TARTU ÜLIKOOL
Arvutiteaduse instituut
Infotehnoloogia mitteinformaatikutele õppekava

Margus Paas

Nõuetetehnika meetodite rakendamine väledas arendusprotsessis ja selle tarbeks kombineeritud töövahendi loomine

Magistritöö (15 EAP)

Juhendajad: Toomas Saarsen, *PhD*

Teet Jagomägi, *MSc*

Tartu 2024

Nõuetetehnika meetodite rakendamine väledas arendusprotsessis ja selle tarbeks kombineeritud töövahendi loomine

Lühikokkuvõte:

Nõuetest sõltub otseselt tarkvara kvaliteet, kuid agiilses ehk väledas arenduses kaheldakse tihti klassikaliste nõuetetehnika meetodite ja tehnikate kasutamise otstarbekuses. Töös püstitati hüpotees, et klassikaliste nõuetetehnika meetodite ja spetsiaalse töövahendi süstemaatiline rakendamine väledas arendusprotsessis suurendab nõuete haldamise efektiivsust, tagab piisava nõuete kvaliteedi ning võimaldab arendada kliendi vajadustele vastava minimaalselt elujõulise toote (MVP) ettenähtud tähtajaks ja realistlikult hinnatud eelarve raames.

Töö eesmärk oli luua lihtne töövahend, mis muudaks nõuete väljaselgitamise protsessi kiiremaks ja efektiivsemaks. Selleks uuriti, kuidas klassikaliste nõuetetehnika meetodite rakendamine mõjutab nõuete haldamise kiirust ja kvaliteeti väledas tarkvaraarenduses. Lisaks hinnati, kas spetsiaalselt loodud töövahend aitab luua kliendi vajadustele vastava minimaalse elujõulise toote (MVP) ettenähtud aja ja eelarve piires.

Loodud töövahend NTKT (nõuetetehnika kombineeritud töövahend) muutis nõuete väljaselgitamise protsessi lihtsamaks, kiiremaks, efektiivsemaks ning võimaldas nõudeid süstematiseerida, kategoriseerida ja filtreerida ning aitas kaasa huvipoolte koostööle COVID-19 tingimustes. Tulemusena valmis terviklik nõuete kogum, mis aitas kaasa toimiva MVP tarnimisele määratud tähtajaks. Kliendi tagasiside MVP-le oli positiivne, mis on oluline tarkvara kvaliteedi hindamise kriteerium. Töövahendit saavad kasutada ka teised ärianalüütikud, et parandada oma töö tootlikkust ja kvaliteeti.

Võtmesõnad:

Tarkvaratehnika, nõuetetehnika, nõuded, ärianalüüs, ärianalüütik, huvipooled, läbirääkimine, prioritiseerimine

CERCS: P170 Arvutiteadus, arvutusmeetodid, süsteemid, juhtimine (automaatjuhtimisteooria)

Application of Requirements Engineering Methods in Agile Development Process and Creation of a Combined Tool for This Purpose

Abstract:

The quality of software directly depends on its requirements, but in agile development, the use of classical requirements engineering methods and techniques is often questioned. This thesis hypothesizes that the systematic application of classical requirements engineering methods and a specialized tool in agile development processes increases the efficiency of requirements management, ensures sufficient quality of requirements, and allows for the development of a minimum viable product (MVP) that meets the client's needs within the specified timeframe and realistically estimated budget. The aim of the work was to create a simple tool that would make the process of identifying requirements faster and more efficient. To achieve this, the study examined how the application of classical requirements engineering methods affects the speed and quality of requirements management in agile software development. Additionally, it assessed whether a specially created tool helps to develop an MVP that meets the client's needs within the specified time and budget. The designed tool, RECT (Requirements Engineering Combined Tool), made the process of identifying requirements simpler, faster, and more efficient. It allowed for the systematization, categorization, and filtering of requirements and facilitated stakeholder collaboration under COVID-19 conditions. As a result, a comprehensive set of requirements was developed, which contributed to the delivery of a functional MVP within the specified timeframe. The client's feedback on the MVP was positive, which is an important criterion for assessing software quality. Other business analysts can also use the tool to improve their productivity and quality of work.

Keywords: Software engineering, requirements engineering, requirements, business analysis, business analyst, development, stakeholders, negotiations, prioritization

CERCS: P170 Computer science, numerical analysis, systems, control

Sisukord

1 Sissejuhatus

Tarkvaratehnikat käsitlevad klassikalised raamatud ja artiklid on enamasti üksmeelel - tarkvara kvaliteet sõltub nõuetest. Samas kahtlevad väleda (agiilse) arendusprotsessi, eriti Scrumi, pooldajad klassikaliste nõuetetehnika meetodite ja tehnikate kasutamise otstarbekuses. Väle metoodika väidab, et see ületab traditsioonilise nõuetetehnika piirangud, keskendudes kasutajate jaoks väärtuse maksimeerimisele, ühendades toote avastamise ja tarnimise ning tagades tõelise läbipaistvuse reaalsete tarkvaratoodete demonstreerimise kaudu, mis on loodud pidevas dialoogis huvipooltega. (Jocham 2024) Teisisõnu tähendab agiilne lähenemine omaduste soovilogi (*backlog*) kasutamist, järkjärgulist tarnet ja kliendi tagasiside põhjal muudatuste ning täienduste tegemist. Toote vastavus nõuetele ja kliendi vajadustele selgub pärast tarnet. Kui toode ei vasta ootustele, on see agiilse lähenemise seisukohast positiivne, sest järgmises sprindis on võimalik toodet veelgi paremaks muuta.

Käesoleva magistritöö teema valis autor seetõttu, et töötab IT ettevõttes ärianalüütikuna ning puutub oma igapäevatöös pidevalt kokku tarkvaranõuetega seonduvate probleemide ning võimalustega agiilses arendusprotsessis. Seistes väljakutse ees, kus suurkorporatsioonist klient soovis üheksa kuu jooksul turule tuua keeruka tarkvaratoote, mis pidi asendama eelmise, tuntud Silicon Valley ettevõtte loodud ja hallatava lahenduse, käitlemaks kümnetesse miljonitesse dollaritesse ulatuvaid rahavoogusid, otsustati arenduse algfaasis läbi viia klassikaline nõuete analüüsi protsess.

Autor otsib töös vastuseid küsimustele, kuidas mõjutab klassikaliste nõuetetehnika meetodite rakendamine nõuete haldamise kiirust ja kvaliteeti väledas tarkvaraarenduses ning kas spetsiaalselt selleks loodud töövahend aitab luua kliendi vajadustele vastava minimaalse elujõulise toote (*minimum viable product,* MVP) ettenähtud aja ja kokkulepitud eelarve piires.

Töö teoreetilises osas annab autor ülevaate tarkvara elutsüklist, nõuetetehnika põhimõistetest, protsessi etappidest, esinevatest probleemidest ja poleemikast seoses nõuetetehnika kasutamise vajaduse ning võimalustega agiilses arendusprotsessis.

Empiirilises osas antakse ülevaade fiktiivsest tarkvaralahendusest, mille aluseks on päriselt teostatud tarkvaraprojekt, kirjeldatakse nõuete väljaselgitamise ja analüüsi protsessi selle projekti raames, kehtestatakse nõuded töövahendile, mis aitaks kiiresti ja efektiivselt välja selgitada MVP nõuete komplekti, luuakse töövahend neid nõudeid arvestades ning analüüsitakse selle kasutamise tulemuslikkust.

Autori igapäevatöö toimub inglise keeles ja peamiselt ingliskeelne on ka nõuete analüüsi temaatikaga tegelev kirjandus. Sellele vaatamata otsustas autor kirjutada töö eesti keeles, et kasutada ja rikastada eestikeelset IT-sõnavara. Samuti oleks lihtsam olnud taotleda kinnist kaitsmist, sest siis oleks näitena (töö sisulises osas) saanud kasutada päris tarkvaralahendust, mille nõuete väljaselgitamise ning analüüsiga autor tegeles ning mis päris kliendile tarniti. Autor otsustas siiski avaliku kaitsmise kasuks, et töö oleks avalikult kättesaadav ja selle tulemused taaskasutatavad ka teiste ärianalüütikute poolt.

Selleks, et avalik kaitsmine toimuda saaks, kasutab autor väljamõeldud klienti, väljamõeldud tarkvaralahendust ning fiktiivseid nõudeid. Seega ei ole kirjeldatud lahendus päriselus kasutatav ja tarkvaralahendusena ei pruugi ootuspäraselt toimida. Autor peab oluliseks kasutatud töövõtete ja lahenduste kirjeldamist näitena Infotehnoloogia mitteinformaatikutele õppekava läbimisel omandatud oskusest.

Allikate sisu tõlkimiseks eesti keelde ja sõnastuse lihvimiseks ning valideerimiseks on kasutatud Perplexity AI-s (Perplexity AI) mudelit Claude 3 Opus (www.anthropic.com/news/claude-3-family). Kirjanduse ja viidete otsimiseks ning sisu genereerimiseks tehisaru abi kasutatud ei ole.

2 Ülevaade nõuetetehnikast

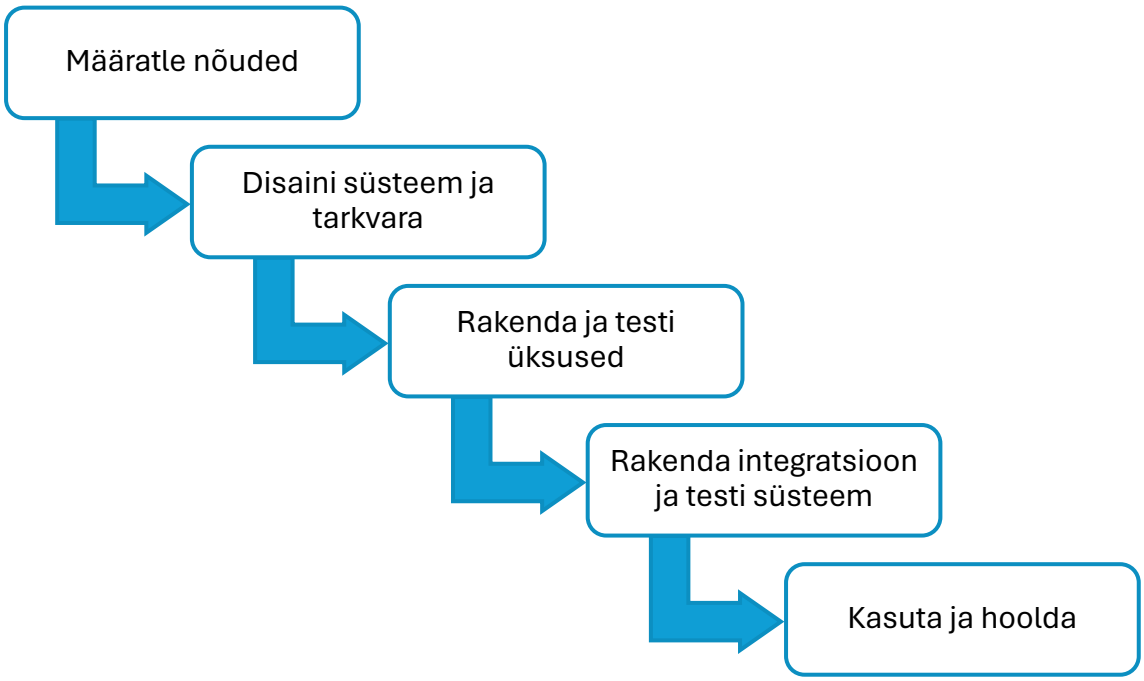
Käesolevas peatükis antakse ülevaade tarkvara elutsüklist, väleda tarkvaraarenduse erisustest nõuete kontekstis, nõuetetehnika protsessi etappidest, nõuetega esinevatest probleemidest ja heade nõuete omadustest. Autor püüab anda tervikliku, kuid kompaktse eestikeelse ülevaate nõuetetehnika raamistikust, meetoditest, protsessidest ja võimalustest, et selle põhjal tekiks arusaamine nõuetetehnikast ka neil, kes sellega igapäevaselt kokku ei puutu. Laiendatult käsitletakse neid nõuetetehnika aspekte, mis on sisendiks praktilisele osale.

2.1 Tarkvara elutsükkel

1960ndate keskel mõisteti mitmel pool vajadust rangema lähenemisviisi järele tarkvaraarenduses. Kuigi termini "tarkvaratehnika" esmakasutuse üle on arutelud, kasutas seda terminit kindlasti Margaret Hamilton, kes oli Apollo programmi juhtivinsener ja vastutas pardatarkvara arenduse eest umbes aastal 1966. Esimene tarkvaratehnika konverents toimus 1968. aastal NATO toetusel ja sellega sündis uus distsipliin.

2.1.1 Koskmudel

Varasemast distsiplineerituma lähenemisviisi aluseks tarkvaraarenduses oli pigem juhtimis- ja organisatsiooniline kui tehniline aspekt. Suuremahulisi projekte arendati etappide kaupa ja 1970. aastal kasutati seda faasilist mudelit (joonis 1) nn tarkvara elutsükli mudeli alusena.

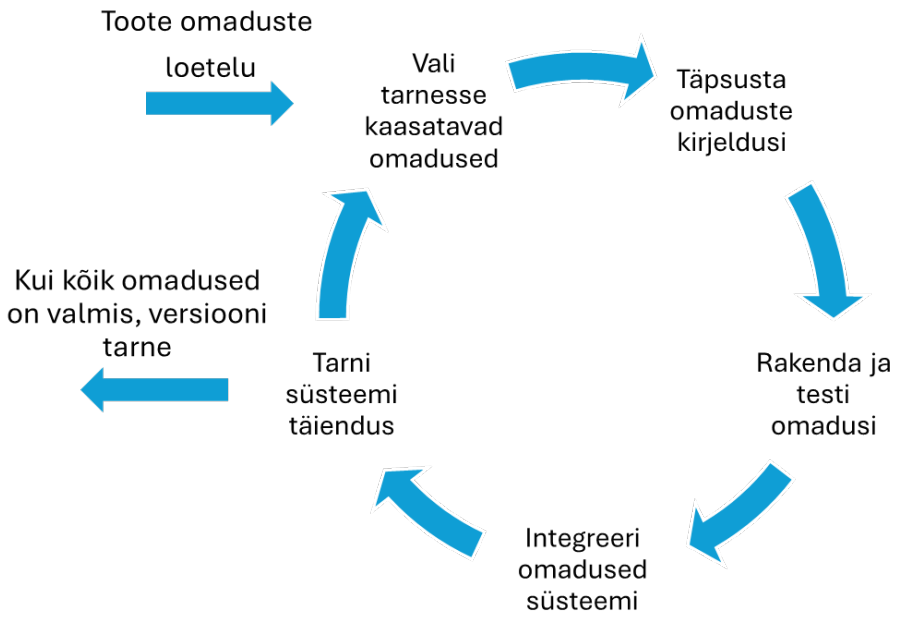


Joonis 1. Tarkvara elutsükkel (koskmudel), autori tõlge ja vormistus (Sommerville 2016)

Elutsükli mudeli juhtimisidee oli progressipõhine, liikudes nõuete faasist mudeli tipus juurutamise faasini allosas (tarkvara kasutamine ja hooldus on enamasti eraldiseisev osa arendusprotsessist). Iga eelnev faas tuleb lõpetada enne järgmise faasi alustamist. Näiteks tarkvara nõuded peavad olema lõplikult paigas enne disainiprotsessi algust. Töö projektiga liigub mudelis astmelisel ülevalt alla, mistõttu nimetatakse seda mudelit mõnikord "koskmudeliks" (*Waterfall*). Selline struktureeritud lähenemine oli eelkõige mõistlik riistvara arendamisel, kuna tol ajal oli riistvara ühiku muutmine pärast selle valmistamist väga kulukas. Seetõttu pidid riistvaranõuded olema väga selged ja enne arendust tuli põhjalikult analüüsida ning kontrollida selle disaini. Tarkvaraarenduses sama lähenemine hästi ei tööta. Tarkvara üks omadusi on muudetavus – seda saab muuta nii arendusprotsessis kui ka pärast tarnimist. Kui nõuded muutuvad või ilmnevad uued nõuded, on mõnikord võimalik neid tarkvaras arvestada. (Sommerville 2016)

2.1.2 Väle mudel

„Väle“ ehk agiilne tarkvaraarenduse mudel on lihtsustatud tarkvaraarenduse meetod, mis võimaldab projekti kulgu pidevalt hinnata ja kohandada kogu arendusprotsessi vältel. Keskendutakse osaliste lahenduste kiirele ja iteratiivsele tarnimisele ning tagasisidet kasutatakse järgmiste sammude korrigeerimiseks. Väle meetod püüab olla kliendi vajadustele reageerides paindlikum kui traditsioonilised meetodid ja pooldajad usuvad, et see viib kokkuvõttes tarkvara kõrgema kvaliteedini, kiirema turule jõudmiseni ning suurema kliendirahuloluni. (O’Regan 2017) Kõik väledad meetodid (joonis 2) põhinevad järkjärgulisel arendusel ja tarnel.



