinkamai reaguoti į žmogaus emocijas, todėl sąveika su mašinomis taptų labiau žmogiška ir intuityvesnė.

  Šio tipo DI tebėra teorinė ir visa idėja ir tikslas šio DI yra išsiaiškinti ir suprasti žmogaus emocijų, ketinimų ir įsitikinimų sudėtingumą. Juo siekiama leisti dirbtinio intelekto sistemoms suprasti žmogaus elgesį ir reaguoti į jį tokiu būdu, kuris atspindėtų gilesnį žmogaus psichologijos ir psichinių būsenų supratimą.
- Self-Aware AI – kalbama apie DI tipą, kuris pažymi bent tam tikro tipo savimonę arba sąmonės būvimo požymius. Tačiau tikrasis DI suvokimas, kuris yra visiškai panašus į žmogų dar tebėra tik teorijos ir filosofijos koncepcija, o ne dabartinė realybė. Pagrindiniai dalykai, kuriuos gali padėti suvokti šį DI yra:
    - Turi supratimą apie savo būseną ir savimonę: turi gebėjimą suvokti savo egzistavimą, tapatybę, sąmonę, mintis ir galbūt net savo esamas emocijas. Šio tipo DI supranta savo galimybes, apribojimus ir aplinką, kurioje jis yra ir veikia. Reikia pabrėžti, kad šiuo metu dar nėra sukurta DI, kuris turėtų tokio tipo funkciją, tai yra savęs suvokimą. Toks dalykas dar viršija mūsų turimas technologijas ir tų technologijų galimybes. Žmogaus sąmonė yra sudėtingas dalykas ir dar nėra visiškai suprantamas. Tai jį atkartoti, kai mes dar jo nesuprantame yra neįmanoma misija.

- Galimybė nuspėti ateitį: turi bent vieno ar kito tipo galimybė nuspėti ateitį.

Apibendrinant galima teigti, kad šio tipo DI dar nėra sukurtas ir prieinamas. Šio tipo DI tipas bus prieinamas tada, kai bus turimos aukštesnio lygio technologijos. Tai ateities technologijos.

- ***Narrow AI*** arba dar kitaip žinomas kaip ***Weak AI*** yra vienas iš labiausiai naudojamų DI tipų. Šis DI specializuojasi tuo, kad jis atlieka konkrečias užduotis ar funkcijas ribotoje srityje. Tai yra, kad jis neturi turėti papildomų funkcijų, leidžiančių jam mokytis, tobulėti ar įsiminti atliktus veiksmus. Visas jo tikslas yra nukopijuoti arba atkartoti jau iš anksto suprogramuotus ir įdiegtus veiksmus. Kartais retais atvejais šis DI yra naudojamas veiksmų tobulinimui, kai norima pasiekti aukščiausio lygio prie vieno tikslaus darbo. Tai gali būtų nuo lėktuvo autopiloto nusileidimo kontrolės, žaidimuose naudojamo priešo, monstro suprogramavimo smegenų daliai arba dalykams kaip roboto valytojo sistemoms. Šio DI pagrindinės savybės yra:
  - Tikslus limito nustatymas: Sukurtas ir išmokytas atlikti konkrečias užduotis ar funkcijas, pvz., kalbos vertimą, vaizdų atpažinimą, žaisti žaidimus ar valdyti konkrečius verslo procesus.
  - Ribota taikymo sritis: Šios DI sistemos veikia tiksliai apibrėžtoje ir ribotoje srityje. Jie neturi bendrųjų žinių ar gebėjimų, viršijančių jiems paskirtą užduotį.
  - Konkrečios užduoties mokymas: siaurieji DI modeliai mokomi naudojant didelius duomenų rinkinius ir specifinius algoritmus, pritaikytus jų numatytoms užduotims, optimizuojant jų veikimą tai konkrečiai funkcijai.
  - Didelis našumas: Savo specializuotose srityse siauros DI sistemos dažnai pasižymi dideliu tikslumu ir efektyvumu, pranokdamos žmogaus galimybes atliekant tam tikras užduotis.

  Vienas iš gerai žinomų atvejų, kur yra naudojama šito tipo DI yra *Siri*, *Alexa* ir *Google Assistant*, kurie puikiai, atpažįsta žmonių balso komandas ir į jas reaguoja su iš anksto suprogramuotais veiksmais, kaip: nustatyti priminimus, teikti informaciją arba valdyti išmaniuosius namų įrenginius. Kiti atvejai apima rekomendacijų sistemas, naudojamas srautinio perdavimo platformose arba internetinės prekybos svetainėse, kuriose siūlomas turinys arba produktai pagal naudotojų pageidavimus, taip pat dirbtinio intelekto pokalbių robotai, tvarkantys klientų aptarnavimo užklausas svetainėse.
- ***General AI*** arba dar žinomas kaip ***Strong AI*** yra DI tipas, kuris pasižymi žmogaus kognityviniais sugebėjimais. Tai yra DI, kuris turi intelektą ir gebėjimus, lygiaverčius

žmogaus intelektui. Tokio tipo DI dažniausiai sugeba atlikti tam tikrus veiksmus, užduotis geriau nei žmogus. Tokio tipo DI galima nusakyti pagal:

- Žmogui panašus intelektas: Šio tipo DI siekia atkartoti žmogaus lygio intelektą ir pažinimo gebėjimus, sugeba imituoti žmogaus įgūdžių, samprotavimo, problemų sprendimo gebėjimus, kūrybiškumo ir gebėjimo prisitaikyti įvairiose užduotyse;
- Universalumas: Šito tipo DI būta universalus ir lengvai pritaikomas, jį galimą pritaikyti įvairiems iššūkiams;
- Supratimas ir mokymasis: Gali mokytis ir tobulėti iš duomenų, kurie yra jam prieinami duomenų bazėje, taip pat sugeba tobulėti ir mokytis iš savo atliktų veiksmų ir gautų rezultatų.

Apibendrinant galima pasakyti, kad ***„General“*** arba ***„Strong AI“*** yra hipotetinė dirbtinio intelekto viršūnė, turinti plataus spektro intelektinius gebėjimus, panašius į žmogaus intelektą. Nors tai tebėra mokslinio tyrimo tikslas ir objektas.

- ***Emotional AI*** – kaip ir pavadinimas jau turbūt nusako, šio tipo DI yra ypatingas tuo, kad jis gali numatyti, nuspėti, nusakyti žmogaus emocijas ir jausmus. Jis yra specifiškai naudingas tuo, kad tokio tipo DI atliktas gerai galėtų padėti nustatyti žmogaus būseną. Tai yra pasakyti ar žmogus meluoja, ar sako tiesą, ar jaudinasi, o gal pyksta. Šios emocijos būtų labai naudingos specialistams dirbantiems su žmonėmis. Tokio tipo DI intelektas pasižymi:
  - Žmonių emocijų atpažinimu;
  - Žmonių emocijų generavimo išraiška;
  - Emocijų supratimu ir jų valdymu.

1.4. Labiausiai naudojami DI tipai

Daugumą gali nustebinti, bet iš visų prieš tai išvardintų DI tipų, labiausiai naudojami yra trys DI ir jie būtų:

- *Narrow AI;*
- *Reactive machine;*
- *Limited memory AI*;

1.5. Labiausiai kompiuteriniuose žaidimuose naudojamas DI

DI, kuris yra naudojamas daugiausiai žaidimuose, būtų *Narrow AI.* Tai šio tipo DI yra naudojamas žaidimuose gan ilgą laiko tarpą. Nuo pačios žaidimų istorijos pradžios su *PING-PONG* iki dabar su tokiais žaidimais kaip *Alien Isolation, Red Dead Redemptions, Witcher 3, GTA V, 7DTD* ir daug daugiau. Priežastis dėl ko šio tipo DI yra labiausiai paplitęs ir naudojamas žaidimuose, gali

būti jo paprastume ir idėjoje. Kai žmonės pradeda kalbėti apie žaidimuose esančius NPC, priešų ir monstrų DI, dauguma asmenų pradeda įsivaizduoti, kad juos sudarantis DI privalo būti kažkas komplikuota ir sudėtinga. Kai kurie asmenys gali net pajuokauti ir prilyginti robotams iš gerai žinomo filmo *Terminatorius*.

1.6. Viešai prieinami žaidimo varikliai

Iš galimų pasirenkamų žaidimo variklių mes turime daug galimų variantų, kaip:

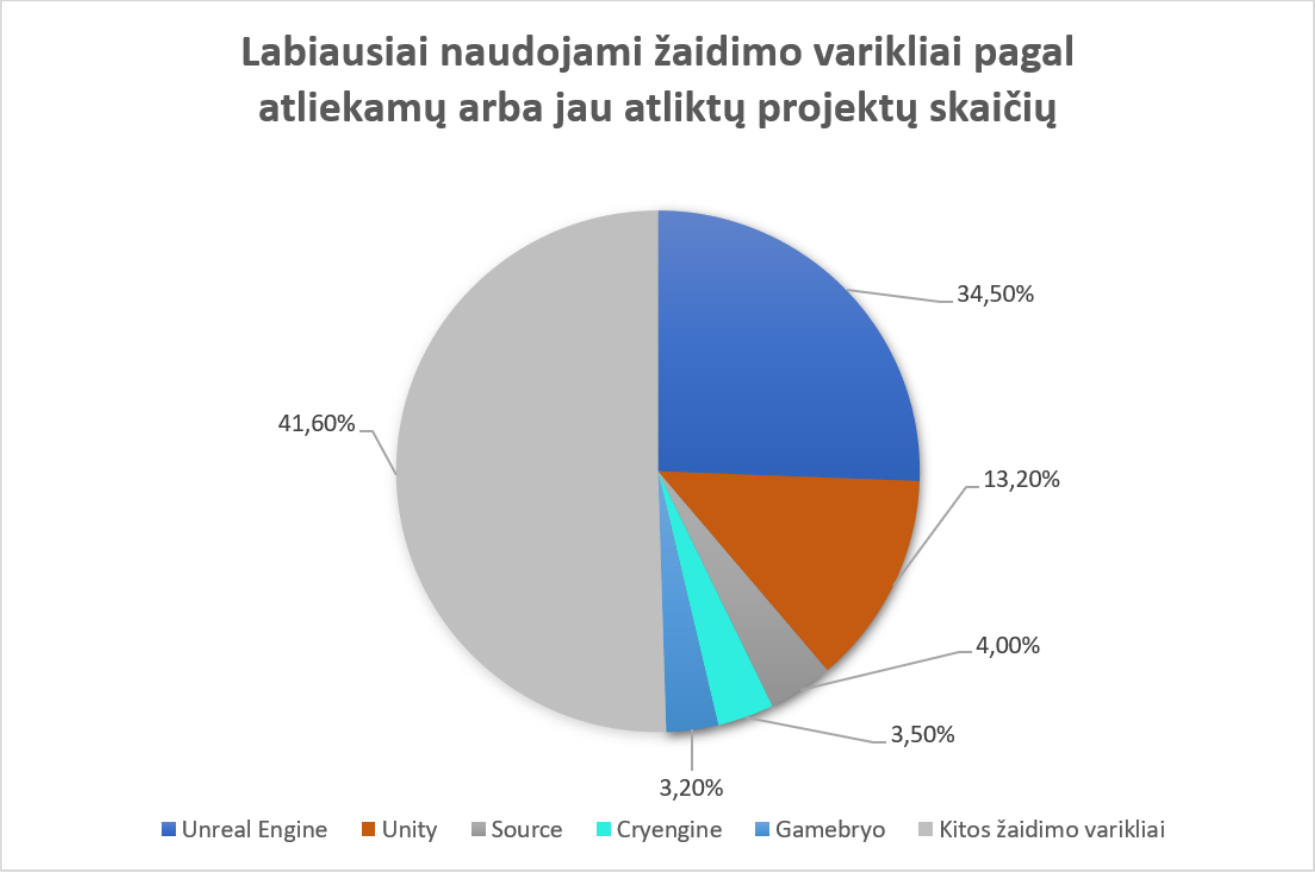
- GameMaker Studio 2
- Unity
- Unreal Engine 5
- Unreal Engine 4
- ANVIL
- Source

Naudodamiesi informacija rasta iš *gamedeveloper* svetainės galime pasakyti, kad pats populiariausias žaidimo variklis pagal atliekamų arba jau atliktų projektų skaičių būtų *UE* 4 ir UE 5, šių dviejų variklių naudojimas yra beveik 35% rinkos. Antrą vietą užimtų *Unity* su 13.2% , o toliau einantys žaidimo varikliai būtų tokie, kurie yra mažiau žinomi ir mažiau naudojami: [5]

- Source su 4.0%
- Cryengine su 3.5%
- Gamebryo su 3.2%
- IW su 2.9%
- ANVIL su 1.7% ir taip toliau

Lengvesniam peržiūrėjimui buvo sudaryta diagramą, kurią galite matyti (**1 pav.**).

1.7. Pasirinktų žaidimo variklių analizė



Šaltinis: sudaryta autoriaus remiantis šaltiniu.

1 pav. Labiausiai naudojami žaidimo varikliai

Pagal jau turimus duomenis, mes matome, kad turime daug skirtingų variantų. Tačiau buvo nuspręsta pasirinkti ir išanalizuoti tik keturis žaidimo variklius. Tai buvo atlikti dėl kelių priežasčių, vienas iš jų buvo dėl laiko stokos, o kitas buvo dėl to, kad mums reikia sukurti lenktyninį automobilį su auto piloto DI funkcijomis. Dėl to mūsų pasirinkti keturi žaidimo varikliai būtų: *GameMaker Studio 2, Unity, Unreal Engine 4 ir Unreal Engine 5*.

Tikrinant skirtumas tarp šių 4 žaidimo variklių randame tai, kad GMS2 yra labiausiai naudojamas, kuriant mobiliojo telefono programas ir žaidimus. Mums reikia sukurti kompiuterinį žaidimą, kuris turėtų dirbtinio intelekto elementų, dėl tos priežasties mums reikia ieškoti žaidimo variklio, kuris turi didesnį kiekį funkcijų ir valdymo priemonių prie žaidimo kūrimo dalies. GMS2 yra naudojamas antros perspektyvos žaidimų kūrimams. Dėl tos priežasties teko atmesti GMS2 naudojimą, iš mūsų pradinių keturių variklių sąrašo. Mums liko tik UN, UE4 ir UN5 [6].

Žaidimo variklis *Unity* yra ypatingas tuo, kad iš žiūrimų žaidimo variklių, šis variklis yra vienintelis, kuris naudoja *C#* ir *JavaScript* programavimo kalbą. UE5 naudoja C++ programavimo kalbą, taip pat kaip UE4.

Papildomas punktas apie UN yra tai, kad šis variklis turi uždėjęs limitą naujiems vartotojams. Tai yra, kai vartotojas nori parsisiųsti žaidimo variklį, jis turi tik du galimus variantus:

- *Personal Unity* – suteikiama ne visi priėjimai prie programinės įrangos, kai žaidimas yra užbaigtas, yra uždedamas logotipas nurodantis, kad žaidimas buvo kurtas su UN. Šis variantas yra nemokamas [11].
- *Professional Unity* - suteikiami visi tie patys priėjimai kaip su *Personal Unity* variantų. Užbaigti žaidimai yra paleidžiami be *Unity* logotipo. Papildomai suteikiama pamokos, skirtos žaidimo variklio mokymuisi, turima dvidešimt keturių valandų prieigą prašyti konsultanto pagalbos. Taip pat yra suteikiami papildomi apmokymai ir konsultacijos prie žaidimo reklamavimo ir skelbimo. Šis variantas UN yra mokamas, galima mokėti kas mėnesinį arba rinktis metinį mokestį [11].

Iš matomos informacijos galime pasakyti, kad UN siūlomas antras paketas yra visai neblogas. Tačiau prieš nuspręsdami dirbti su šituo varikliu, turi dar peržiūrėti likusius žaidimo variklius.

Unreal Engine 4 ir *5* siūlo pilną priėjimą prie savo C++ *source code*. Taip pat šie žaidimo varikliai suteikia priėjimą prie visų savo žaidimo paketų kūrimo bibliotekų, pagalbos įrankių be jokių papildomų mokesčių. Papildomai šie varikliai yra labai populiarūs, daug garsių ir populiarių įmonių dirba su šio tipo žaidimo varikliu. Dauguma iš tų įmonių yra sukūrusios žaidimus, kur yra naudojamas DI su *Narrow AI* funkcijomis. Šito žaidimo variklis pasižymi labai įdomiais įrankiais ir prietaisais, kurie padeda kūrėjams iš kelių skirtingų vietų. Nuo laiko sutaupymo iki aktyvios naudotojų grupės, kuri padeda surasti klaidą į iškilusias problemas. Dėl tos priežasties buvo nuspręsta, kad naudosime *Epic Game Launcher* siūloma žaidimo variklį UE 5.

