TALLINNA TEHNIKAÜLIKOOL
INSENERITEADUSKOND
Virumaa kolledž

Parkimisrakenduse integratsioonikihi evolutsiooni analüüs ja kavandamine

Analysis and design of the evolution of a parking application's integration layer

ARUKAD SÜSTEEMID JA RAKENDUSINFOTEHNOLOOGIA ÕPPEKAVA
LÕPUTÖÖ

Üliõpilane: Merilin Lidmets
Üliõpilaskood: 207552EDTR
Juhendaja: Avar Pentel, lektor

Kohtla-Järve, 2023

AUTORIDEKLARATSIOON

Olen koostanud lõputöö iseseisvalt.

Lõputöö alusel ei ole varem kutse- või teaduskraadi või inseneridiplomit taotletud. Kõik töö koostamisel kasutatud teiste autorite tööd, olulised seisukohad, kirjandusallikatest ja mujalt pärinevad andmed on viidatud.

"...." ..... 20.....

Autor: Merilin Lidmets
/ allkiri /

Töö vastab rakenduskõrgharidusõppe lõputööle/magistritööle esitatud nõuetele
"...." ..... 20.....

Juhendaja: .....
/ allkiri /

Kaitsmisele lubatud
"...." ..... 20.....

Kaitsmiskomisjoni esimees .....
/ nimi ja allkiri /

LIHTLITSENTS LÕPUTÖÖ ÜLDSUSELE KÄTTESAADAVAKS TEGEMISEKS JA REPRODUTSEERIMISEKS

Mina, Merilin Lidmets (sünnikuupäev: 18.03.1994)

1. Annan Tallinna Tehnikaülikoolile tasuta loa (lihtlitsentsi) enda loodud teose

„Parkimisrakenduse integratsioonikihi evolutsiooni analüüs ja kavandamine“, mille juhendaja on Avar Pentel,

1.1. reprodutseerimiseks säilitamise ja elektroonilise avaldamise eesmärgil, sealhulgas Tallinna Tehnikaülikooli raamatukogu digikogusse lisamise eesmärgil kuni autoriõiguse kehtivuse tähtaja lõppemiseni;

1.2. üldsusele kättesaadavaks tegemiseks Tallinna Tehnikaülikooli veebikeskkonna kaudu, sealhulgas Tallinna Tehnikaülikooli raamatukogu digikogu kaudu kuni autoriõiguse kehtivuse tähtaja lõppemiseni.

2. Olen teadlik, et punktis 1 nimetatud õigused jäävad alles ka autorile.

3. Kinnitan, et lihtlitsentsi andmisega ei rikuta kolmandate isikute intellektuaalomandi ega isikuandmete kaitse seadusest ja teistest õigusaktidest tulenevaid õigusi.

TalTech Inseneriteaduskond Virumaa kolledž

LÕPUTÖÖ ÜLESANNE

Üliõpilane: Merilin Lidmets, 207552EDTR

Õppekava, peaeriala: EDTR17/18 Arukad süsteemid ja rakendusinfotehnoloogia

Juhendaja(d): lektor, Avar Pentel, avar.pentel@taltech.ee

Lõputöö teema:

(eesti keeles) *Parkimisrakenduse integratsioonikihi evolutsiooni analüüs ja kavandamine*

(inglise keeles) Analysis and design of the evolution of a parking application's integration layer

Lõputöö põhieesmärgid:

1. Olemasoleva integratsioonikihi analüüs ja kokkuvõte
2. Lisanduva integratsiooni nõuete kaardistamine
3. Integratsioonikihi evolutsiooni kava

Lõputöö etapid ja ajakava:

Nr	Ülesande kirjeldus	Tähtaeg
1.	Olemasoleva integratsioonikihi analüüs	31.10.2023
2.	Evolutsiooni nõuete kaardistamine ja analüüs	15.11.2023
3.	Lõputöö vormistamine ja presentatsioon	10.12.2023

Töö keel: eesti keel **Lõputöö esitamise tähtaeg:** "20"detsember 2023a

Üliõpilane: Merilin Lidmets..... "01"oktoober 2023a
/allkiri/

Juhendaja: Avar Pentel..... ".....". ..... 20.....a
/allkiri/

Programmijuht: Žanna Gratšjova..... ".....". ..... 20.....a
/allkiri/

SISUKORD

JOONISTE LOETELU

TABELITE LOETELU

EESSÕNA

Diplomitöö on kirjutatud, kavandamaks parkimisrakenduse ja majandustarkvara vahelise integratsiooni arendust. Töö eesmärk on kaardistada parkimisrakenduse taustsüsteemi hetkeolukord, analüüsida ja kirjeldada vajalikke nõudeid, määrata kõrgeima prioriteetsusega probleemid ning püstitada arendusülesanded. Eesmärgi saavutamiseks uurib töö autor eri tarkvaraarenduse ja nõuete kogumise meetodeid ning kirjeldab nende abil nõudeid ja koostab arendusülesanded.

Töö tulem on nõuete kogum ja arendajatele esitatavad nõuetele vastavad ülesanded integratsiooni loomiseks.

Lõputöö on kirjutatud eesti keeles ning sisaldab teksti 34 leheküljel, 3 peatükki, 2 joonist ja 11 tabelit.

LÜHENDITE JA TÄHISTE LOETELU

API – ingl *Application Programming Interface*, rakendusliides [1]

Eepos – ingl *epic,* mahukas kasutajalugu [1]

ERP-süsteem – ingl *Enterprise Resource Planning*, majandustarkvara [1]

Kasutuslugu – ingl *user story,* lihtne jutustus, mis illustreerib kasutajanõuet mingi isiku vaatepunktist [1]

Käsitlusvõlg – ingl *backlog,* tootenõuete ja lõpetamisele kuuluvate saaduste prioriteediloend [1]

Metoodika – korrastatud lähenemine suuremale tegevusele

MoSCoW – nõuete prioriseerimistehnika

MVP – ingl *Minimal Viable Product*, minimaalne elujõuline toode [1]

NAV – ingl *Microsoft Dynamics Navision*, majandustarkvara

PCC – ingl *Parking Control Center*, Belgia veebipõhise parkimise haldusteenus

Tööseminar – ingl *workshop*, õppeotstarbeline või probleemi lahendamisele pühendunud interaktiivne üritus [1]

Välearendus – ingl *agile development*, tarkvaraarenduse metoodika [1]

Veebiteenus – ingl *web service,* meetod ja/või vahendikogum elektronseadmete suhtluseks võrgu kaudu ning veebipõhiste rakenduste integreerimiseks [1]

SISSEJUHATUS

Elame tehnoloogiliste edusammude ajastul, mis on suuresti muutnud ettevõtete toimimist, mille tulemuseks on kiiremad ja tõhusamad äriprotsessid. Veebipõhiste teenuste kulutõhusamaks muutmisel mängib suurt rolli eri protsesside automatiseerimine.

Koos veebitehnoloogiate arenguga on APIde ehk rakendusliideste kasutamine tõusnud veebiteenuste arendamise standardiks, olles veebirakenduste automatiseerimise keskmes. APIde kasutamine võimaldab ühel rakendusel saata päringuid teisele rakendusele ja saada vastuseid ilma inimsekkumiseta. See vähendab olulisel hulgal vajadust manuaalsete toimingute järele ja võimaldab süsteemidel efektiivselt andmeid vahetada ja eri ülesandeid täita.

Eri süsteemide rakendusliidestega integreerimiseks luuakse integratsioonikiht, mille ülesehitus ja rakendus võivad sõltuvalt süsteemist ning selle vajadustest varieeruda. Integratsioonikihi arendamine nõuab põhjalikku ülevaadet ja arusaama süsteemi protsessidest ning ettevõtte äriprotsessidest.

Lõputöö eesmärk on analüüsida ja kavandada mobiilse parkimisrakenduse integratsioonikihi laiendamist. Nimelt on parkimisrakendust arendava ettevõtte eesmärk ühtlustada eri riikides pakutavate teenuste seadistamist ja müümist, mis eeldab ühtse ERP-süsteemi kasutamist ja sellega integreerumist.

Lõputöö esimeses peatükis kirjeldatakse käsitletavat ettevõtet ja probleemi tausta, samuti antakse ülevaade süsteemi hetkeolukorrast. Lisaks selgitatakse töö käsitlusala ja töö autori rolli tulemuste saavutamisel.

Lõputöö teises peatükis uuritakse, kirjeldatakse ja võrreldakse eri nõuete kogumise, prioriseerimise ja tarkvaraarenduse metoodikaid. Lisatud on põhjendus metoodikate valikute kohta, mida on kasutatud äri- ja süsteemianalüüsiks.

Kolmas peatükk on pühendatud lahenduse leidmisele. Seal kirjeldatakse kasutatud tööriistu ning ärianalüüsi tulemusi. Ärianalüüsist saadud sisendi ja selgunud ärinõuete põhjal tehti süsteemianalüüs, mille käigus kaardistati ning prioriseeriti süsteemi funktsionaalsed ja mittefunktsionaalsed nõuded. Formuleeritud nõuete alusel jaotati eduka integratsiooni saavutamiseks vajalikud arendusülesanded kahte etappi ning määrati kindlaks ja sõnastati arendusmeeskonna esimeses faasis ellu viidavate funktsioonide ülesanded. Peatükk sisaldab ka informatsiooni tulevaste arendustegevuste kohta, mis on suunatud integratsiooniga seotud taustsüsteemi tööde parendamisele.

1. PROBLEEMI PÜSTITUS JA ÜLESANDE KIRJELDUS

Selles peatükis tutvustatakse ettevõtet, kelle integratsioonikihi evolutsiooni kavandatakse, ning antakse ülevaade hetkeolukorrast ja projekti eesmärgist.

1.1 Ettevõtte kirjeldus

Ettevõte, kus autor töötab, tegeleb parkimisrakenduse taustsüsteemi arendusega. Ettevõte on parkimisrakenduse ettevõtte tütarfirma, kelle Eesti kontor asub Tallinnas, kus töötab 16 inimest, nende seas arendusmeeskond, tarkvara testijad, süsteemiadministraatorid ning tootemeeskond.