Joonis 2. Järkjärguline (väle) arendus, autori tõlge ja vormistus (Sommerville 2016)

Tarkvara arendatakse järk-järgult ja ühe etapi arendamine võib alata enne, kui teised on täielikult määratletud. See lähenemine oli väleda mudeli aluseks ja on nüüd standardne lähenemine kõikidele tarkvaratehnika tüüpidele.
Kuigi koskmudelit peetakse vananenuks ja tänapäeval on enamik arendusi väledad, on arendustegevused, mis on toodud koskmudelis, fundamentaalsed kõigile tarkvaratehnikatele ja kasutatavad tarkvaratoodete ja tarkvaratehnikate võrdlemiseks. Näiteks kindel nõuetetehnika on üha enam tunnustatud kui võti kvaliteetse, õigeaegse ja eelarve piiresse jääva tarkvaralahenduse tarnimiseks. (Laplante 2017)

2.2 Nõuetetehnika põhimõisted

Nõue (*requirement*) on: (1) Tingimus või võimekus, mida kasutaja vajab probleemi lahendamiseks või eesmärgi saavutamiseks. (2) Tingimus või võimekus, mis peab olema täidetud või süsteemil või süsteemi komponendil olema, et rahuldada lepingut, standardit, spetsifikatsiooni või muid ametlikult kehtestatud dokumente. (3) Tingimuse või võimekuse dokumenteeritud esitus nagu punktides (1) või (2). (Pohl 2010)
Nõuetetehnika (*Requirements Engineering*) on süsteemne ja distsiplineeritud lähenemine nõuete spetsifitseerimisele ja haldamisele järgmiste eesmärkidega:

- tunda asjakohaseid nõudeid, saavutada huvipoolte vahel konsensus nõuete osas, dokumenteerida nõuded ja neid süsteemselt rakendada;
- mõista ja dokumenteerida huvipoolte soove ning vajadusi, seejärel spetsifitseerida ja hallata nõudeid (Pohl and Rupp 2015).

Nõuetetehnikat peetakse üheks kõige olulisemaks, kuid keerulisemaks faasiks tarkvaraarenduses. Tähelepanu pööramine tarkvaraarendusprojektide nõuete väljaselgitamisele, defineerimisele ja haldamisele aitab ennetada vigu, parandada kvaliteeti ja minimeerida riski tarnida süsteem, mis ei vasta huvipoolte soovidele ja vajadustele. (Damian and Chisan 2006)

2.2.1 Peamised nõuete tüübid:

- **funktsionaalsed nõuded** - nõuded, mis puudutavad tulemust või käitumist, mida süsteemi funktsioon peab tagama;

- **kvaliteedinõuded** - (mittefunktsionaalsed nõuded) on nõuded, mis puudutavad kvaliteediküsimusi, mida funktsionaalsed nõuded ei kata (Pohl and Rupp 2015). Kolm peamist kvaliteedinõuete faktorit on:
    - **jõudlus -** kui kiiresti peaks tarkvara reageerima, kui palju arvutiressursse peaks kasutama, kui täpne peaks tulem olema, kui palju andmeid peaks suutma talletada;
    - **kasutatavus -** kui efektiivselt peaks süsteem kasutajatega töötama, kui lihtne peaks olema seda kasutama õppida, kui tõhus peaks see olema igapäevasel kasutamisel;
    - **hooldatavus -** kui lihtne peaks olema vigade parandamine, uue funktsionaalsuse lisamine jne (Lauesen 2002).
- **piirangud** (*constraints*) - nõuded, mis piiravad lahendusruumi rohkem kui on vajalik funktsionaalsete nõuete ja kvaliteedinõuete täitmiseks (Pohl and Rupp 2015).

2.2.2 Nõuete allikad

Kirjanduse põhjal on peamised nõuete allikad:

- huvipooled,
- dokumendid,
- kasutusel olevad süsteemid (Pohl and Rupp 2015).

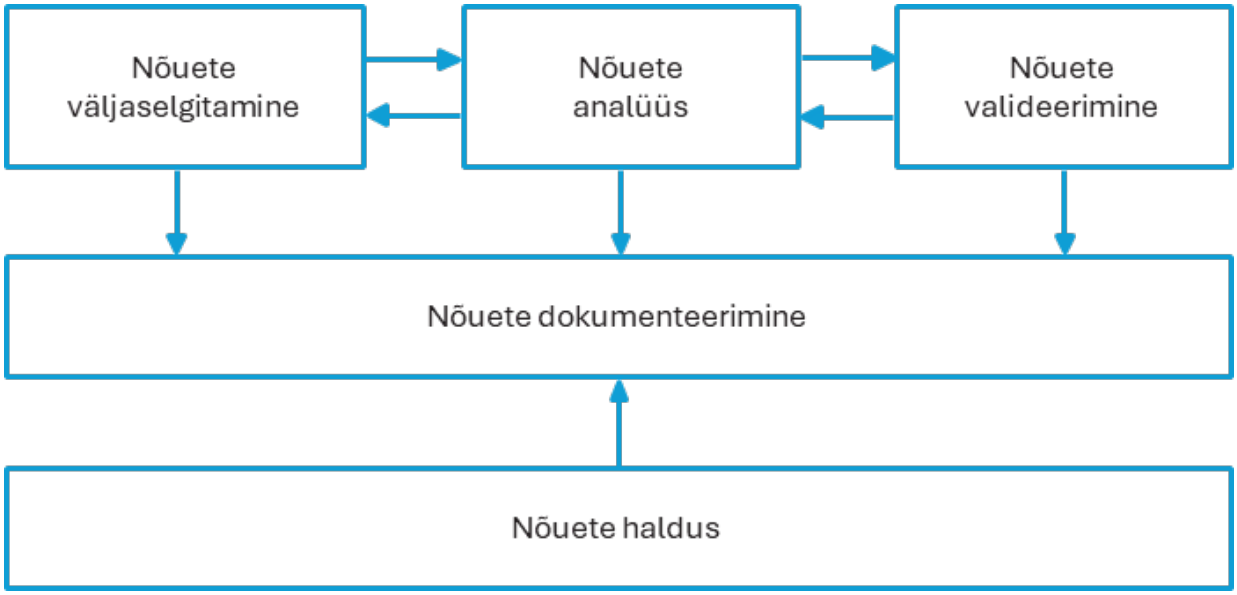
Huvipooled (*stakeholders*) on inimesed või organisatsioonid, kes (otseselt või kaudselt) mõjutavad süsteemi nõudeid. Huvipoolte näideteks on süsteemi kasutajad, operaatorid, arendajad, arhitektid, kliendid ja testijad.

Dokumendid sisaldavad sageli olulist informatsiooni, millest võivad selguda nõuded. Dokumentide hulka kuuluvad nii universaalsed dokumendid, nagu standardid ja õigusaktid, kui ka valdkonna- või organisatsioonispetsiifilised dokumendid, nagu nõuete dokumendid, pärandtarkvara (*legacy software*) veateated, e-kirjad, kasutajate tagasiside ja andmete kirjeldused. (Pohl and Rupp 2015) Pärandsüsteemi dokumentide analüüs võib aidata analüütikul esitada täpsustavaid küsimusi ning koguda informatsiooni, mis on abiks uue süsteemi nõuete tuvastamisel, täpsustamisel või lahenduse soovilogi (*backlog*) (Arendussõnastik 2024) väljatöötamisel (Paul and Cadle 2020).

Käigus olevad süsteemid, mis võivad olla nii loodav või arendatav süsteem ise praeguses arengufaasis, arendatavale lahendusele eelnev pärandtarkvara või konkureerivad süsteemid. Osapoolte kirjeldused kasutuskogemusest on väärtuslik materjal ja nende põhjal saab tuletada uusi või parandada ja täiendada olemasolevaid nõudeid. (Pohl and Rupp 2015)

2.3 Nõuetetehnika raamistiku etapid

Nõuetetehnika protsessi põhilised etapid on toodud joonisel 3.



Joonis 3. Nõuetetehnika raamistik, autori tõlge (Paul and Cadle 2020)

2.3.1 Nõuete väljaselgitamine

Nõuete väljaselgitamine (*requirements elicitation*), eesti keeles kasutatakse ka nõuete hõive mõistet, on protsess, mille käigus tuvastatakse kavandatava süsteemi nõuded. See hõlmab **arutelusid** huvipooltega, et kindlaks teha nende vajadused ning selgelt määratleda, milliseid funktsionaalsusi süsteem peaks pakkuma. (O'Regan 2017) Oluline on protsessi kaasata **huvipooled**, kes töötavad organisatsioonis või on tarkvaratoote tulevased kasutajad (Paul and Cadle 2020). Ärianalüütik ja asjakohased huvipooled viivad läbi **ajurünnakuid**, et määratleda kõrgtaseme nõuded kavandatavale süsteemile või olemasoleva süsteemi muudatusele. Nõuete väljaselgitamine võib hõlmata **intervjuusid** huvipooltega, et lubada neil rääkida, kuidas nad praegu oma tööd teevad, ja tuvastada nende nõuded kavandatavale süsteemile. See võib hõlmata ka **vaatlusessiooni**, kus ärianalüütik jälgib kasutajaid, et näha, kuidas tööd praegu tehakse. (O'Regan 2017)

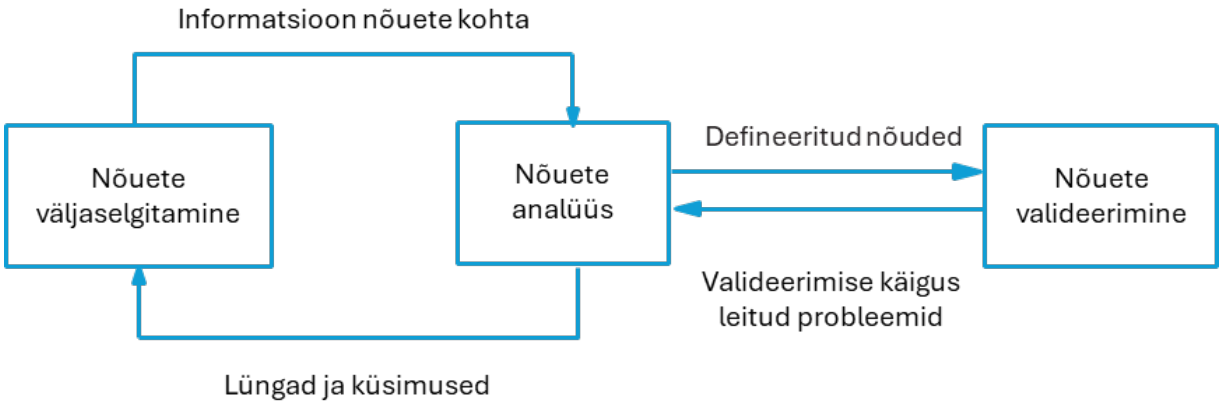
Nõuete väljaselgitamise tehnikaid käsitletakse erinevates allikates väga erinevas kvantitatiivses ja kvalitatiivses mahus. Kuna erinevate tehnikate kasutamine ja analüüs võiks olla eraldi uurimistöö teema, siis autor neil käesolevas töös pikemalt ei peatu, vaid toob lühidalt välja need, mida on ise käesolevas uurimistöös või teistes projektides kasutanud.

Töötubasid (*workshop*) saab kasutada nõuete esitamise alustamiseks, valdkonnast ülevaate saamiseks, keeruliste nõuete analüüsimiseks, vastuoluliste nõuete lahendamiseks ja pakutava lahenduse prototüüpide hindamiseks.

Dokumentide kogumine ja analüüs on eelkõige analüütiku iseseisev töö projektiga seotud dokumentidega. Formaalsed dokumendid on ärinõuded, kasutajapoolsete nõuete kirjeldused, eelnevate ja liidestuvate süsteemide dokumentatsioon jne. (Paul and Cadle 2020)
Mitteformaalsed dokumendid, nagu kirjad, uuringud, tegevusloendid ja muud kirjeldavad materjalid võivad sisaldada dokumentatsioonis mitteleiduvaid nõudeid. Sellised kasutajanõuded ei tohiks jääda varjatuks, vaid need tuleb välja tuua. (Hull, Jackson, and Dick 2010)
Andmenõuded on samuti oluline osa nõuetest. Milliseid andmeid peaks süsteem sisestama ja väljastama ning milliseid andmeid peaks süsteem seesmiselt salvestama? Enamik süsteeme liidestuvad väliste andmestikega ja andmeid tekib ka süsteemi töötamise käigus. Süsteem peab talletama vastavad andmed mõnda andmebaasi või muusse seesmisesse objekti. Nende andmete spetsifitseerimine on oluline. Mõned analüütikud väidavad, et andmedetailidega tegeldakse arendusprotsessis ja neid ei peaks nõuetes spetsifitseerima. Siiski on valdkonnas leiduva teabe ja süsteemi talletatud andmete vahel peaaegu üksühene seos (Lauesen 2002). Käesoleva töö autor on seisukohal, et tarkvaraprojektide puhul, mis kasutavad sisendina erinevaid andmeallikaid, on andmenõuetega tegelemine võtmetähtsusega.
Prototüüpe ja visandeid võib kasutada kahel viisil: neid võib luua töötoa käigus osana tegevusest, et visualiseerida ekraani, aruannet või stsenaariumit. Teise võimalusena võib ühiselt läbi käia eelnevalt välja töötatud prototüübi, et valideerida praegust mõtteviisi, selgitada välja nõuded, mis on tähelepanuta jäänud, või genereerida nõuete kohta täiendavaid üksikasju. (Paul and Cadle 2020)
Käesolevas magistritöös on autor kasutanud peamiselt online-koosolekute teel läbiviidud töötubasid ja nii formaalsete kui ka mitteformaalsete dokumentide läbitöötamist ning andmeanalüüsi.

2.3.2 Nõuete analüüs

Nõuete analüüs on väljaselgitatud nõuete ülevaatamine, klassifitseerimine ja korrastamine, et kõrvaldada kattuvusi või vigu, lahendada vastuolusid ja vasturäkivusi, hinnata teostatavust ning määrata prioriteete. Nõuete väljaselgitamise ja analüüsi eraldamine on nõuetetehnika raamistiku oluline tugevus. Nõuete väljaselgitamise faasis kogutakse aktiivselt teavet, avastatakse varjatud teadmisi ja tuvastatakse nõudeid. Seejärel kogutud informatsioon analüüsitakse, et defineerida lahendusele esitatavad nõuded. Nõuete väljaselgitamise ja analüüsi protsess on iteratiivne - nõuded tuvastatakse, analüüs tuvastab puudujäägid ja järgneb nõuete täiendav väljaselgitamine. (Paul and Cadle 2020)



Joonis 4. Nõuete väljaselgitamise ja nõuete analüüsi vahelised seosed (Paul and Cadle 2020), autori tõlge

Enne projekti alustamist võidakse teha esialgne toimiv prototüüp (POC), et tagada kavandatava süsteemi tehniline teostatavus ning määratud eelarve ja ajakulupiirangute täitmine. Vastuolud lahendatakse arutelude ja **läbirääkimiste** teel huvipooltega. Nõuete analüüs on iteratiivne protsess, kus tagasiside läheb nõuete väljaselgitamise protsessi huvipooltele. (O'Regan 2017) Nõuete väljaselgitamise, analüüsi ja järgmises alapeatükis käsiteldava valideerimise seoseid väljendab joonis 4.

2.3.3 Nõuete valideerimine ja verifitseerimine

Nõuete valideerimise ja verifitseerimise vahelist erinevust saab illustreerida fraasiga "teeme õiget asja" versus "teeme seda õigesti". Teisisõnu, valideerimine kontrollib, et rakendatakse õigeid nõudeid, samas kui verifitseerimine jälgib, et määratletud nõudeid rakendatakse korrektselt. (Stec 2023)

Nõuete valideerimine on protsess, mille käigus kontrollitakse, kas dokumenteeritud nõuded määratlevad täpselt süsteemi, mida klient tegelikult soovib. See on kriitilise tähtsusega tegevus, sest vead nõuetes võivad hiljem arendusetapis või juurutamisel kaasa tuua ulatuslikud ümbertegemise kulud. Nõuete probleemide parandamine on palju kulukam kui disaini- või koodivigade kõrvaldamine, sest see nõuab sageli muudatusi süsteemi disainis, uut arendust ja testimist. Valideerimise käigus kontrollitakse nõuete **kehtivust, järjepidevust, täielikkust, realistlikkust ja testitavust**. (Sommerville 2016)

Nõuete valideerimise protsess aitab parandada tarkvara kvaliteeti ning tagada, et nõuded peegeldaksid tegelike kasutajate vajadusi (Ramadan and Megahed 2016). Nõudeid võidakse valideerida erinevate dokumenteerimis- ja modelleerimistehnikate abil. On soovitav, et valideerimise viib läbi projektiväline huvipoolte rühm. (Paul and Cadle 2020)

Nõuete verifitseerimine viiakse läbi kontrollimaks, et toode rahuldab nõudeid. Minimaalselt tehakse seda vastuvõtutestis (*acceptance test*), kus huvipooled käivad nõuded ükshaaval läbi ja kontrollivad, et rakendus neid täidab. Samuti on hea mõte verifitseerida, et vaheprodukti töö rahuldab nõudeid. Nii arendajad kui ka kliendid peavad veenduma, et arenduse käigus arvestatakse kõiki nõudeid (Lauesen 2002).