1.8. Pasirinkto žaidimo variklio pagrindinės funkcijos

Nuspręsta naudotis *Epic Game Launcher* išleista *Unreal Engine 5* variklį. Pagrindinės priežastys, dėl kurios buvo pasirinkta naudotis šitą variklį buvo kelios. Viena iš priežasčių buvo tai, kad šis žaidimo variklis yra išleistas publikai. Šis variklis bent pagal tai, ką mes žinome, neturi jokių trūkstamų funkcijų arba įrankių, kuriuos būtu galima rasti UE4. Visos esamos funkcijos buvo atnaujintos ir pakeistos taip, kad jos taptų paprastesnės naudoti. Kad naujiems vartotojams būtų lengviau išmokti ir naudoti šituos įrankius. Taip pat buvo pridėta naujų dalykų, kurie gali būti naudingi ateityje, tolesniam žaidimo tobulinimui ir atnaujinimui. Naujos funkcijos, kurios buvo

sukurtos ir pridėtos, labiau yra naudojamos prie animacijos, specialiųjų efektų ir filmų kūrimo, tačiau šių ir senų funkcijų praplėtimai palengvins mums atliekamą darbą. Pagrindiniai įrankiai ir funkcijos, kurios mus sudomino buvo:

1) *Blueprint – Unreal Engine* funkcija, kuri veikia kaip žaidimo scenarijaus sistema, jis yra pagrįstas mazgų ir sąsajos naudojimo koncepcija, kuriant žaidimo elementus naudojančius *Unreal Editor*. Kaip ir daugelis įprastų scenarijų kalbų, šis dalykas yra naudojamas objekto orientuotoms klasėms arba objektams apibrėžti variklyje. Kai naudojate UE4 arba UE5, dažnai pastebėsite, kad objektai, apibrėžti naudojant *Blueprint*, šnekamojoje kalboje vadinami tiesiog *Blueprints.* Ši sistema yra itin lanksti ir galinga, nes suteikia galimybę dizaineriams praktiškai naudoti visas priemones ir koncepcijas. Dažniausiai prieinamas tik programuotojams, bet su *Unreal Engine* esančiu *Blueprint* specifiniu žymėjimu yra leidžiama programuotojams sukurti bazines sistemas, kurias dizaineriai gali išplėsti [17].
2) *Framework* – tai yra funkcija, kuri yra naudojama, kai skirtingos žaidimo sistemos dalys, susijungia. Paprastas to pavyzdys būtų, triušis lenktyniauja su sraige. Žaidimo rėžimas nustato žaidimo taisykles, pavyzdžiui, taisyklę, kad tas žaidėjas, kuris pirmas kirs finišo liniją, yra nugalėtojas. Žaidimo sistemos pagrindas yra žaidimo rėžimas (angl. *GameMode)*. Taip pat ši funkcija gali turėti savo judėjimo ir kitos žaidimo logikos taisykles, tačiau ši funkcija taip pat gali būti įtraukta į valdiklį. Valdiklis gali būti žaidėjo kontroleris (angl. *PlayerController*), DI (angl. *AIController)* arba priimantis įvestį iš žmogaus grotuvo, su automatiniu kompiuterio valdymu. Šiame pavyzdyje žaidėjas valdytų sraigę, todėl sraigės pėstininkas būtų valdomas žaidėjo kontroleriu (angl. *PlayerController*). Triušis būtų valdomas DI, o taisyklės, kada jis turėtų nusnūsti, sustoti ar sprukti, būtų nustatytos DI (angl. *AIController*), turinčiame triušio charakterį. Kameros teikiamas vaizdas rūpi tik žmonėms, todėl žaidėjo kamera naudotų tik vieną kameros komponentą, tai yra sraigės matomą vaizdą.
3) *Multiplayer mode* – žaidimo funkcija, kuri padeda, kūrėjau įdiegti kelių žaidėjų režimą savo žaidime. Aišku, tokius žaidimus, gali žaisti daugiau nei vienas žaidėjas, dėl to asmuo dažnai turi nustatytas taisykles, kurios kaip ir nustato žaidimo rėžimą. Paprasčiausiu lygiu šios taisyklės gali apimti nuo:
    - Dalyvaujančių žiūrovų ir žaidėjų skaičius, taip pat didžiausias leistinas žiūrovų ir žaidėjų skaičius.
    - Kai žaidėjai patenka į žaidimą, jie gali būti nukelti į kitas vietas.

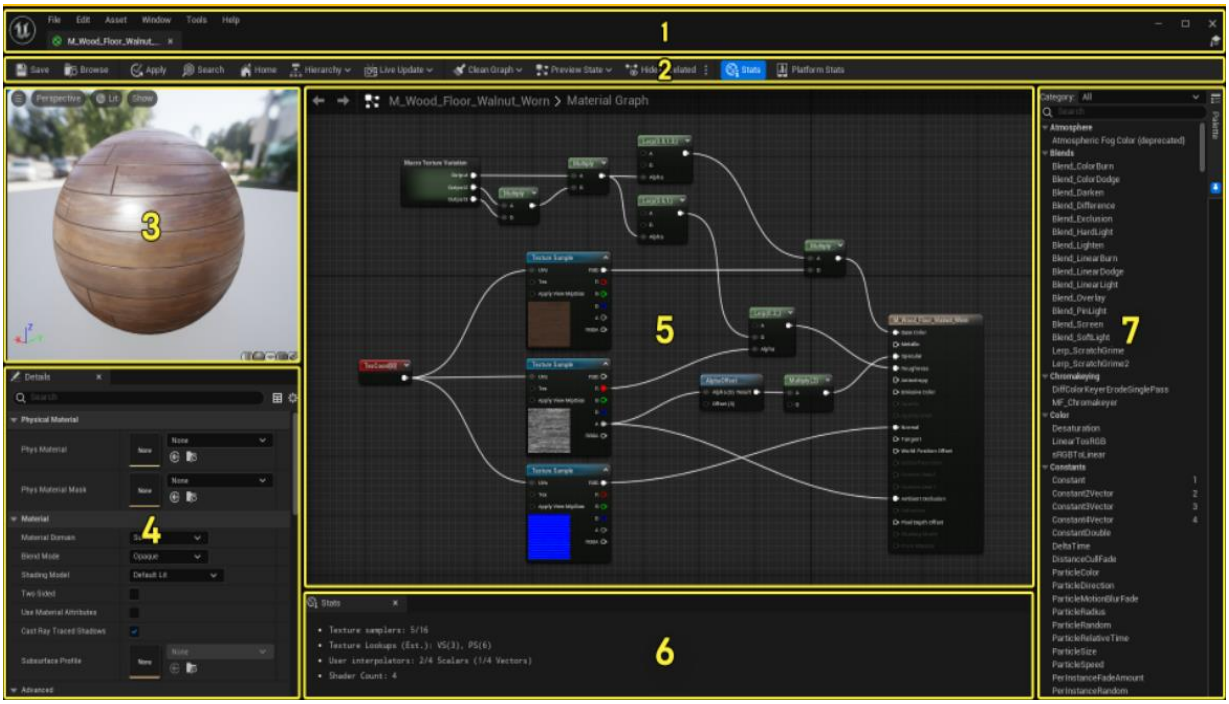
- Žaidimą galima pristabdyti, naudojantis tam tikras funkcijas, taip pat yra aprašyta kaip bus tvarkomas žaidimo pristabdymas.
- Perėjimai tarp lygių, įskaitant tai, ar žaidimas turėtų prasidėti kitu režimu.
- Žaidime įvyksta situacijos ir įvykiai, kuriuos reikia įrašyti ir išsaugoti. Turima informacija reikia stebėti bei dalytis su visais žaidėjais, ta informacija išsaugoma ir sinchronizuojama žaidimo būsenoje. Ši informacija apima:
    - Kiek laiko buvo vykdomas žaidimas (įskaitant laiką iki vietinio žaidėjo prisijungimo);
    - Kai kiekvienas žaidėjas prisijungė prie žaidimo ir savo žaidžiamo veikėjo;
    - Skirtingos režimo klasės, kaip žaidimo pabaiga ir pradžia.

4) *Film Quality Post-Process Effects – Unreal Engine* teikia vadinama *Post Process Effects*, kad menininkai ir dizaineriai galėtų pakoreguoti bendrą scenos vaizdą ir jausmą. Elementų ir efektų pavyzdžiai: žydėjimas (žydėjimo efektas ryškiems objektams HDR), ypač padeda prie atmosferos ir žemėlapiu kūrimo, už kuria būtu atsakingi pagrinde žemėlapių ir atmosferos kūrėjai arba programuotojai [4].
5) *VFX ir Particle Simulation* – įdomi nauja metodika, skirta realių laikų vaizdo efektų tvarkymui, tai yra filmuojant tiesioginį veiksmą. Ši technologija pagrįsta LED tiesioginio kameros stebėjimo, apšvietimo ir atvaizdavimo realiuoju laiku su ašies projekcija mišiniu, kad būtų sukurta sklandi pirmojo plano veikėjų ir virtualaus fono integracija. Šios funkcijos tikslas buvo žalio ekrano pašalinimas.
6) *Extensive Animation Toolset* – čia yra žaidimo variklio dalis, kuri yra naudojama kuriant naujus veikėjus, fizinius aspektus ir animacijas. Kaip kuriamo veikėjo kūno struktūra. Iš viso yra trys pagrindiniai įrankiai, kurie dirba vienas su kitu ir tie įrankiai būtų:
    - *Skeleton Editor* tai yra vieta, kur viskas prasideda, nes ji naudojama kaulų kūrimui ir valdymui.
    - Taip pat turime *Skeletal Mesh* ir animaciją. Skeleto tinklelio redaktorius yra naudojamas modifikuoti skeleto tinklelį, kuris yra susietas su skeletu. Animacijos rengimo langas leidžia kurti ir modifikuoti animacijos išteklius, joje koreguojami animacijos veiksmai.
    - Paskutinis įrankis būtų *Physics Asset Editor,* kuris yra naudojamas redaguojant ir kuriant fizinius korpusus, kurie vėliau bus naudojami skeleto tinklelio susidūrimui [2,16].

7) *Flexible Meterial Editor* – medžiagų ir objektų tvarkymo funkcija, kuri veikia kaip atskira meniu juosta, suteikianti žmonėms priėjimą prie objektų tvarkymo. Šis įrankis, kuris veikia kaip atskiras meniu laukas, turi penkis skirtingus regionus, visi šie skirtingi regionai yra atskirti į atskiras skirtingas dalis kaip:

- *1. Menių bar;*
- *2.Toolbar;*
- *3. Viewpoint;*
- *4.Detail panel;*
- *5.Material Graph panel;*
- *6.Stats panel;*
- *7. Pallette panel;*

Visus išvardytus regionus ir valdymo kontrolės vietas galima matyti pavaizduotas (***2* pav.**). [12] (Skeleton mesh editor)



Šaltinis: Unreal Engine tinklalapis. *Material Editor* UI. Žiūrėta 2023-11-03 per internetą < https://docs.unrealengine.com/5.0/en-US/unreal-engine-material-

2 *pav. „Flexible material editor“ laukas ir jos įrankiai*

1.9. Įmonių ir jų sukurti DI analizė

Viena iš geriausiai žinomų ir tvarių įmonių yra bendrovė *OpenAI*. Ši įmonė yra atsakinga už *ChatGPT* sukūrimą. Kiek yra žinoma, *ChatGPT* sukurtas remiantis gilaus mokymosi modelio

architektūra, žinoma kaip transformatorius, pats GPT yra pagrįstas ir suprogramuotas taip, kad naudotų neuroninių tinklų pagrindą. Pagrindiniai komponentai ir metodai, naudojami *ChatGPT* DI modelyje yra šie:

- Transformatoriaus architektūra: *ChatGPT* naudoja transformatoriaus architektūrą, kuri leidžia suprasti ir generuoti tekstą, pagrįstą dideliu duomenų kiekiu. Ši architektūra puikiai tvarko nuoseklius duomenis ir fiksuoja kontekstinius ryšius tarp žodžių ar žetonų.
- Išankstinis mokymas ir koregavimas: modelis iš pradžių yra apmokytas didžiuliu teksto korpusu iš įvairių šaltinių, naudojant neprižiūrimą mokymąsi. Šis parengiamojo mokymo etapas padeda dirbtiniam intelektui išmokti bendrųjų kalbos modelių, kontekstus ir santykius. Be to, tikslinant konkrečius modelius tokioms užduotims kaip pokalbio DI dar labiau patobulinamos modelio galimybės generuoti į žmones panašius atsakymus pokalbiuose.
- Dėmesio mechanizmas: transformatoriai naudoja dėmesio mechanizmus, leidžiančius modeliui sutelkti dėmesį į skirtingas įvesties teksto dalis generuojant atsakymus. Šis mechanizmas padeda užfiksuoti ilgalaikes priklausomybes ir kontekstą tekste.
- Didelio masto kalbos modeliavimas: *ChatGPT* DI modelis parengtas remiantis didžiuliu duomenų rinkiniu, leidžiančiu suprasti kalbos niuansus, sakinių struktūras ir semantines reikšmes, kad būtų generuojami nuoseklūs ir kontekstui tinkami atsakymai.
- Nuolatinis mokymasis: modelis gali būti nuolat tobulinamas periodiškai perkvalifikuojant naujus duomenis, leidžiant jam prisitaikyti prie besikeičiančių kalbos modelių ir geriau suprasti esamą kontekstą.

Galima pasakyti, kad *ChatGPT* naudoja giluminio mokymosi modelį, pagrįstą *transformer* architektūra, kuri iš anksto išmokytu naudoti didžiulius tekstinių duomenų kiekius ir tiksliai sureguliuotą konkrečioms užduotims. Kaip pokalbio sąveikai, šis metodas leidžia *ChatGPT* suprasti, apdoroti ir generuoti į žmones panašius atsakymus natūralios kalbos pokalbiuose.

Kita įmonę, kurią peržiūrėsime yra *Creative Assembly* ir jų sukurtą žaidimą *Alien: Isolation*. Įmonė buvo įkurta 1987 m. ir jos pagrindinė būstinė yra Jungtinėje Karalystėje. Creative Assembly įmonė geriausiai žinoma dėl savo strateginių ir veiksmo nuotykių žaidimų. Iš pradžių įmonė buvo sutelkusi dėmesį į strateginius žaidimus, tokius kaip *Total War* serija, tačiau laikui einant įmonė išplėtė savo portfelį įtraukdama papildomus skirtingus žanrus.

Vienas iš labiausiai žinomų dalykų apie įmonę yra, jos milžiniškas pasiekimas DI srityje su žaidimu *Alien: Isolation.* Pagrindinė žaidimo idėja yra tai, kad žaidėjo žaidžiamas veikėjas, tai yra Ellen Ripley įstrigo kosminiame laive, kuris yra apsėstas visokiausių ateivių ir monstrų. Ellen Ripley, tai yra žaidėjas naudodamasis turima aplinka ir įrankiais turi sekti numatyta istorija, kurios metu gali

pasitaikyti skirtingo tipo iššūkiai. Su šiais iššūkiais turi susidoroti žaidėjas, kai tuo pačiu metu yra persekiojamas ir medžiojamas ateivio, dar vadinamo *Xenomorph*.