Parkimisrakenduse ettevõte, mille Eestis asuvas tütarfirmas autor töötab, on Prantsuse ettevõte. Prantsusmaal Pariisis asuvas kontoris töötab ligikaudu 80 inimest, kelle seas on lõppkasutajavaate arendustiim, disainerid, tootetiim, turundustiim, klienditugi, kliendihaldurid, tarkvara testijad, ärianalüütikud ja ettevõtte juhatus.

Parkimisrakenduse ettevõtte emafirma on globaalne organisatsioon, millel on parkimismaju ja -platse üle kogu maailma. Parkimisrakendus võimaldab lõppkasutajal oma mobiili- ja veebirakenduse kaudu parkida nii linnade ja omavalitsuste avalikel tänavatel kui ka eraettevõtete halduses olevates parkimismajades ja -platsidel. Parkimisrakendus, mille arendamise kallal autor töötab, on pühendunud emafirma Euroopa turul pakutavatele teenustele.

Läbi parkimisrakenduse on kliendil võimalik tarbida mitmeid parkimisega seotud tooteid ja teenuseid, mille hulka kuuluvad näiteks parkimise perioodikaardid, parkimiskoha broneerimine, tunnipõhine parkimine, soodustingimustel parkimise kasutamine jpm.

1.2 Probleemi kirjeldus ja eesmärk

Parkimisrakenduse emafirma portfell sisaldab väga suures koguses parkimismaju ja -platse üle kogu maailma. Igas riigis on emafirmal eraldi filiaal, mille töötajad tegelevad asukohariigi parklatega seonduva haldamisega.

Oma teenuse pakkumine parkimisrakenduse vahendusel ei ole praegu emafirma põhiline tuluallikas – see on vaid lisandväärtus, mida klientidele pakutakse. Võttes aga arvesse parkimistööstuse üldist trendi kolida oma teenuste müügiga järjest enam internetti, on emafirma võtnud eesmärgiks parkimisrakenduse rolli oma äritegevuses üha suurendada. Selleks, et parkimisrakenduses pakutavate teenuste ja toodete müük oleks sujuv ja võimalikult vähenõudlik kõikide osapoolte jaoks, on oluline, et nende toodete ning teenuste seadistamine oleks kõikide asukohariikide töötajate jaoks tsentraalne ja lihtne ning vähe aega nõudev.

Ei ole mõistlik, et asukohariikide töötajad seadistaksid pakutavaid tooteid ja teenuseid parkimisrakenduse administraatori platvormil, kuna äritegevust toetava majandustarkvara väljatöötamine ja arendamine olemasolevale platvormile on liiga keerukas ja suur väljakutse. Arvestades, et emafirma on taolisi raamatupidamis- ja haldussüsteeme lasknud endale juba minevikus arendada ning et turul on mitmeid kolmanda osapoole ERP-süsteeme, mida on vastavalt ettevõtte vajadustele lihtne kohandada, on mõttekam parkimisrakenduse taustsüsteemil olemasolevate süsteemidega integreeruda.

Emafirma eesmärk on tulevikus kasutada parklate haldamiseks ühtainsat ERP-süsteemi kõikides Euroopa riikides, kus oma teenuseid pakutakse. Selleks on valitud Microsoft Dynamics Navisioni lahendus.

Diplomitöö eesmärk on analüüsida ja kavandada arendustöid parkimisrakenduse taustsüsteemi ja NAV-i veebiteenuste integreerimiseks. Autori roll on lahenduse nõuete kogumine ja kaardistamine, vormindamine, lahenduse kirjeldamine ning arendustööde prioriseerimine.

Arendatava integratsiooni puhul on oluline, et see ühilduks eksisteeriva lahendusega, kasutades kaasaegseid tehnoloogiaid.

1.3 Olemasoleva integratsioonikihi kirjeldus

Parkimisrakenduse integratsioonikiht toimib vahendajana, korraldades rakendusliidese abil sujuvat andmevahetust ja funktsionaalsust eri süsteemide vahel, tagades kogu platvormi ladusa toimimise.

See kiht liidestab parkimisrakenduse serveripoole mitme konfiguratsiooniteenuse pakkujaga, kust hangitakse olulisi andmeid ja sätteid, mis tagavad rakenduse tõhusa töö ja lõppkasutajale kuvatavate andmete õigsuse. Konfiguratsioonipakkujad pakuvad andmeid parkimisteenuse hinna, geograafiliste asukohtade, pakutavate toodete ja mitmesuguste muude parameetrite kohta, mis võimaldavad rakendusel dünaamiliselt kohaneda parkimist pakkuva partneri poolt muudetud operatiivsete nõuete, reeglite või teenustega.

Samal ajal loob integratsioonikiht ka ühenduse mitmete veebipõhiste parkimise haldamise süsteemidega, millega vahetatakse reaalajas teavet käimasolevate parkimisseansside kohta.

Integratsioonikiht tagab, et süsteemi taustsüsteemide ja kasutajaliidese vahel edastatakse täpset ja asjakohast teavet.

1.3.1 Haldustarkvara

Tarkvara A on parkimisrakenduse emafirma tsentraalne haldustarkvara Prantsusmaa turul. Seda kasutatakse eri parklate ja nende teenuste hindade seadistamiseks. Kuna emafirma haldab üle 650 parkimismaja ja -platsi, mille teenuseid ja tooteid müüakse nii veebi vahendusel eri platvorme kasutades kui ka parkimismajade ja -platside kioskites kohapeal, on üliolulise, et müümiseks kasutatav informatsioon on igal pool ühtne ning täpne.

Parkimisrakenduse taustsüsteem hangib rakendusliidese kaudu haldustarkvarast infot Prantsusmaa parklate hindade kohta.

1.3.2 Organisatsioonikataloog

Tarkvara B on parkimisrakenduse emafirma tsentraalne organisatsioonikataloog, milles hallatakse mitmesuguseid parklatega seotud organisatsioonilisi andmeid. Selles süsteemis seadistatakse kõikide emafirma parklate korralduslikke üksikasju, alustades parklaga seotud finantsandmetest ja lõpetades parkimisala lahtiolekuaegadega.

Parkimisrakenduse taustsüsteem hangib rakendusliidese kaudu organisatsioonikataloogist infot Prantsusmaa, Belgia, Šveitsi, Hispaania ja Luksemburgi parklate ning linnade korralduslike üksikasjade kohta.

1.3.3 Be-Mobile Parking Control Center

Be-Mobile Parking Control Center (PCC) on platvorm, mis võimaldab parkimisteenuse pakkujatel hankida Belgia linna- ja omavalitsusspetsiifilisi seadistusi, mida hallatakse tsentraalselt Be-Mobile Parking Control Centeri süsteemis.

PCC rakendusliides tagab, et veebipõhise parkimisteenuse pakkujad kasutavad parkimisteenuse müümisel korrektseid ärireegleid ja tariife.

Parkimisrakenduse taustsüsteem hangib rakendusliidese kaudu parkimistsooni geograafilise ala, parkimisreeglid ja -hinnad ning soodustingimustel parkimise õiguse.

1.3.4 EasyPark

Parkimisrakendus pakub muude linnade seas parkimist ka Pariisi tänavatel. Pariisis toimub pidev õhusaaste seire. Kui õhusaaste tase tõuseb üle kokkulepitud piiri, kuulutatakse välja õhusaastevastane päev, mil parkimine linnatänavatel on tasuta ning ühistranspordipileteid saab osta soodushinnaga. Neid päevi ei ole võimalik pikalt ette planeerida, kuna õhusaastuse tase tõuseb ette teavitamata. Reeglina teatab Pariisi linnavalitsus õhusaastevastasest päevast ette maksimaalselt 24 tundi.

Selleks, et parkimisrakendust kasutavad kliendid saaksid osa tasuta parkimise hüvest, hangib parkimisrakendus EasyParki rakendusliidese kaudu infot õhusaastevastaste

päevade kohta. Seda infot kasutades muutub määratud päeval parkimine Pariisi tänavatel tasuta parkimiseks ning kõiki käimasolevaid tasustatud parkimisseansse pikendatakse õhusaastevastase perioodi pikkuse võrra.

1.3.5 Google Places API

Google Places API on teenus, mis tagastab teavet geograafiliste asukohtade, asutuste asupaikade ning eri huvipunktide kohta.

Parkimisrakendus kasutab Google Places API teenust mitmel moel, näiteks aadresside automaattäitmisel ja valideerimisel, parkimismajade leidmisel ning otsingusoovituste pakkumisel.

1.3.6 Parkimise haldussüsteemid

Parkimisrakenduse taustsüsteem edastab enda rakendusliidese kaudu teavet eri parkimise haldussüsteemidesse, mida kasutavad parkimiskontrolörid parkimisseansside kontrollimiseks.

Samuti on parkimisrakendus integreeritud parklate automaatsete tõkkepuudega, mille saadetud teabe põhjal kas alustatakse või lõpetatakse parkimisseansse.

1.4 Autori roll ja töö ulatus

Autor töötab töö raames ettevõttes tooteomanikuna, kelle ülesanne on tegeleda nõuete kogumisega, struktureerimisega ning nõuetele tuginedes arendusülesannete koostamisega. Tooteomanikuna vastutab autor arendatud toote väärtuse maksimeerimise ning tootelogi haldamise eest.

Töö käsitlusalasse kuuluvad järgmised tegevused: analüüsida rakenduse taustsüsteemi olemasolevat andmestruktuuri, kirjeldada uue integratsiooni loomiseks vajaminevaid nõudeid ning nende põhjal luua kasutuslood reaalsete arendustöödega alustamiseks.

2 METOODIKAD

Selles peatükis kirjeldatakse täpsemalt töö valmimiseks valitud protsesse.