2.3.4 Nõuete dokumenteerimine

Nõuete dokumenteerimise (spetsifikatsiooni loomise) etapil tegeldakse erineva täpsus- ja täielikkusastmega nõuete kirjelduste, tabelite ning diagrammide koostamisega (Paul and Cadle 2020).

Kasutajanõuete dokument on tavaliselt kirjutatud loomulikus keeles (*natural language*), kirjeldab süsteemi välist käitumist ning määratleb funktsionaalsed ja mittefunktsionaalsed nõuded mittetehnilises keeles. **Süsteeminõuete dokument** on kasutajanõuete dokumendi laiendatud versioon ning see annab üksikasjaliku teabe selle kohta, kuidas kasutajanõudeid süsteemis täidetakse. (O'Regan 2017) Nõuete dokument peaks olema **selge, järjepidev, kokkuvõtlik ja teostatav**. Nõuetetehnika protsess ei ole ainult süsteemi kohta teabe kogumine, vaid ka oluline suhtluskanal tarkvaraarenduse protsessis osalevate huvipoolte vahel. Nõuete dokumentatsioon aitab analüüsida, ümber kirjutada ja kinnitada neid nõudeid. Kuigi modelleerimine ja dokumentatsioon on väledates meetodites vähem kasutusel kui nõuetetehnikas, kasutatakse neid siiski süsteemi paremaks selgitamiseks huvipooltele. Dokumentatsiooni puudumine võib pikas perspektiivis probleeme kaasa tuua. (Ramadan and Megahed 2016)

2.3.5 Nõuete haldus

Nimetatud etapp on seotud määratletud nõuete muudatuste haldamisega ja soovitud jälgitavuse taseme saavutamisega (Paul and Cadle 2020).

Hoolimata katsetest teha asjad juba esimese korraga õigesti, muutuvad nõuded arenduse käigus. Spetsifikatsiooni peab olema lihtne uuendada, lisada uusi nõudeid ja hinnata muudatuste tagajärgi (Lauesen 2002).

Nõuete haldus peab tagama, et projekt säilitaks kogu elutsükli jooksul ajakohase ja kinnitatud nõuete kogumi. On väga oluline, et projekti tarneobjektid oleksid kooskõlas nõuete uusima versiooniga ning kui nõuete dokument muutub, siis ka kõik muud projekti tarneobjektid, nagu kujundusdokument, tarkvaramoodulid ja testikirjeldused, oleksid vastavuses nõuete uue versiooniga. (O'Regan 2017)

2.4 Nõuete jälgitavus

Nõuete jälgitavuse etapi eesmärk on kontrollida, et kõik projekti jaoks määratletud nõuded on rakendatud ja testitud. Üks viis on vaadata iga nõuet eraldi ja käia läbi kogu disainidokument, et leida, kus seda konkreetset nõuet disainis rakendatakse. Samuti käia läbi testidokument, kui see on olemas ja leida viited nõude numbrile, et näha, kas ja millisel sammul seda testitakse. (O'Regan 2017)

Nõuete jälgimiseks saab kasutada näiteks nõuete jälgimismaatriksit, (*Requirements Traceability Matrix*, RTM). See on dokument, mis aitab jälgida, et kõik nõuded on süsteemi spetsifikatsioonides kajastatud ja korrektselt testitud (Pohl 2010).

Järgneva tabelina on toodud näide lihtsast nõuete jälgimismaatriksist (tabel 1).

Tabel 1. Näide nõuete jälgimismaatriksist

Nõude ID	**Nõude kirjeldus**	**Algallikas**	**Süsteemi moodul**	**Testi-juhtumi ID**	**Staatus**
REQ-001	Kasutaja peab saama sisse logida kasutades e-posti ja parooli.	Kasutaja-lood	Autentimis-moodul	TC-101	Testitud
REQ-002	Süsteem peab kuvama veateateid ebaõnnestunud sisselogimis-katsete korral.	Kasutaja-lood	Autentimis-moodul	TC-102	Testitud
REQ-003	Administraatorid saavad luua, muuta ja kustutada kasutaja-kontosid.	Spetsifikat-siooni-dokument	Kasutaja-halduse moodul	TC-103, TC-104	Töös
REQ-004	Süsteem peab kasutaja paroole krüpteerima.	Turva-nõuded	Autentimis-moodul	TC-105	Testitud
REQ-005	Süsteem peab toetama vähemalt 1000 samaaegset kasutajat.	Jõudlus-näitajad	Taristu	TC-106	Testimata
REQ-006	Andmed peavad automaatselt varunduma iga 24 tunni järel.	Avariitaaste kava	Andme-halduse moodul	TC-107	Testimata

Maatriksi põhielemendid:

- **Nõude ID**: iga nõude jaoks unikaalne identifikaator.
- **Nõude kirjeldus**: lühikirjeldus selle kohta, mida nõue hõlmab.
- **Algallikas**: dokument või arutelu, kus nõuet esmalt tutvustati.
- **Süsteemi moodul**: süsteemi osa, kus nõuet rakendatakse.
- **Testijuhtumi ID**: viited spetsiifilistele testijuhtumitele, mis kinnitavad nõude täitmist.
- **Staatus**: nõude praegune staatus (nt "Testitud", "Töös", "Testimata").

Toodud maatriks aitab huvipooltel tagada, et iga funktsionaalne ja mittefunktsionaalne nõue on mitte ainult rakendatud, vaid ka testitud vastavalt algspetsifikatsioonile. Samuti aitab see hallata süsteemi muudatusi, näidates, millised süsteemi osad ja testid on mõjutatud nõuete muutumisest. (Chemuturi 2012)

2.5 Nõuete prioritiseerimine

Prioriseerimine on tarkvaraarenduses oluline, kuna nõudeid on palju ning alati kehtivad aja- ja eelarvepiirangud. Kõik nõuded ei ole võrdselt tähtsad ja prioriseerimine aitab määratleda, millega tuleks iga iteratsiooni jooksul tegeleda või mis peaks minema reaalajas kasutusse. (O'Regan 2017) Prioriseerimise viise on palju. Mõned neist on lihtsad, näiteks **kõrge, keskmine või madal**, samas kui teised on sügavamad ja keerukamad. Oluline on märkida, et peamine probleem prioriseerimisel on meeskonna võime kokku leppida prioriteetsuse tasemes ja lihtsa struktuuri kasutamisel selles, mida tasemed tähendavad.

MoSCoW prioriseerimisskeem pakub struktureeritud viisi projektide arendamisel ja tarnimisel nõuete haldamiseks. Skeem määratleb neli prioriseerimiskategooriat.

1. **Peab olema** (*must have*) **nõuded:** need on esmaseks kasutuselevõtuks hädavajalikud ja moodustavad nõuete „miinimumkasutatava alamhulga".
2. **Peaks olema** (*should have*) **nõuded:** need nõuded lisavad projektile paindlikkust ja varu. Kui neid nõudeid ei saa algse ajakava kohaselt tarnida, saab neid edasi lükata järgmisse etappi ja võib ajutiselt rakendada manuaalsete lahendustena.
3. **Võiks olla** (*could have*) **nõuded:** on vähem kriitilised, kuid neid saab lisada juhul, kui on lihtsad ja odavad integreerida. Pakuvad täiendavat varu, kuna saab välja jätta, kui ajapiirangud muutuvad probleemiks.

4. **Tahaks olla, kuid seekord ei ole** (*won't have (this time)*) **nõuded:** on määratletud hilisematesse projektifaasidesse ja ei prioritiseerita kohe tarneks. See kategooria aitab tulevaste etappide planeerimisel, kus nõuete prioriteetsus võib muutuda kõrgemaks.

MoSCoW meetod on kasulik mitte ainult tarkvaraarenduses, vaid ka teistes ärimuudatustes, nagu protsessidokumentatsioon ja meeskonna võimekuse arendamine, aidates tuvastada olulisi elemente, edasilükatavaid valikuid ja mittevajalikke üksusi (Milani 2019).

2.6 Probleemid nõuetega

Viimase kolmekümne aasta jooksul tehtud uuringud IT-projektide nurjumiste kohta on järjepidevalt toonud esile sarnaseid probleeme:

- üle 80% IT arendusprojektide vigu ilmneb nõuete analüüsi faasis;
- nõuete analüüsi faasile on eraldatud vähem kui 12% projektide teostamise ajast (Yeates et al. 2006);
- arendusfaasis esineb vigu alla 10% – arendajad programmeerivad asju enamasti õigesti, kuid sageli mitte õigeid asju;
- enamus IT projektidele kuluvast ajast on eraldatud arendus- ja testimisfaasidele;
- arendatud süsteemide kooskõla äristrateegiate ja eesmärkidega kaldub olema nõrk;
- nõuete juhtimine ja haldamine on enamasti ebapiisav (Paul and Cadle 2020).

Walia ja Carver tuvastasid oma metauuringus, mis hõlmas 149 teaduslikku tööd, üle 90 nõuete vigade põhjuse. Tüüpilised probleemid nõuetega on:

- puuduv seos projekti eesmärkidega;
- sõnastuse ebamäärasus;
- nõuete mitmemõttelisus;
- nõuete dubleerimine ja kattumine;
- vastuolud nõuete vahel (Walia and Carver 2009).

Davey ja Parker toovad kirjanduse süstemaatilise analüüsi tulemusena välja järgmised nõuetetehnika probleemid koondatuna üheksasse kategooriasse:

1. nõuete väljaselgitamise käigus ilmneb inimlikke aspekte, mis muudavad suhtluse konsultandi ja kliendi vahel keeruliseks;
2. inimeste vahelise suhtlemise keel ei ole alati tehnoloogilise lahenduse jaoks sobiv;
3. projekti käigus nõuded muutuvad;
4. mõnikord esitavad kliendid nõudeid, mida organisatsioon tegelikult ei vaja;

5. huvipooled ei oska öelda, mida nad süsteemilt tahavad ja millised on tegelikud ärivajadused;
6. mõned kliendid ei ole koostöövalmid ja ei taha analüütikut projekti juures aidata. (Ka autor on puutunud kokku huvipoolte vastutöötamisega – kardetakse, et õnnestunud tarkvaralahenduse tulemusena võivad nad töö kaotada);
7. nõuetetehnika protsess ebaõnnestub, sest seda ei tehta korralikult;
8. sageli teatatakse sümptomitest, mis ei ole tegelikud probleemid;
9. nõuetetehnika ei ole deterministlik (Davey and Parker 2015).

Nõuete väljaselgitamine huvipooltelt on keeruline, sest:

- huvipooled ei pruugi teada, mis on tehniliselt teostatav ja mis mitte, ning neil võivad olla ebareaalsed ootused;
- erinevatel huvipooltel võivad olla erinevad ja vastandlikud ootused süsteemile (O'Regan 2017).

Zeil toob välja veel mõned ärikeskkonnaga seotud probleemid seoses nõuete väljaselgitamisega:

- organisatsioonilised ja poliitilised tegurid võivad mõjutada süsteemi nõudeid;
- võivad ilmneda uued huvipooled;
- ärikeskkond võib muutuda (Zeil 2023).

Mõned tüüpilised probleemid, mis võivad tekkida nõuete hõive käigus väledatel meeskondadel:

- ei tehta huvipoolte analüüsi ega arvestata oluliste sidusrühmadega; usaldatakse täielikult ühte isikut, kes esindab nn kliendi häält;
- ei kaasata laiemat huvipoolte gruppi ega mõisteta erinevaid nõuete vaatenurki;
- keskendutakse liialt funktsionaalsuse ja omaduste väljaselgitamisele, jättes tähelepanuta mittefunktsionaalsed nõuded;
- ei analüüsita, kuidas uus lahendus muudab praeguseid tööpraktikaid, ametirolle, juhtimisstruktuure ja äriprotsesse (Girvan and Paul 2017).

Uuringud on näidanud, et nõuete analüüs on arendustsükli kõige veaohtlikum osa. Nõuetele vastavusega seotud vigade parandamiskulud kasvavad oluliselt, kui need avastatakse arendustsükli hilisemates etappides (Paul and Cadle 2020).

2.7 Heade nõuete omadused

Üldtunnustatud definitsioon, mis määratleb **heade nõuete** spetsifikatsiooni, on leitav IEEE standardist 830-1998. Hea nõuete kogum vastab järgmistele kvaliteedikriteeriumidele:

- korrektne - iga nõue peegeldab huvipoolte vajadust.
- täielik - kõik vajalikud nõuded on kaasatud.
- ühemõtteline - iga nõue peab olema selgelt kirjeldatud ja üheselt mõistetav kõigile huvipooltele.
- järjepidev - nõuete komplekt on täielik ja järjepidev ning ei ole üksteisega vastuolus.
- tähtsuse ja/või stabiilsuse järgi järjestatud - igale nõudele on määratud prioriteet, mis näitab selle suhtelist tähtsust.
- muudetav - nõuded on vajadusel lihtsasti muudetavad nii, et kooskõla teiste nõuetega säilib.
- kontrollitav - hea nõue on sõnastatud viisil, mis võimaldab seda testida ja näha, kas nõue on edukalt rakendatud ning täidetud.
- jälgitav - hea nõuete kogumi aspekt on see, et on võimalik jälgida nii nõuete kogumit kui üksikuid nõudeid kõrgemate ja madalamate tasemete nõueteni. See puudutab ka võimet nõudeid muuta ja kaaluda, milliseid teisi nõudeid see mõjutab. eesmärkide/otstarvetega, disaini/koodiga seotud.

(IEEE Computer Society and Software Engineering Standards Committee 1998)

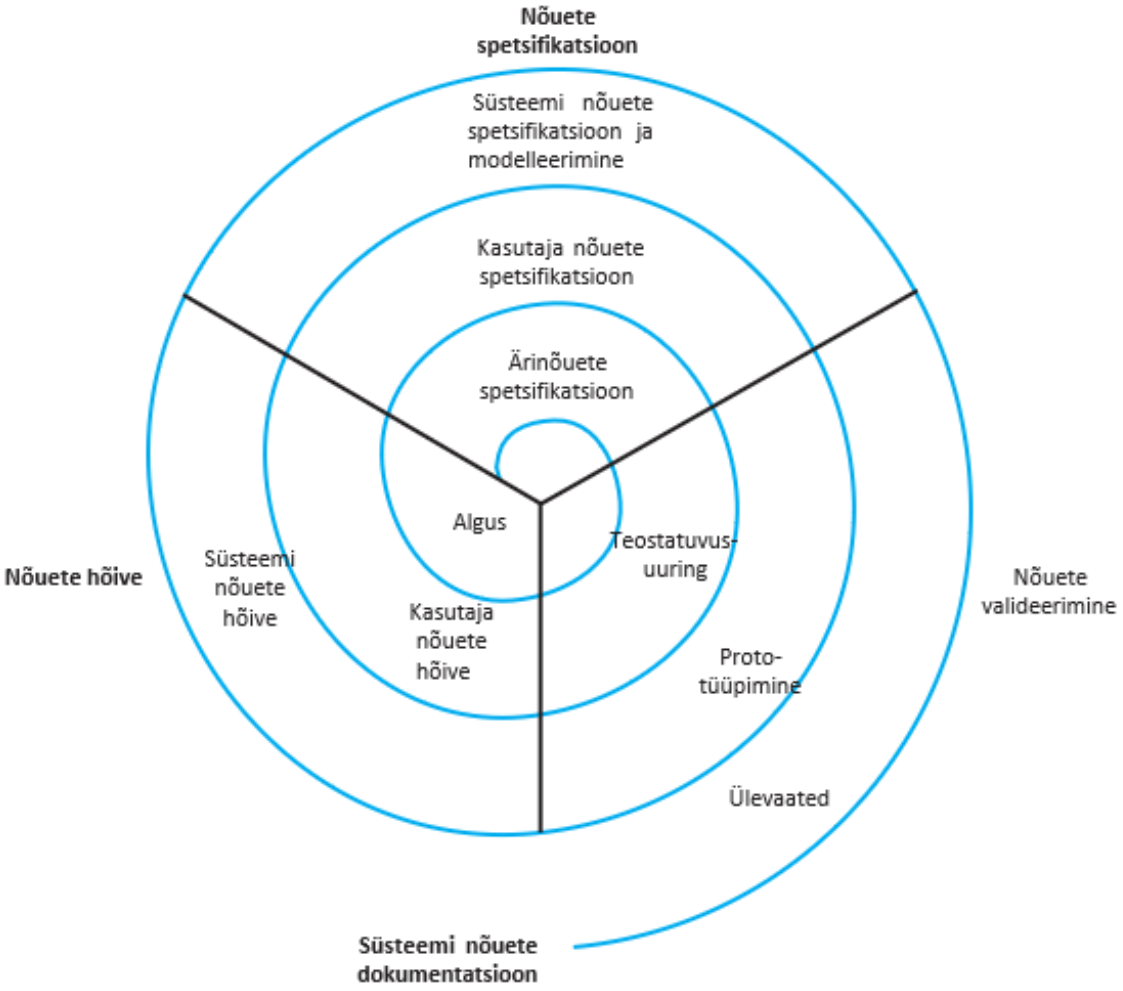
Lisaks veel mõned headele nõuetele iseloomulikud tunnused, mida on erinevates allikates mainitud:

1. iga nõue on süsteemi funktsionaalsuse või omaduse tagamiseks vajalik, on kooskõlas projekti eesmärkidega (O'Regan 2017);
2. iga nõuet on võimalik rakendada - see peab olema teostatav. Tavaliselt esinevad piirangud riskide, eelarve ja/või aja osas, mida tuleb arvesse võtta. Hea nõue arvestab neid piiranguid, et tagada teostatavus (Milani 2019).