Sukurtas DI, kuris yra atsakingas už *Xenomorph* valdymą, dar vadinamas kaip *Dynamic Alien AI* yra viena iš svarbiausių žaidimo savybių. Ši savybė yra pagrindinė dalis, kuri pritraukia žaidėją prie įtraukiančios ir intensyvios žaidimo patirties. Ateivio DI buvo sukurtas remiantis sudėtingu elgesio modelių ir algoritmų rinkiniu, skirtu imituoti tam tikrų grobuonių instinktus ir elgesį. Buvo įdėta labai daug pastangų ir minčių prie ateivio grobuoniškų instinktų ir gebėjimo prisitaikyti prie žaidėjo žaidimo stiliaus ir jo veiksmų. Todėl jis yra nuolatinis ir nenuspėjamas priešas, ateivis sugeba pakeisti savo elgesį. Visa tai vyksta tiesiogiai žaidimo metu ir žaidėjas gali matyti ateivio tobulėjimą ir mokymąsi. Visa šita dalis ir buvo labiausiai atsakinga už žaidėjui sukeltus įtampos ir baimės jausmą, nes *Xenomorph* judesiai ir elgesys atrodė organiški. Pagrindinės *Xenomorph* DI savybės:

- Aplinkos sąmoningumas: DI valdomas ateivis buvo aprūpintas sudėtingu aplinkos sąmoningumu. *Xenomorph* naudoja jutimo signalus, įskaitant garsą, vaizdą ir žaidimo aplinkos išdėstymą, kad galėtų sekti žaidėjo vietą ir atitinkamai reaguoti. *Xenomorph* aplinkos suvokimas leido jam dinamiškai ieškoti žaidėjo, tirti trikdžius ir naudoti įvairius maršrutus bei įėjimo taškus žaidimo aplinkoje, kad pagerintų medžioklės galimybes.
- Mokymasis ir prisitaikymas: Esminis DI aspektas buvo jo gebėjimas mokytis ir prisitaikyti. Buvo sukurta dvi smegenų dalys, tai būtų du skirtingi DI, kurie turėjo skirtingas užduotis. Viena smegenų dalis, dar žinoma kaip didžiosios smegenys, buvo atsakinga už žaidėjo stebėjimą, jo taktikos analizę ir reakciją į pavojus. Kol kita smegenų dalis, dar žinoma kaip mažosios smegenys, buvo atsakinga už žaidėjo buvimo vietos užuominų davimą, tai yra, jei žaidėjas pakartotinai naudotų tą pačią strategiją, slepiasi toje pačioje vietoje, naudoja tuos pačius įrankius arba vengimo būdus, tada mažosios smegenys duoda didžiajai smegenų daliai užuominą, kuri padeda *Xenomorph* mokytis ir tobulėti. Jo mokymosi galimybės užtikrino, kad susitikimai su *Xenomorph* kiekvieną kartą jaustųsi skirtingai, todėl žaidėjams reikėjo nuolat kurti naujoves ir prisitaikyti, kad išgyventų.
- AI reagavimas ir realizmas: Ateivį valdanti dirbtinio intelekto sistema buvo jautri, realiu laiku reaguodama į žaidėjo judesius ir veiksmus. Toks reagavimas padidino įtampos ir panardinimo jausmą, sukuriant intensyvią ir tikrovišką patirtį.
- Poveikis žaidėjo patirčiai: Labai detalus ir sudėtingas DI valdomas *Xenomorph* elgesys padarė didelę įtaką žaidėjo patirčiai. Tai padidino baimės faktorių, sukurdama jaudinančią ir nervus vejančią atmosferą viso žaidimo metu. Jo pažangus programavimas ir elgsenos modeliavimas siekė sukurti organišką, protingą ir prisitaikantį priešą, kuris buvo neatsiejama žaidimo,

siaubo patirties dalis. Šito tipo DI buvo vienas iš pirmųjų, kuris nustatė naują standartą ir įvairovę.

Taip pat reikia paminėti, kad įmonės *Creative Assembly* yra privati įmonė ir kaip visos privačios įmonės, juos yra labai atsargios su duomenų atskleidimų. Dėl tos priežasties dauguma dalykų kaip: komandos sudėtį, turimų darbuotojų skaičių, žaidimui skirtas lėšos ir kiti panašūs dalykai buvo neprieinami išoriniams asmenims.

Tačiau pasinaudodami informacija ir kitų žmonių nuomone ir perspektyvomis, galima bandyti nuspėti kelis išvardintus punktus. Toliau esama informacija buvo sudaryta naudojantis socialinių tinklalapių ir *Youtube* vaizdo įrašų pagalba.

Yra numanoma, kad kuriant „Alien: Isolation“ žaidimą, įmonė *Creative Assembly* skyrė nuo 430 iki 490 milijonų dolerių. Dalis tų lėšų, tai yra nuo 50 iki 100 milijonų dolerių, buvo skirta žaidimo reklamavimui. Papildomi keli milijonai buvo skirti žaidimo tvarkymui ir privalomiems atnaujinimams. Tačiau didžiausia lėšų dalis sudarė pats žaidimo kūrimas. Žaidimo kūrimo procesas užtruko nuo 3 iki 4 metų. Iš tų keturių metų, 6 mėn. buvo skirti komandos sudarymui, biudžeto apskaičiavimui ir komisijų peržiūrai. Papildomai dar apie 6 mėn. galima nuimti, žaidimo reklamavimui ir žaidimo testavimui, po kurio ėjo žaidimo koregavimas. Taip paties žaidimo kūrimas jau tebeužtruko apie 2-3 metus.

Toliau žaidimui sukurti reikia peržiūrėti, paties žaidimo kūrimo procesą. Žaidimą sudaro ne tik DI kūrimas, bet ir tokie dalykai kaip žemėlapio kūrimas, žaidimo istorijos rašymas ir tvarkymas, muzikos ir garsų kūrimas, 3D efektai, animacijos ir daug daugiau. Pašalinus visus papildomus pirminius dalykus galima nuspėti, kad paties DI kūrimas, diegimas, testavimas ir tvarkymas prasidėjo tik žaidimo kūrimui įsipusėjus. Tai pasinaudojus informacija turima iš kitų įmonių ir kitų žaidimo kompanijų galimą nuspėti, kad laikas skirtas DI daliai buvo apie 1 metus.

Komandą, sudarančių žmonių skaičius, kuri yra atsakinga už DI kūrimą nėra žinomas, tačiau iš kelių vaizdo įrašų iš YouTube ir socialinių tinklų, buvo galima spėti, kad komandą sudaro minimumas 20 darbuotojų. Čia ir užsibaigia mūsų rasta informacija apie *Creative Assembly* komandos struktūrą, darbuotojų skaičių ir biudžetą.

1.10. Analizės dalies išvada

Baigiamojo darbo analizėje buvo išanalizuoti keli moksliniai literatūros šaltiniai, kuriuose kalbama apie DI ir jo prasmę. Toliau išsiaiškinome, kas yra DI, peržiūrėjome žinomus DI tipus, kur jie yra naudojami, kaip jie naudojami ir kas reikalinga jų veikimui, tuo pačiu metu buvo sudaromos UML diagramos, kurios bus panaudotos mūsų DI kūrimui. Atlikus tai išsiaiškinome mums tinkamus dirbtinio intelekto tipus, tai yra *Narrow AI* ir *General AI*. Toliau susiradome žaidimo variklį UE 5, kuris, mūsų nuomone, atitiks visus mūsų reikalavimus. Pasirinkome gerai žinomą įmonę, kuri yra sukūrusi bent vieną populiarų kompiuterinį žaidimą, naudojant DI vieno arba kelių tipų funkcijas. Empiriniu būdu bandyta nustatyti žmogiškųjų resursų, reikalingų DI elementų kūrimui santykį su gauto produkto kokybe. Tam atlikti palyginsime mūsų sukurtą DI su įmonės sukurtu DI. Gauti rezultatai bus vėliau panaudoti projekto išvadų sudarymui.

SUDERINTA

Vadovas:_____

Magistrantas:_____

Data:_____

2. TECHINĖ UŽDUOTIS

1. DARBO PAVADINIMAS (TEMA):
   Dirbtinio intelekto elementų naudojimas kompiuteriniame žaidime.
2. ANALITINIO IR TIRIAMOJO DARBO TURINYS:
   2.1. DI tipų ir naudojimo analizė;
   2.2. Viešai prieinamos žaidimo kūrimo variklių analizė;
   2.3. Aprašyti pasirikto žaidimo variklio analizė;
   2.4. Įmonių ir tų įmonių sukurti DI analizė;
3. PROJEKTUOJAMOS SISTEMOS FUNKCIJOS:
   3.1. DI diagramų sukūrimas;
   3.2. DI kūrimo bandymai;
   3.3. DI pridėjimas prie automobilio;
   3.4. Atliekamas DI testavimas;
   3.5. Užsirašoma patirtos pastabos, klaidos ir iššūkiai;
   3.6. Atliekamą kodo modifikavimas ir testuojama iš naujo;
   3.7. Užsirašoma patirtos pastabos, klaidos ir iššūkiai;
4. SISTEMOS APRAŠYMO DOKUMENTAI IR INSTRUKCIJOS:
   4.1. DI instaliavimo ir paleidimo instrukcijos;
5. SISTEMOS PROJEKTAVIMO ĮRANKIAI, PROGRAMINĖS IR TECHINĖS ĮRANGOS REIKALAVIMAI:
   5.1. Unreal Engine 5
   5.2. Draw.io
6. SISTEMOS TESTAVIMAS IR ĮVERTIIMAS
   6.1. Baigto DI *Pre-Alpha* stadijos testavimas;
   6.2. Savo sukurtas DI bus palyginamas su pasirinktos įmonės DI, rezultatai bus surašomi į sudarytą lentelę;
7. DARBŲ PRISTATYMO REIKALAVIMAI:

7.1. Darbo gynimas nuo 8 iki 10 min. žodinis pranešimas ir pateiktas (pptf formatu). žmogiškųjų resursų, reikalingų DI elementų kūrimui, santykį su gauto produkto kokybe.

3. PROJEKTO DALIS

Magistrinio darbo temos pasirinkimas įvyko 2023-09-01 dieną. Tuo metu buvo žinoma tik tai, kad turime apytiksliai apie 4-5 mėnesius atlikti darbą. Taip pat buvo žinoma, kad bus du gynimai, tai yra katedrinis ir galutinis gynimas. Pradinis mėn. buvo skirtas duomenų ir informacijos rinkimui. Po to buvo skirtas laikas žaidimo variklio kodo mokymuisi ir kodo rašymo testavimai. 2 mėn. buvo skirti UML diagramų kūrimui, DI ir įmonių analizei, po kurios buvo pradėti pirminiai DI kūrimo testavimai. Likęs laikas buvo paskirtas magistrinio darbo rašymui. Taip pat kelios dienos buvo paliktos pasiruošti skaidres darbo pristatymui.

3.1. Projekto planas

Prieš pradedant kurti mašinėlę su DI funkcijomis, buvo poreikis pirmiausia išsiaiškinti, kas sudaro DI, kaip jis veikia, kokio tipo DI egzistuoja ir kuo jie vienas nuo kitų skiriasi. Taip pat buvo reikalinga peržiūrėti kelias įmones, kurios yra žinomos dėl gerai sukurto DI. Viena arba kelios įmonės bus aprašytos ir vėliau panaudotos DI rezultatų palyginimui. Atlikus tuos darbus, reikės pasirinkti kokį žaidimo variklį naudosime ir kokio tipo DI kursime. Taip pat reikėjo išanalizuoti, kokios funkcijos mums yra privalomos ir bus reikalingos kūrimo proceso metu. Dėl to buvo sudarytas projekto planas ir UML diagramos, kurio laikantis buvo kuriamas magistrinio darbo aprašas bei pats projektas.

Projekto pradžia: 2023-09-20;

Projekto pabaiga: 2023-12-31;

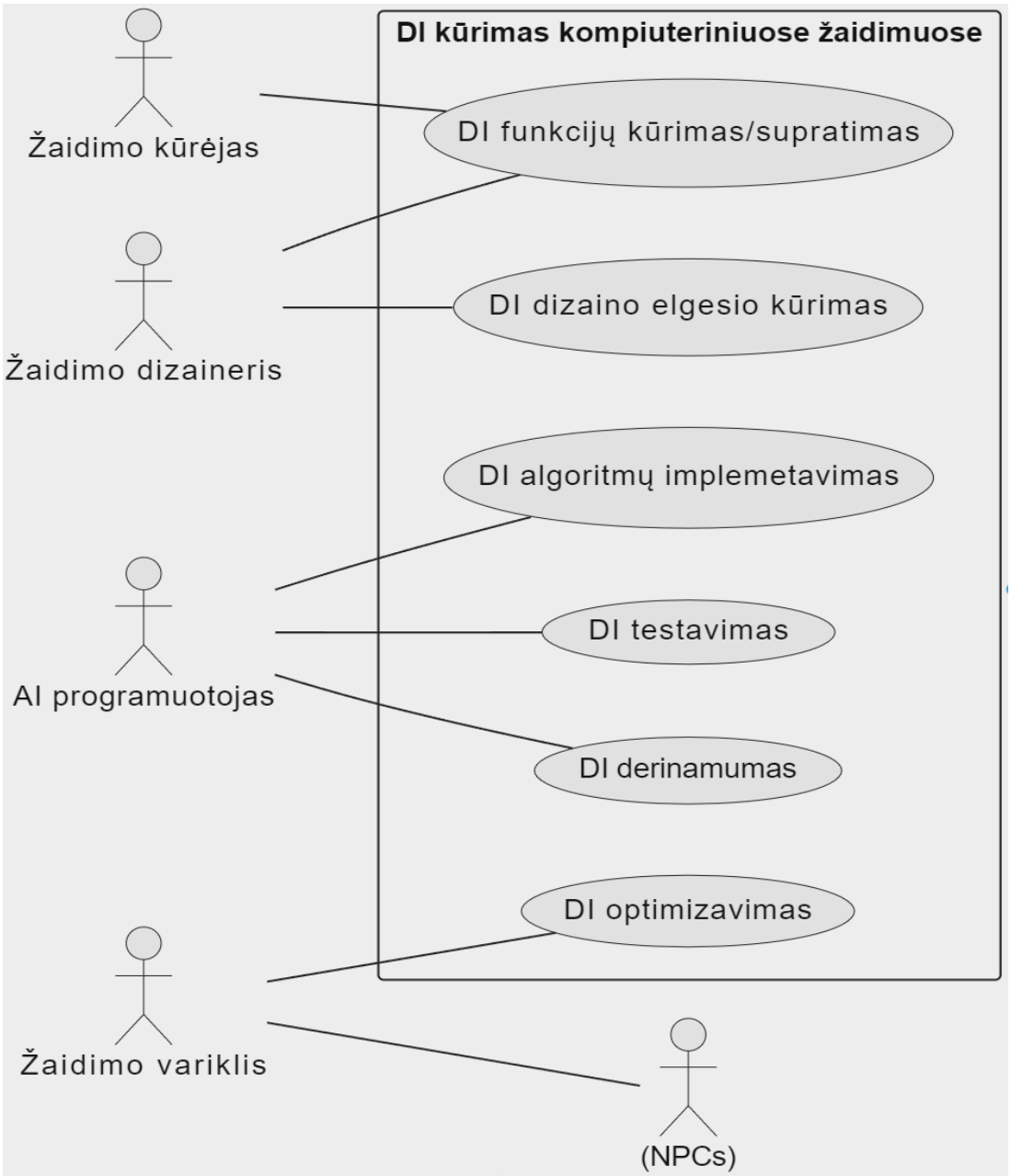
Projektas užtruko: 102 dienas;

3.2. Sistemos projektavimas

Sistemos projektavimo metu buvo sudaryta DI specifikacija, kuri pateikiama prieduose (1 PRIEDAS). Šios svetainės specifikacijos išgryninimas padėjo nenutolti nuo plano ir sukurtas DI atitiko daugumą reikalavimų.