2.1 Nõuete kogumise metoodikad

Nõuete kogumise protsess on tarkvaraarenduse kriitiline aspekt, mis hõlmab tarkvaraprojekti funktsionaalsete ja mittefunktsionaalsete nõuete tuvastamist ning dokumenteerimist. [2]

Nõuete kogumise protsess algab projekti kontseptsiooni ja ulatuse määratlemisega ning lõpeb nõuete dokumenteerimisega. Nõuetes väljendatakse, millist ülesannet peab arendatav tarkvara täitma hakkama, ning need on aluseks kogu järgnevale arendustööle. [3]

Nõuete kogumise tehnikaid on hulgaliselt, kuid kõik ei ole ühtmoodi efektiivsed. Nõuete kogumisel tuleb arvestada analüüsitava projekti karakteristikuid ning kombineerida nõuete kogumise tehnikate seast asjakohased valikud. [4]

Tabelis 2.1 on kirjeldatud nõuete kogumise eri meetodeid. Tabel ei ole täielik, vaid selles on kirjeldatud vaid enimsoovitatud tehnikaid.

Tabel 2.1 Enimsoovitatud nõuete kogumise meetodid [2][3][4][5]

Nõuete kogumise meetod	**Kirjeldus**	**Tugevused**	**Piirangud**
Dokumentide analüüs	Dokumentide analüüsi käigus uuritakse ärivajaduste paremaks mõistmiseks taustteavet. See võib hõlmata ka olemasoleva lahenduse uurimist.	Analüüsi alusena saab kasutada olemasolevat materjali.	Olemasolev dokumentatsioon võib olla vananenud või ebaoluline.
Kasutajalood	Kasutajalugu on lühidalt ja konkreetselt sõnastatud väide, mis kirjeldab vajalikku funktsionaalsust või omadust, mis pakub väärtust huvigruppidele, väljendades nende vajadusi.	Lihtsasti mõistetav huvigruppide jaoks, kuna keskendutakse väärtustele.	Tavaliselt on kasutajalood kavandatud ajutiseks vajaduste kirjeldamiseks ja prioriseerimiseks. Selle meetodi kasutamine pikemaajalise teadmiste säilitamise või üksikasjaliku analüüsi vahendina tekitab probleeme dokumentatsiooni haldamisel.
Nõuete kogumise tööseminar	Tööseminar on konkreetse eesmärgiga sündmus, millest võtavad osa huvigrupid ja valdkonna eksperdid ning mis toimub etteantud ajaperioodil. See on organiseeritud ühise eesmärgi saavutamise nimel läbi koostöö.	Ühiste otsusteni jõudmine on võimalik suhteliselt lühikese aja jooksul.	Võib olla keeruline leida sobivat aega, mis kõikidele huvigruppidele sobib.
Intervjuu	Intervjuu on metoodiline viis kogumaks ärianalüüsi andmeid. Seda tehakse inimestega rääkides ja neilt asjakohaseid küsimusi küsides, dokumenteerides nende vastuseid.	See on mitmesugustes olukordades kasutatav lihtne ja sirgjooneline tehnika.	Intervjueerimine nõuab osalistelt pühendunud kaasalöömist.
Mõttetalgud	Mõttetalgute siht on genereerida arvukalt ja erinevaid ideid. Protsessis keskendutakse spetsiifilisele teemale, püüdes leida sellele võimalikult palju eri lahendusi.	Võimaldab koguda hulgaliselt innovaatilisi ideid lühikese aja jooksul.	Osavõtt on oluliselt seotud iga osavõtja individuaalse loomingulisusega ja valmisolekuga panustada lahenduste leidmisse.
Protsessi modelleerimine	Protsessi modelleerimise käigus koostatakse standardiseeritud graafiline mudel, mis illustreerib töö tegemise viisi. See näitab tegevuste kulgu tööprotsessis.	Pakub huvigruppidele selge pildi vastutusaladest, üleandmiskohtadest ning tegevuste järjestusest.	Struktuuri ebakorrektsel organiseerimisel võib see muutuda ülimalt keeruliseks ja seeläbi kasutuskõlbmatuks.
Prototüüpimine	Prototüüpimine on toote kujundamise lähenemisviis, mille keskmes on töötava varase mudeli ehk prototüübi esitamine.	Pakub visuaalset ettekujutust tulevasest tootest.	Keerukate süsteemide või protsesside puhul on prototüübi loomine aeganõudev.
Toote teekaart	Toote teekaart kirjeldab teekonda, kuidas saavutada seatud eesmärke, ja toote edasist arenguplaani vastavalt toote nõuetele ning funktsionaalsustele.	Ligipääsetav ja arusaadav kõikidele huvigruppidele.	Projektide puhul, mille eesmärgid on pidevas muutuses, on toote teekaart kasutu.
Minimaalne elujõuline toode	MVP-ga määratakse miinimumarv nõudeid, mis on vajalikud huvigruppidele ja esimestele kasutajatele oluliste väärtuste loomiseks. Selle käigus keskendutakse põhifunktsionaalsusele, mis on vajalik esialgse versiooni juurutamiseks.	Vähendab kulusid ja riske, saades klientidelt varajast tagasisidet. On soodsam arendada kui täielikku lahendust korraga.	Nõuab eelnevat üksikasjalikku analüüsi, et määratleda vajalik funktsionaalsus.
Tööde käsitlusvõla viimistlus	Eesseisvate ehk tegemata tööde detailne nimekiri.	Pakub selgust ja ühist mõistmist eelseisvate tööde osas.	Toote visiooni ja eesmärkide pideva muutuse puhul ei ole see meetod efektiivne.
Planeerimise tööseminar	Planeerimise tööseminaride käigus määratakse kindlaks tarnitava väärtuse laad ja ajakava. Sellised tööseminarid toimuvad reeglina vahetult pärast tööde käsitlusvõla viimistlust.	Osapoolte pidev suhtlus ja koostöö tagab ühtse visiooni säilitamise.	Vajalik on kõigi meeskonnaliikmete osalemine, et töötuba toimiks plaanipäraselt.
Ülevaade	Ülevaate käigus esitatakse ja inspekteeritakse koos kõikide huvipooltega lahenduse eelseisvat osa. Seda tehakse, et kontrollida, kas lahendus vastab huvigruppide vajadustele, ja saada tagasisidet juba tehtud töö osas.	Võimaldab saada varajast ja asjakohast tagasisidet huvigruppidelt.	Kui osapooli on väga palju, võib tagasiside järelduste tegemiseks olla liiga hajutatud ja laialivalguv.
Tagasivaade	Tagasivaade on meetod protsessi pidevaks parendamiseks, analüüsides, mis sujus hästi ja mida saaks parendada. See annab kõigile tiimiliikmetele võimaluse keskenduda hiljutistele tarnetele.	Võimaldab varajast probleemide käsitlemist ja keskendub protsessi optimeerimisele.	Efektiivne ainult siis, kui tiim rakendab tagasisidet ja liikmed usaldavad üksteist.

2.1.1 Dokumentatsiooni uurimine

Nõuete kogumise protsess algas dokumentatsiooni uurimisega. See tehnika valiti, kuna dokumentatsioon oli Microsoft Dynamics meeskonna poolt olemas ning see võimaldas olemasolevaid veebiteenuseid sügavuti mõista.

Dokumentatsiooni uurimise käigus selgus, millised veebiteenused on väljatöötatava integratsiooni loomisel aktuaalsed ning millised mitte. Lisaks kaardistati dokumentatsiooni uurimise käigus, millised veebiteenuste kaudu hangitavad parameetrid on integratsiooni loomisele olulised ja millised mitte ning millised parameetrid on toimiva lahenduse jaoks puudu.

2.1.2 Nõuete kogumise tööseminarid

Töö autor valis nõuete kogumise meetodina tööseminaride tegemise. Seminare korraldati eri sidusrühmadega nii individuaalselt kui ka kollektiivselt, et tagada ühine visioon ja mõistmine.

Tööseminaride käigus keskenduti konkreetsete rakendatavate nõuete kaardistamisele ja valideerimisele. Seminare iseloomustasid interaktiivsed sessioonid, kus kõik osapooled said aktiivselt osaleda, esitades küsimusi, jagades mõtteid ja pakkudes lahendusi.

Kõikide seminaride jooksul dokumenteeriti aset leidnud asjakohased arutelud, ideed ja ettepanekud, et vältida olulise teabe kadu.

Pärast tööseminari analüüsiti kogutud teavet, tuvastati projektile seatud nõuded ja määratleti järgmised sammud. Tööseminarid andsid väärtuslikku infot nõuete määratlemiseks ning aitasid kaasata eri huvigruppe, tagamaks, et kõiki olulisi vajadusi oleks arvesse võetud.

2.1.3 Intervjuud

Autor valis nõuete kogumise lõppetapiks intervjuude tegemise asjakohaste spetsialistidega, kes olid samuti lahenduse lõppkasutajad. Valitud intervjuu stiil oli vaba vorm, mis tähendab, et formaat ega küsimuste järjestus ei olnud jäigalt määratud, võimaldades paindlikkust ja spontaansust vestlustes. Küsimused kujunesid ja arenesid dünaamiliselt, tuginedes intervjueeritavate vastustele ja tekkivatele alateemadele.

Selline meetod valiti eelkõige seetõttu, et see võimaldab sügavamat mõistmist ja paremat sisulist arusaamist intervjueeritavate vajadustest, kogemustest ja väljakutsetest. Vabas vormis intervjuud pakkusid intervjueeritavatele võimalust jagada oma mõtteid ja kogemusi avatult ning ausalt, ilma et nad tunneksid end piiratuna või

suunatuna. Samuti võimaldas meetod küsimusi ja teemasid reaalajas kohandada, et tekkivaid ideid ja probleeme paremini mõista ning uurida.

Intervjuude käigus keskenduti lisaks tehnilistele ja funktsionaalsetele nõuetele ka kasutajate kogemustele, eelistustele ja potentsiaalsetele takistustele, mis võivad mõjutada lahenduse lõplikku vastuvõtmist ning kasutamist. Erilist tähelepanu pöörati ka kasutajate ootustele ja soovidele seoses uue süsteemiga, tagamaks, et lõplik lahendus ei oleks mitte ainult tehniliselt täpne, vaid ka kasutajasõbralik ja väärtust pakkuv.