2.8 Nõuetetehnika protsesside iteratiivsus

Praktikas on nõuetetehnika iteratiivne protsess, kus tegevused on korraldatud vastavalt spiraalmudelile (joonis 5). Väljundiks on süsteemi nõuete dokument, mille puhul iga tegevuse aja- ja ressursikulu varieerub sõltuvalt projekti etapist, süsteemi tüübist ja eelarvest. Spiraalmudel

võimaldab nõuete arendamisel erinevat detailsust ja erinevat arvu iteratsioone enne spiraalist väljumist, kui kasutaja nõuded on piisavalt välja selgitatud.



Joonis 5. Nõuetetehnika protsessi spiraalne vaade, autori tõlge (Sommerville 2016)

Nõuded muutuvad sageli, kuna tekib parem arusaamine, toimuvad organisatsioonilised muudatused või süsteemi modifikatsioonid. Muudatuste haldamine on oluline, et hinnata nende mõju teistele nõuetele ning kaasnevaid kulusid ja tagajärgi. (Sommerville 2016)

2.9 Nõuete analüüs väledas arendusprotsessis

Täpsete nõuete väljendamine täiesti uue tarkvaralahenduse jaoks võib olla keeruline. Huvipooled keskenduvad tihti lahendustele, mitte põhivajadustele. Ärianalüütikud peavad neid nõudeid kriitilise pilguga vaatama, analüüsima ja hindama muudatuste mõjusid, et määratleda täpsed ning asjakohased vajadused. Algideed võivad projekti edenedes ja mõistmise süvenedes areneda.

Seetõttu ei määratleta nõudeid väledates projektides alguses täielikult, vaid täpsustatakse vajaduspõhiselt arendusiteratsioonide käigus. (Girvan and Paul 2017)

Ärieesmärkidele keskendunud väledas tarkvaraarenduse protsessis ei nähta sageli ette piisavalt aega ja eelarvet nõuete väljaselgitamiseks ja analüüsiks, ning nõuetetehnika meetodite piisava põhjalikkusega kasutamiseks. Kliendid on harjunud, et nõuete kogumist praktiliselt ei toimu ja selleks ei ole vahendeid ette nähtud. Arendaja kogub esialgsed nõuded kokku, tarnib midagi ja siis hakatakse rakendust muutma selliselt, et see vastaks tegelikele ootustele ja vajadustele. Kliendile on keeruline, kuid vajalik seletada, et nõuete analüüsile eraldi aja ja vahendite leidmine aitab kaasa arendusprotsessi kiirendada ning tulemus vastab paremini ootustele. Ümbertegemist on vähem ning järgmiste arendusfaaside ära jäämise tõttu saavutatakse pikemas perspektiivis kokkuhoid koos ärivajadustele paremini vastava tarkvaralahendusega. (Rasheed et al. 2021)

Väledad mudelid põhinevad järkjärgulisel arendusel ja tarnel kus tarkvaratoodet käsitletakse nõuetel põhinevate funktsioonide kogumina, mida järk-järgult rakendatakse. Alguses prioriseeritakse olulisemad funktsioonid, mille üksikasjad defineeritakse ja mis seejärel rakendatakse. Kasutajad või testkasutajad saavad funktsiooni proovida ja anda arendusmeeskonnale tagasisidet. Seejärel liigutakse edasi järgmise funktsiooni defineerimise ja rakendamisega. (Sommerville 2019)

Käesoleva töö autor leiab siiski, et isegi väleda arendusprotsessi puhul on mõttekas projekti algfaasis ära kaardistada nõuete kogum, mis katab kliendi jaoks oluliseks tähtajaks tarnitavad vajalikud funktsionaalsused. Siiski ei saa väledas arendusprotsessis nõudeid käsitleda arendusest ja tarnest eraldi ja seetõttu on töö praktilises osas käsitletud arenduse ning tarnega seotud teemasid. Käesolevas töös näiteks võetava tavapäraselt väleda arendusprotsessiga tarkvarafirmas Reach-U oli vaadeldava arenduse puhul mõttekas teha erand ja defineerida MVP (M*inimum Viable Product*) nõuded võimalikult täpselt ja põhjalikult, sest MVP kasutuselevõtu tähtaeg oli suhteliselt lühike ja ootused tarkvara funktsionaalsuse osas kõrged ning konkreetsed. Üheksa kuu pärast pidi tarkvaral olema sadu kasutajaid ja teenindama kümnetesse miljonitesse dollaritesse ulatuvaid rahavoogusid.

2.10 Näited nõuetetehnikas kasutatavatest valmistarkvaradest

Nõuetetehnika protsessi toetamiseks on loodud palju erinevaid tarkvaralahendusi või lisamooduleid arenduse haldustarkvarale.

Nõuete haldustarkvarana vaadeldi järgmisi valmistarkvara alternatiive ning hinnati nende eeliseid ja puudusi väledas arenduskeskkonnas:

1. Sparx systems Enterprise Architect

2. Jama Connect
3. Atlassin Jira
4. Jira nõuete halduseks mõeldud lisamoodul R4J
5. Atlassin Confluence

Enterpsise Architect (EA)

EA eelised:

1. EA võimaldab nõudeid koguda, süstematiseerida ja siduda muude tarkvaramudeli elementidega nagu kasutuslood, äriprotsessid ja süsteemi komponendid. Aitab tagada nõuete jälgitavuse ja terviklikkuse kogu arendusprotsessi vältel;
2. EA pakub mitmekülgseid võimalusi nõuete dokumenteerimiseks, sealhulgas tekst, tabelid ja diagrammid. See muudab nõuded arusaadavaks erinevatele huvipooltele;
3. EA toetab nõuete versioonihaldust ja muudatuste jälgimist, mis on oluline väledas arenduses, kus nõuded võivad kiiresti muutuda. (Enterprise Architect 2024)

EA puudused tarkvaranõuete haldamisel väikeste väledate meeskondade puhul:

1. EA on keerukas tööriist, mille tõhus kasutamine nõuab koolitust ja aega harjumiseks. See võib olla väljakutse väikestele meeskondadele, kellel on piiratud ressursid;
2. EA on mõeldud suurte ja keerukate süsteemide modelleerimiseks. Väiksemate projektide puhul võib EA kasutamine olla ülemäärane ja aeganõudev;
3. EA litsentsid ja hoolduskulud on märkimisväärsed. Väiksematel meeskondadel ei pruugi olla piisavalt eelarvet, et EA täielikku potentsiaali ära kasutada. (Askarinejad 2012)

Jama Connect

Jama Connecti peetakse tugevaks nõuete haldamise ja jälgitavuse lahenduseks, mis on eriti sobilik keerukate toodete arendamiseks kõrgelt reguleeritud tööstusharudes. Selle peamised eelised hõlmavad reaalajas jälgitavust, testihaldust, ülevaatuste haldust ning riski- ja nõuete haldust, mis võimaldavad meeskondadel tõhusalt koostööd teha ja kohandada tööprotsesse vastavalt organisatsiooni vajadustele. Rakendus toetab erinevaid integratsioone, nagu Slack, Jira Software ja Microsoft Teams, mis lihtsustab ülesannete tõhusat täitmist. Siiski, Jama Connecti on keeruline kasutada ja selle kasutama õppimine võtab aega. Lisaks võib see olla väikestele organisatsioonidele kallis valik. (The Daily Egg 2023b)

Jira

Atlassin Jira on võimas tööriist lihtsamate tarkvaratoodete arenduse haldamiseks, kuid see ei sobi keerukate süsteemide nõuete korral, mis peavad vastama keskmisest rangematele standarditele.

Jira ei võimalda nõudeid mugavalt dokumenteerida, versioonida, visualiseerida ega jälgitavust tagada. Samuti on Jiras keeruline genereerida nõuete dokumente ülevaatusteks. (ReqView 2024)

R4J

Jira R4J (Requirements for Jira) on tõhus vahend mis võimaldab luua ja hallata nõuete hierarhilisi struktuure, pakkudes selget jälgitavust ja muudatuste mõju analüüsi. Integreerub sujuvalt Jira olemasoleva infrastruktuuriga, võimaldades meeskondadel hallata nõudeid samas keskkonnas, kus nad tegelevad projektijuhtimise ja veaotsinguga, parandades koostööd ja suhtlust.

Siiski, R4J võib olla keeruline kasutada oma laialdaste kohandamisvõimaluste ja funktsioonide tõttu, mis võivad tekitada segadust ja vajavad põhjalikku väljaõpet. Lisaks võib R4J olla ressursimahukas ning nõuda tugevat infrastruktuuri. Suuremate rakenduste korral võib esineda jõudlusprobleeme. (Software Advice 2024)

Confluence

Confluence on tugev nõuetetehnika tööriist, mis sobib hästi väledate arendusmeeskondade jaoks, pakkudes intuitiivset kasutajaliidest ja võimalust hõlpsalt koguda, korraldada ja jagada nõudeid meeskonnaga (The Daily Egg 2023a).

Jira ja Confluence'i koos kasutades on võimalik ühendada Jira tugevused projektijuhtimises Confluence'i eelistega nõuete dokumenteerimisel (Simplilearn.com 2022).

Siiski on Confluencel ka miinused, nagu suhteliselt piiratud funktsionaalsus, vähesed ajahaldusvõimalused ja ressursside haldamise toetus, mis võivad mõjutada selle sobivust keerukamate projektijuhtimis- ja koostöövajadustega ettevõtete jaoks (The Daily Egg 2023a).

Kokkuvõttes nõuab spetsiifilise nõuete haldustarkvara kasutuselevõtmine ajaressurssi selle häälestamiseks ja tundmaõppimiseks ning finantse. Tarkvaraprojektide eluiga tänapäeval suhteliselt lühike ja sõltub äriprotsessidest ning nende muutumisest. Suurte ja suhteliselt staatiliste IT süsteemide, nagu ettevõtte tuumikinfosüsteemid ja ERP tarkvarad, mida kasutavad paljud ettevõtted, on multifunktsionaalse nõuete haldustarkvara kasutuselevõtt õigustatud, aga agiilses arenduses, kus piiratud ressurssidega soovitakse pidevat ja kiiret tulemust, võivad need süsteemid olla kohmakad ja käigushoidmine ressursimahukas. (Software Advice 2024)

3 Rakenduslik uurimistöö

Selles peatükis sõnastatakse hüpotees, uurimisküsimused, metoodika, töö eesmärk, antakse ülevaade fiktiivsest tarkvaralahendusest, mille näitel autor nõudeid välja selgitas ning esitatakse nõuetetehnika protsessi haldamiseks loodud töövahendile esitatud nõuded ja nende alusel loodud NTKT töövahendi funktsionaalsused.

3.1 Kontekst

Siinkohal tutvustab autor põgusalt fiktiivset tarkvaraprojekti, mis on avaliku kaitsmise huvides loodud analoogse tegeliku tarkvaraprojekti baasil, milles autor osales ärianalüütikuna ja kasutas käesolevas töös kirjeldatud töövõtteid ja meetodeid.

Tarkvararakenduse tellija on **Multimart**, fiktiivne Ladina-Ameerika hulgimüügikorporatsioon, mis tegeleb jaemüügikettide varustamisega, edaspidi **Klient**.

Eesti tarkvarafirma Reach-U ja Kliendi vahel on kujunenud koostöösuhe, kus arendatakse väledat arendusmudelit kasutades tarkvaraplatvorme, mis on kasutuses juba aastaid. Arendus toimub sprintidena, mille eel lepitakse kokku uued lisanduvad funktsionaalsused ja eelarve ning nõuded kirjeldatakse enamasti formaliseerimata kujul SOW (*statement of work*) dokumendis. Seega oli tegemist väleda arendusprotsessiga, mis põhines pikaajalisel edukal koostööl ja usalduslikul suhtel.

Seekord soovis Klient täiesti uut ja peaaegu nullist ehitatavat tarkvaralahendust (liidestusi eelnevalt loodud tarkvarakomponentidega oli siiski võimalik kasutada), mis pidi Kliendi ärikeskkonnas välja vahetama pärandtarkvara **SMP** platvormi, mis teenindab sadu suurkliente ning sadadesse miljonitesse dollaritesse ulatuvaid eelarveid.

SSMP (*Strategic Sales Management Platform*) on Kliendi poolt tellitav fiktiivne tarkvaraplatvorm. See on iseteenindusbüroo, kus jaemüüjast **Ostja** saab juurdepääsu olemasolul pärast sisselogimist lasta genereerida enda ettevõttele müügipakkumise, mis on optimeeritud just Ostjat huvitavale tarbijate sihtrühmale, tagades, et pakkumine sisaldab just sellele sihtrühmale optimaalses sortimendis ja koguses kaupu, mis on võimalik ka kasumlikult maha müüa. Ostja suhtleb pakkumise saamiseks kasutajaliidesega **Ostja UI**, kuhu pärast sisselogimist sisestab pakkumise eelarve ja teised vajalikud parameetrid. Ostja UI saadab päringu **Pakkumiste Genereerijale** (edaspidi **PG**), mis genereerib ostja poolt valitud parameetrite ja sihtrühma põhjal Pakkumise, kasutades selleks erinevaid andmeallikaid ja API-sid. Osa andmetest on anonümiseeritud suurandmed, mis sisaldavad inimeste eelnevat ostukäitumist ja nende põhjal

tehtud ennustusi. Neid andmeid kasutades peab PG optimeerima Pakkumise valitud sihtrühma kuuluvate tarbijate suhtes. PG saadab pakkumise Ostja UI-le, mis teeb selle visualiseeritult Ostjale kättesaadavaks. Ostja vaatab pakkumise üle ja kui see sobib, kiidab pakkumise heaks ja Ostja UI saadab selle välisele süsteemile **ISSL** (*Integrated Stock, Sales and Logistics Management System*), mis tegeleb Pakkumise põhjal tehtud tellimuse kokkupaneku ja tarnega. Käesoleva magistritöö autor osales projektis Reach-U **ärianalüütikuna**.
Võrreldes pärislahendusega on kasutatud fiktiivne näide tunduvalt lihtsustatud. Näiteks oli nõuete arv ning piirangute mõju kirjeldus teatud nõuetele ja visualiseerimine oluliselt komplekssem kui käesolevas näites.

3.2 Probleem

Tegemist oli suure ja keerulise arendusega, mis pidi asendama eelmise, tuntud Silicon Valley ettevõtte loodud ja hallatava lahenduse, käitlemaks kümnetesse miljonitesse dollaritesse ulatuvaid rahavoogusid. Autori arvates oli projektile algselt ette nähtud ebarealistlikult väike eelarve ja lühike tähtaeg, milleks toode pidi lõppkasutajateni jõudma ning teenindama kümnetesse miljonitesse dollaritesse ulatuvaid rahavooge.
Projekti arendusprotsess sattus COVID-19 pandeemia aega, mis tähendab, et töö toimus kodukontorites, suheldi veebi teel suhtlusrakenduste (näiteks MS Teams, Webex, jne) kaudu ning füüsilisi kokkusaamisi huvipoolte vahel ei toimunud.
Reach-U-l ja Kliendil ei olnud projekti algul ühist koostöötamise keskkonda ja ühiskasutuses olevat nõuete haldustarkvara.

3.3 Hüpotees

Püstitati hüpotees, et klassikaliste nõuetetehnika meetodite ja spetsiaalse töövahendi süstemaatiline rakendamine väledas arendusprotsessis suurendab nõuete haldamise efektiivsust, tagab piisava nõuete kvaliteedi ning võimaldab arendada kliendi vajadustele vastava minimaalselt elujõulise toote (MVP) ettenähtud tähtajaks ja realistlikult hinnatud eelarve raames.

3.4 Uurimisküsimused

1. Kuidas mõjutab klassikaliste nõuetetehnika meetodite süsteemne rakendamine nõuete haldamise kiirust ja kvaliteeti väledas tarkvaraarenduses?

2. Kas klassikaliste nõuetetehnika meetodite süsteemne rakendamine spetsiaalse tööriista abil võimaldab luua kliendi vajadustele vastava vähima toimiva toote (MVP) ettenähtud aja ja kokkulepitud eelarve piires?
3. Milline nõuete haldamise töövahend sobib oma funktsionaalsuse, kasutusmugavuse, koostöövõimaluste, integreerimisvõimaluste ja kasutamise töömahukuse poolest rakendamiseks agiilses tarkvaraarendusprotsessis, võimaldades tõhusalt toetada nõuete kogumist, analüüsi, prioritiseerimist ja jälgimist kogu arendusprotsessi vältel ning soodustades koostööd ja kommunikatsiooni huvipoolte vahel, ilma liigselt koormamata meeskonna ajaressurssi?