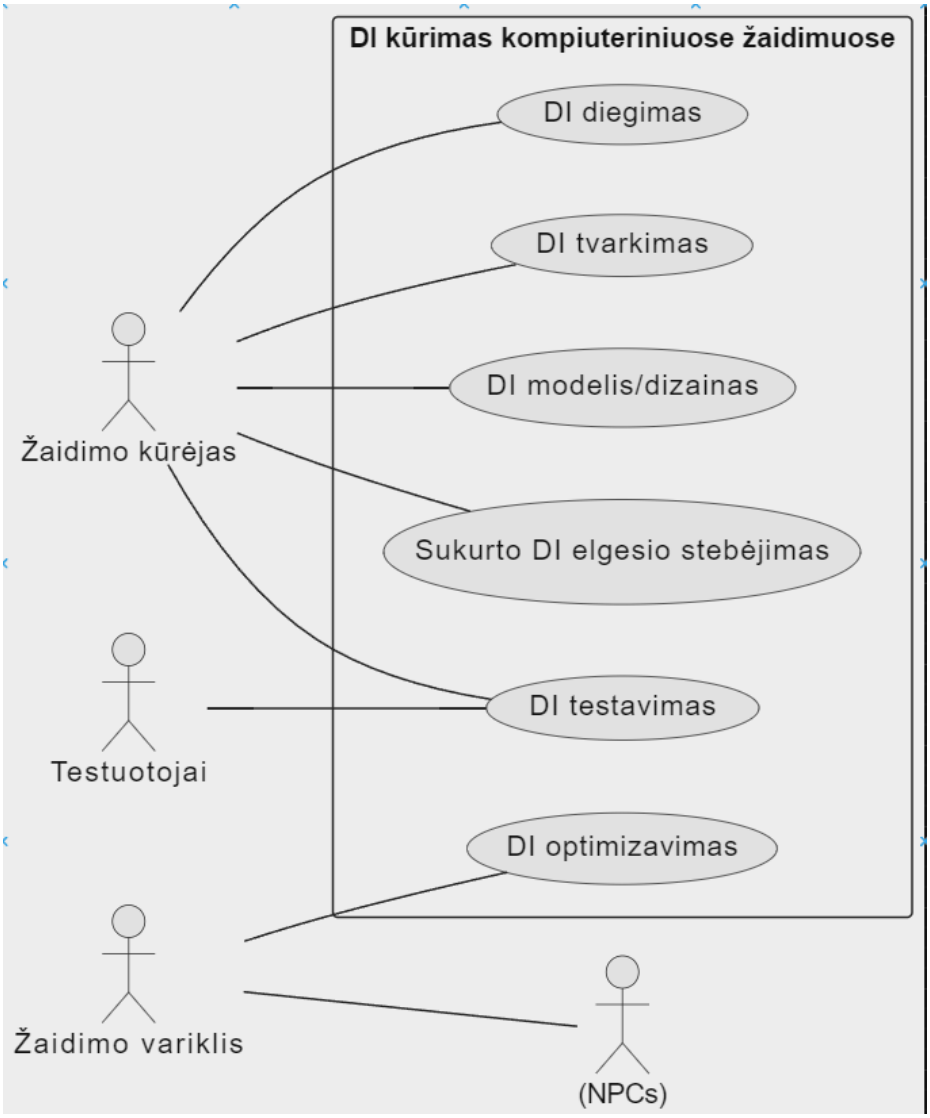
3.3. UML diagramų kūrimas

Prieš pradedant kurti DI, buvo sukuriama numanoma bendra „DI kūrimas kompiuterinius žaidimus“ *USE-CASE* diagrama, kurią galite matyti (**3 pav.**). Pagal sukurtą diagramą, galime matyti skirtingų tipų specialistus, kurie yra reikalingi DI kūrimui, diegimui ir testavimui. Visi šie skirtingi specialistai yra svarbūs DI diegimui, tačiau mūsų atveju iškyla problema tokia, kad DI bus kuriamas, diegiamas ir gal net testuojamas vieno asmens. Dėl tos priežasties mums reikia koreguoti mūsų turimą



3 pav. DI kūrimo use-case diagrama

diagramą į tokią, kuri labiau atitiktų mūsų atliekamą darbą. Naujai perdaryta *use-case* diagramą, galime matyti (**4 pav.**)

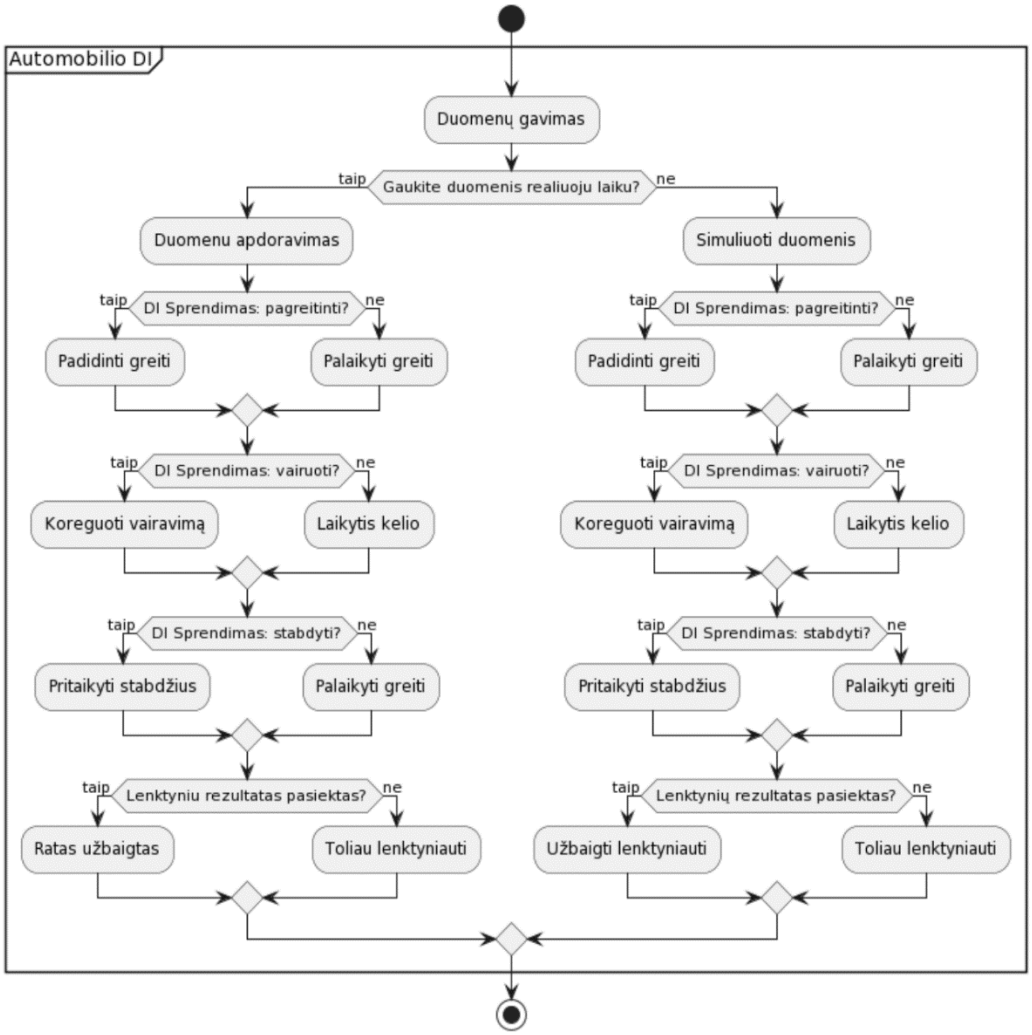


4 pav. Perdaryta DI kūrimo use-case diagrama

Pagal perdarytą diagramą matome jau vieną iš iššūkių, su kuriuo teks susidurti. Tai bus visų skirtingų specialistų darbų atlikimas vienam asmeniui. Įmonė kaip *Creative Assembly* turi didelius lėšų išteklius, leidžiančius nusamdyti skirtingų tipų specialistus ir sugrupuoti juos į skirtingas komandų dalis.

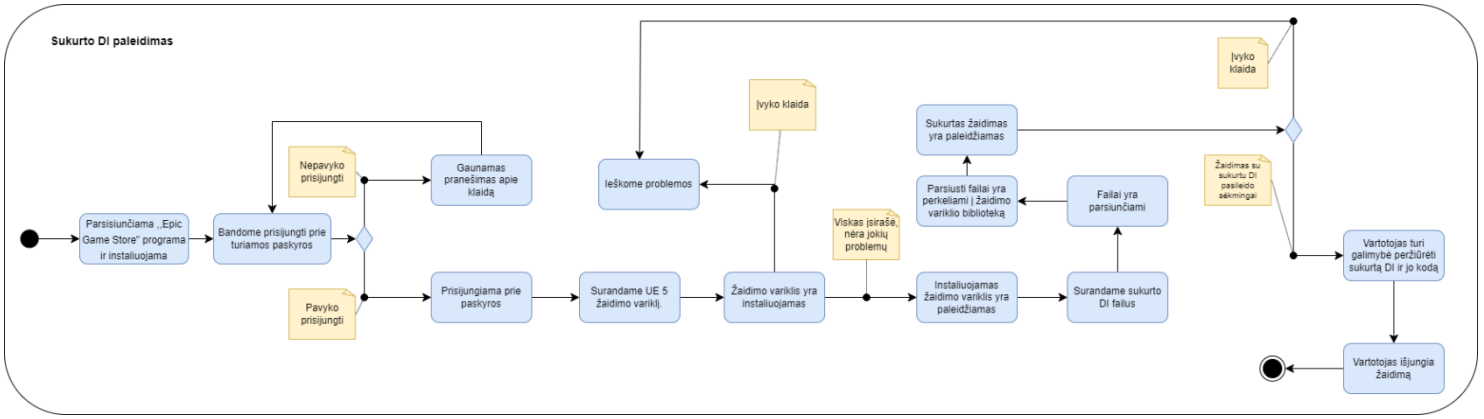
Toliau buvo sudaryta lenktyninio automobilio DI veiklos diagrama (**5 pav.**). Diagrama buvo sudaryta su tikslu apibrėžti automobilyje esančio DI veiksmus. Taip pat diagrama buvo sudaroma su ta idėja, kad kuriamas dirbtinis intelektas bus paremtas naudojantis *Narrow AI* tipo funkcijomis. Dėl

šios priežasties pagrindinės ir svarbiausios sukurtos funkcijos bus sudarytos paties programuotojo, o vėliau atsižvelgus į turimą likusį laiką, būtų galimą bandyti pridėti ir *General AI* tipo funkcijas.



5 pav. Veiklos diagrama - DI sprendimo priėmimas

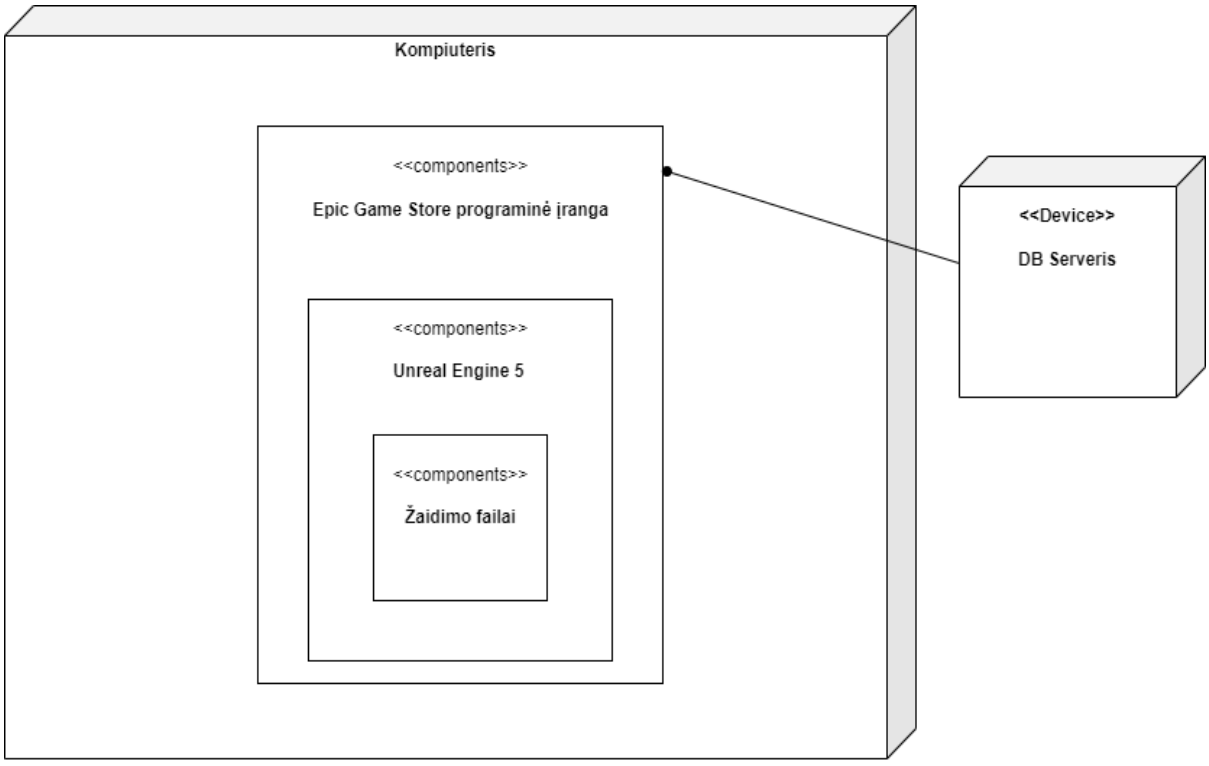
Buvo sukurta veiklos diagrama, (**6 pav.**) nurodanti vartotojui DI paleidimo instrukcijas. Ši diagrama atvaizduoja žingsnius, kurių asmuo turės imtis norėdamas išbandyti sukurtą DI.



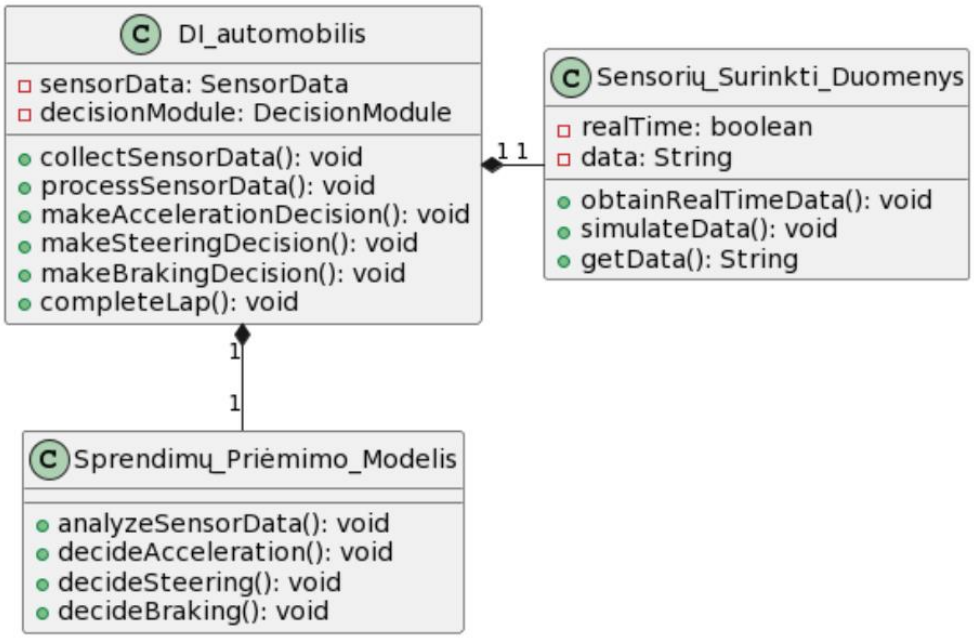
6 pav. Veiklos diagrama - DI paleidimo instrukcijos

Kadangi tuo metu buvo dirbama su UML diagramomis, buvo stengiamasi sukurti viską, kas gali būti naudinga arba panaudota vėlesniam laikui. Dėl šios priežasties buvo sukurtos dar papildomos 3 diagramos. Dvi iš sukurtų diagramų yra klasių diagramos pavaizduojančios DI klasės galimus duomenis ir renkamą informaciją, žiūrėti (**9 pav.** ir **8 pav.**). Paskutinė diagrama, kurią galite matyti (**7 pav.**), buvo sukurta tik vienai užduočiai parodyti naudojamo žaidimo variklio serverio priėjimo svarbą. Tai yra, jei vartotojas nori išbandyti sukurtą DI, jis privalo prisijungti prie EGL programos. Tik po to, kai prisijungimas bus sėkmingas, vartotojas gaus priėjimą prie UE 5 žaidimo variklio. Neatlikus prieš tai išvardintų veiksmų, nebus galima išbandyti sukurto DI. Sukūrėme kelias papildomas diagramas, kurios turėjo galimybę būti vėliau panaudotos DI kūrimui.

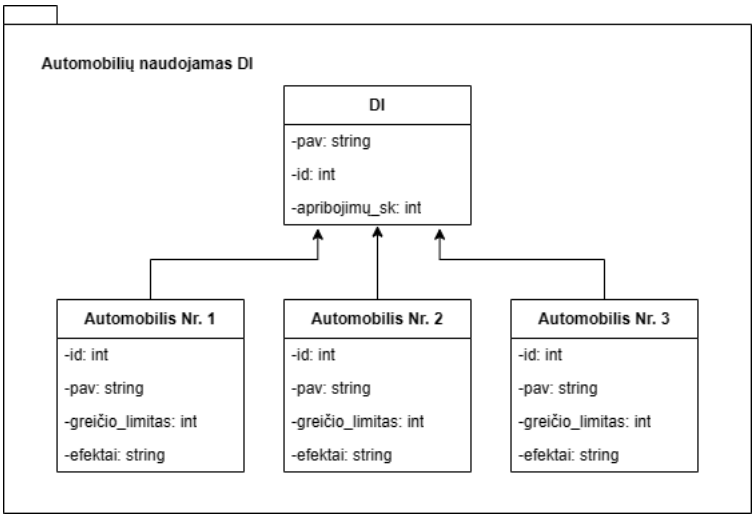
Vadovausimės sukurtomis diagramomis, kuriant DI. Aišku, visos sukurtos funkcijos ir priedai bus modifikuojami ir tvarkomi, tačiau bus stengiamasi nenukrypti nuo pagrindinės struktūros. Kuriant DI sukurta specifikacija užtikrins, jog bus laikomasi nustatytų nurodymų ir visi reikalavimai nenukryps per toli nuo plano.



7 pav. Žaidimo variklio DB interaktyvavimas su mūsų žaidimo failais



9 pav. DI klasės diagrama (2 modelis)



8 pav. DI klasės diagrama (1 modelis)

4. PROGRAMINĖS REALIZACIJOS DALIS

Šiame skyriuje bus aptariama, kokios programos, funkcijos ir įrankiai buvo naudoti atliekant projekto realizaciją. Kaip buvo sukurti iššūkiai ir kokios funkcijos buvo naudotos jų realizacijai. Šitoje dalyke aprašysime, kokius pluginus mums prireikė naudoti ir kokia buvo jų paskirtis. Taip pat aprašysime, kokius objektus ir dizainus panaudojome iš publikai prieinamos bibliotekos, testavimas ir galiausiai, kokią techninę įrangą naudojome atliekant šį darbą.