2.2 Tarkvaraarenduse metoodikad

Tarkvaraarenduse metoodikad on reeglid või juhised, mida kasutatakse tarkvaraarenduse protsessi juhtimiseks ja struktureerimiseks. [6]

Kaasaegses tarkvaraarenduses eristuvad kaks peamist arendusmeetodite kategooriat – konventsionaalne arendus ja välearendus –, mis mõlemad hargnevad veel mitmeks spetsiifilisemaks lähenemisviisiks. [7]

Tabelis 2.2 on kirjeldatud tarkvaraarenduse eri metoodikaid. Tabel ei ole täielik, vaid selles on kirjeldatud vaid enimkasutatavaid metoodikaid.

Tabel 2.2 Enimkasutatud tarkvaraarenduse metoodikad [6][7][8][9][10][11][12]

Metoodika	**Kirjeldus**	**Tugevused**	**Piirangud**
Koskmudel	Koskmudeli meetod on lineaarne – iga järgmine etapp algab alles pärast eelmise lõppu. See on üks vanimaid metoodikaid, mida esmakordselt kirjeldas 1970. aastal dr Winston W. Royce.	Selged etapid, mida on lihtne mõista ja kasutada. Sobib hästi väiksemate projektide arendamiseks, mille nõuded on väga selged.	Ei ole paindlik muutuste suhtes, mis arendusprotsessi käigus tekkida võivad. Projekti õnnestumiseks on oluline teha õigeid otsuseid varajases faasis.
Agiilne	Agiilne metoodika keskendub iteratiivsele arendusele, kus projekt jagatakse väikesteks osadeks ja seda arendatakse tsükliliselt. Agiilne metoodika sai alguse 2001. aastal, kui 17 tarkvaraarendajat koostasid Agiilse manifesti.	Suur paindlikkus, klient on arendusprotsessi kaasatud, kiirem toote turuletoomine.	Võib olla raske projekti ulatust kontrollida, vajab kõrgelt kvalifitseeritud meeskonda.
Scrum	Scrum on agiilse tarkvaraarenduse raamistik, mis keskendub lühiajalistele arendustsüklitele, mida nimetatakse sprintideks. Scrumi metoodika arendasid 1990ndate alguses välja Ken Schwaber ja Jeff Sutherland.	Suurepärane meeskonnatöö ja koostöö, kiire reageerimine muutustele, toote pidev parandamine.	Võib olla keeruline rakendada suuremates organisatsiooni-des, vajab kogenud Scrum Masterit.
Lean	Lean keskendub klientide jaoks väärtuse loomisele, minimeerides samal ajal raiskamist. Lean pärineb Toyota tootmissüsteemist ja seda hakati tarkvaraarenduses kasutama 2000ndate alguses.	Raiskamise minimeerimine, klientide väärtuse maksimeerimine, pidev parendamine.	Võib olla keeruline mõista ja rakendada, vajab organisatsiooni-list muutust.
V-mudel	V-mudel keskendub arendusprotsessi iga etapi paralleelsele testimisetapile. V-mudel töötati välja 1980ndatel ja on tuntud ka kui Saksamaa föderaalvalitsuse süsteemiarenduse standard.	Selged etapid, paralleelne testimine arendusprotsessi ajal.	Ei ole paindlik muutuste suhtes, riskid avastatakse hilja.
XP	Agiilne metoodika, mis loodi, et parandada tarkvaraarendusprojektide kvaliteeti. Keskendub tehnilisele täiuslikkusele ja headele disainipraktikatele.	Klient on kaasatud arendusprotsessi, paarprogrammeerimine, pidev testimine ja integreerimine, mis tagab arenduse kvaliteedi.	Eeldab kõrgelt kvalifitseeritud ja omavahel hästi toimivaid meeskondi. On kõige tõhusam väiksemate meeskondade ja projektide puhul.
Kanban	Visuaalne süsteem, mis kasutab kaarte või muid märgiseid, et jälgida töövoogu.	Paindlik, vähendab raiskamist, parandab töövoogu.	Ei ole struktureeritud, võib olla keeruline suurte meeskondade jaoks.

Tarkvaraarenduse metoodika valik sõltub mitmest tegurist, sealhulgas projekti laadist, meeskonna suurusest ja oskustest, huvigruppide vajadustest ja organisatsioonilisest kontekstist. [7]

2.2.1 Agiilne arendus Kanbani meetodil

Ettevõttes, kus töö autor töötab, toimub välearendus Kanbani metoodikat kasutades.

Agiilne tarkvaraarendus ehk välearendus on tarkvaraarenduse lähenemisviis, mille põhiline eesmärk on olla paindlik ja reageerida kiiresti muutuvatele nõuetele ning olukordadele. Välearenduse protsessis jagatakse arendusprotsess lühikesteks iteratsioonideks, mille käigus toimub tarkvara astmeline arendus ning mille tulemusena valmivad toimivad tarkvaraversioonid. [10]

Kanbani tehnika osutus ettevõttes valituks mitmel põhjusel.

Kanban pakub hea visuaalse ülevaate töövoost, näidates selgelt, millised ülesanded on tehtud, millised on töös ja millised alles ootavad arendustööde algust. See selgus aitab tuvastada pudelikaelu ja tõhusust takistavaid elemente. [13]

Lisaks seab Kanban piirangud sellele, kui palju tööülesandeid saab teatud etapis korraga olla, tagades, et korraga ei võeta liiga palju tööd ette, parandades sellega keskendumist ning vähendades ülekoormust. Piirates korraga töös olevate ülesannete arvu, julgustatakse pooleliolevate tööde lõpetamist enne uue alustamist, mis omakorda vähendab ressursside raiskamist, elimineerides mitme ülesande vahel hüppamise. [13]

Kanbani meetodi puhul tõstetakse ülesandeid Kanbani tahvlil ülespoole vastavalt nende prioriteetidele, mis tagab, et kõige olulisemad ülesanded saavad esmajärjekorras tähelepanu ning vajalikke ressursse. Keskendudes vaid kõige vajalikumatele töödele, väheneb järjekordselt ressursside raiskamine, sest meeskonnad ei raiska aega ega energiat madala prioriteediga ülesannete peale. [13]

Kokkuvõtlikult, kuna ettevõte soovib olla paindlik, kohaneda kiiresti muutustega ning optimeerida ressursse, on agiilse arenduse ja Kanbani kombinatsioon kasulik lähenemine.

2.3 Nõuete prioriseerimise metoodikad

Nõuete prioriseerimine on kriitiline samm tarkvaraarenduse protsessis, tagamaks, et kõige olulisemad ja kriitilisemad nõuded rahuldatakse esmajärjekorras. [14]

Prioriseerimise protsessi käigus hinnatakse ja järjestatakse süsteemi, tarkvara või toote arendamiseks esitatud nõuded selle põhjal, kui vajalikud on need projekti edukaks toimimiseks. [14]

Tabelis 2.3 on kirjeldatud nõuete prioriseerimise eri tehnikaid. Tabel ei ole täielik, vaid selles on kirjeldatud populaarsemaid metoodikaid.

Tabel 2.3 Nõuete prioriseerimise metoodikad [15][16][17][18]

Metoodika	**Kirjeldus**	**Tugevused**	**Piirangud**
MoSCoW	Nõuded jagatakse nelja kategooriasse, mis iseloomustavad nõude olulisust ja vajalikkust süsteemi edukaks toimimiseks.	Lihtne ja kiire, selgetesse kategooriatesse jagamine aitab kaasa otsustamisele.	Puudub detailsus, võib tekitada vaidlusi nõuete kategooriatesse paigutamisel.
Kano mudel	Meetod, mis keskendub sellele, millised toote või teenuse omadused mõjutavad kõige rohkem kliendi rahulolu.	Keskendub kliendi rahulolule, aitab teha vahet baasootustel ja konkurentsieelist pakkuvatel nõuetel.	Nõuab laialdast arusaama kliendipsühholoogiast, tehnilised nõuded võivad jääda selgumata.
$100 test	Meetod, mille käigus antakse igale huvipoolele kujuteldavad 100 dollarit ja palutakse neil see summa „investeerida" eri nõuetesse, jagades oma 100 dollarit nende vahel vastavalt nende isiklikule arvamusele nõuete prioriteetide kohta.	Lihtne ja intuitiivne meetod, mis sunnib osalejaid tegema reaalsete kaalutlustega valikuid. Aitab selgitada, millised nõuded on eri huvipooltele kõige olulisemad.	Ei pruugi olla piisavalt täpne, kui raha on vaja jagada suure hulga või väga detailsete nõuete vahel. Raha jaotamise kontseptsioon võib mõne osaleja jaoks olla liialt kunstlik ning mitte peegeldada tegelikke ärivajadusi.
Kolmetasandiline skaala	Hinnatakse nõuete olulisust kolmel tasandil: kõrge, keskmine, madal.	Lihtne mõista ja rakendada, ei nõua keerukaid vahendeid ega palju aega. Kasulik suurte nõuete	Kategooriad võivad olla liiga üldised, jättes tähelepanuta nõuete spetsiifilisemad erisused. Samuti võib

Metoodika	Kirjeldus	Tugevused	Piirangud
		kogumite puhul kiireks sorteerimiseks. Kategooriatesse jagamine võimaldab kiiresti teha järeldusi prioriteetide kohta.	prioriteedi määramine olla subjektiivne, mis ei anna selget ülevaadet reaalsetest vajadustest.
Analüütiline hierarhiate meetod	Matemaatiline meetod keerukate otsustusprobleemide süstemaatiliseks lahendamiseks, mille käigus jaotatakse probleem hierarhiateks ja võrreldakse prioriteedi määramiseks nõudeid paarikaupa.	Struktureeritud ja süstemaatiline lähenemine, mis on väga tõhus keerukate ja mitmetahuliste otsustuskriteeriumite prioriseerimise puhul.	Nõuab õigeks implementeerimiseks kõrget ekspertiisi ja võib olla väga ajamahukas.