3.5 Lähtekoht, eesmärk

Uurimistöö lähtekohaks oli vajadus lühikese ajaga välja selgitada ning hallata suurt hulka nõudeid, mille rakendamine on eelduseks tarkvara tarnimisele kliendile kindlaksmääratud kuupäeval. Arvestada tuli sellega, et kliendipoolseteks huvipoolteks olid mitte-IT taustaga inimesed ja kliendi puhul on tegemist teisel mandril asuva korporatsiooniga, millel on ranged IT turvareeglid ning kliendil ja arendusmeeskonnal puudus ühine nõuete ja arendusprotsessi haldamise keskkond.
Käesoleva magistritöö eesmärk on uurida ja analüüsida klassikaliste nõuetetehnika meetodite ja süstemaatilise rakendamise mõju nõuete haldamise efektiivsusele, kvaliteedile ja kiirusele agiilses tarkvaraarendusprotsessis. Töös uuritakse, milline nõuete haldamise töövahend on kõige sobivam väleda tarkvaraarendusprotsessi toetamiseks piiratud ajaressursi juures, arvestades selle funktsionaalsust, kasutusmugavust, koostöö- ja integreerimisvõimalusi ning töömahukust, et optimeerida koostööd ja kommunikatsiooni arendusmeeskonna ning teiste huvipoolte vahel. Töö keskendub sellele, kuidas nimetatud lähenemised aitavad kaasa minimaalselt elujõulise toote (MVP) arendamisele, mis vastab kliendi vajadustele ettenähtud tähtajaks ja eelarve raames.

3.6 Metoodika

Nõuetetehnika protsessis käesoleva töö raames otsustati kasutada metoodikat, mis katab töö teoreetilises osas toodud nõuetetehnika protsessi peamised etapid ja tegevused ning lähtub konkreetse tarkvaralahenduse spetsiifikast. Autor lähtus hüpoteesist, et klassikalise nõuetetehnika metoodika kasutamine teatud erisustega on antud projekti puhul õigustatud. Metoodika lühikirjelduse koostamisel, mida on lihtne protsessi käigus jälgida, on kohaldatud näidet Murali

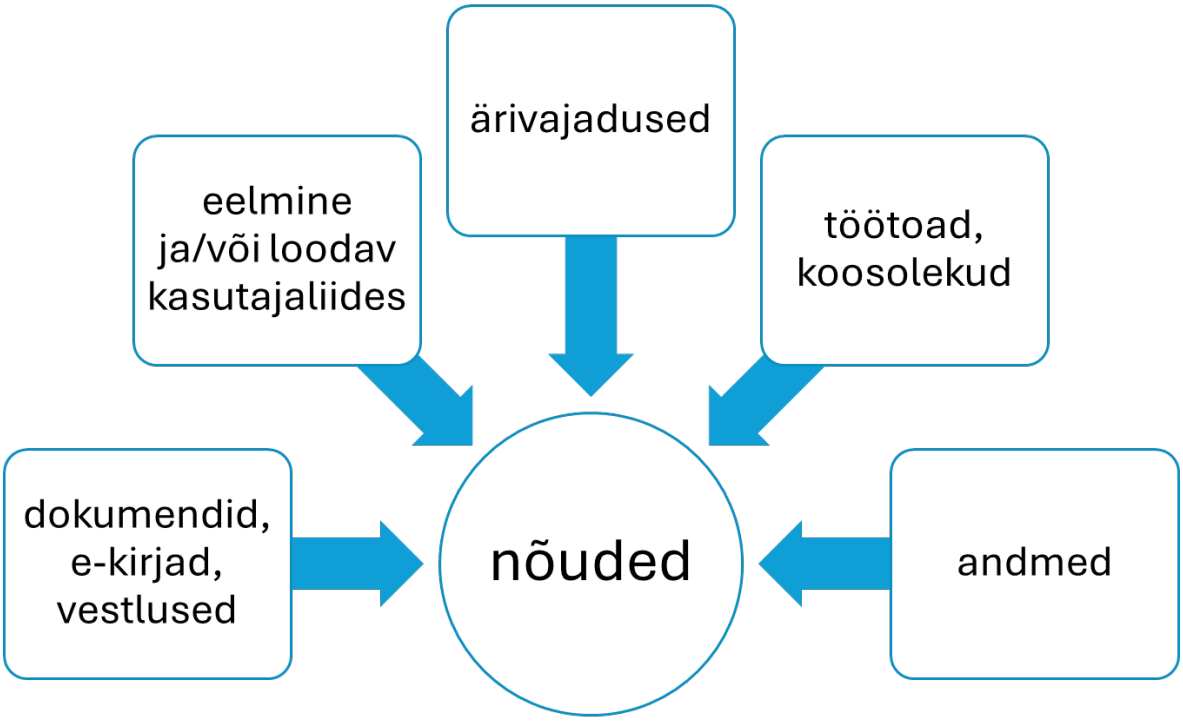
Chemuturi raamatust „Requirements Engineering and Management for Software Development Projects“ vastavalt käesoleva töö teooriaosas esitatud nõuetetehnika protsessi etappidele:

1. tuvastada nõuded kõikvõimalikest allikatest – dokumendid, koosolekute märkmed, e-kirjad, suhtlustarkvara Slack vestlused jne;
2. tuvastada nõuete protsessis olulised huvipooled, vajadusel paluti kliendi esindajal kaasata huvipooli teatud nõuete täpsustamiseks;
3. loetleda kõik esitatud, leitud ja tuletatud nõuded;
4. grupeerida nõuded loogilistesse rühmadesse;
5. kategoriseerida vastavalt tüübile:
    1. funktsionaalsed
    2. mittefunktsionaalsed
6. tuvastada korduvad nõuded;
7. tuvastada omavahel vastuolus olevad nõuded;
8. hinnata iga nõude teostatavust tehnilisest, finantsilisest ja ajakava seisukohast;
9. tuvastada süsteemi liidesed, millega süsteem peab integreeruma ja andmeid vahetama;
10. nõuded prioritiseerida vastavalt ajakavale ning tehnilisele ja finantsilisele teostatavusele;
11. kontrollida iga nõuet ja kogu nõuete komplekti täielikkuse osas;
12. määrada nõuete rakendamise ajakava (projektijuhtide poolt);
13. lahendada ülaltoodud tegevustes avastatud probleemid ja vastuolud (Chemuturi 2012).

Kehtestati nõuded, millele peab vastama tarkvara või muu töövahend, mis toetab nõuetetehnika protsessi läbiviimist projekti raames.
Vastavavalt analüüsi tulemustele, töövahendile esitatavatele nõuetele ja metoodika järgimise toetamiseks loodud nõuetetehnika kombineeritud töövahendit (NTKT) kasutades uuriti, kas see vastab kehtestatud nõuetele ning toetab nõuete haldamise protsessi käesoleva projekti raames.

3.7 Nõuete väljaselgitamine SSMP näitel

Käesolevas alapeatükis käsitletakse nõuete väljaselgitamise etappi SSMP projekti näitel.
Nõuete peamised allikad SSMTP projekti puhul on toodud järgneval joonisel (joonis 6).



Joonis 6. Peamised nõuete allikad SSMP projekti raames, autori koostatud joonis

Projekti algmaterjaliks oli Kliendi poolt saadetud üldine nõuete dokument, mis kirjeldas soovitud tulemust üldsõnaliselt. Lisaks edastati e-maili teel erinevas formaadis funktsionaalsuste nõudeid, piirangute ja reeglite kirjeldusi, mis põhinesid suuresti eelmisel tarkvaral, aga ei olnud süstematiseeritud. Projekti algfaasis toimusid iganädalased koosolekud kliendipoolse tootetiimiga, kus autor osales Reach-U esindajana enamasti üksinda.

Kliendi esindajad olid peamiselt äripoolelt ja neil puudus varasem tarkvaranõuete kirjeldamise kogemus. See-eest osati sõnastada üldised ärivajadused, milleks uut tarkvaraplatvormi vaja on.

Nõuete valiidsus ja teostatavus sõltusid suuresti andmetest. Autor analüüsis ja modelleeris SSMP projekti käigus andmenäidiseid, et tagada nõuete vastavus andmetele. Näiteks oli suure mõjujõuga huvipoole nõue, et rakendus peab toetama pakkumise väljastamisel kuupäevalist granulaarsust, aga selgus, et sisendandmed, mida väljundi genereerimisel tuleb kasutada, on nädalase granulaarsusega. Kui seda ei oleks nõuete valideerimisel kindlaks tehtud, oleks tegemist olnud kokkulepitud nõudega, mida ei oleks olnud võimalik arendusfaasis täita.

Osapooled lähtusid nõuete kirjeldamisel tihti pärandtarkvara SMP kasutajaliidesest, mis oli paindlik ja lubas sisestada palju erinevaid, kohati üksteisega sobimatuid või vastukäivaid sisendparameetreid. Samas ei olnud huvipooled eelmise lahenduse poolt genereeritud pakkumistega rahul, sest need ei vastanud kõigile parameetritele. Reach-U poolse ärinalüütiku eesmärgiks ei olnud osatada, et võib-olla ei suutnud eelmine süsteem sisendparameetritele vastavat tulemust tagastada, kuna seda ei olnudki võimalik saavutada, vaid leppida kokku mõttekad

parameetrid SSMP jaoks ja teisendada need tarkvaranõueteks, arvestades, kas need on üheselt kodeeritavad. Vajadusel konsulteeriti kogenud arendajatega.

Platvormi disain sai alguse Kliendi keskkonnas kasutajaliidese nõuete kirjeldamisest. Kliendi esindajate poolt tehti teatavaks, et loomisel on kasutajaliides, milles kasutaja saab sisestada järgmised parameetrid kasutajasõbralikul viisil. Reach-U ärianalüütik, kes pidi välja selgitama tagasüsteemi (back-end) nõuded, kaasati sellesse protsessi ajalise viitega, kui kasutajaliidese esialgne disain oli prototüüpimistarkvaras Figma juba valmis ja kasutajaliidese arendaja tööd alustanud. Autoril oli siiski kaasarääkimisõigus, tänu millele muudeti vajadusel disaini selliselt, et parameetrid saadeti tagasüsteemile sobival kujul ja neile vastavad funktsionaalsused olid tarkvaraliselt teostatavad.

Autor koondas koosolekutel tehtud märkmed, e-mailide kaudu saadud erinevad nõuded ja funktsionaalsuste kirjeldused ühte kausta, mis sisaldas 58 dokumenti PG nõuete kohta, lisaks koosolekutel tehtud märkmed ja kümned näidised potentsiaalsetest sisendandmetest. Koosolekutel Kliendi esindajatega selgus, et osad esitatud nõuetest olid vasturääkivad, mittetäielikud või aegunud. Puudus ülevaade, kas nõuete komplekt on terviklik ja piisav.

3.8 Nõuetetehnika protsessi tarbeks sobiva töövahendi loomine

Autori töökeskkonnas polnud kasutusel sobivat töövahendit ega tarkvara nõuete koondamiseks, süstematiseerimiseks, valideerimiseks ja haldamiseks. Nõudeid oli seni kirjeldatud loomulikus keeles kirjutatud tekstidokumentides standardiseerimata ja süstematiseerimata kujul ning nende põhjal toimus soovilogi hoidmine ning järgnevaks sprindiks kandidaatide valik.

Kuna SSMP projekti nõudeid oli kavantitatiivselt mahus, mida ei olnud võimalik hallata mittesüsteemselt, tekkis vajadus töövahendi järgi, mis seda võimaldaks. Nõuete väljaselgitamise ja haldamise toetamiseks otsustati võtta kasutusele sobiv tarkvaralahendus.

Projekti iseloomust tulenevalt püstitati nõuete haldustarkvarale järgmised nõuded. Nõuete prioritiseerimisel on kasutatud teoreetilises osas viidatud MoSCoW mudelit:

1. peab olema kasutusele võetav maksimaalselt kahe nädala (10 tööpäeva) jooksul;
2. peab võimaldama suure hulga nõuete haldamist (rohkem kui 100);
3. peab olema võimalik online- koosolekute ajal ekraani jagades teha jälgitavaks ja hoomatavaks kõigile osalevatele huvipooltele;
4. uusi nõudeid ja muudatusi ning täiendusi olemasolevatesse nõuetesse peab saama sisse viia veebikoosolekutel reaalajas kõigi osalevate huvipoolte silme all;
5. peab toetama järgmisi nõuetetehnika protsessi etappe:

    a. nõuete väljaselgitamine
    b. nõuete kategoriseerimine
    c. nõuete analüüs
    d. nõuete läbirääkimine
    e. konfliktide ja duplikaatide haldamine
    f. nõuete valideerimine
    g. nõuete haldus
    h. nõuete jälgimine
    i. nõuete prioritiseerimine
    j. arendusprotsessi etapi skoobis olemine või mitteolemine
6. peab võimaldama filtreerida nõudeid vastavalt:
    a. nõude kategooriale
    b. nõude staatusele
    c. nõude prioriteedile
    d. tähtaja või tarne skoobis olemisele või mitteolemisele
    e. nõudele rakenduvatele piirangutele
7. peab toetama piirangute rakendumise määramist ja visualiseerimist nõuetele;
8. peab kasutajale andma kiire ja selge ülevaate nõuetest ja nende haldusprotsessi hetkeseisust;
9. peaks olema hõlpsasti jagatav e-maili või pilveteenuste kaudu (nt. Google Drive);
10. peaks olema lihtne kasutada ja arusaadav ka mitte-IT taustaga huvipooltele;
11. võiks olla kasutatav ka abivahendina projektijuhtimisel;
12. võiks võimaldada nõuetele järgmiste parameetrite lisamist ning nende alusel filtreerimist:
    a. arenduse progress
    b. tarne staatus
    c. viide arendusülesandele (nt. Jira task)
    d. planeeritav tarne kuupäev
    e. tegelik tarne kuupäev
13. Tahaks olla integreeritud projekti-ja versioonihaldustarkvaraga (Jira, Confluence jne).

Alapeatükis 2.10 on toodud näited nõuete halduses kasutusel olevatest valmistarkvaradest. Autor ei pretendeeri toodud loetelu ammendavusele ega keskendu erinevate tarkvarade võrdlusele, kuna see oleks eraldi uurimistöö teema. Viidatud on vaid mõnele, et tuua välja põhjused, miks

käesoleva töö raames ei otsustatud spetsiaalselt nõuete halduseks mõeldud valmistarkvara kasuks, vaid otsustati lihtne lahendus ise luua.

Autor ostis isikliku raha eest **Enterprise Architecti** aastase tudengilitsentsi, kuna EA pakkus huvi põhjusel, et käesoleva magistritöö aluseks oleva tarkvaralahenduse puhul on tegemist suhteliselt suure ja keeruka projektiga Reach-U kontekstis. EA kogemust saaks sobivusel ka teiste Reach-U arenduses olevate platvormide haldamisel kasutusele võtta.

Pärast tarkvaraga tutvumist kahe päeva jooksul jõudis autor tõdemusele, et aega ja inimressurssi Reach-U-s selle kasutusele võtmiseks, seadistamiseks ning vajaliku infoga täitmiseks ei ole ning seda poleks ka projekti tähtajad võimaldanud.

Jama Connecti kasutuselevõttu ei kaalutud selle tarkvaraülevaadetes välja toodud keerukuse ning kõrge litsentsitasu tõttu.

Atlassin Jira on Reach-U töökeskkonnas ülesannete halduses, vea- ja projektijuhtimises edukalt kasutusel juba pikka aega, kuid spetsiifiliselt nõuete haldamiseks seda ei kasutata, vaid soovilogist võetud nõuded esitatakse Jiras juba ülesandena.

Jira nõuete mooduli R4J litsentsi Reach-U soetatud ei ole ja see ei ole ka plaanis. R4J on saadaval vaid Jira pilveversioonile. Reach-U-s on kasutusel Jira serveriversioon ja seda enam Atlassin ei toeta, mistõttu juhtkond on võtnud vastu otsuse Jirast lähitulevikus loobuda. Ka R4J hinnastus ei ole võrreldes sellest eeldatava kasuga mõistlik.

Confluence on Reach-U-s samuti kasutusel, aga enamus Reach-U analüütikuid eelistab Confluencele Google Docs keskkonda, sest seal on võimalik luua ja hallata erinevat tüüpi dokumente, neid vajadusel jagada või saata e-maili teel erinevate klientide huvipooltele, kellel juurdepääs konkreetsele Confluencele puudub, ja saada kommenteeritud tagasisidet.

Confluence on kasutusel ka Kliendi ärikeskkonnas, aga pikka aega polnud autoril sellele juurdepääsu, samuti ei ole kliendil juurdepääsu Reach-U Jirale ega Confluencele.

Confluence mittekasutamise otsus võeti vastu nii autori isiklike eelistuste, ettevõttes üldiselt tavaks oleva praktika kui ka koostöövõimaluse toe puudumise tõttu.

Lisaks ei vasta ükski viiest enamusele ülalpool esitatud nõuetest.