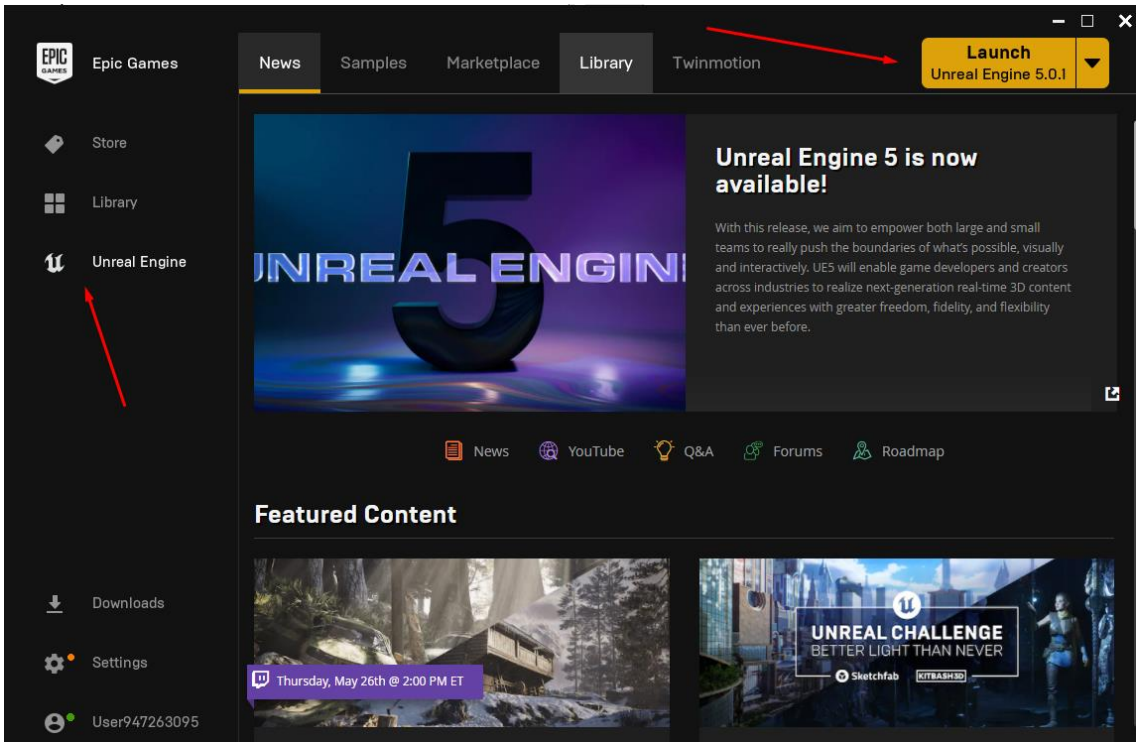
4.1. Programinė ir techninė įranga

Projekto realizacijai buvo pasirinktas naudoti stacionarus kompiuteris, kurį surinkau pats. Kompiuteris turi procesorių i9-13900K, vaizdo plokštę *MSI Nvidia RTX Geforce 4900*, 64 GB DDR5 RAM (4x16 GB), 2T SSD. Taip pat yra naudojama *Windows* 64 bitų operacinė sistema. Šis kompiuteris yra galingas, tačiau, kuriamam DI šitokios galios šiuo metu nereikės. Ateityje, jei DI bus toliau tobulinamas ir plečiamas, gali prireikti tokio galingo arba net galingesnio kompiuterio. Tačiau šis kompiuteris yra mano pagrindinis ir buvo nuspręsta, kad ieškoti kitokio neapsimoka ir nėra laiko.

Dėl to nusprendžiau, kad įdiegsiu visas reikalingas programas, žaidimo variklius ir pluginus į šį kompiuterį. Visi toliau išvardinti įrankiai bus reikalingi, kuriant kompiuterinį žaidimą ir jo testavimui:

- *Epic Game Launcher*
- *Unreal Engine 5*

Taip pat yra galimybė, kad gali būti reikalingas vienas pluginas, nes kadangi žaidimo variklis dar yra naujas, būna atvejų, kad iškyla neaiškių problemų ar trūkumų, kurie dažniausiai būna dėl SDK nebuvimo arba senos versijos. Taip pat yra galimybė, kad net atnaujintus arba iš naujo įkėlus SDK pluginą, gali sukurtas DI paleidimas neveikti. Tačiau testavimo metu buvo rasta, kad tai dažniausiai atsitinka asmenims, kurių kompiuteriai būna senesni nei 2-4 metų. Dėl to, jei susidursite su tokia situacija, patariama susirasti kitą kompiuterį ir bandyti iš naujo arba laukti, kol *Unreal Engine 5* išleis problemos sprendimo būdą.



10 pav. Epic Game Launcher ir Unreal Engine

Kad būtų galima šiuo metu žaisti arba ištestuoti DI, yra privaloma susikurti paskyrą naudojant *Epic Game Launcher* arba jų tinklalapyje, tada reikia prisijungti ir parsisiųsti *Unreal Engine 5* (**10 pav.**). Kai *Unreal Engine 5* yra instaliuotas, paleiskite žaidimo variklį ir galima pradėti dirbti.

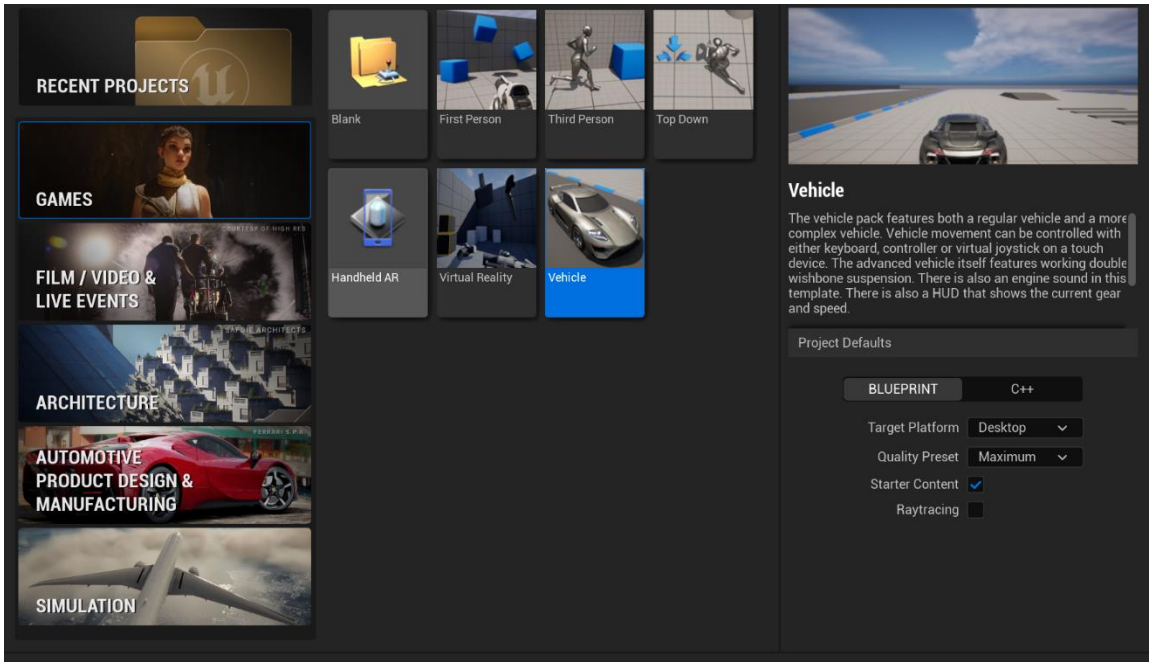
4.2. Projekto pasirinkimas

Visų pirma reikėjo tiksliai peržiūrėti UE 5 siūlomas paketų bibliotekas. Iš viso buvo galimi pasirinkti iš 7 skirtingų bibliotekų - projektų variantų (**13 pav.**). Tų projektų variantai buvo išskaidyti į 7 dalis:

- VR – sukuriamas projektas su jau iš anksto sukurtais ir įdiegtais virtualios realybės įrankiais.
- FP – sukuriamas projektas su papildomais įrankiais ir modeliais, kurie yra skirti padėti kurti žaidimus su pirmo asmens perspektyva.
- Top Down – sukuriamas projektas, skirtas antros perspektyvos žaidimų kūrimams.
- Handhel AR – sukuriamas projektas, skirtas telefoninių žaidimų kūrimams.
- Vehicle – sukuriamas projektas, turintis iš anksto sukurtą lenktyninę trasą ir mašinėlė, kurią vartotojas gali valdyti.

- Blank – atidaromas tuščio tipo projektas, kur nėra sukurta jokių papildomų įrankių ir funkcijų. Šio tipo projektas yra naudojamas didelių projektų kūrimo metu arba testavimams.

Kadangi mūsų kuriamas DI bus mašinėlė su auto piloto funkcija, mes pasirinkome *Vehicle* tipo projektą. Pasirinkus mums tinkamą projekto variantą, pirmas dalykas, ką turime atlikti yra visų funkcijų ir turimų įrankių išsitestavimas. Pirmiausia išbandėm sukurtą automobilį (**12 pav.**) ir paskui peržiūrėjome sukurtą trasą (**11 pav.**).



13 pav. UE 5 projektų variantai

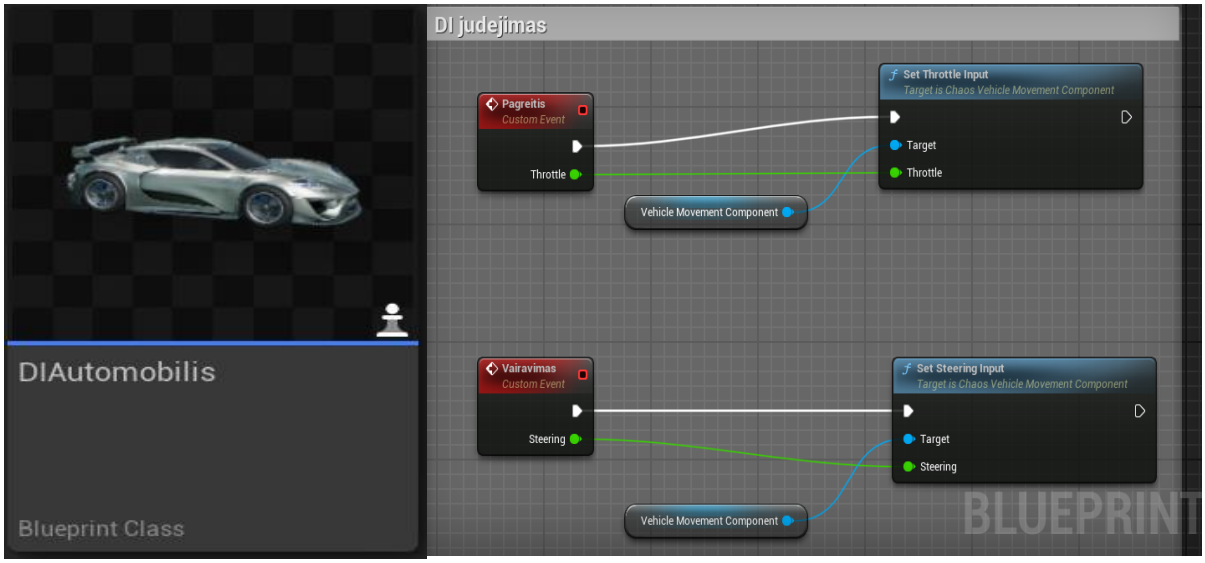
12 pav. UE 5 sukurtas automobilio modelis

11 pav. UE 5 sukurta lenktyninė trasa

4.3. DI kūrimas

Peržiūrėjus sukurtą žemėlapį, ištestavus sukurtus objektų modelius ir išbandžius visus turimus įrankius, buvo pradėti pirmieji DI kūrimo eksperimento bandymai. Vienas iš eksperimentų buvo turimo automobilio modelio, dublerio kūrimas. Sukurtas dublikatas bus panaudotas kaip modelis, kur bus diegiamas sukurtas DI (**14 pav.**). Prie to paties sukūrėme atskirą plano pradžią, kur bus rašomas,

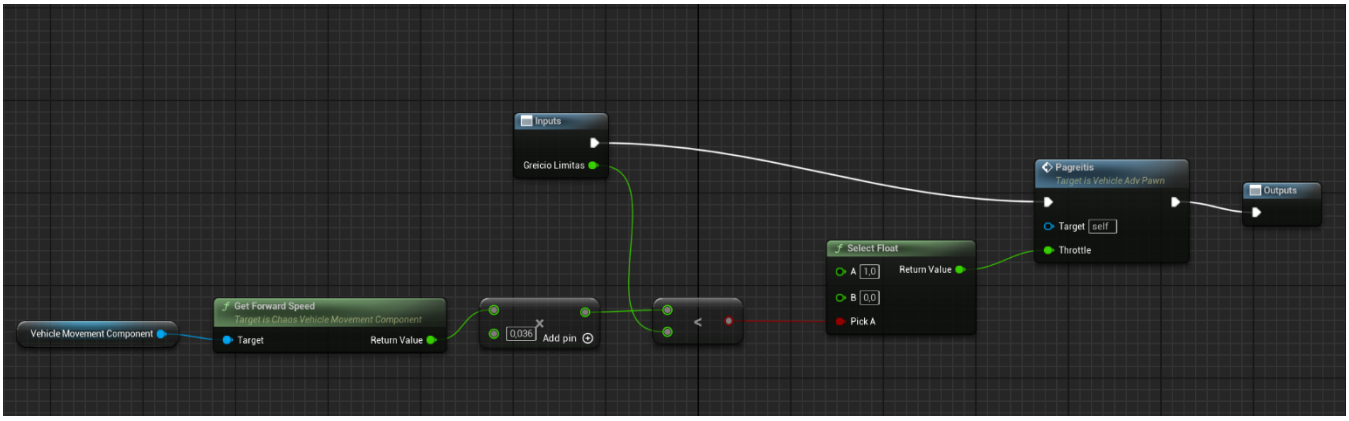
modifikuojamas ir kuriamas naujas *blueprint*. Viskas yra ištestuojama ir peržiūrima taip, kad būtų įsitikinta, kad visos senos *blueprint* jungtys ir funkcijos yra ištrintos. Toliau bandėme sukurti funkcijas, kuriuos atitiktų *Narrow AI* tipą. Tai būtų greičio valdymas, vairo valdymo kontrolė, kelio kontrolės laikymosi funkcijos, gebėjimas atpažinti važiuojamo paviršiaus kelią bei bausmės priskyrimas dėl nurodymų nesilaikymo.



14 pav. Automobilio modelio duplikantas

4.4. DI greičio valdymas

Pirmiausia funkcija, kurią bandėm sukurti, mūsų DI yra greičio valdymas ir reguliavimas. Pirmas sukurtas modelis (**14 pav.**) buvo labai paprastas ir tiesiai prie tikslo. Sukurtas *blueprint* leido automobiliui važiuoti į priekį. Tačiau tai buvo vienintelė automobilio funkcija, šiuo metu automobilis neturėjo jokių funkcijų ar sensorių, leidžiančių jam žinoti apie pasitaikančias kliūtis. Dėl tos priežasties teko daryti papildomus keitimus ir tvarkymus. Buvo stengiamasi sukurti naują *blueprint*, kuris būtu padarytas taip, kad vėliau nereikėtų grįžti ir daug tvarkyti. Dėl šios priežasties buvo sukurti



15 pav. Greičio valdymo blueprint

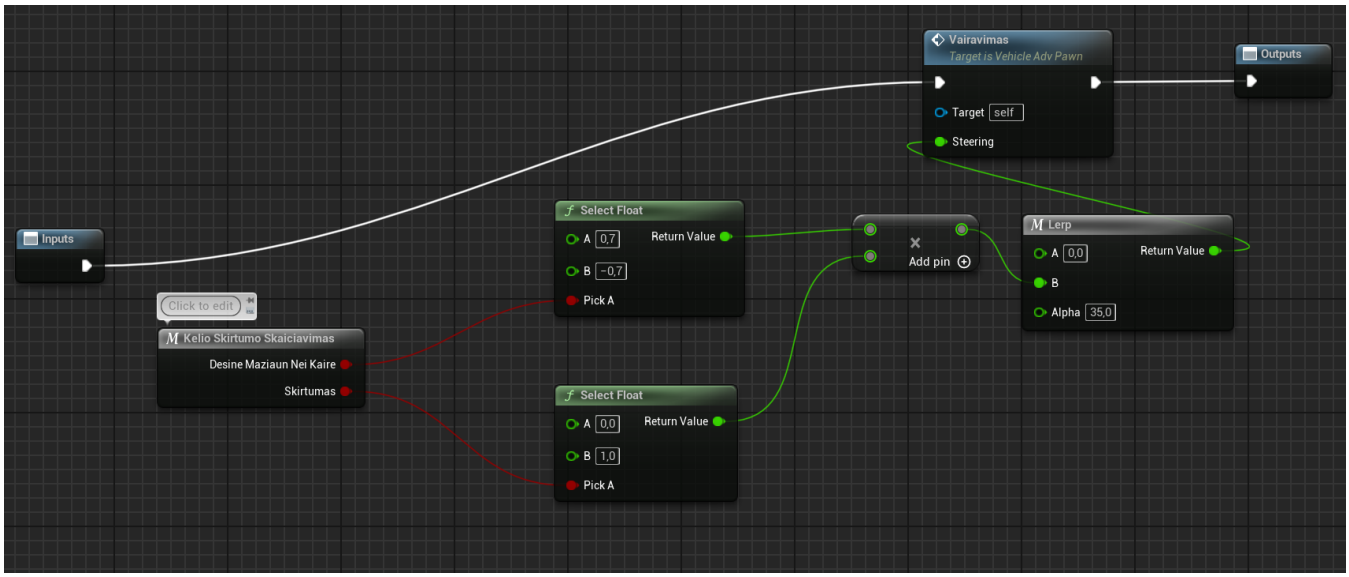
ir įdiegti tokie dalykai kaip greičio limitai, pagreičio valdymas ir net mašinėlės pozicijos greičio matavimas, naujai sudarytą *blueprint* galima matyti (**15 pav.**).

