TALLINNA TEHNIKAÜLIKOOL
Infotehnoloogia teaduskond
Tarkvarateaduse instituut

Merit Timmas 162793IABM

# TESTIMISE PROTSESSI VÄLJATÖÖTAMINE RAHANDUSMINISTEERIUMI INFOTEHNOLOOGIAKESKUSE NÄITEL

Magistritöö

Juhendaja: Margus Noormaa
Magistrikraad

Kaasjuhendaja: Gunnar Piho
Doktorikraad

Tallinn 2018

# Autorideklaratsioon

Kinnitan, et olen koostanud antud lõputöö iseseisvalt ning seda ei ole kellegi teise poolt varem kaitsmisele esitatud. Kõik töö koostamisel kasutatud teiste autorite tööd, olulised seisukohad, kirjandusallikatest ja mujalt pärinevad andmed on töös viidatud.

Autor: Merit Timmas

[04.05.2018]

# Annotatsioon

Magistritöö on kirjutatud teemal „Testimise protsessi väljatöötamine Rahandusministeeriumi Infotehnoloogiakeskuse näitel“. Kuna IT arendusprojektides läbiviidavate testimistega seotud tegevusi kajastavad täna erinevad korrad organisatsioonis killustatult ning osade testimiste kontekstis protsesside kirjeldused puuduvad, siis on käesoleva töö ajendiks vajadus uurida ja kaardistada asutuses IT arenduste raames toimuvaid testimisi ning läbi kogutud info luua parendatud testimise protsessid. Selle kaudu panustada testimise tööde sujuvamasse laabumisse IT arendusprojektides.

Käesoleva töö eesmärgid on:

- kaardistada tänane olukord Rahandusministeeriumi Infotehnoloogiakeskuses IT arendusprojektides toimuvate testimistega;
- tuua välja testimise protsessiga seotud probleemkohad testimiste liikide lõikes;
- võttes arvesse tuvastatud probleeme määratleda optimaalne testimise korralduse protsess;
- võttes arvesse testimisega seotud praktikaid ning korraldust teistes sarnastes asutustes, analüüsida tuvastatud probleeme ning väljapakutud parendusettepanekuid.

Eesmärgi saavutamiseks on viidud läbi mitmeid intervjuusid kaasates asutuse seest spetsialiste. Lisaks on uuritud testimise valdkonnaga seotud praktikaid ning IT arendusprojektides toimuvaid testimisi nelja sarnase asutuse lõikes.

Töö tulemusena kaardistab autor tänased Rahandusministeeriumi Infotehnoloogiakeskuse IT arendusprojektides toimuvad testimised ja intervjuude tulemusena selgunud probleemid. Loob parendatud testimise protsessid, analüüsib tulemeid ning toob välja testimise kasu äri aspektist.

Lõputöö on kirjutatud Eesti keeles ning sisaldab teksti 83 leheküljel, 6 peatükki, 15 joonist, 3 tabelit.

# Abstract

Development of software testing process by example of IT Centre of the Ministry of Finance

Due to the aspect that the rules of procedures in IT Centre of the Ministry of Finance reflect only fragments of descriptions that are connected with organizing and carrying out software testing in development projects and some of the procedures are not properly described at all, master's thesis author conducted a survey to investigate software testing and connected procedures in the organization. Considering the information collected, author of this thesis worked out improved software testing process to thereby contribute to have smoother and better workflow in different software development projects.

The main purposes of this master's thesis are:

- to map the current situation in the IT Center of the Ministry of Finance with software testing in IT development projects;
- identify issues related with software testing process;
- taking into account the problems identified, determine the optimal testing process for the organization;
- analyze identified problems and proposed improvements considering software testing practices and arrangements with software testing in other similar organizations.

To achieve above mentioned purposes master's thesis author carried out several interviews within the organization and investigated software testing practices including the practices in four similar organizations.

The result of this thesis author maps the current situation with software testing in the organization and brings out problems connected with the testing process. Author develops improved testing process and analyses the improvement propositions, also brings out the benefit of the software testing for the business point of view.

The thesis is in Estonian and contains 83 pages of text, 6 chapters, 15 figures, 3 tables.