Kokkuvõttes ei olnud käesoleva töö raames spetsiifilise nõuete haldustarkvara kasutuselevõtmiseks piisavalt aja- ega finantsressurssi. Tarkvaraprojektide eluiga on tänapäeval suhteliselt lühike ja sõltub äriprotsessidest ning nende muutumisest. Suurte ja suhteliselt staatiliste IT süsteemide, nagu ettevõtte tuumikinfosüsteemid ja ERP tarkvarad, mida kasutavad paljud ettevõtted, on multifunktsionaalse nõuete haldustarkvara kasutuselevõtt õigustatud, aga agiilses arenduses, kus piiratud ressurssidega soovitakse pidevat ja kiiret tulemust, on need süsteemid kohmakad ja käigushoidmine ressursimahukas.

Tulenevalt eelnevast otsustati ise luua kontoritarkvaras MS Excel töövahend, mis vastaks esitatud nõuetele, oleks kiiresti välja töötatav ja rakendatav ning vastaks kujunenud olukorras kõige paremini projekti vajadustele. Oluliseks sai lihtsus, kerge muudetavus, jagatavus nii ekraanil kui faili kujul ning kohandatavus.

Nimetagem seda Nõuetetehnika kombineeritud töövahendiks (NTKT), (inglise keeles *Requirements Engineering Combined Tool (RECT)*).

Kõige rohkem sarnanes esialgne töövahend Nõuete jälgimise maatriksiga (RTM), aga selle algne eesmärk oli teine – kattuvate nõuete koondamine, süstematiseerimine, konfliktsete nõuete väljaselgitamine ja nõuete läbirääkimine Kliendiga MVP seisukohast – millised nõuded peavad ja saavad olla MVP skoobis, millised saab edasi lükata teise faasi ja millised nõuded jäävad ootama järgmisi arendustsükleid, kuid peaksid lõpplahenduses siiski täidetud saama.

Esialgses NTKT-s olid järgmised veerud, mis on kirjeldatud järgnevas tabelis (tabel 2).

Tabel 2. Esialgse NTKT versiooni veergude kirjeldused

ID	***Veeru nimi***	***Sisestatava info kirjeldus ja võimalikud väärtused***	***Kaetavad funktsio-naalsused***
B	**Kategooria**	Näitab, millisesse üldisemasse nõuete kategooriasse konkreetne nõue kuulub. Algselt polnud see väli kõigi nõuete puhul täidetud, vaid lisati kategoriseerimise käigus. Aitab nõudeid süstematiseerida ja filtreerida.	nõuete analüüs, süstemati-seerimine, filtreerimine
C	**Nõude lühikirjeldus**	Nõude lühikirjeldus huvipooltele arusaadaval kujul. Tegemist ei ole nõude ammendava kirjeldusega ega kasutajalooga (*user story*), vaid huvipooltele üheselt arusaadava nimetusega, mis kirjeldab tarkvara funktsionaalsust, parameetrit, reeglit, piirangut või integratsiooni. Nõuete detailsed kirjeldused olid eraldi dokumendis, mille käsitlus jääb käesoleva töö raamidest välja.	nõuete välja-selgitamine, analüüs
D	**Nõude staatus**	Näitab, milline on nõudega tegelemise seis. Kasutatud on tingimuslikku vormindust, mis aitab erinevas staatuses nõudeid visuaalselt eristada.	nõuete haldus, filtreerimine,

		Võimalikud rippmenüüst valitavad variandid on: **duplikaat** – nõue dubleerib kas osaliselt või täielikult mõnda teist nõuet ja see olukord on vaja lahendada; **esitatud** – nõue on Kliendi või analüütiku poolt esitatud ja nimekirja lisatud, aga sellega ei ole veel tegeldud; **heaks kiidetud** – nõudega seotud info on kõigi seotud huvipoolte poolt heaks kiidetud ja seda kas rakendatakse või ei rakendata kokkulepitud projekti faasis; **konfliktne** – nõue on vastuolus kas ühe või mitme teise nõudega ja see olukord vajab läbirääkimisi huvipoolte vahel ning lahendust; **ootel** – on võetud vastu otsus, et selle nõudega hetkel ei tegelda ja selle juurde pöördutakse tagasi kunagi hiljem, kui huvipooled selle tõstatavad; **pole täidetav** – nõuet ei ole võimalik täita ja huvipooled on sellest aru saanud ning ühel meelel, isegi kui leidub huvipooli, kes peavad seda äriliselt vajalikuks. Nõue jääb siiski nimekirja, et seda mõni huvipool jälle ei tõstataks või selleks, et tingimuste ja võimaluste muutumisel selle nõude rakendamise võimalust uuesti analüüsitaks; **töös** – nõudega seonduvaga parasjagu tegeldakse, see on kas täpsustamisel, rakendatavuse väljaselgitamisel, arendusetapi määratlemisel jne; **vajab kliendi tegevust** – vajab kliendipoolseid tegevusi nt. siseringis arutamist, äriprotsesside muutmist, täiendavaid andmeid, kolmandate osapooltega kooskõlastamist jms.	ülesannete omistamine

M1	**MVP skoobis**	On kokku lepitud, kas nõude täitmine on või ei ole minimaalse elujõulise toote skoobis ning seotud funktsionaalsuse testitud ja verifitseeritud tarne peab olema tehtud hiljemalt enne kuupäeva, mil rakendus tehakse reaalsetele kasutajatele kättesaadavaks. Kui vastus on jaatav, värvub lahter heleroheliseks, eitava vastuse korral helepunaseks ja vastuse puudumisel silmatorkavalt helekollaseks, indikeerides, et otsus on vaja langetada. jah/ei	nõuete haldus ja skoop
M2	**Faas 2 skoobis**	Näitab, kas nõuet rakendatakse MVP-le järgnevas faasis (eelduseks on siiski projekti jätkamine st MVP edukaks tunnistamine ja jätkuv finantseering). jah/ei	nõuete haldus ja skoop
M3	**Lõpptoote skoobis**	Näitab, kas kliendi nägemuse kohaselt võiks nõue olla lõpliku lahenduse skoobis isegi, kui selle teostatavuse ja ajakava suhtes puudub hetkel piisav informatsioon (põhimõtteliselt sooviloend). jah/ei	nõuete haldus ja skoop
N	**Märkused, kommen-taarid**	Siia lisatakse vajadusel kommentaar, millistel tingimustel on konkreetne nõue rakendatav, millised on piirangud ja lisatingimused või põhjendus, miks ei ole antud nõuet võimalik või vajalik rakendada.	nõuete haldus
O1	**Piirang - tarbijarühm (TR)**	Piirang, tingimus, millisel juhul antud nõuet saab rakendada AV-andmevaluuta, EOA - esimese osapoole andmed, mõlemad - kohaldatav mõlemal juhul	nõuete analüüs ja läbirääkimine
O2	**Piirang - eelarve/ kontaktid**	Piirang, tingimus, millisel juhul antud nõuet saab rakendada - Eelarve, kontaktid, mõlemad	nõuete analüüs ja haldus

Järgneval joonisel (joonis 7) on nõuete väljaselgitamise ja prioritiseerimise töövahendi esimese iteratsiooni näide, mida kasutati nõuete väljaselgitamise faasis ja sisaldab ülalpool kirjeldatud veerge.

Projekt	Faas
SSMP	Nõuete kogumise faas
Ärianalüütik	Projektijuht
John Doe	Jane Doe
john.doe@company.com	jane.doe@company.com

Legend:	
jah	kokku lepitud, on projekti skoobis
ei	kokku lepitud, skoobist väljas
	otsust ei ole veel tehtud

Kategooria	Nõude lühikirjeldus	Nõude staatus	MVP skoobis	Faas 2 skoobis	Lõpptoote skoobis	Märkused, kommentaarid, kitsendused	Optimeeritav tarbijarühm (TR)	Kohaldatav mahunõudele eelarve/ kontakti
Mahu sisendparameetrid	Tellimuse eelarve $	Heaks kiidetud	jah	jah	jah	Eelarvet ja kontaktide arvu ei saa kasutajaliideses samaaegselt defineerida ja API kaudu pärida	Mõlemad	Eelarve
Mahu sisendparameetrid	Kontaktide arv Andmevaluuta (AV) alusel	Esitatud			jah	"---"	AV	Kontaktid
Mahu sisendparameetrid	Kontaktide arv esimese osapoole andmete (EOA) alusel	Vajab kliendi tegevust			jah	"---"	EOA	Kontaktid
Esmatasandi sisendparameetrid	Tellimuse ID	Esitatud	jah	jah	jah		Mõlemad	Mõlemad
Esmatasandi sisendparameetrid	Tarneperioodi algus	Heaks kiidetud	jah	jah	jah		Mõlemad	Mõlemad
Esmatasandi sisendparameetrid	Tarneperioodi lõpp	Heaks kiidetud	jah	jah	jah		Mõlemad	Mõlemad
Esmatasandi sisendparameetrid	Tootekoguste valik (tk, kg, l)	Vajab kliendi tegevust		jah	jah		Mõlemad	Mõlemad
Esmatasandi sisendparameetrid	Sihtturu tüüp (Rahvuslik/kohalik)	Duplikaat	ei	ei	ei	Lahendatakse sihtturu valikuga	Mõlemad	Mõlemad
Esmatasandi sisendparameetrid	Sihtturu valik	Töös	jah	jah	jah		Mõlemad	Mõlemad
Esmatasandi sisendparameetrid	Müüja kategooria (Multimart/Partner)	Ootel	ei	jah	jah		Mõlemad	Mõlemad
Esmatasandi sisendparameetrid	Esmase tarbijarühma (TR) valik (AV/EOA)	Heaks kiidetud	jah	jah	jah		Mõlemad	Mõlemad
Esmatasandi sisendparameetrid	Esmane TR AV	Heaks kiidetud	jah	jah	jah	ainult vanus/sugu	Mõlemad	Mõlemad
Esmatasandi sisendparameetrid	Teisene TR EOA	Heaks kiidetud	jah	jah	jah		Mõlemad	Mõlemad
Esmatasandi sisendparameetrid	Teisene TR AV	Heaks kiidetud	jah	jah	jah		Mõlemad	Mõlemad
Esmatasandi sisendparameetrid	Teisene TR EOA	Konfliktne	ei	ei	TBD		Mõlemad	Mõlemad
Teise tasandi sisendparameetrid	Esmane TR kontaktide hulk (+-10%)	Ootel	ei	jah	jah		Mõlemad	Eelarve
Teise tasandi sisendparameetrid	Esmane TR- minimaalne indeks	Ootel	ei	jah	jah		Mõlemad	Mõlemad
Teise tasandi sisendparameetrid	Esmane TR hind 100 kontakti kohta (+-10%)	Ootel	ei	jah	jah	Ei ole kohandatav EOA puhul	AV	Mõlemad
Teise tasandi sisendparameetrid	Teisene TR kontaktide hulk (+-10%)	Ootel	ei	jah	jah		AV	Eelarve
Teise tasandi sisendparameetrid	Teisene TR minimaalne indeks	Ootel	ei	jah	jah		Mõlemad	Mõlemad
Teise tasandi sisendparameetrid	Teisene TR CPM - Goal (+-10%)	Ootel	ei	jah	jah		AV	Mõlemad
Optimeerimise eesmärk	Esmane TR - Maksimeerida kontaktide hulk (vaikimisi)	Heaks kiidetud	jah	jah	jah		EOA	Eelarve
Optimeerimise eesmärk	Esmane TR - Maksimeerida indeks	Heaks kiidetud	jah	jah	jah		EOA	Mõlemad
Optimeerimise eesmärk	Esmane TR - Maksimeerida TR katvus	Vajab kliendi tegevust			jah		EOA	Mõlemad
Optimeerimise eesmärk	Teisene TR - Maksimeerida kontaktide hulk	Heaks kiidetud	jah	jah	jah		AV	Eelarve
Optimeerimise eesmärk	Teisene TR - Maksimeerida indeks	Heaks kiidetud	jah	jah	jah		AV	Mõlemad

Joonis 7. Nõuete läbirääkimise töövahendi NTKT esimene versioon (autori kuvatõmmis)

Nõude staatuse kategoriseeringu võttis autor veidi muudetud kujul üle sarnasest töövahendist, mis on leitav ettevõtte Smartsheet kodulehel, kus pakutakse dokumendipõhju erinevate ülesannete lahendamiseks (Smartsheet 2024).

Järgmisel etapil toimus nõuete prioritiseerimine MVP raames ja sellele vastavalt täiendati ka tabelit, mille käigus lisati tabelis 3 kirjeldatud väljad nõuete analüüsi, halduse ja tarne protsessi tarbeks.

Tabel 3. NTKT tabelisse lisatud täiendavate veergude kirjeldused

Veeru ID	***Veeru nimi***	***Kirjeldus ja võimalikud väärtused***	***Kaetavad funktsio-naalsused***
A	**Nr**	Nõude number, mille alusel seda identifitseerida ja sellele viidata.	nõuete identifit-seerimine
E	**Kasutajaloo ID/allikas**	Viide nõudega seotud kasutajaloole või muule nõude allikale. Kasutajalood saab kirjutada ka otse samasse tabelisse, aga antud juhul ei olnud see mõistlik ega vajalik, sest Kliendi nõudel sõnastati kasutajalood kliendipoolsete analüütikute poolt ja hoiti kliendi Confluences, et Kliendil oleks ülevaade nõuetest ning võimalus neid testida ja verifitseerida.	nõuete haldus ja dokumen-teerimine

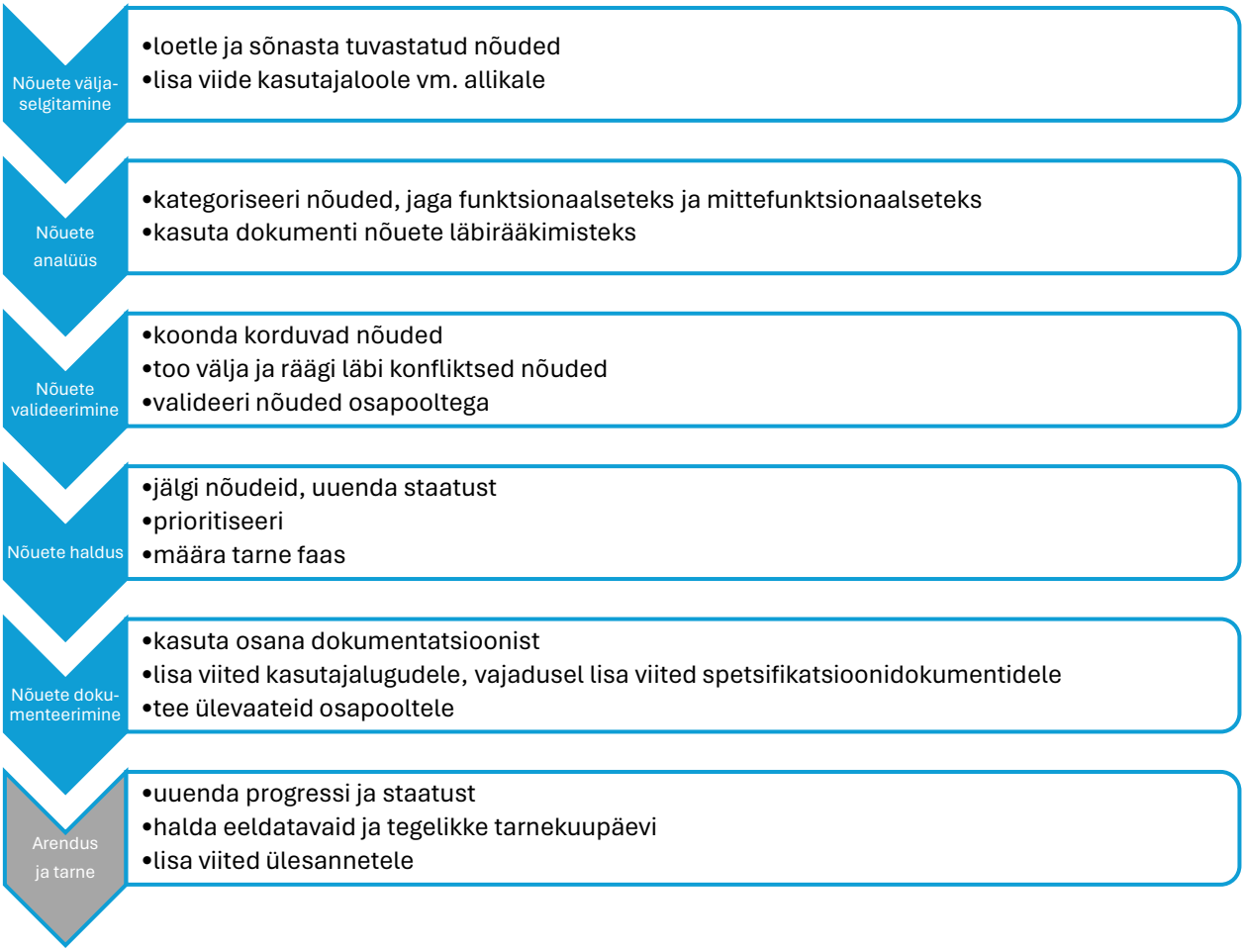
F	**Prioriteet**	Läbirääkimiste käigus määratud prioriteet nõudele, selle olulisus ja täitmise järjekord. kõrge keskmine madal pole oluline	nõuete haldus ja prioriti-seerimine
G	**Progress**	Saab lisada hinnangulise protsendi, kui palju nõudega seonduva omaduse tarnimiseks vajalikust tööst on tehtud.	nõuete haldus
H	**Tarne staatus**	Informatsioon nõudele vastava funktsionaalsuse arenduse kohta. **pole alustatud** – funktsionaalsuse arendamist pole veel alustatud; **töös** – funktsionaalsus on arendamisel; **valmis** - funktsionaalsus e arendus on lõppenud ja valmis tarneks; **ootel** – arendus on ootel, kuna vaja on täiendavat informatsiooni või ressurssi; **tähtaeg üle** – funktsionaalsus ei saanud tarnetähtajaks valmis; **vajab ülevaatamist** – nõue või lahenduse kirjeldus vajab arendaja tagasiside põhjal ülevaatamist ja täpsustamist; **testimisel** - funktsionaalsus on testimisel.	arendus ja tarne
I	**Eeldatav tarne kuupäev**	Planeeritud tarne kuupäev.	arendus ja tarne
J	**Tegelik tarne kuupäev**	Kuupäev, mil toimus tegelik tarne, vastavalt kokkuleppele kliendiga kas kliendi testkeskkonda või tootekeskkonda.	arendus ja tarne
K	**Sprint**	Vajadusel saab lisada sprindi numbri, mille jooksul funktsionaalsust arendatakse	arendus ja tarne
L	**Ülesanne**	Soovi korral saab lisada ka arenduse haldustarkvara ülesande (näiteks Jira taski) numbri, aga käesoleva lahenduse puhul seda ei tehtud.	arendus ja tarne

Valminud töövahendi NTKT kuvatõmmis on toodud Lisas 1.