4.5. DI vairavimas

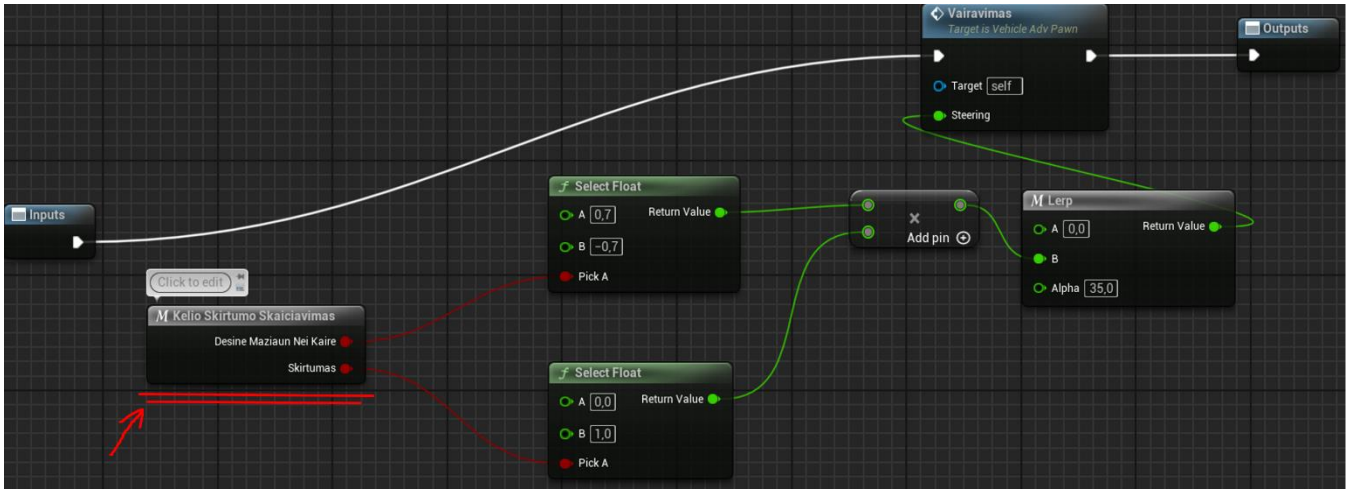
Toliau buvo dirbama prie vairavimo kontrolės *blueprint* modelio. Pirmiausia sukūrėme jau žinomus reikalingus kontroles punktus, tai būtų tokie dalykai kaip: duomenų rinkimas apie ratų sukimosi koeficientą, kelio skirtumo skaičiavimas (**22 pav.**), *Spline* apskaičiavimas ir vairavimo kontrolės priskyrimas objektui, kuris mūsų atveju būtų UE 5 sukurtam automobiliui. Daugumą iš šių kontrolės funkcijų bazės buvo galima sudėti į vieną bendrą *blueprint* (**17 pav.**), tačiau tų bazių išsišakojimai bus atskirti į savo atskirus *blueprint* modelių šakas.

Vienas iš iššūkių buvo tai, kad nepavyko atlikti efektyvaus priekinių ratų sukimosi koeficientų apskaičiavimo. Iškilo tokia problema, kad norint apskaičiuoti ratų sukimosi koeficientą, atsirasdavo vienos arbo kitos formos paklaida. Dėl atsiradusių paklaidų iškildavo skaičiavimo klaidos, kurios nulemdavo automobilio netaisyklingą vairavimą. Vienas iš galimų pavyzdžių būtų tai, kad jei automobilis važiuoja daugiau nei 50 kilometrų per valandą greičiu, ant sekančio daromo posūkio dėl didelio greičio vienas arba daugiau ratų pakiltų nuo kelio. Dėl tokių klaidų atsirasdavo paklaida, dėl kurios vairavimo kontrolės sistema pradėdavo daryti nesąmones. Tokiai paklaidai sutvarkyti buvo išbandomi keli skirtingi dalykai. Dauguma pradinių išbandytų dalykų neturėjo teigiamų rezultatų, tačiau vienas iš jų padėjo išspręsti mums turimą problemą. Tai buvo atskiras kairės ir dešinės pusės padangų posūkio koeficientų skaičiavimas. Skaičiuojant priekines padangas kaip 2 skirtingus objektus, mes turėjome 2 skirtingų tipo duomenis (**16 pav.**). Gautų padangų duomenys su gaunama automobilio pozicijos vieta leido mums sukurti gan sudėtingą vairavimo kontrolę. Mūsų vairavimo kontrolė leidžia mūsų DI atlikti tikslius posūkius, kuriuos mes galime nustatymuose lengvai kontroliuoti. Šios lengvos kontrolės pagalba vėliau būtų galima sukurti skirtingų tipo DI valdomus

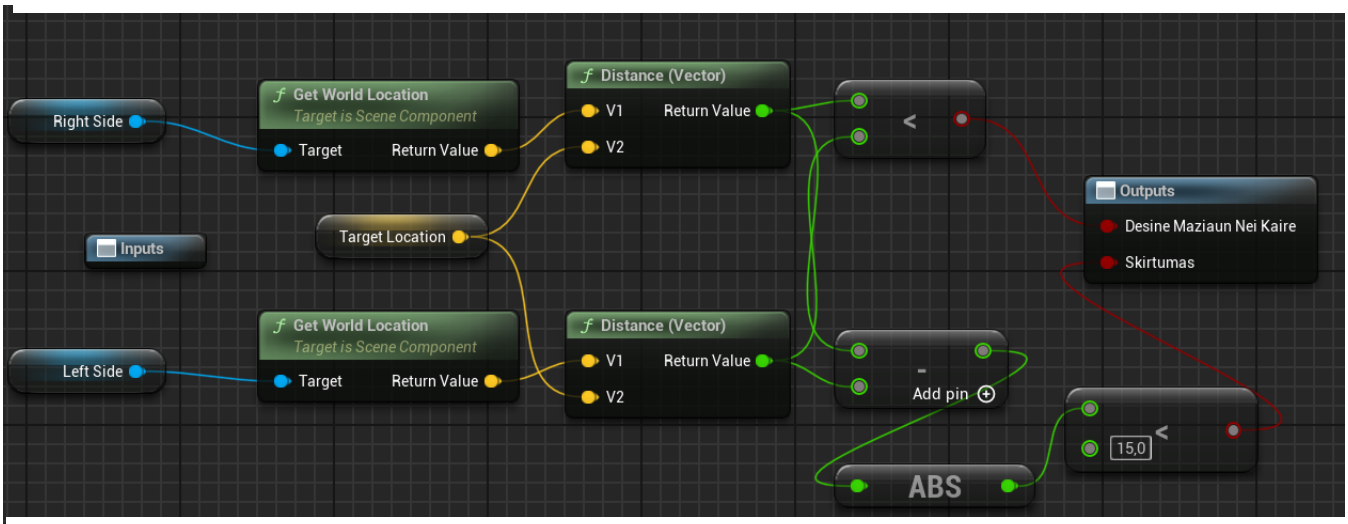
lenktyninius automobilius su skirtingo tipo sudėtingumais. Automobilis, kuris gali daryti didesnio tipo posūkius, turėtų galimybę apvažiuoti tą pačią trasą greičiau.



18 pav. Vairavimas ant sukurto spline



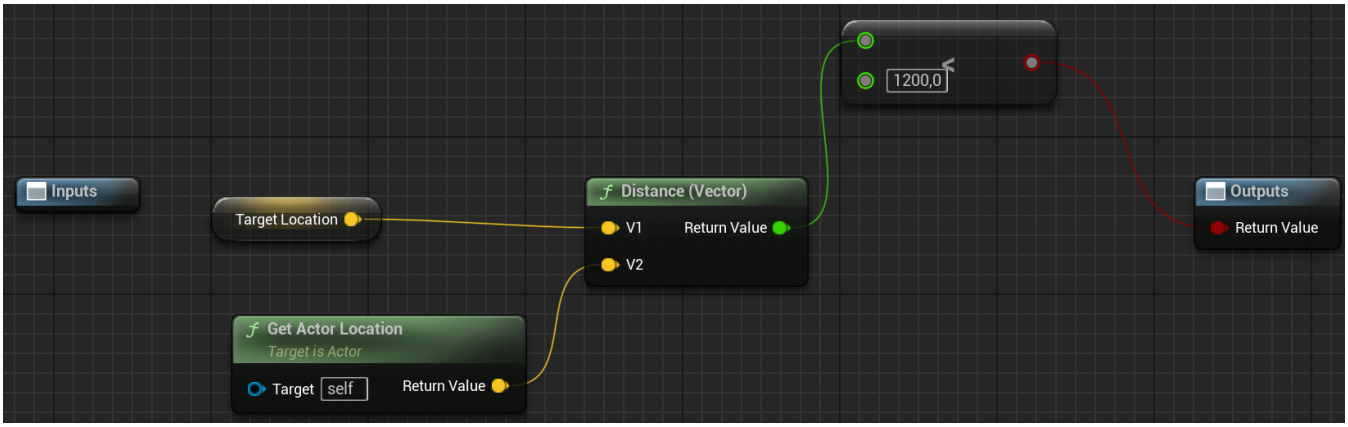
17 pav. Blueprintas atsakingas už sukurto DI vairavimo kontrolę



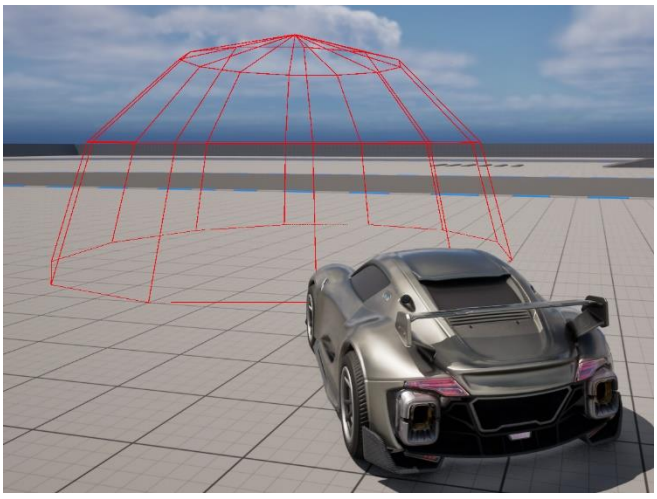
16 pav. Trasos ir kelio skirtumo skaičiavimas

4.6. DI kelių ribojimų, taisyklių sukūrimas

Kita užduotis yra sukurti kelias funkcijas, kurios mūsų sukurtam DI būtų perspėjamasis ženklas. Tai yra, kad turime sukurti dalykus, kurie paskatintų mūsų sukurtą DI neišvažiuoti iš trasos kelio ribų. Taip pat prie to dar reikia padaryti taip, kad mūsų sukurtas *Spine* neveiktų kaip viena ištisa linija. Reikia jai priskirti apribojimą (**20 pav.**), kurio pagalba turimas sukurtas *Spine* būtų padalintas į trumpesnes dalis. Padaryti tai atliksime kelis dalykus, vienas iš jų bus resursų naudojimo sumažinimas, o kitas yra tai, kad Spine padalinimas į mažesnes dalis, leis mūsų sukurtiems DI mašinėlėms turėti lengviau pasiekiamą tikslo tašką. Vietoj to, kad važiuotų aplink visą trasos kelią, pasiektų finišo liniją be galimybės pasitikrinti nuvažiuotą atstumą ir rezultatą. Bus sukurti keli saugūs taškų punktai, kurių pasiekiamumas informuotų DI apie teisingai atliktus veiksmus. Šie saugos taškai leis DI žinoti, kad jo atliekami veiksmai yra teisingi ir toliau gali lenktyniauti. Turėdami visas šias idėjas, mintis, sukūrėme atskirą *blueprint*, žiūrėti (**18 pav.).**



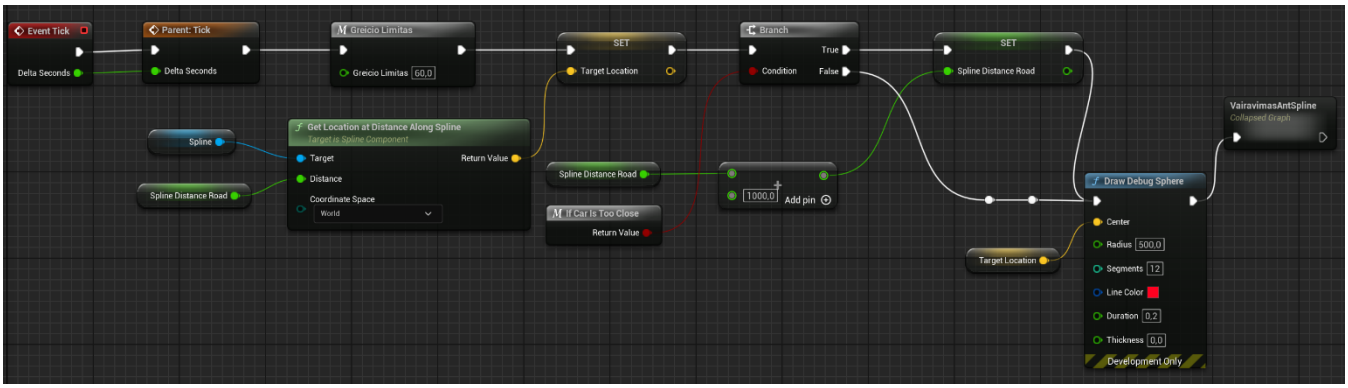
19 pav. Sukurto DI automobilio vietos duomenų rinkimas ir perdavimas



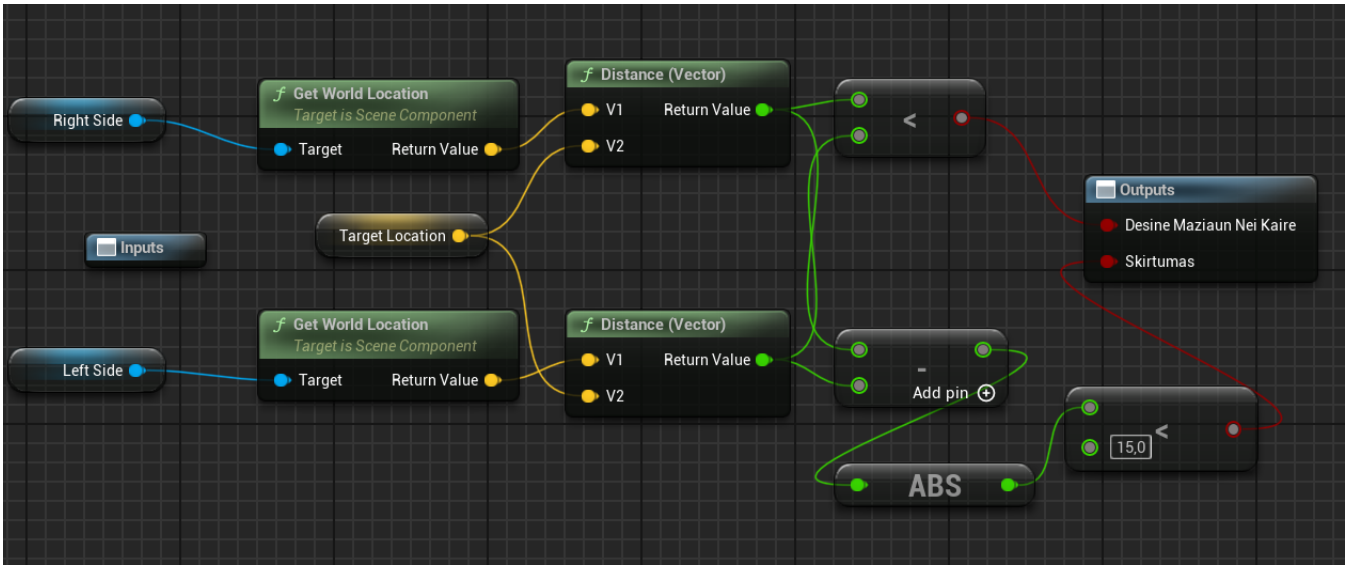
20 pav. Sukurtas taškas, kurį DI turi pasiekti

4.7. Sukurto DI pagrindinės smegenys

Visus sukurtus atskirus *blueprint* ir jų gaunamus duomenis, reikėjo sujungti į vieną tašką žinomą, kaip centrinė smegenų dalis, žiūrėti (**21 pav.**). Šį tašką mes pavadinome smegenų centrine dalimi dėl to, kad čia gavosi pagrindinė vieta, kur buvo sudarytos pagrindinės ir svarbiausios jungtys. Tai yra jungtys, atsakingos už duomenų perdavimą ir siuntimą. Čia iš dalies būtu vos ne tas pats kaip ***General AI*** tipo neurono tinklai, kurie yra atsakingi už gautų duomenų apdorojimą, rezultatų išvedimą ir tų rezultatų išvedimą į atliekamus veiksmus.