2.3.1 Nõuete prioriseerimine MoSCoWi meetodil

Nõuete prioriseerimiseks kasutati MoSCoWi prioriseerimise meetodit, mis on populaarne tehnika projekti nõuete järjestamiseks eesmärgiga maksimeerida toodetavat väärtust.

MoSCoW on akronüüm, mis kirjeldab prioriseerimise eri kategooriaid (*Must have, Should have, Could have, Won't have – MoSCoW*). [14]

MoSCoWi meetodit järgides jagatakse esitatud nõuded nelja kategooriasse (vt tabel 2.4):

Tabel 2.4 Prioriteetide kategooriad [14]

Kategooria	Tähendus
Must have (peab olema)	Esmaoluline nõue, mis tuleb täita, et lahendus saaks toimida
Should have (peaks olema)	Oluline nõue, mis on vajalik, kuid ilma milleta saab lahendus siiski töötada
Could have (võiks olla)	Lisaväärtust pakkuv nõue, mida lahendus võiks sisaldada, kuid mis ei ole vajalik

Kategooria	**Tähendus**
Won't have (ei pea olema)	Ebaoluline nõue, mida ei ole vaja realiseeritavasse projekti kaasata

Iga nõuet tuleks hinnata vastavalt etteantud kategooriatele, et määrata selle olulisus. Kuna projektis esitatud nõuete elluviimise juures mängib olulist rolli ajakava, siis ajalise ressursi muutumisel võib olla vajalik nõuete hindamise kordamine. [14]

3 LAHENDUS

3.1 Kasutatud tööriistad

Selles alapeatükis kirjeldatakse tööriistasid, mida käsitletava integratsiooni loomisel kasutati.

3.1.1 Jira

Jira tarkvara on Atlassiani välja töötatud tööriist, mida kasutatakse projektide juhtimiseks ja meeskondlikuks tööks. Jira abil saavad meeskonnad organiseerida, seirata ja luua tarkvaraprojekte. [19]

Siin lõputöös rakendati Jira platvormi, et kavandada ja jälgida arendustegevusi. Lõputöö raames loodi Jira keskkonnas eepos, mille alla koondusid kõik konkreetse projekti jaoks olulised arendusülesanded, tagades nii struktureeritud ja süsteemse lähenemise projekti loomiseks.

3.1.2 Confluence

Confluence on Atlassiani välja töötatud mitmekülgne platvorm, mida kasutatakse meeskondade sisemiseks teabejagamiseks ja dokumentatsiooni loomiseks ning haldamiseks. [20]

Lõputöö raames kasutati Confluence'i platvormi projekti planeerimisel ja dokumenteerimisel, mis aitas kogu meeskonnal projekti nõuete ning arengutega kursis püsida.

3.1.3 Slack

Slack on populaarne meeskondade suhtlustööriist, mis on loodud e-kirjade asendamiseks, pakkudes platvormi reaalajas saadetavate ja säilivate sõnumite jaoks. Selle eesmärk on suurendada produktiivsust, julgustades avatud ja tõhusat suhtlust. [21]

Lõputöö käigus kasutati Slacki, et tagada töö autori ja arendustiimi vahel tõhus suhtlus, kui tekkis vajadus konsulteerida või arutleda integratsiooni nõuete üle.

3.1.4 Teams

Microsoft Teams on Microsoft Office 365 paketti kuuluv koostööplatvorm, mis toimib meeskondade ja organisatsioonide suhtlus- ning koostöökeskkonnana. See loob kasutajatele võimaluse suhtlemiseks, koosolekute pidamiseks ja failide jagamiseks ühtse rakenduse kaudu. [22]

Lõputöö raames kasutati Microsoft Teamsi platvormi sidusrühmadega kohtumiste pidamiseks ja koostöö tegemiseks.

3.2 Ärianalüüsi tulemused

Ärianalüüs keskendus ärisektori vajaduste kindlakstegemisele.

Ärianalüüsi protsess sai alguse Microsoft Dynamicsi meeskonna esitatud dokumentatsiooni läbivaatamisest. See kirjeldab mitmeid NAV-i veebiteenuseid, mida on võimalik kasutada parkimisrakenduse ja NAV-i vahelise integratsiooni loomiseks.

Järgmisena tuvastati kõik olulised sidusrühmad. Nendega tehti vabas vormis intervjuud, mille kaudu hinnati praegust olukorda ja selgitati välja ärisektori ootused ning projekti ulatus. Intervjuude põhieesmärk oli koguda teavet ärinõuete kujundamiseks. Vajaduse korral täpsustati küsimusi, et saada ülevaade ka funktsionaalsetest ja mittefunktsionaalsetest nõuetest.

Selleks, et saavutada sügavamat mõistmist ja saada detailsemat teavet, kasutati järgnevas ärianalüüsi etapis tööseminare, mis võimaldasid nõudeid veelgi põhjalikumalt kaardistada ja selgitada.

Ärianalüüsi tulemusena kaardistati tabelis 3.1 kirjeldatud ärinõuded. Need on kirjutatud juhuslikus järjekorras ega määra nõude kaardistamise hetke ega olulisust.

Tabel 3.1 Ärinõuded

Ärinõude tähis	**Ärinõude kirjeldus**
ÄN01	Peab olema võimalik müüa ühe-, kolme- ja kaheteistkuulisi abonemente.
ÄN02	Peab olema võimalik müüa NAV-is seadistatud abonemente.
ÄN03	Abonemendi eest tasumine peab olema võimalik kliendi valitud intervalliga, kas iga ühe, kolme, kuue või kaheteistkümne kuu tagant.
ÄN04	Peab olema võimalik müüa eri ajapiirangutega abonemente.
ÄN05	Peab olema võimalik müüa erihindadega abonemente vastavalt kliendi kategooriale.
ÄN06	Abonemendi esimese kuu eest tasumine peab olema arvutatud *pro rata* alusel.
ÄN07	Abonementi peab saama tühistada ühekuulise etteteatamisega.
ÄN08	Ühte abonementi peavad saama kasutada kuni kolm sõidukit, kuid mitte samaaegselt.
ÄN09	Esialgne integratsioon peab toimima vaid Belgia turu toodete müümisega, kuid edaspidi peab sama integratsioon võimaldama parkimisrakenduse kaudu müüa ka teiste Euroopa riikide tooteid.

Ärinõude tähis	Ärinõude kirjeldus
ÄN10	Peab olema võimalik müüa erihindadega abonemente vastavalt sõiduki tüübile.

Ärinõuded on aluseks uue integratsiooni funktsionaalsuse planeerimisel. Seega kirjeldavad need otsest kasu, mida süsteem peab pakkuma.

3.3 Süsteemianalüüsi tulemused

Süsteemianalüüsi raames analüüsiti kasutajate nõudeid, mis põhinesid huvigruppide ootustel.

Süsteemianalüüsi aluseks olid ärianalüüsi järeldused ja intervjuude kaudu saadud andmed süsteemi funktsionaalsuse kavandamiseks.

Arendusprotsesside hajutamise eesmärgil jagati projekti nõuded kahte skoopi: esmane ning hilisem. See otsus tulenes vajadusest toota võimalikult kiiresti toimiv lahendus lõppkasutajate jaoks. Ajalise ressursi piirangute tõttu oli seega vajalik eristada nõudeid, mida oleks vajalik ja võimalik täita juba esimese arendusetapi jooksul, eraldades need hilisemate arendusetappide nõuetest.

Süsteemianalüüsi tulemused on esitatud funktsionaalsete ja mittefunktsionaalsete nõuetena esimese arendusetapi kohta.

Süsteemianalüüs loodi eesmärgiga, et seda saaks kasutada loodava integratsiooni arendamisel. Lisaks rakenduse arendajatele on süsteemianalüüs mõeldud ka eri lahendustega seotud huvigruppidele, selgitamaks süsteemi funktsionaalsust.

3.3.1 Funktsionaalsed nõuded

Funktsionaalsed nõuded määratlevad, mida tarkvara peaks suutma teha. Need on spetsifikatsioonid, mis kirjeldavad konkreetseid funktsioone, mida süsteem peaks suutma teha, et täita seatud eesmärke. [2]

Tabelis 3.2 on kirjeldatud nõuete kogumise käigus kaardistatud arendatava integratsiooni funktsionaalseid nõudeid. Need on kirjutatud suvalises järjekorras ega määra nõude olulisust.

Tabel 3.2 Funktsionaalsed nõuded

Funktsionaalse nõude tähis	Funktsionaalse nõude kirjeldus
FN01	Ühendus parkimisrakenduse taustsüsteemi ja NAV-i vahel.
FN02	Taustsüsteem peab importima NAV-is seadistatud Belgia abonemente.

Funktsionaalse nõude tähis	Funktsionaalse nõude kirjeldus
FN03	Abonementide importimine toimub API kaudu.
FN04	Taustsüsteem uuendab juba imporditud abonementide andmeid, kui need muutuvad NAV-i süsteemis.
FN05	Taustsüsteem võimaldab lõppkasutajal abonemendi tühistamist ühekuulise etteteatamisega.
FN06	Taustsüsteem võimaldab kuni kolmel sõidukil kasutada sama abonementi, kuid mitte samaaegselt.
FN07	Taustsüsteem logib kõik imporditud andmed.
FN08	Taustsüsteem logib kõik toodete andmete muudatused.
FN09	Taustsüsteem logib kõik veateated.
FN10	Taustsüsteem võimaldab valida eri makseperioodide vahel.

3.3.2 Mittefunktsionaalsed nõuded

Mittefunktsionaalsed nõuded, mida vahel nimetatakse ka süsteemi kvaliteediatribuutideks, defineerivad tarkvaraarenduses süsteemi käitumist ja omadusi, kuid mitte selle konkreetset funktsionaalsust. Teisisõnu, need nõuded täpsustavad, kuidas süsteem peaks toimima, mitte mida see peaks täpselt tegema. Mittefunktsionaalsed nõuded on süsteemi disainimisel, arendamisel ja testimisel hädavajalikud, kuna need mõjutavad oluliselt kasutajate kogemust ja süsteemi üldist edukust. [2]

Tabelis 3.3 on kirjeldatud nõuete kogumise käigus kaardistatud arendatava integratsiooni mittefunktsionaalsed nõuded.