Oluliseks põhjuseks, miks on töövahendit mõttekas hoida Excelis või sellega hästi ühilduvas Google Sheets'is, on võimalus kasutada ja lisada sisseehitatud visualiseerimisvahendeid, nagu tingimuslik vormindamine ja Gantti graafik sprintide tähtaegade visualiseerimiseks.

Töövahend MS Excel formaadis, mille sisus on toodud käesolevas töös toodud fiktiivse tarkvaralahenduse nõuete analüüsi tulemus, on allalaetav sellelt Google Drive lingilt ja manustatud käesoleva töö eraldioleva lisana.

Kokkuvõtvalt katab NTKT töövahend nõuetetehnika protsessi järgmised etapid nagu toodud järgneval joonisel (joonis 8).



Joonis 8. Funktsionaalsused nõuetetehnika protsessis, mida NTKT katab, autori joonis

Järgnevalt on toodud reeglid ja protseduurid, mida peeti oluliseks silmas pidada tabeli täitmisel. Arendus-ja tarneprotsessi käigus oli kasulik, et NTKT tabelisse lisati ka huvipoolte poolt välja toodud nõuded, mis ei jäänud nõuete analüüsi tulemusena skoopi. See aitas lahendada olukordi, kus pärast tarnet huvipool võib väita, et „ma eeldasin, et selline funktsionaalsus tuleb“ või „ma pidasin selle nõude täitmist iseenesestmõistetavaks“.

NTKT saab kasutada väleda arendusprotsessi soovilogi hoidmise vahendina, kuhu saab juurde lisada uusi nõudeid ning näidata, millises sprindis selle eeldatav tarne võiks olla. Konkreetse projekti raames on, soovilogi nõuded määratud täitmiseks faasis 2 või lõpptootes ja neid lisandub veel. Vajadusel saab lisada sprindi tulba, kus määratakse, millise konkreetse sprindi tarnes sellele nõudele vastav funktsioon peaks sisalduma.

Kuigi projektijuhtimine ja tarneprotsess ei oleva käesoleva töö skoobis ning arenduse, projekti ja tarnete haldus toimub eraldi tarkvaras, on mõttekas lisada tarnega seotud info ka NTKT tabelisse,

sest selle kaudu saab Kliendile, kellel puudub ligipääs arenduskeskkonnale, anda jooksvalt ülevaadet nõuetega seonduvate funktsionaalsuste arendus- ja tarneprotsessi seisust.

4 Tulemused, arutelu

Käesolevas peatükis võtab autor kokku töö tulemused ja analüüsib, kas leitud lahendus vastas nõuetele ja püstitatud eesmärkidele ning kuidas hindas töö tulemuslikkust klient.

4.1 NTKT kasutuselevõtuga saavutatud tulemus

NTKT tabeli esialgse versiooni loomiseks kulus 2 tööpäeva, mis kompenseeris sobiva töövahendi otsinguks kulunud aja ning jäi nõuetes toodud kahe töönädala piiresse. Tabelile lisafunktsioonide tekitamine võttis kokku veel umbes 1,5 tööpäeva. Esialgse nõuete nimekirja tabelisse koondamine erinevatest allikatest võttis 2-3 tööpäeva. Nõuete tervikliku kogumi loomine ja oluliste huvipoolte kinnituse saamine võttis pärast tabeli kasutusele võtmist neli nädalat, mis on lühem kui eelnev nõuete läbirääkimisele kulunud aeg. Pärast NTKT tabeli kasutuselevõtmist kiirenes nõuete väljaselgitamise ja analüüsi töötempo oluliselt. Kui projekti varases staadiumis suudeti tunnise koosoleku jooksul läbi arutada 2-3 nõuet, siis pärast tabeli kasutuselevõttu käidi nõuded järjestikku läbi ja määrati neile staatus. Kui nõue olid selge, kõigile arusaadav ning aktsepteeritav, siis märgiti staatuseks „Heaks kiidetud“. Kui selgus, et nõude puhul on vaja täiendavaid tegevusi, siis määrati staatuseks, kes sellega tegeleb ja mindi järgmise juurde. Ühe koosolekuga käidi kõik kogutud nõuded üle, määrati jooksvad staatused ja otsustati, kes millega järgnevalt tegeleb. Järgmisel koosolekul lauale jäänud nõuete hulk oli juba oluliselt väiksem ja osaga neist oli juba vahepeal tiimide siseselt tegeletud, mis tähendas, et iga koosolekuga kahanes mitteheakskiidetud nõuete hulk oluliselt. Kui vahepeal olid ilmnenud uued nõuded, siis need lisati. NTKT tabeli kasutuselevõtuga muutus nõuete analüüsi protsess iteratiivseks, mis sobib väledasse arendusprotsessi. See näitab, et klassikaliste nõuetetehnika meetodite süsteemne rakendamine kiirendab nõuete haldamise protsessi ja parandab kvaliteeti.

Üks NTKT töövahendi funktsioone oli nõuete terviklikkuse kontroll. Pärast dokumendi ülevaatust oluliste huvipoolte poolt ja kinnitust, et tabelis esitatud MVP nõuete loetelu on täielik, lisati tabeli PDF MVP tööülesannete kirjelduse (Statement Of Work, SOW) dokumendile ja allkirjastati Reach-U ning Kliendi vastutavate huvipoolte poolt.

Esialgselt oli PG-le ette nähtud piiratud eelarve. PG kogueelarve kujunes oluliselt suuremaks kui klient oli algselt eeldanud, sest nõuete väljaselgitamise ning analüüsi käigus selgusid MVP tegelik skoop ning selleks vajaliku arenduse maht ja keerukus. Seda peab autor oluliseks nõuete väljaselgitamise etapi kõrvaltulemiks. Puhtväleda lähenemise korral oleks tõenäoliselt jõutud olukorda, kus raha oleks teatud sprintide järel otsa saanud (esialgne eelarve moodustas umbes

viiendiku MVP kogueelarvest) ja ning eelarve suurendamine projekti käigus arvestades, et tegemist on suurkorporatsiooniga, oleks olnud keeruline.

Projekti esialgselt määratud eelarvet otsustati Kliendiga kokkuleppel kasutada prototüübi ehk POC loomiseks. Nõuete väljaselgitamisega paralleelselt algas ka POC arendusprotsess. Arendusmeeskonna poolt võeti vastu otsus teha POC selline, et selle koodi oleks võimalik hiljem pärisrakenduse tuumana kasutada. POC ei võtnud arvesse kõiki sisendparameetreid ja nõudeid, vaid ainult peamised, mis aitasid pakkumise piiritleda ja tagastas optimeeritud pakkumise peamiste väljundparameetritega. PG põhifunktsionaalsuse kood oli loodud. Edasi sai järk-järgult arendada lisafunktsionaalsusi, vastavalt sellele, kuidas nõuded selgusid ja täpsustusid.

Tagasüsteemi PG MVP tarne toimus kokkulepitud ajal ja mahus, jättes kliendile aega ka kasutajapoolseks testimiseks ning nõuete verifitseerimiseks. Seega said käesoleva töö peatükis 3 esitatud uurimisküsimused positiivse vastuse. Klassikaliste nõuetetehnika meetodite süsteemne rakendamine kiirendas nõuete haldamise protsessi ja NTKT tööriist aitas kaasa MVP tarnele üheksa kuu pärast projekti algust korrigeeritud eelarve piires.

Hinnanguliselt aitas tabeli juurutamine projektile kuluvat aega lühendada vähemalt ühe kalendrikuu võrra. Täpsemat mõõdikut on keeruline tuua, sest puudub võrdlusmoment analoogsete projektidega, aga varasemate arendusprojektidega võrreldes, milles autor on osalenud, oli projekti mahu ja kulutatud aja suhe oluliselt parem.

COVID-19 tingimustes toimus osapoolte vaheline suhtlus veebikoosolekute teel – töövahendit sai üle vaadata ja täiendada reaalajas huvipoolte silme all ja sellega tagada, et huvipooled olid kursis tehtavate muudatustega ning said vajadusel sekkuda või küsida, kui midagi jäi arusaamatuks. Nõuete aktsepteerimine või tagasi lükkamine, skoopi või soovilogisse määramine ning prioritiseerimine toimus kõigi osalevate huvipoolte silme all. See vastab uurimisküsimusele, mis puudutab koostööd ja kommunikatsiooni huvipoolte vahel.

Nõuete väljaselgitamise protsessi tugevuseks antud projekti raames peab autor ka andmekeskset lähenemist ja sünteetiliste andmete kasutamist arendusprotsessis, mis kiirendas rakenduse valmimist ning vähendas andmete erinevusest tulenevat ümbertegemise vajadust arendusfaasis.

Loodud töövahend on reaalses kasutamises tõestatud abivahend analüütikutele (ja hilisemas faasis ka projektijuhtidele) nõuete kogumiseks, koondamiseks, läbirääkimiseks, süstematiseerimiseks, prioritiseerimiseks, kasutuslugude lisamiseks ning arendusprotsessi ja tarnete planeerimiseks. Töövahend võiks olla eriti sobiv kasutamiseks olukorras, kus puudub eelnev ühtne nõudeid kirjeldav dokument, ühine koostöötarkvara nõuete haldamiseks ja/või tellija poole esindajateks on huvipooled, kellel puudub spetsiifiline haridus ja kogemus IT projektidega tegelemiseks.

Autori hinnangul on NTKT kergelt kasutusse võetav, lihtne kasutada, paindlik, ühendab endas erinevad nõuetetehnika protsessiks vajalikud funktsionaalsused ja võimaldab sama töövahendit kasutada arendusprotsessi erinevates etappides.

4.2 Kliendi rahulolu tarnitud tarkvaralahendusega

Reach-U juhtimissüsteemi käsiraamatus on kirjas, et ettevõttes peetakse tööd kvaliteetseks siis, kui klient on tulemusega rahul ja see on saavutatud majanduslikult ökonoomselt. Kliendid ei hinda üldjuhul töö alguses fikseeritud tähtajast, skoobist ja hinnast paindumatut kinnipidamist, vaid pigem paindlikkust, kui nende vajadused on muutunud. Kvaliteeti hindab alati klient. Kui vähegi võimalik, analüüsitakse tegevust koos kliendiga nii projekti käigus kui ka lõpus. (Reach-U 2023)
Kliendi poolt esitatud oluline mittefunktsionaalne nõue oli, et PG genereeriks pakkumise oluliselt kiiremini kui eelmine. Eelmine süsteem pöördus erinevate andmeallikate poole, mis on vajalikud pakkumise kokkupaneku protsessi käigus. Tõenäoliselt oli see põhjuseks, miks pakkumise koostamine võttis eelmisel süsteemil aega 20 minutit kuni 2 tundi. Ootus uuele PG süsteemile oli, et pakkumise genereerimisele kulub vähem kui 10 minutit.
Et pakkumise kokkupanekut kiirendada, võttis arendustiim vastu otsuse, et andmed kogutakse erinevatest allikatest üks kord ööpäevas kokku ja eelgenereeritakse andmebaas PG-le ette kujul, et see oleks võimalikult kompaktne ning kiiresti loetav. Kuna rakendus ja andmed pidid asuma „neutraalsel“ pinnal st. mitte Kliendi omanduses oleval riistvaral, otsustati nii andmete hoiustamiseks ja vahetamiseks kui ka rakenduse virtuaalmasinate platvormiks kasutada Google pilveteenust. PG algoritm pidi samuti olema hästi optimeeritud ja kiire. Selle eest hoolitses arendaja, kellel on varasemate sarnaste projektidega suurandmete kiirel töötlemisel kogemusi.
Esialgne, ilma piiranguteta POC algoritm suutis Pakkumise kokku panna vähem kui 10 sekundiga. Pärast kõigi sisendparameetrite kasutuselevõttu vastas PG tavaliselt vähem kui 30 sekundi jooksul, eriti keeruliste päringute puhul vähemalt kolme minuti jooksul. Vaid kolmandate osapoolte API-de tõrked, mille poole PG pöördus, võisid vastamisaega pikendada, aga see probleem sai lahendatud hiljem, kus pärast kolmanda osapoole API-lt vastuse mittesaamist 120 sekundi jooksul PG väljastab tulemuse ilma kolmandalt osapoolelt saadava parameetri väärtuseta. Seega ületas tarnitud lahendus kiiruse osas Kliendi ootused oluliselt ja selle kohta on tulnud positiivset tagasisidet erinevatelt huvipooltelt.
Samuti on saanud positiivset tagasisidet süsteemi stabiilsus. Juba mitu kuud on Reach-U saanud kliendile raporteerida, et tagasüsteemi kättesaadavus (vastus päringutele saadakse kokkulepitud aja jooksul) on 4 kuud tagasi alanud monitooringuperioodi jooksul olnud 100%.

Käesoleva fiktiivse rakenduse aluseks olev päris tarkvara on olnud kliendi ärikeskkonnas kasutusel juba peaaegu aasta. Positiivse vs. negatiivse tagasiside suhe on selgelt positiivse kasuks. Nädal pärast projekti tutvustamist avalikkusele pidulikul galaüritusel saabusid tänusõnad ka Kliendi valdkonna asepresidendilt. Ühemõtteliselt negatiivset tagasisidet PG kohta saadud ei olegi. Üksikuid vigu avastatakse ja kliendilt tuleb ettepanekuid muudatuste osas, aga see on tarkvaraarenduse loomulik osa. Funktsioonide lisamise arendustöö väledas vormis jätkub. Kliendil on plaanis kaasata platvormile ka teisi sarnase profiiliga ettevõtteid, mis soovivad sarnast lahendust.

4.3 Edasise uurimistöö võimalused

Kuna NTKT puhul on tegemist on laiatarbetarkvaraga MS Excel ja Google Sheets ühilduva tootega, saab töövahendit jooksvalt muuta ja täiendada. Lisada saab tulbad lahendusega tegeleva arendaja nime, testjuhtumite viited, lahenduseks kuluva aja ja eelarve hinnangud, Gantti diagrammi tarnetähtaegade haldamiseks jne. Kindlasti ei ole igas arenduses vaja kõiki välju või on vajadus nende teistsuguse sisu järele. Fail on lihtsasti muudetav ja kasutaja vajadustele kohandatav. Selle saab tõsta Google pilveteenusesse ja jagada linki oluliste huvipooltega, et nad saaksid nõuetetehnika protsessi reaalajas jälgida ja infot lisada või staatuseid muuta.

NTKT oluliseks puuduseks on integratsiooni puudumine arenduse haldusvahendite (nt. Jira, Confluence) vahel. Teoreetiliselt saab tulevikus vastavaid integratsioone siiski luua eksport-import lahendusi juurutades.

Kuna dokumendile on juurdepääs paljudel osapooltel, on keeruline ka nõuete juhusliku või tahtliku muutmise välistamine ühe osapoole poolt ilma kõigi osapoolte nõusolekuta. Seda riski saab vähendada, kui salvestada mingiks vaheetapiks kokku lepitud seis kaitstud PDF vormingus.

Huvitav oleks kasutada sarnases protsessis siiski ka mõnda peatükis 3.10 toodud valmistarkvaralahendust ja võrrelda sel viisil saavutatut käesoleva tulemusega, aga see eeldaks spetsiifilise huvi ja vajaduse olemasolu.