21 pav. Sukurto DI pagrindinė bazinė kontrolė



22 pav. Trasos ir kelio skirtumo skaičiavimas

4.8. Sukurtos lenktyninės mašinėlės testavimas

Po visų atliktų funkcijų kūrimo ir galutinių smegenų dalies sujungimo, liko paskutinis svarbiausias darbas. Tai yra sukurtos mašinėlės DI testavimas, reikia įsitikinti, kad visos turimos funkcijos veikia taip, kaip norime, ar nėra netikėtų klaidų, kirmėlių ir lūžių. Testavimo metu patirtos

klaidos ir problemos yra fiksuojamos sukurtoje lentelėje. Visos sukurtos lentelės ir jų atsakymai buvo pridėti prie Testavimo skyriaus, kuriuos galima rasti prie priedų. Jei norite peržiūrėti toliau nagrinėjamas sukurtas lenteles ir tyrimo rezultatus reikia peržiūrėti 2 PRIEDAS (***Lentelė* 1, Sukurtos mašinėlės DI testavimas**).

Pirmas testavimas buvo atliktas rankiniu būdu, tai yra kai pagrindinis asmuo atsakingas už DI kūrimą atlieka visus numatytus DI funkcijų testavimus. Pirmiausia tokiam testavimui atlikti reikėjo sudaryti lentelę, kurioje būtų 3 punktai. Pirmas punktas yra klausimų eilės numeris, antras punktas yra vieta kur užrašoma mums svarbūs testuojamo DI klausimai ir trečias punktas yra gautų rezultatų atsakymas. Sekanti užduotis buvo scenarijų klausimų sukūrimas ir trečias etapo žingsnis buvo DI testavimas ir gautų rezultatų užrašymas sukurtoje lentelėje.

Kuriant testavimo klausimus, buvo stengiamasi padaryti taip, kad gaunamas rezultatas būtų skirstomas į 2 tipų atsakymus, tai būtų teigiamas ir neigiamas. Teigimas atsakymas reiškia tai, kad užduotas klausimas, kaip „Automobilis su DI, be išorinių asmenų pagalbos, sugeba pajudėti iš vietos.“ Veikia taip kaip yra numatyta programuotojo. Neigiamas atsakymas reiškia tai, kad testuojamas DI nesugeba atlikti tai ko iš jo yra tikimasi arba norima. Taip pat turime ir nenumatytą trečią rezultatą, kuris buvo priskirtas prie nenuspėjamų. Tai yra rezultatas, kuris dėl nežinomos priežasties pastoviai keičiasi ir iš atliktų šimtų skirtingų testų, gali būti penkios, dešimt arba nors ir visos šimtas situacijų, kai testuojamas dalykas gali veikti ir neveikit. Testuojamas DI funkcijos dalis neturėjo pastovaus nesikeičiančio galutinio rezultato. Dėl tos priežasties galutiniai rezultatų atsakymai buvo suskirstyti į trejus skirtingus tipus: teigiamas, neigiamas ir nenuspėjamas.

Sudarytą lentelę su klausimais ir gautais testavimo rezultatais galima peržiūrėti 2 PRIEDAS (***Lentelė* 1, Sukurtos mašinėlės DI testavimas**).

Antras testavimas buvo beveik toks pat kaip ir pirmas. Pagrindinis skirtumas buvo tai, kad testavimas buvo atliktas ne pagrindinio programuotojo, bet dviejų savanorių, kurie buvo rasti per socialinius tinklus. Savanoriams buvo nurodytos sukurto DI instaliavimo ir paleidimo instrukcijos. Po sėkmingo žaidimo paleidimo, žaidėjams buvo nurodyta jų užduotis. Užduotis buvo per sutartą savaitę išbandyti sukurtą DI. Savanoriams buvo suteiktos ir parodytos kelios iš svarbiausių DI sukurtų valdomų funkcijų, kurias buvo galima koreguoti. Savanoriai naudodamiesi laisvos formos testavimu su galimybe keisti nurodytas funkcijas, galėjo per savaites laikotarpį išbandyti sukurtą DI. Savaitės pabaigoje, naudojantis Microsoft *TEAM pagalba, savanoriai buvo sukviesti į nuotolinio tipo pokalbį. Pokalbio metu naudojant iš anksto sukurtą lentelę* (***Lentelė* 1, Sukurtos mašinėlės DI testavimas**), buvo užduodami klausimai apie jų atliktą testavimą. Gauti savanorių atsakymai buvo

palyginti su prieš tai turimais kūrėjo gautais rezultatais. Neradus jokių klaidų buvo užfiksuota, kad atliktas kūrėjo pirminis testavimas buvo sėkmingas.

4.9. Pasirinktos įmonės sukurto DI testavimas

Pasirinktos įmonės *Creative Assembly* sukurto *Xenomorph* DI testavimas buvo atliktas naudojantis tokiu pat standartu kaip ir sukurtos mašinėles DI. Buvo sudaryta lentelė iš 3 punktų: Eil. Nr., Scenarijaus klausimai ir gautų rezultatų atsakymai. Sekanti užduotis buvo scenarijų klausimų sukūrimas ir trečio etapo žingsnis buvo DI testavimas ir gautų rezultatų užrašymas sudarytoje lentelėje. Lentelę ir gautų rezultatų atsakymus galite rasti 2 PRIEDAS (***Lentelė* 2, *Įmonės* DI testavimas**).

4.10. Gautų rezultatų palyginimas su pasirinktos įmonės sukurtu DI.

Paskutinis tyrimo žingsnis buvo sukurtų DI rezultatų palyginimas. Tam atlikti buvo sudaryta lentelė, kurioje buvo palyginti patys svarbiausi dalykai kuriant DI. Sudaryta ir užpildyta lentelė su visais prieš tai turimais duomenimis buvo panaudota išvadų sudarymui. Lentelę galite rasti 2 PRIEDAS (***Lentelė* 3, Sukurto DI palyginimas su „Creative Assembly“ įmonės sukurtu DI**).

IŠVADOS

1) DI kūrimo proceso analizė parodė, kad projekto nesėkmę gali sukelti tiek organizacinės, tiek technologinės priežastys, tokios kaip pernelyg aukšti reikalavimai, srities, kurioje bus taikoma AI sudėtingumas, kūrimui skirto biudžeto ar kitokių resursų trūkumas.
2) Išanalizavus populiariausius žaidimo variklius buvo nuspręsta, kad UE 5 yra geriausias variantas, norint sukurti kompiuterinį žaidimą su DI elementais.
3) Atlikus informacijos rinkimo ir analizės dali, buvo nuspręsta sukurti lenktyninę mašinėlę turinčia DI auto piloto elementų.
4) Pasirinkta įmonė dėl didesnių turimų žmogiškųjų ir biudžetinių išteklių, turi sukūrusį DI turinti daugiau ir geresnių žmonių imituojančių elementų.
5) Palyginus mūsų ir įmonės sukurtą DI, buvo matomi aiškus skirtumai tarp naudojamų DI elementų ir jų veikimo rezultatų.

LITERATŪROS ŠALTINIAI

1. Alexander Nareyak, AI in Computer Games: (February 2004), [žiūrėta 2023-9-15] Prieiga internete: https://dl.acm.org/doi/pdf/10.1145/971564.971593
2. Animation Editor, unrealengine, [žiūrėta 2023-11-20] Prieiga internete: https://docs.unrealengine.com/4.26/en-US/AnimatingObjects/SkeletalMeshAnimation/Persona/Modes/Animation/
3. Barreto, A., de O. Barros, M., Werner, C., Staffing a software project: A constraint satisfaction and optimization-based approach, Computers & Operations Research: (2008), [žiūrėta 2023-12-29] Prieiga internete: https://doi.org/10.1016/j.cor.2007.01.010
4. Basic Scripting with Blueprint, [žiūrėta 2023-11-29] Prieiga internete: https://docs.unrealengine.com/5.0/en-US/basic-scripting-with-blueprints-in-unreal-engine/
5. Blueprint Quick Start Guide, [žiūrėta 2023-12-31] Prieiga internete: https://docs.unrealengine.com/5.0/en-US/quick-start-guide-for-blueprints-visual-scripting-in-unreal-engine/
6. Blueprint Visual Scripting, [žiūrėta 2023-10-12] Prieiga internete: https://docs.unrealengine.com/5.0/en-US/blueprints-visual-scripting-in-unreal-engine/
7. Brooks, M., The many worlds of AI, New Scientist., 2023
8. Chen, R., A competence-time-quality scheduling model of multi-skilled staff for IT project portfolio, Computers & Industrial Engineering: (2020), [žiūrėta 2023-12-30] Prieiga internete: https://doi.org/10.1016/j.cie.2019.106183
9. Darius Amilevičius, Dirbtinis intelektas ir besiformuojančių technologijų etika: (2017), [žiūrėta 2023-12-31] Prieiga internete: https://portalcris.vdu.lt/server/api/core/bitstreams/d8a99d3b-9e4c-4c65-84fc-7d479221eba3/content
10. GameMaker: (2022), [žiūrėta 2023-12-31] Prieiga internete: https://en.wikipedia.org/wiki/GameMaker
11. Get started on Unity Pro today, [žiūrėta 2023-12-20] Prieiga internete: https://unity.com/pages/unity-pro-buy-now?ds_rl=1295837&ds_rl=1295837&gclid=CjwKCAjwyryUBhBSEiwAGN5OCJGA-t6nHpO7Dgmnj-eN7w4Ov8A7iYxVyg09WMkCuLmpoaqRdcaU9RoCT7UQAvD_BwE&gclsrc=aw.ds
12. Giedrimas, V., AI-enchanced Crowdsourcing as an Element of Information Systems Development. In: Rocha, Á., Ferrás, C., Ibarra, W. (eds) Information Technology and

Systems: (2023), [žiūrėta 2023-12-31] Prieiga internete: https://doi.org/10.1007/978-3-031-33261-6_27

13. M. Toftedahl, Which are the most commonly used Game Engines?: (2019 09 30)., [žiūrėta 2023-12-31] Prieiga internete: https://www.gamedeveloper.com/production/which-are-the-most-commonly-used-game-engines-
14. Philip Bontrager, Deep Reinforcement Learning for General Video Game AI: (June 2018), [žiūrėta 2023-12-31] Prieiga internete: https://arxiv.org/abs/1806.02448
15. Skeleton Editor, [žiūrėta 2023-12-26] Prieiga internete: https://docs.unrealengine.com/4.26/en-US/AnimatingObjects/SkeletalMeshAnimation/Persona/Modes/Skeleton/
16. Unreal Engine H.P.:(2022), [žiūrėta 2023-11-19] Prieiga internete: https://www.unrealengine.com/
17. Unreal Engine:(2022), [žiūrėta 2023-12-22] Prieiga internete: https://lt.wikipedia.org/wiki/Unreal_Engine
18. Westenberger, J., Schuler, K., Schlegel, D., Failure of AI projects: understanding the critical factors, Procedia Computer Science: (2022), [žiūrėta 2023-12-31] Prieiga internete: https://doi.org/10.1016/j.procs.2021.11.074

SUMMARY

The aim of this Master's thesis is to empirically use the UE5 game engine to create a self-driving car with elements of artificial intelligence. Then using the AI we created, try to find out the relationship between the created AI machine, quality and human resource with AI created by other individuals and companies.

To achieve this goal, I first had to find out what other game engines were publicly available, what made those game engines different from each other. After we decide on what kind of engine we will use, we need to find out about AI. What was AI, what it consisted of, and how we were going to use it. After a complete and detailed analysis of AI and a review of several well-documented scientific articles describing the use and development of AI is required. We also had to analyze a few well-known companies and their products that use AI. During this phase, we will create several UML diagrams that will also be used to facilitate the AI we are developing. The artificial intelligence being developed will be tracked in detail, all steps, challenges, problems, glitches and shortcomings will be described and tracked. The generated AI and performance description data will later be used to perform a comparison between it and the AI developed by company Creative Assembly. The final work will be to review all the available information, data and use the obtained results to answer the research question.

PRIEDAI

1 PRIEDAS

DI SPECIFIKACIJOS

Projektuojamas objektas – Dirbtinio intelekto elementų naudojimas kompiuteriniame žaidime.

Projektuojamo objekto paskirtis – Sukurti lenktyninį automobilį, turintį DI savybių. Sukurtą DI palyginti su pasirinktos įmonės sukurtu DI. Gautus rezultatus panaudoti išvadų sudarymui.

1. **DI specifikacijos:**
   1.1. Turi būti sukurtas automobilis su DI elementais;
   1.2. Sukurtas DI turi turėti tokias savybes;
      1.2.1. Važiuoti į prieki;
      1.2.2. Atlikti manevringus posūkius į kairę ir į dešinę:
      1.2.3. Pagal situaciją sugeba didinti ir mažinti greitį;
      1.2.4. DI žino, kur yra trasos kelias;
      1.2.5. Pagal situacija aplinkybes sukurtas DI automobilis gali aplenkti trasoje esančias, kitas transporto priemones;
   1.3. Sukurtas DI turi turėti tokius apribojimus;
      1.3.1. Nustatytus tikslius trasos kelio limitus;
      1.3.2. Greičio kontrolės valdymas;
      1.3.3. Pagreičio kontrolės valdymas;
      1.3.4. Pastovus duomenų perdavimas apie esamą poziciją žemėlapyje;
      1.3.5. Kontrolės sistema, skirta informuoti DI apie pasiektą kelio poziciją, tikslą;
   1.4. Sukurtas DI gali būti įdiegtas skirtingų modelių automobiliams.
   1.5. DI turi nustatytą finišo liniją.