Tabel 3.3 Mittefunktsionaalsed nõuded

Mittefunktsionaalse nõude tähis	Mittefunktsionaalse nõude kirjeldus
MFN01	Taustsüsteem peab suutma toime tulla suure hulga andmete importimise ja uuendamisega ilma jõudluse kaotuseta.
MFN02	Integratsioon peab olema stabiilne ja töötama katkestusteta.
MFN03	Kõik andmevahetused API kaudu peavad olema turvatud, et vältida volitamata juurdepääsu või andmete leket.

Mittefunktsionaalse nõude tähis	Mittefunktsionaalse nõude kirjeldus
MFN04	Lõppkasutajale mõeldud funktsioonid, nagu abonemendi ostmine ja tühistamine, peavad olema intuitiivsed ja kasutajasõbralikud.
MFN05	Süsteem peab olema kergesti hooldatav, võimaldades tulevikus muudatusi ja täiendusi.
MFN06	Süsteem peab olema skaleeritav, et toime tulla kasvavate andmemahtudega.

3.4 Nõuete prioriseerimise tulemused

Kogutud nõuded prioriseeriti vastavalt MoSCoWi kategooriatele (vt tabel 3.4).

Tabel 3.4 Prioriseeritud nõuded

Kategooria	Nõude tähis	Kirjeldus
Must have (peab olema)	FN01	Ühendus parkimisrakenduse taustsüsteemi ja NAV-i vahel.
Must have (peab olema)	FN02	Taustsüsteem peab importima NAV-is seadistatud abonemente.
Must have (peab olema)	FN03	Abonementide importimine toimub API kaudu.
Must have (peab olema)	FN04	Taustsüsteem uuendab juba imporditud abonementide andmeid, kui need muutuvad NAV-i süsteemis.
Must have (peab olema)	MFN01	Taustsüsteem peab suutma toime tulla suure hulga andmete importimise ja uuendamisega ilma jõudluse kaotuseta.
Must have (peab olema)	MFN02	Integratsioon peab olema stabiilne ja töötama katkestusteta.
Must have (peab olema)	MFN03	Kõik andmevahetused API kaudu peavad olema turvatud, et vältida volitamata juurdepääsu või andmete leket.
Should have (peaks olema)	FN05	Taustsüsteem võimaldab lõppkasutajal abonemendi tühistamist ühekuulise etteteatamisega.
Should have (peaks olema)	FN06	Taustsüsteem võimaldab kuni kolmel sõidukil kasutada sama abonementi, kuid mitte samaaegselt.

Kategooria	**Nõude tähis**	**Kirjeldus**
Should have (peaks olema)	MFN04	Lõppkasutajale mõeldud funktsioonid, nagu abonemendi ostmine ja tühistamine, peavad olema intuitiivsed ja kasutajasõbralikud.
Should have (peaks olema)	MFN05	Süsteem peab olema kergesti hooldatav, võimaldades tulevikus muudatusi ja täiendusi.
Should have (võiks olla)	MFN06	Süsteem peab olema skaleeritav, et toime tulla kasvavate andmemahtudega.
Should have (võiks olla)	FN10	Taustsüsteem võimaldab valida eri makseperioodide vahel.
Could have (võiks olla)	FN07	Taustsüsteem logib kõik imporditud andmed.
Could have (võiks olla)	FN08	Taustsüsteem logib kõik toodete andmete muudatused.
Could have (võiks olla)	FN09	Taustsüsteem logib kõik veateated.

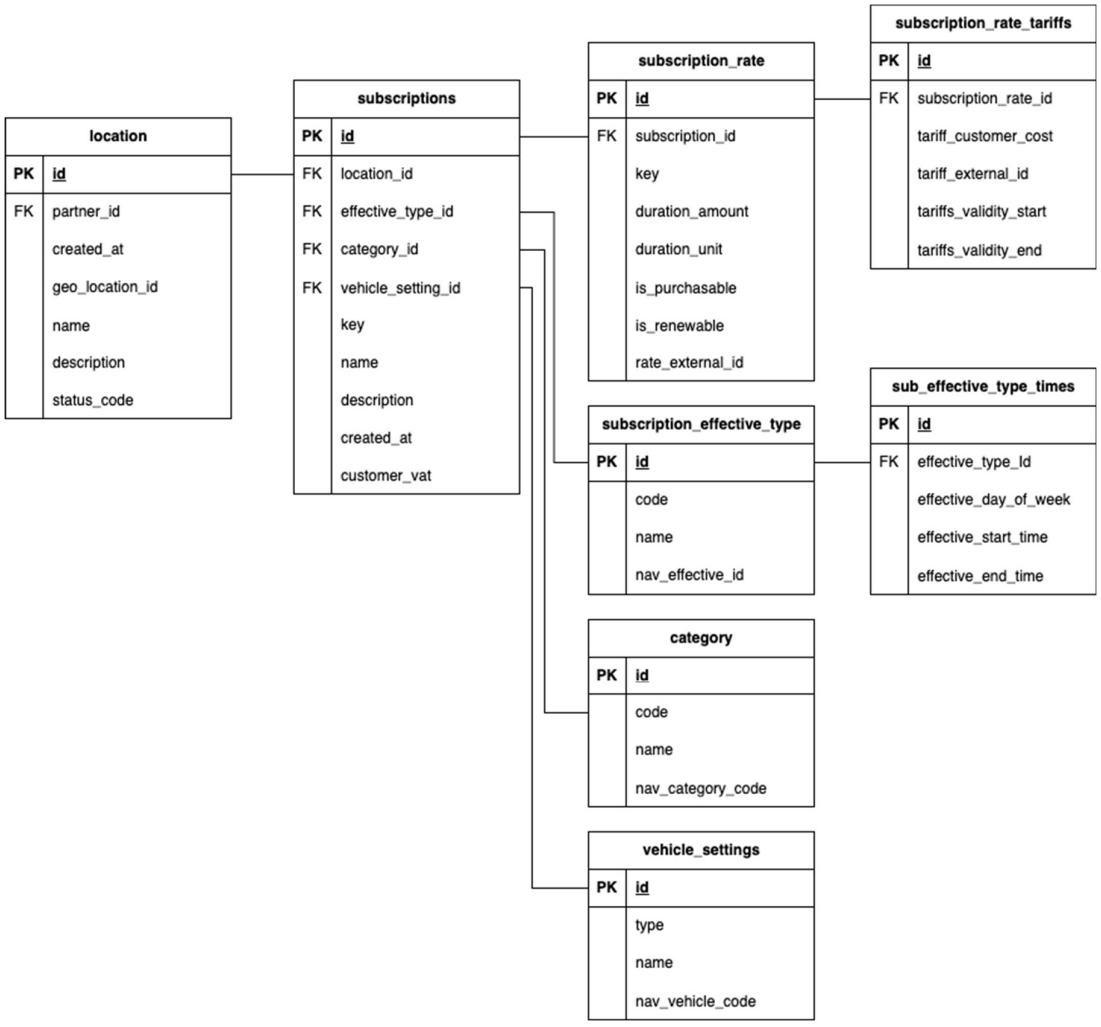
Prioriseerimise käigus ei selgunud ühtegi nõuet, mis kuuluks kategooriasse *Won't have* (ei pea olema).

3.5 Lahenduse tehnilised aspektid

3.5.1 Füüsiline andmemudel

Joonisel 3.1 on kujutatud taustsüsteemi abonementidega seotud lihtsustatud olemi-suhte diagrammi ehk ERD-d. Koostatud diagramm ei ole täielik, vaid hõlmab ainult asjakohaseid tabeleid ja parameetreid.

Mudel koostati, arvestades analüüsi käigus selgunud vajadustega.



Joonis 3.1 Abonementide lihtsustatud ERD (Allikas: autori koostatud)

3.5.2 Navisioni veebiteenuste vastavus taustsüsteemi tabelitega

3.5.2.1 Products

NAV-i veebiteenus Products tagastab NAV-i andmebaasist tabeli Productsi andmed *read-only* kujul.

Veebiteenus Products varustab parkimisrakenduse taustsüsteemi abonemendi definitsiooniga ehk paneb paika müüdava abonemendi baasparameetrid.

Tabelis 3.5 on välja toodud veebiteenuse Products parameetrid, mis on seatud vastavusse taustsüsteemi abonementi defineerivate parameetritega.

Tabel 3.5 Veebiteenuse Products vastusparameetrid

NAV-i parameetri nimi	**Taustsüsteemi vastavus**	**Kommentaar**
ProductCode	subscription_rate.id	NAV-i süsteemis kasutatakse tootekoodi, mis koosneb toote omadusi kirjeldavatest lühenditest. Kuigi tootekood aitab tooteid kirjeldada, määravad toote omadused siiski konkreetsed parameetrid. Tootekoodi kasutatakse raamatupidamisraportites. Näide tootekoodist: A_5D7__07H19A_MM
Description	subscriptions.description	See väli on NAV-i kasutajatele vabatekstiväli, mis tähendab, et selle väärtus sõltub sellest, mis riigi abonemente me NAV-ist impordime. Taustsüsteemis peame abonemendi nimetuse esitama nõnda, et see oleks arusaadav ja kasutatav kõikides rakenduses pakutavates keeltes. Taustsüsteemis tuleb seega abonemendi nimetus genereerida, järgides järgmist mustrit: asukoha nimi, abonemendi kehtivusaeg. Näide: Antwerpen, 24/7
VehicleType	subscriptions.vehicle_setting_id	Defineerib, millistele sõidukitele abonement laieneb. CAR – auto ELC – elektriauto MOTO – mootorratas BIKE – jalgratas PMR – kaubik Kui väli on NAV-is tühi, siis taustsüsteemis on vaikimisi väärtus „auto“.
VehicleTypeName	-	Ei kasutata taustsüsteemis.
EngagementDuration	subscription_rate.duration_unit subscription_rate.duration_amount	M – kuu K – kvartal S – poolaasta J – aasta
EngagementDurationName	-	Ei kasutata taustsüsteemis.
CategoryCode	subscriptions.category_id	Defineerib kliendi, kellel on õigus abonementi osta, profiili. Näiteks: RES – resident COM – kaupmees VIS – külaline Kui väli on NAV-is tühi, siis taustsüsteemis on vaikimisi väärtus „külaline“.
EffectiveTime	subscriptions.effective_type_id	Defineerib, mis päevadel ja kellaaegadel on abonement kasutatav. Defineeritakse koodina. Link NAV-i EffectiveTimesi tabelile.
EffectiveTimeNumberOfDays	-	Ei kasutata taustsüsteemis.
InvoicePeriodicity	-	Defineerib arvete esitamise perioodilisuse. Praegu on taustsüsteemis võimalik arveldada ainult kuuajaliste intervallidega. Esimeses arendusetapis impordime ainult kuu kaupa arveldatavaid abonemente.