5 Kokkuvõte

Käesoleva töö raames läbi viidud nõuetetehnika protsessi tulemusena valmis planeeritava tarkvaralahenduse terviklik nõuete kogum, mis aitas kaasa toimiva MVP tarnimisele määratud tähtajaks.

Töö käigus loodud nõuetetehnika kompleksne töövahend NTKT:

1. muutis nõuete väljaselgitamise protsessi kiiremaks ja efektiivsemaks kui enne selle kasutuselevõttu;
2. võimaldas nõudeid süstematiseerida, kategoriseerida ning filtreerida;
3. aitas nõudeid läbi rääkida, hallata duplikaate ja konflikte nõuete vahel;
4. toetas nõuete analüüsi protsessi iteratiivseks muutmist;
5. aitas kaasa huvipoolte koostööle COVID-19 tingimustes, kuna oli ekraanil lihtsalt jagatav;
6. võimaldas nõudeid prioritiseerida, määrata skoopi või soovilogisse, tagada nõuete terviklikkust;
7. tekkinud soovilogi nõuetest, mis ei läinud MVP skoopi, saab kasutada järgnevate sprintide sisendina;
8. töövahend leidis kasutust ka arendusfaasis ning projektijuhtimises.
9. oli tagasiside põhjal arusaadav ka mitte-IT taustaga huvipooltele.

Kliendi tagasiside tähtajaks valminud MVP-le oli positiivne ja kliendi rahulolu peetakse Rach-U-s oluliseks tarkvara kvaliteedi hindamise kriteeriumiks.

Töö käigus loodud lihtsa, kuid multifunktsionaalse töövahendi abil saab autor parandada nii oma töö tootlikkust, kvaliteeti kui ka koostööd klientidega. Soovi korral saavad ka teised ärianalüütikud kohaldada töövahendit vastavalt enda vajadustele ja kasutada nõuete analüüsi protsessi haldamiseks.

6 Viidatud kirjandus

Arendussõnastik. 2024. “Arendussõnastik.” Retrieved April 25, 2024 (https://agiil.github.io/sonastik/).

Askarinejad, Zahra. 2012. “Challenges and Weaknesses of Agile Method in Enterprise Architecture.” *International Journal of Computer Science & Engineering Survey* 3:37–45. doi: 10.5121/ijcses.2012.3603.

Chemuturi, Murali. 2012. *Requirements Engineering and Management for Software Development Projects*. 2013th edition. Springer.

Claude AI Model 3. 2024. “Introducing the next Generation of Claude.” Retrieved April 29, 2024 (https://www.anthropic.com/news/claude-3-family).

Damian, D., and J. Chisan. 2006. “An Empirical Study of the Complex Relationships between Requirements Engineering Processes and Other Processes That Lead to Payoffs in Productivity, Quality, and Risk Management.” *IEEE Transactions on Software Engineering* 32(7):433–53. doi: 10.1109/TSE.2006.61.

Davey, Bill, and Kevin Parker. 2015. “Requirements Elicitation Problems: A Literature Analysis.” *Issues in Informing Science and Information Technology* 12. doi: 10.28945/2211.

Enterprise Architect. 2024. “Requirements | Enterprise Architect User Guide.” Retrieved May 11, 2024 (https://sparxsystems.com/enterprise_architect_user_guide/16.1/modeling_domains/requirementsmanagement.html).

Girvan, Lynda, and Debra Paul. 2017. *Agile and Business Analysis: Practical Guidance for IT Professionals*. BCS, The Chartered Institute for IT.

Hull, Elizabeth, Ken Jackson, and Jeremy Dick. 2010. *Requirements Engineering*. 3rd ed. 2011 edition. London Heidelberg: Springer.

IEEE Computer Society, and Software Engineering Standards Committee. 1998. *IEEE 830 :IEEE Recommended Practice for Software Requirements Specifications*. New York, NY: IEEEE Press.

Jocham, Ralph. 2024. “Agile Done Right Eliminates the Need for Classical Requirements Engineering. | Scrum.Org.” Retrieved May 14, 2024 (https://www.scrum.org/resources/blog/agile-done-right-eliminates-need-classical-requirements-engineering).

Laplante, Phillip A. 2017. *Requirements Engineering for Software and Systems*. 3rd edition. Boca Raton London New York: Auerbach Publications.

Lauesen, Soren. 2002. *Software Requirements: Styles and Techniques*. 1st edition. London Munich: Addison-Wesley Professional.

Milani, Fredrik. 2019. *Digital Business Analysis*. Cham: Springer International Publishing.

O'Regan, Gerard. 2017. *Concise Guide to Software Engineering: From Fundamentals to Application Methods*. 1st ed. 2017 edition. New York, NY: Springer.

Paul, Debra, and James Cadle. 2020. *Business Analysis*. 4th ed. edition. BCS, The Chartered Institute for IT.

Perplexity AI. 2024. "Perplexity." Retrieved April 29, 2024 (https://www.perplexity.ai/).

Pohl, Klaus. 2010. *Requirements Engineering: Fundamentals, Principles, and Techniques*. 1st ed. Springer Publishing Company, Incorporated.

Pohl, Klaus, and Chris Rupp. 2015. *Requirements Engineering Fundamentals: A Study Guide for the Certified Professional for Requirements Engineering Exam - Foundation Level - IREB Compliant*. 2nd edition. Santa Barbara, Calif: Rocky Nook.

Ramadan, Nagy, and Salwa Megahed. 2016. "Requirements Engineering in Scrum Framework." *International Journal of Computer Applications* 149:24–29. doi: 10.5120/ijca2016911530.

Rasheed, Aqsa, Bushra Zafar, Tehmina Shehryar, Naila Aiman Aslam, Muhammad Sajid, Nouman Ali, Saadat Hanif Dar, and Samina Khalid. 2021. "Requirement Engineering Challenges in Agile Software Development." *Mathematical Problems in Engineering* 2021:e6696695. doi: 10.1155/2021/6696695.

Reach-U. 2023. "Reach-U Juhtimissüsteemi Käsiraamat." *Google Docs*. Retrieved May 10, 2024 (https://docs.google.com/Reach-U sisedokumendid).

ReqView. 2024. "Requirements Management for Jira | ReqView." Retrieved (https://www.reqview.com/blog/jira-integration/).

Simplilearn.com. 2022. "Jira vs Confluence: Which Is a Better Project Management Tool? | Simplilearn." *Simplilearn.Com*. Retrieved May 13, 2024 (https://www.simplilearn.com/tutorials/project-management-tutorial/jira-vs-confluence).

Smartsheet. 2024. "Requirements Management 101 | Smartsheet." Retrieved April 24, 2024 (https://www.smartsheet.com/content/requirements-management).

Software Advice. 2024. "R4J Software Reviews, Demo & Pricing - 2024." Retrieved May 14, 2024 (https://www.softwareadvice.com/requirements-management/r4j-profile/).

Sommerville, Ian. 2016. *Software Engineering*. Tenth edition, global edition. Boston Columbus Indianapolis New York San Francisco Hoboken Amsterdam Cape Town Dubai London Madrid Milan Munich Paris Montreal Toronto Delhi Mexico City São Paulo Sydney Hong Kong Seoul Singapore Taipei Tokyo: Pearson.

Sommerville, Ian. 2019. *Engineering Software Products: An Introduction to Modern Software Engineering*. 1st edition. Hoboken, NJ: Pearson.

Stec, Albert. 2023. "Software Engineering: V-Shaped Model | Baeldung on Computer Science." Retrieved April 27, 2024 (https://www.baeldung.com/cs/v-model).

The Daily Egg. 2023a. “Confluence Review - The Good and Bad for 2024.” *The Daily Egg*. Retrieved May 14, 2024 (https://www.crazyegg.com/blog/confluence-review/).

The Daily Egg. 2023b. “Jama Connect Review - The Good and Bad for 2024.” *The Daily Egg*. Retrieved May 14, 2024 (https://www.crazyegg.com/blog/jama-connect-review/).

Walia, Gursimran Singh, and Jeffrey C. Carver. 2009. “A Systematic Literature Review to Identify and Classify Software Requirement Errors.” *Information and Software Technology* 51(7):1087–1109. doi: 10.1016/j.infsof.2009.01.004.

Yeates, Don, Debra Paul, Tony Jenkins, Keith Hindle, and Craig Rollason. 2006. *Business Analysis*. London: British Computer Society.

Zeil, Steven J. 2023. “Eliciting Requirements.” Retrieved April 26, 2024 (https://www.cs.odu.edu/~zeil/cs350/latest/Public/eliciting/index.html).

Lisad

LISA 1 Töövahendi NTKT kuvatõmmis

Projekt	Faas
SSMP	Arendusfaas
Ärianalüütik	Projektijuht
John Doe	Jane Doe
john.doe@company.com	jane.doe@company.com

legend:	jah	kokku lepitud, on projekti skoobis
	ei	kokku lepitud, skoobist väljas
		otsust ei ole veel tehtud

A. Nr.	B. Kategooria	C. Nõude lühikirjeldus	D. Nõude staatus	E. Kasutaja-loo ID	F. Prioriteet	G. Progress	H. Tarne staatus	I. Eeldatav tarne kuupäev	J. Tegelik tarne kuupäev	M1. MVP skoobis	M2. Faas 2 skoobis	M3. Lõpptoote skoobis	N. Märkused, kommentaarid	O1. Piirang - tarbijarühm (TR)	O2. Piirang - eelarve/ kontaktid
1	Mahu sisendparameetrid	Tellimuse eelarve $	Heaks kiidetud	SSMPRQ_001	kõrge	100%	valmis	06.07.2023	06.07.2023	jah	jah	jah	Eelarvet ja kontaktide arvu ei saa kasutajaliideses samaaegselt defineerida ja API kaudu pärida	Mõlemad	Eelarve
2	Mahu sisendparameetrid	Kontaktide arv Andmevaluuta (AV) alusel	Esitatud	SSMPRQ_003	keskmine	0%				jah	jah	jah	"---"	AV	Kontaktid
3	Mahu sisendparameetrid	Kontaktide arv esimese osapoole andmete (EOA) alusel	Vajab kliendi tegevust	SSMPRQ_003	keskmine	65%	töös	10.10.2023		jah	jah	jah	"---"	EOA	Kontaktid
4	Esmatasandi sisendparameetrid	Tellimuse ID	Esitatud	SSMPRQ_003	keskmine	0%				jah	jah	jah		Mõlemad	Mõlemad
5	Esmatasandi sisendparameetrid	Tarneperioodi algus	Heaks kiidetud	SSMPRQ_003	kõrge	100%	valmis	04.08.2023	04.08.2023	jah	jah	jah		Mõlemad	Mõlemad
6	Esmatasandi sisendparameetrid	Tarneperioodi lõpp	Heaks kiidetud	SSMPRQ_003	kõrge	30%	töös	10.10.2023		jah	jah	jah		Mõlemad	Mõlemad
7	Esmatasandi sisendparameetrid	Tootekoguste valik (tk, kg, l)	Vajab kliendi tegevust	SSMPRQ_007	kõrge	90%	testimisel	04.08.2023	04.08.2023	jah	jah	jah		Mõlemad	Mõlemad
8	Esmatasandi sisendparameetrid	Turu tüüp (Rahvuslik/kohalik)	Duplikaat	SSMPRQ_003	pole oluline					ei	ei	ei	Lahendatakse turu valikuga	Mõlemad	Mõlemad
9	Esmatasandi sisendparameetrid	Turu valik	Töös	SSMPRQ_003	kõrge	100%	valmis	06.07.2023	06.07.2023	jah	jah	jah		Mõlemad	Mõlemad
10	Esmatasandi sisendparameetrid	Müüja kategooria (Multimart/Partner)	Ootel	SSMPRQ_011	madal					ei	jah	jah		Mõlemad	Mõlemad
11	Esmatasandi sisendparameetrid	Esmase tarbijarühma (TR) valik (AV/EOA)	Heaks kiidetud	SSMPRQ_011	kõrge	100%	valmis	06.07.2023	06.07.2023	jah	jah	jah		Mõlemad	Mõlemad
12	Esmatasandi sisendparameetrid	Esmane TR AV	Heaks kiidetud	SSMPRQ_011	kõrge	70%	töös	10.10.2023		jah	jah	jah	ainult vanus/sugu	Mõlemad	Mõlemad
13	Esmatasandi sisendparameetrid	Teisene TR EOA	Heaks kiidetud	SSMPRQ_011	kõrge	50%	tähtaeg üle	06.07.2023		jah	jah	jah		Mõlemad	Mõlemad
14	Esmatasandi sisendparameetrid	Teisene TR AV	Heaks kiidetud	SSMPRQ_011	kõrge	0%	ootel			jah	jah	jah		Mõlemad	Mõlemad
15	Esmatasandi sisendparameetrid	Teisene TR EOA	Konfliktne	5	pole oluline	0%	pole alustatud			ei	ei	TBD		Mõlemad	Mõlemad
16	Teise tasandi sisendparameetrid	Esmane TR kontaktide hulk (+-10%)	Ootel	SSMPRQ_011	madal	0%	ootel			ei	jah	jah		Mõlemad	Eelarve
17	Teise tasandi sisendparameetrid	Esmane TR- minimaalne indeks	Ootel	SSMPRQ_011	madal	0%	ootel			ei	jah	jah		Mõlemad	Mõlemad
18	Teise tasandi sisendparameetrid	Esmane TR hind 100 kontakti kohta (+-10%)	Ootel	SSMPRQ_011	madal	0%	ootel			ei	jah	jah	Ei ole kohandatav EOA puhul	AV	Mõlemad
19	Teise tasandi sisendparameetrid	Teisene TR kontaktide hulk (+-10%)	Ootel	SSMPRQ_011	madal	0%	ootel			ei	jah	jah		AV	Eelarve
20	Teise tasandi sisendparameetrid	Teisene TR minimaalne indeks	Ootel	SSMPRQ_021	madal	0%	ootel			ei	jah	jah		Mõlemad	Mõlemad
21	Teise tasandi sisendparameetrid	Teisene TR CPM - Goal (+-10%)	Ootel	1	madal	0%	ootel			ei	jah	jah		AV	Mõlemad
22	Optimeerimise eesmärk	Esmane TR - Maksimeerida kontaktide hulk (vaikimisi)	Heaks kiidetud	SSMPRQ_022	kõrge	0%	vajab ülevaatamist	03.11.2023		jah	jah	jah		EOA	Eelarve
23	Optimeerimise eesmärk	Esmane TR - Maksimeerida indeks	Heaks kiidetud	SSMPRQ_021	kõrge	85%	testimisel	04.08.2023	04.08.2023	jah	jah	jah		EOA	Mõlemad
24	Optimeerimise eesmärk	Esmane TR - Maksimeerida TR katvus	Vajab kliendi tegevust	SSMPRQ_024	keskmine	55%	töös	03.11.2023		jah	jah	jah		EOA	Mõlemad
25	Optimeerimise eesmärk	Teisene TR - Maksimeerida kontaktide hulk	Heaks kiidetud	SSMPRQ_021	kõrge	30%	töös	03.11.2023		jah	jah	jah		AV	Eelarve
26	Optimeerimise eesmärk	Teisene TR - Maksimeerida indeks	Heaks kiidetud	SSMPRQ_021	kõrge	100%	valmis	06.07.2023	06.07.2023	jah	jah	jah		AV	Mõlemad
27	Optimeerimise eesmärk	Teisene TR - Maksimeerida TR katvus	Vajab kliendi tegevust	SSMPRQ_027	keskmine	30%	tähtaeg üle	04.08.2023		jah	jah	jah		AV	Mõlemad
28	Ranged tagasüsteemi piirangud	Mitte ületada laoseisu	Heaks kiidetud	8	kõrge	100%	valmis	10.10.2023	04.08.2023	jah	jah	jah		Mõlemad	Mõlemad

Litsents

Lihtlitsents lõputöö reprodutseerimiseks ja üldsusele kättesaadavaks tegemiseks

Mina, Margus Paas,

1. annan Tartu Ülikoolile tasuta loa (lihtlitsentsi) minu loodud teose „Nõuetetehnika meetodite rakendamine väledas arendusprotsessis ja selle tarbeks kombineeritud töövahendi loomine“, mille juhendajad on Toomas Saarsen, PhD ja Teet Jagomägi, MSc, reprodutseerimiseks eesmärgiga seda säilitada, sealhulgas lisada digitaalarhiivi DSpace kuni autoriõiguse kehtivuse lõppemiseni.
2. Annan Tartu Ülikoolile loa teha punktis 1 nimetatud teos üldsusele kättesaadavaks Tartu Ülikooli veebikeskkonna, sealhulgas digitaalarhiivi DSpace kaudu Creative Commonsi litsentsiga CC BY NC ND 3.0, mis lubab autorile viidates teost reprodutseerida, levitada ja üldsusele suunata ning keelab luua tuletatud teost ja kasutada teost ärieesmärgil, kuni autoriõiguse kehtivuse lõppemiseni.
3. Olen teadlik, et punktides 1 ja 2 nimetatud õigused jäävad alles ka autorile.
4. Kinnitan, et lihtlitsentsi andmisega ei riku ma teiste isikute intellektuaalomandi ega isikuandmete kaitse õigusaktidest tulenevaid õigusi.

Margus Paas
15.05.2024