TESTAVIMAS

Lentelė 1, Sukurtos mašinėlės DI testavimas

Sukurtos mašinėlės DI testavimas

Eil. Nr.	Scenarijaus klausimai	Rezultatas
1	Automobilis su DI, be išorinių asmenų pagalbos, sugeba pajudėti iš vietos.	Teigiamas
2	Automobilis su DI sugeba atlikti manevringus posūkius.	Teigiamas
4	Automobilis su DI sugeba apvažiuoti viena pilna ratą aplink turimą lenktyninę trasą.	Neigiamas
5	Automobilis su DI laikosi nustatytų greičio limitų.	Teigiama
6	Automobilis su DI, pasitaikius kelio kliūtims, sugeba sulėtinti greiti.	Nenuspėjamas
7	Automobilis su DI, sugeba lenktyniauti su kitais automobiliais turinčiais DI.	Teigiama
8	Automobilis su DI, sugeba atpažinti trasos finišo linija.	Teigiamas
9	Automobilis su DI, pasiekęs numatytą finišo linija sustoja.	Nenuspėjamas
10	Automobilis su DI, sugeba važiuoti pagal turimos trasos kelius.	Neigiamas
11	Automobiliui su DI, nukrypus nuo pagrindinės kelio trasos, pavyksta greitai ir efektyviai grįžti atgal.	Neigiamas
12	Automobilis su DI, sugeba pereiti į aukštesnes greičio pavaras.	Teigiamas
13	Sukurtas DI gali būti priskirtas ir kitiems automobiliams.	Teigiamas
14	Sukurtas DI, sugeba atpažinti *Spine* funkcijos egzistavimą?	Nenuspėjamas
15	Sukurtas DI sugeba mokytis iš savo veiksmų istorijos.	Neigiamas
16	Sukurtas DI sugeba tobulėti iš prieš tai atliktų veiksmų.	Neigiamas
17	Sukurtas DI gali reaguoti į aplink esančius objektus ir garsus?	Neigiamas
18	Sukurtas DI automobilis sugeba gauti duomenis iš aplinkoj esančių objektų?	Neigiamas

Lentelė 2, Įmonės DI testavimas

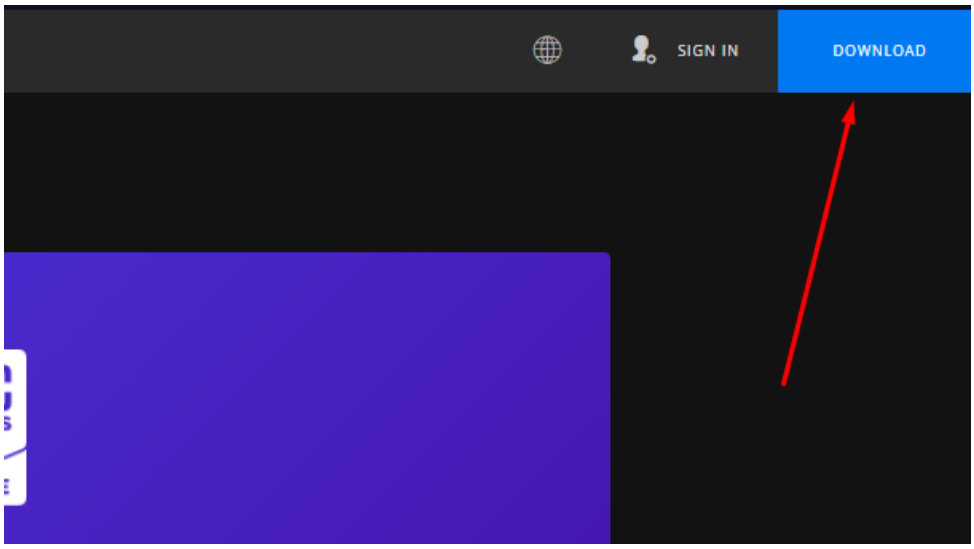
Eil. Nr.	Scenarijaus klausimai	Rezultatas
1	Sukurtas Xenomorph ateivis sugeba judėti ir manevruoti aplink kosminę stotį.	Teigiamas
2	Xenomorph ateivis neturi problemų ir vietų, kur pastrigtų ar užlūžtų.	Neigiamas
3	Xenomorph ateiviui iš esamos aplinkos pareina vieno arba kito tipo duomenis.	Teigiamas
4	Xenomorph ateiviu išgirdus keistą garsą arba signalą, bus imamasi tyrinėjimo, peržiūros veiksmai.	Teigiamas
5	Xenomorph ateivis, naudodamasis aplinka ir gautais signalais, sugeba sekti žaidėją aplink kosminę stotį.	Teigiamas
6	Sukurtas Xenomorph sugeba pagal situaciją skleisti skirtingų tipo garsus.	Teigiamas
7	Xenomorph, pamačius pagrindinį žaidėją, imamasi gaudymo veiksmų.	Teigiamas
8	Xenomorph, pagavus pagrindini žaidėją, žaidėjas yra nužudomas. Žaidėjas sugrąžinamas į paskutinę išsaugotą vietą.	Teigiamas
9	Xenomorph ateivis, bėgant laikui, moka mokytis ir tobulėti.	Teigiamas
10	Xenomorph ateivio mokymosi greitis priklauso nuo žaidėjo sugebėjimo vengti pagrindinės grėsmės, tai yra Xenomorph.	Teigiamas
11	Išorinių įrankių naudojimas ant Xenomorph sukelia vieno arba kito tipo poveikį, į kurį Xenomorph DI sugeba sureaguoti.	Teigiamas
12	Xenomorph ateivis, naudodamasis paskutine turima informacija apie žaidėją, gali bandyti nuspėti jo einama kelio trajektorija?	Teigiamas
13	Xenomorph ateivis, naudodamasis gaunama informacija apie žaidėją, sugeba sukurti situaciją, kur Xenomorph gali sukurti žaidėjui spąstus.	Teigiamas
14	Xenomorph ateivis, sugeba pakeisti savo nuomonę, po nuomonės pakeitimų Xenomorph imsis atitinkamų veiksmų.	Teigiamas

Lentelė 3, Sukurto DI palyginimas su „Creative Assembly“ įmonės sukurtu DI

Mūsų sukurtas DI	Rezultatų palyginimas	Įmonės sukurtas DI
300+	Testavimo metu patirtas klaidų skaičius	13
10	Testavimo metu, neveikiančių funkcijų skaičius	1
7	Testavimo metu, veikiančių funkcijų skaičius	13
6 mėn.,	Skirtas laikas, kuriant DI	6-12 mėn.
1	Asmenų skaičius, kuris dirbo prie DI sukūrimo	20+

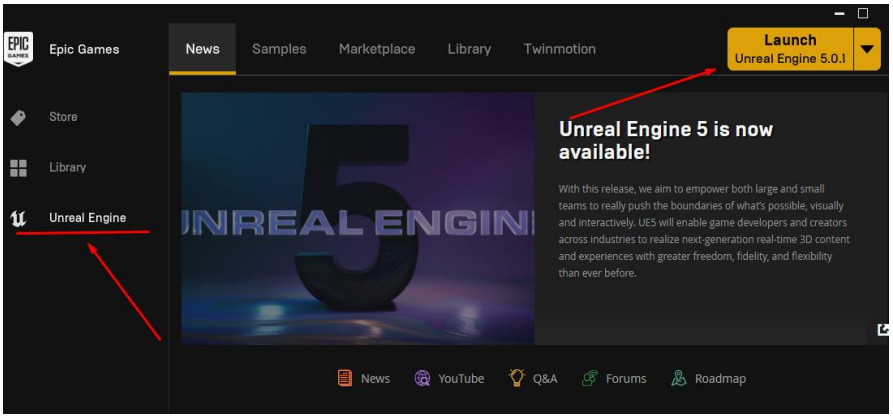
Informacija apie tai, kaip galima parsisiųsti DI failus. Kaip ir kur juos reikia įkelti ir kaip paleisti viską.

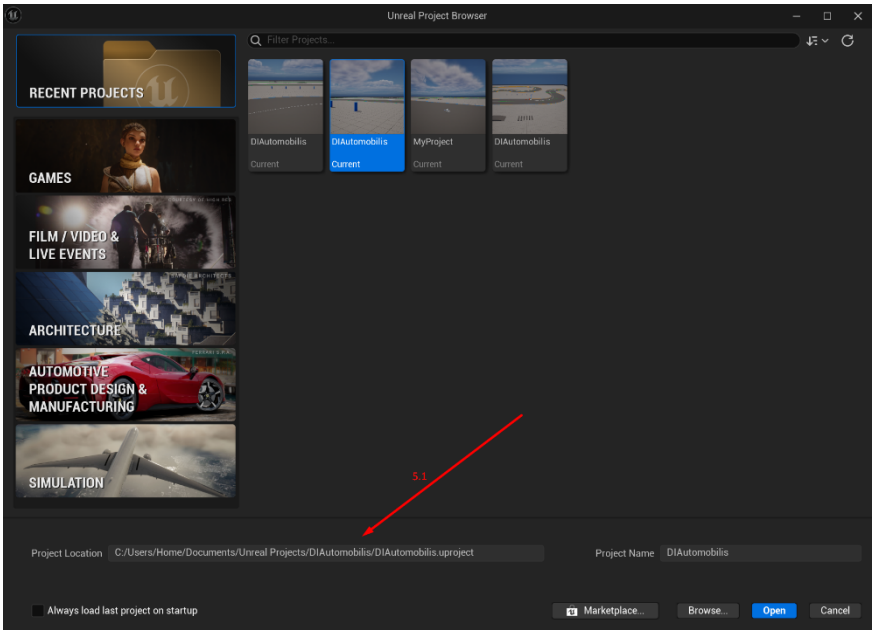
1. Visų pirma turite nueiti į nurodytą puslapį ir parsisiųsti visą dokumentą su visais failais, kai tai padarysite, toliau turėsite naudojant *winrar* iškelti failus. Atlikus tai, turėtumėte turėti vieną dokumentą su pavadinimu „Magistrinis_projektas“, kurioje viduje rasite kitą failą su pavadinimu „DI kūrimo projektas“. Šitas failas jums bus reikalingas vėliau.
   1.1. → Spauskite čia ← . kad parsisiųstumėte projekto failus.
2. Toliau turėsite nueiti į *Epic Game Launcher* puslapį → Spauskite čia ←, svetainėje turėsite susirasti mygtuką su žodžiu *DONWLOAD*, nuspaudę jį, galėsite parsisiųsti žaidimo variklį. Parsiųstą failą reikės instaliuoti, jei instaliacija įvyks sėkmingai, tada teliks paleisti programą.



Programai pasileidus privalote prisijungti su *Epic Game Store* paskyrą. Jei neprisimenate savo paskyros duomenų, tada galite paspausti mygtuką *forgot password*, kuris jums į paštą atsiųstų laišką su nauja svetainės nuoroda, ši nuoroda perkels jus ten, kur bus galima pasikeisti slaptažodį. Bet jei paskyros išvis neturite, tada teks susikurti naują paskyrą.

3. Atlikus prieš tai išvardintus veiksmus teisingai, programa pasileis ir jos kairėje pusėje rasite lauką su pavadinimu *Unreal Engine*, nuspaudus tą teksto lauką jūs būsite perkeliamas į *Unreal Engine* puslapio vietą. Programos dešinėje pusėje jūs matysite mėlyną mygtuką su užrašu *DOWNLOAD*. Nuspaudus šį mygtuką bus parsiunčiama *Unreal Engine 5* programa. (Perspėjimas, kad parsiųsdami programą, automaitiškai bus siunčiama naujausia programos versijos dalis.)

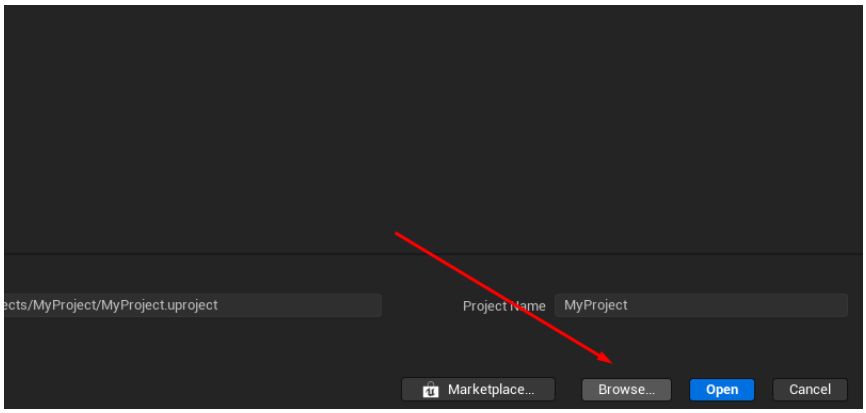




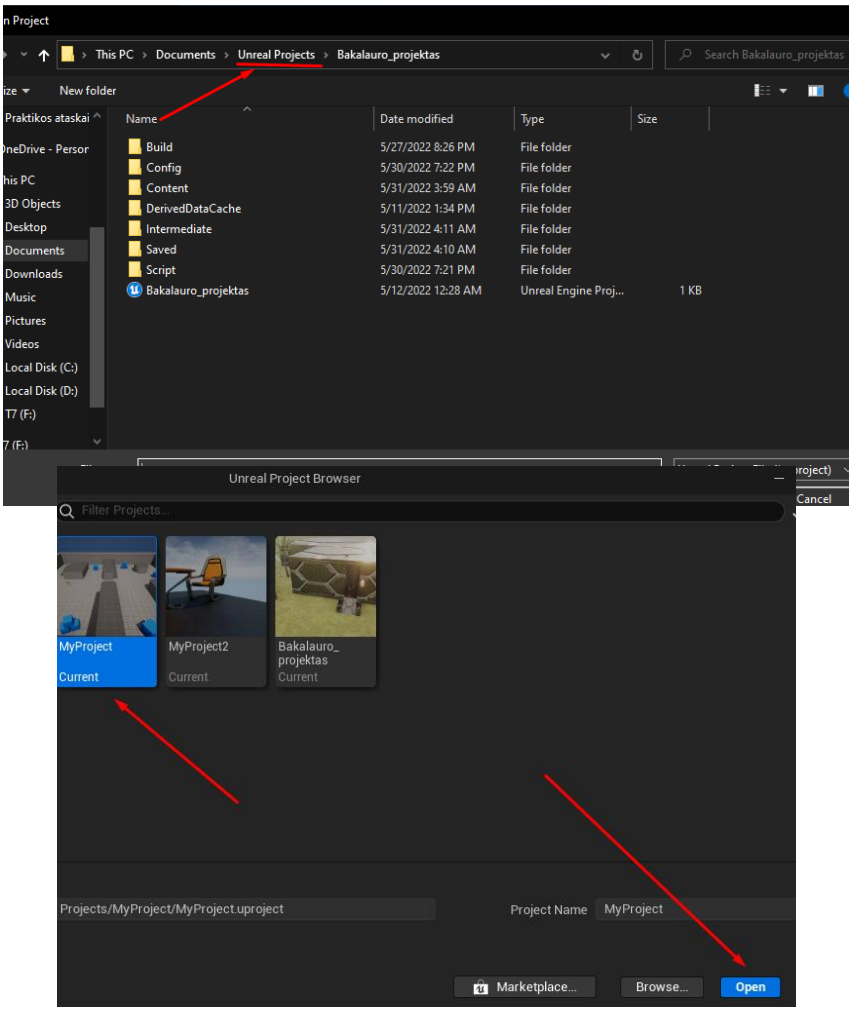
4. Kai *Unreal Engine 5* baigiasi instaliuotis, tame pačiame programos lauke ir vietoje, kur radote „DONWLOAD“ mygtuką, atsiras naujas mygtukas, tai yra „LAUNCH“ mygtukas. Tą mygtuką reikės nuspausti, kad būtų paleidžiamas UE 5 žaidimo variklis.
5. Kai nuspausite *Launch* mygtuką, turėsite palaukti, kol išmes naują langą. Šitame lange turėsite atlikti kelis žingsnius, tai yra:
   5.1. Failo įkėlimo vieta. Mums to reikės, nes ten bus vieta, kur turėsite įkeltą prieš tai minėtą failą „DI_kūrimo_projektas“. Kur yra projektai saugomi, tai galite rasti pažiūrėję į pateiktą

paveikslėli. Šita informacija gali būti reikalinga tokiu atveju, jeigu užmiršote projektų laikymo vietą, tokiu atveju žinosite, kur eiti ir įkelti „DI_kūrimo_projektas“.

5.2. Paskui spauskite mygtuką „Browse“, atlikus tai, jums bus iškviestas naujas langas.



5.3. Naujame lange privalote nueiti į „Unreal Projects“ lauką, kur turėsite įkelti prieš tai parsisiųstus magistro projekto failus „DI_kūrimo_projektas“. Atlikus tai sėkmingai bus galima uždaryti langą ir išjungti Unreal Engine 5. Kai Unreal Engine 5 išsijungs, galite išjungti ir paleisti kompiuterį iš naujo.



5.4. Jeigu atlikote visus veiksmus teisingai, tada, kai pasileisite *Epic Game Launcher* ir *Unreal Engine 5* iš naujo, turėtumėte matyti naują projektą su pavadinimu „ DI_kūrimo_projektas“. Pasižymėkite jį ir aktyvuokite projektą.

6. Tada turėsite palaukti, kol projektas bus pilnai įkrautas. Tai gali užtrukti nuo 30 min. iki 2 valandų. Paskutinis žingsnis būtų DI paleidimas/testavimas. Visa tai galima atlikti nuspaudus viršuje žalią mygtuką.

Viskas, tai buvo visi reikalingi žingsniai, kuriuos turėjote atlikti, kad galėtumėte išbandyti sukurtą DI. Kadangi sukurtas DI yra be jokių ribojimų, jūs galite visus failus, programos kodą ir duomenis, galite modifikuoti ir pertvarkyti į jums norimą formą. Tačiau esate įspėjami, kad bet kokie atlikti pakeitimai, bus galimai išsaugomi ir nesugrąžinami. Tokiu atveju, jei norėsi sugrįžti prie DI pirmosios būsenos, teks kartoti visus prieš tai atliktus veiksmus. Sėkmės!