3.5.2.2 ProductPrices

NAV-i veebiteenus ProductPrices võimaldab *read-only* ligipääsu NAV-i andmebaasi ProductsPricesi tabelile, milles on defineeritudProducts tabeli eri elementide hinnainfo vastavalt sellele, kus teatud toode kehtib.

Tabelis 3.6 on ProductPricesi vastuste parameetrid pandud vastavusse parkimisrakenduse taustsüsteemi asjakohaste parameetritega.

Tabel 3.6 Veebiteenuse ProductPrices vastusparameetrid

NAV-i parameetri nimi	**Taustsüsteemi vastavus**	**Kommentaar**
ProductCode	subscription_rate.id	Nende kahe parameetri kombinatsioon viitab konkreetsele tootele.
LocationID	subscriptions.location_id	
StartDate	subscription_rate_tariffs.tariffs_validity_start	Defineerib, mis kuupäevast kehtib konkreetne hind. Ajavöönd sõltub serverist, kus hostitakse NAV-i instantsi, mis käitab veebiteenust. Üldiselt on serverid seadistatud UTC-sse.
EndDate	subscription_rate_tariffs.tariffs_validity_end	Defineerib, mis kuupäeval konkreetne hind kehtivuse kaotab.
Price	subscription_rate_tariffs.tariff_customer_cost	Abonemendi kuumaksumus, mille sisse on arvestatud käibemaks.
CurrencyCode	-	Defineerib, millises valuutas abonemendi maksumus on toodud.

NAV-i parameetri nimi	Taustsüsteemi vastavus	Kommentaar
		Ei impordita veebiteenuse kaudu, vaid defineeritakse taustsüsteemis partneri tasemel.
UniqueID	subscription_rate_tariffs.id	Defineerib abonemendi hinnakonteineris oleva konkreetse tariifi. Vajalik, kuna on võimalus, et eri ajavahemikel on samale abonemendile seadistatud eri tariifid.
AllowToSell	subscription_rate.is_purchasable	Defineerib, kas NAV-i toode on parkimisrakenduses müüdav või mitte.
IsRenewable	subscription_rate.is_renewable	Defineerib, kas NAV-i toode on parkimisrakenduses pikendatav või mitte.

3.5.2.3 EffectiveTimes

NAV-i veebiteenus EffectiveTimes võimaldab *read-only* ligipääsu NAV-i andmebaasi EffectiveTimesi tabelile, milles on defineeritudProducts tabeli eri elementide kehtivusajad.

Tabelis 3.7 on EffectiveTimesi vastuste parameetrid pandud vastavusse parkimisrakenduse taustsüsteemi asjakohaste parameetritega.

Tabel 3.7 Veebiteenuse EffectiveTimes vastusparameetrid

NAV-i parameetri nimi	**Taustsüsteemi vastavus**	**Kommentaar**
EffectiveTimeCode	subscriptions.effective_type_id	Link Productsi tabeliga.
DayOfWeek	sub_effective_type_times.effective_day_of_week	Defineerib kehtivusaja nädalapäeva 1 – esmaspäev 2 – teisipäev jne.
StartTime	sub_effective_type_times.effective_start_time	Defineerib kehtivusaja alguskellaaja.
EndTime	sub_effective_type_times.effective_end_time	Defineerib kehtivusaja lõppkellaaja.

3.6 Püstitatud arendusülesanded

Alapeatükis kirjeldatakse esimese arendusetapi raames püstitatud ülesandeid arendustiimile.

3.6.1 Integratsiooni loomine Navisioni veebiteenustega

NAV-i ja parkimisrakenduse taustsüsteemi abonementidega seotud andmemudelite põhiline erinevus seisneb abonemendi kõrgeimas defineerivas tasemes.

NAV-i süsteemis on abonemendi kõrgeim tase toote definitsioon Productsi tabelis. Selles tabelis defineeritakse toote põhilised omadused, mis kehtivad ja on muutumatud igas asukohas, kus toodet müüakse. Hierarhialt järgmised on NAV-is ProductPricesi tabelid, milles defineeritakse, kus, millal ja millise hinnaga Productsi tabelis seadistatud tooted müüakse, ning EffectiveTimesi tabel, kus toimub toote kehtivusaja reeglite määratlemine.

Taustsüsteemi andmemudeli hierarhia tipus on aga asukohta määravad tabelid. Taustsüsteemi loogika järgi on igas asukohas müüdav abonement eraldi toode – see on NAV-i loogikaga vastupidine.

Seega on taustsüsteemi imporditav abonement unikaalselt tuvastatav kahe veebiteenuse, Productsi ja ProductPricesi vastusest saadud parameetrite kombinatsiooniga:

- ProductCode'i parameeter tabelist Products, mis viitab tootele;
- LocationID parameeter tabelist ProductPrices, mis viitab, kus toode kehtib.

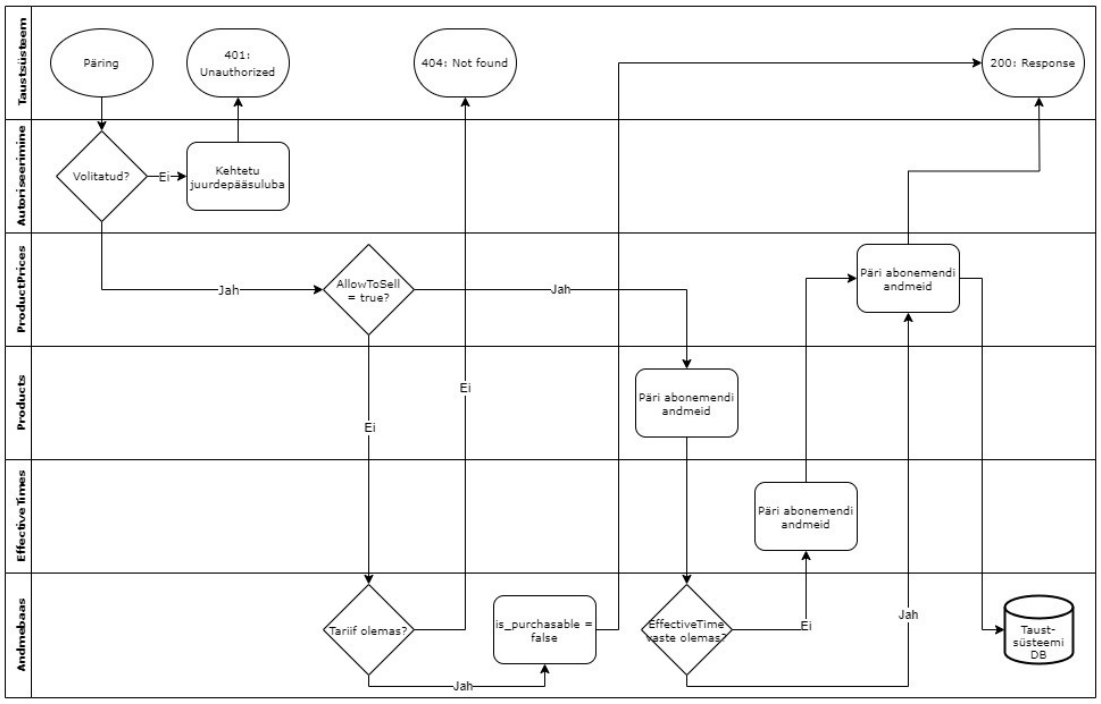
Lisaks on iga abonement seadistatud konkreetse kehtivusajaga, mille reeglid on kättesaadavad veebiteenuse EffectiveTimes vastusest.

Tuleb teha igapäevane sünkroniseerimistoiming, mis impordib kõik asjakohased tooted ja nende andmed Navisionist parkimisrakenduse taustsüsteemi.

Importimisel tuleb järgida järgnevaid reegleid.

- Toote importimise otsus põhineb AllowToSelli parameetril veebiteenuse ProductPrices vastusest.
    - Kui AllowToSell on tõene, siis jätkatakse importimise protsessiga.
    - Kui AllowToSell on väär, ei jätkata toote impordiga.
- Kui on saadaval on imporditav tariif ehk ProductPricesi sissekanne konkreetse asukoha ehk LocationID jaoks, on kõigepealt tarvilik importida toote definitsioon ehk veebiteenuse Products vastusparameetrid selle abonemendi jaoks.
- Kui Productsi tabelis on defineeritud EffectiveTime, mis ei ole taustsüsteemis saadaval, tuleb see importida, kasutades veebiteenust EffectiveTimes.
- Pärast toote loomist taustsüsteemis jätkata veebiteenuse ProductPrices vastuse parameetrite importimisega, et luua tootele taustsüsteemis „subscription_rate" ja „subscription_rate_tariffs" sissekanded.
- Kui imporditud tariifi parameeter AllowToSell muudetakse NAV-is vääraks, kustutatakse tariif taustsüsteemist.
- Kui hinnakonteinerile ei vasta enam ükski tariif tõese väärtusega AllowToSelli parameetriga, seatakse taustsüsteemis hinnakonteineri parameeter „is_purchasable" vääraks.

Joonis 3.2 kirjeldab abonemendi importimise protsessi.



Joonis 3.2 Abonemendi importimise protsess (Allikas: autori koostatud)

Taustsüsteemis on eraldi tabelid abonementide defineerimisel kasutatavate parameetrite jaoks, mis sätestavad kliendi- ja sõiduki kategooria, millele abonement laieneb.

Selleks, et tagada taustsüsteemi ja NAV-i andmete ühilduvus, tuleb luua võimalus kaardistada NAV-is kasutatavaid koode taustsüsteemis kasutatavate kategooriate vastu.

Et seda eesmärki täita, tuleb taustsüsteemi vastavatesse tabelitesse lisada väli NAV-i koodide vastendamiseks, mida peaks saama seadistada taustsüsteemi haldusliidese kaudu.

3.6.2 Logimine

Selleks, et luua ülevaade Navisionist imporditud andmete kohta, tuleb luua järgnev logimine:

- loodud tooted
- loodud hinnakonteinerid
- loodud tariifid
- muudetud tooted
- muudetud hinnakonteinerid
- muudetud tariifid
- veateated, milles on kirjeldatud tekkinud viga ja seotud toote asukohakood

3.7 Võimalikud edasiarendused

Teises arendusetapis on kavas luua taustsüsteemis võimalus siduda üks abonement kuni kolme sõidukiga. Samas saab abonemendi soodustusi samaaegselt kasutada vaid üks sõiduk. Praegu saab taustsüsteem seostada iga abonemendi ainult ühe sõidukiga. Selle funktsionaalsuse laiendamise keerukus seisneb selles, kuidas kohaldada parkimistasu abonemendi alla kuuluvatele sõidukitele olukorras, kus üks sõiduk juba kasutab abonementi samal hetkel parkimiseks.

Järgneva olulise edasiarendusena on kavas muuta paindlikumaks abonementide eest tasumise perioodilisus. Praegu võimaldab süsteem tasuda abonementide eest ainult ühe kuu kaupa, isegi juhul, kui abonemendi kehtivusaeg on pikem. Arenduse eesmärk on võimaldada klientidel valida, kas nad soovivad tasuda oma abonemendi eest iga kuu, kolme kuu, kuue kuu või aasta kaupa.

KOKKUVÕTE

Lõputöö eesmärk oli analüüsida ja kavandada mobiilse parkimisrakenduse integratsioonikihi evolutsiooni arendustööd, mille siht on ühtlustada parkimisteenuste seadistamise protsessid ettevõtte eri asukohariikides.

Lõputöö alguses tutvustati põgusalt parkimisrakenduse emaettevõtet, probleemi tausta ja olemust ning süsteemi hetkeseisu. Kirjeldati lõputöö käsitlusalasse kuuluvat ning töö autori rolli.

Teises peatükis analüüsiti nõuete kogumise, nõuete prioriseerimise ja tarkvaraarenduse eri metoodikaid ning põhjendati valikuid, mis toetasid töö raames äri- ja süsteemianalüüsi ning arendustöid.

Kolmandas peatükis keskenduti lahenduse väljatöötamisele, kirjeldades kasutatud tööriistu, ärianalüüsi ja selle tulemusi. Ärianalüüsi käigus selgunud ärinõuetele tuginedes tehti süsteemianalüüs, mille raames kaardistati ja prioriseeriti süsteemi funktsionaalsed ja mittefunktsionaalsed nõuded. Analüüsi tulemusena jaotati arendustegevused etappideks ning üksikasjalikult määratleti ja kirjeldati esimese etapi arendusülesandeid. Põgusalt anti ka ülevaade planeeritud järgnevatest sammudest integratsiooni parendamiseks.

Autori hinnangul vastab lõputöö püstitatud eesmärkidele ja ettevõtte vajadustele, lihtsustades ja tõhustades parkimisabonementide seadistamise ja müümisega seotud protsesse, luues väärtust nii ettevõttele töö kulutõhususe tõstmisega kui ka kliendile laiema tootevaliku pakkumisega.

SUMMARY

The title of this thesis is "Analysis and design of the evolution of a parking application's integration layer". The thesis's author Merilin Lidmets is a product owner at a company developing a mobile parking application.

The aim of the thesis was to analyse and design the development work for the evolution of the integration layer of the mobile parking application, with the goal of standardising the setup processes of parking products in the different countries where the company operates.

At the beginning of the thesis, a brief introduction was given to the parent company of the parking application. Additionally, the nature of the problem, the scope of the thesis and the role of the author were described. The first chapter also provides a brief overview of the current system.

In the second chapter, the author analysed various methodologies for requirement gathering, prioritisation and software development, justifying the choices that supported the work of analysing the business and system requirements and executing the development within the thesis scope.

The third chapter focused on finding the solution to reach the set goal, describing the tools used, conducting the business analysis and documenting its outcomes. Based on the gathered business requirements, the author performed a system analysis during which the functional and non-functional requirements of the system were mapped and prioritized. As a result of the analysis, the development tasks were divided into two stages. The tasks for the first phase were detailed and a brief overview was also given of the planned next steps for improving the integration.

The author concludes that the thesis met its objectives and the needs of the company by simplifying and enhancing the processes related to setting up and selling parking subscriptions, thus creating value for the company by increasing cost efficiency, and also for the customer by offering a wider range of products.

KASUTATUD KIRJANDUSE LOETELU

[1] Cybernetica AS, „Andmekaitse ja infoturbe leksikon," [Võrgumaterjal]. Available: https://akit.cyber.ee/.

[2] A. Stellman and J. Greene, *Applied Software Project Management*, O'Reilly Media, Inc., 2005.

[3] R. R. Young, *The Requirements Engineering Handbook*, Artech House, Inc., 2004.

[4] R. R. Young, *Effective Requirements Practices*, Addison-Wesley Professional, 2001.

[5] International Institute of Business Analysis, *Agile Extension to the BABOK Guide: Version 2*, Toronto: International Institute of Business Analysis, 2017.

[6] M. Lotz, „Waterfall vs. Agile: Which is the Right Development Methodology for Your Project?," Segue Technologies, [Võrgumaterjal]. Loetud aadressil: https://www.seguetech.com/waterfall-vs-agile-methodology/, 2018. [Kasutatud 11.10.2023]

[7] R. Shaydulin and J. Sybrandt, „To Agile, or not to Agile: A Comparison of Software Development Methodologies," ArXiv, 2017. DOI: 10.48550/arXiv.1704.07469

[8] „Scrumbanfall: An Agile Integration of Scrum and Kanban with Waterfall in Software Engineering," 2020. DOI: 10.35940/ijitee.d1437.029420.

[9] V. Hema, S. Thota, S. N. Kumar, C. Padmaja, C. R. Krishna, and K. Mahender, „Scrum: An Effective Software Development Agile Tool," 2020. DOI: 10.1088/1757-899X/981/2/022060.

[10] K. Beck *et al*., „Manifesto for Agile Software Development," Agile Alliance, [Võrgumaterjal]. Loetud aadressil: https://agilemanifesto.org/, 2001. [Kasutatud 11.10.2023]

[11] K. Schwaber and J. Sutherland, „Scrum Guides," [Võrgumaterjal]. Loetud aadressil: https://scrumguides.org/, 2020. [Kasutatud 11.10.2023]

[12] O. Sohaib, H. R. Solanki, N. Dhaliwa, W. Hussain, and M. Asif, „Integrating design thinking into extreme programming," 2018. DOI: 10.1007/S12652-018-0932-Y

[13] GeeksforGeeks, „Kanban – Agile Methodology," 2022. [Võrgumaterjal]. Loetud aadressil: https://www.geeksforgeeks.org/kanban-agile-methodology/.

[14] K. Model and G. Herzwurm, „Software-Supported Product Backlog Prioritization in Scrum Software Development Projects," 2022. [Võrgumaterjal]. Loetud aadressil: https://dblp.org/rec/conf/icsob/ModelH22.html. [Kasutatud 14.10.2023]

[15] E. Miranda, „MoSCoW Rules: A quantitative exposé," 2021. [Võrgumaterjal]. Loetud aadressil: https://www.researchgate.net/publication/356836488_MoSCoW_Rules_A_quantitative_expose_Accepted_for_presentation_at_XP2022. [Kasutatud 14.10.2023]

[16] J. Moorman, „Leveraging the Kano Model for Optimal Results," UX Magazine, vol. 882, 2012. [Võrgumaterjal]. Loetud aadressil: http://uxmag.com/articles/leveraging-the-kano-model-foroptimal-results. [Kasutatud 15.10.2023]

[17] K. Wiegers, „Five Requirements Prioritization Methods," Medium: Analyst's Corner, 2020. [Võrgumaterjal]. Loetud aadressil: https://medium.com/analysts-corner/five-requirements-prioritization-methods-86f4c5e0433e. [Kasutatud 15.10.2023]

[18] A. Hudaib, R. Masadeh, M. H. Qasem, and A. Alzaqebah, „Requirements Prioritization Techniques Comparison," Modern Applied Science, vol. 12, no. 2, 2018. DOI: 10.5539/mas.v12n2p62

[19] „Welcome to Jira Software," Atlassian. [Võrgumaterjal]. Loetud aadressil: https://www.atlassian.com/software/jira/guides/getting-started/introduction#what-is-jira-software. [Kasutatud 20.10.2023]

[20] „Confluence basics," Atlassian. [Võrgumaterjal]. Loetud aadressil: https://www.atlassian.com/software/confluence/resources/guides/get-started/overview#hosting-options. [Kasutatud 20.10.2023]

[21] „About Slack," Slack. [Võrgumaterjal]. Loetud aadressil: https://slack.com/about. [Kasutatud 20.10.2023]

[22] „Get started with Microsoft Teams," Microsoft Office Support. [Võrgumaterjal]. Loetud aadressil: https://support.microsoft.com/en-us/office/get-started-with-microsoft-teams-b98d533f-118e-4bae-bf44-3df2470c2b12. [Kasutatud 20.10.2023]