# Lühendite ja mõistete sõnastik

| | |
|---|---|
| Jira | *Atlassian Jira* on veebipõhine arendusprojektide haldamise tarkvara, mis võimaldab jälgida projekti kulgemise protsessi kogu projekti elutsükli vältel, jälgida aruandlust, teatada vigadest, määrata tööülesandeid, teha tellimusi ja täita muid funktsioone, mis on vajalikud ühe projekti töö kulgemiseks. [36] |
| Confluence | *Atlassian Confluence* on veebipõhine tarkvara meeskonna või ettevõtte siseseks informatsiooni ja ressursside jagamiseks. [37] |
| ISTQB | *International Software Testing Qualifications Board.* Rahvusvaheline tarkvara testimise kvalifikatsiooniga tegelev juhtorgan. Asutatud aastal 2002. Kaasates sadu vabatahtlikke testimise eksperte üle maailma- haldab, määratleb ja arendab see juhtfiguur tarkvara testimisega seotud suundi ja võimaldab testijaid sertifitseerida testimise parimaid praktikaid arvesse võttes. [34] |
| SLA | *Service Level Agreement*. Tellija ja teenuse osutaja vaheline teenustaseme kokkulepe. |
| WCAG 2.0 | *Web Content Accessibility Guidelines 2.0.* Standard, mis kehtestab nõuded veebilehekülgede juurdepääsetavusele. WCAG 2.0 nõudeid tuleb järgida, et tagada erivajadusega inimestele samaväärsed võimalused veebi kasutamiseks nagu on teistel kasutajatel [35] |
| CTP | *Critical Testing Processes*. Kriitilise testimise protsess, raamistik testimise protsessi hindamiseks ja parendamiseks. |
| TMMi | *Testing Maturity Model Integration*. Testimisprotsesside parendusmudel. |
| TPI Next | *Test Process Improvement Next*. Testimisprotsessi parendamise mudel ja metoodika. |
| STEP | *Systematic Test and Evaluation Process*. Süstematiseeritud testimise hindamise protsess. Raamistik testimise protsessi hindamiseks ja parendamiseks. |
| RIK | Registrite ja Infosüsteemide Keskus. |
| KEMIT | Keskkonnaministeeriumi Infotehnoloogiakeskus. |
| SMIT | Siseministeeriumi infotehnoloogia-ja arenduskeskus. |
| TEHIK | Tervise ja Heaolu Infosüsteemide Keskus. |
| IKT | Info- ja kommunikatsioonitehnoloogia. |
| SOAP UI | Tööriist veebiteenuste testimiseks. |
| JMeter | Tööriist jõudlustestide tegemiseks. |

| | |
|---|---|
| TestRail | Tööriist testimise haldamiseks, jälgimiseks ja korraldamiseks. |
| Scrum | Agiilne tarkvaraarendamise metoodika. |
| Kogu omamiskulu | *Total Cost of Ownership*. Vara ostu-, installeerimise-, kasutamise-, uuendamise- ja haldamiskulud kokku [38] |
| MFN | Mittefunktsionaalne nõue. |

# Sisukord

# Jooniste loetelu

# Tabelite loetelu

# 1 Sissejuhatus

Tarkvara testimine on oluline osa tarkvaraarenduse projektist. Testimine aitab tagada usaldusväärsema ja kvaliteetsema tarkvara toote. Selleks, et kvaliteeti tagada, peab testimise korraldus olema läbimõeldud, planeeritud ja ka jälgitav. Testimine ei ole ainult testide käivitamine pärast kodeerimise lõpetamist. Testimise protsess hõlmab mitmeid erinevaid tegevusi alates planeerimisest kuni testimise lõpetamiseni.

Tarkvara toote kvaliteet sõltub väga palju toote valmistamise protsessist - tarkvara arendusprotsessi kvaliteedist ning toote arendajate (analüütikute, arhitektide, programmeerijate, testijate, projektijuhtide jt) teadmistest, oskustest ning ka motivatsioonist. Seega üheks tarkvara kvaliteedi tõstmise viisiks on parendada protsesse ja teiseks viisiks on testida lõpptulemust (käivitada koodi). [5]

Süsteemid, kus tarkvaral on oluline roll kanda, muutuvad üha keerukamaks ning kvaliteediküsimusele keskendumine on seetõttu samuti üha olulisem. Uued meetodid, tehnikad ja tööriistad mõjutavad meie tööd ja tegemisi ning seetõttu on loogiline, et aja jooksul tuleb organisatsioonis erinevaid protsesse üle vaadata, kaasajastada ja probleemkohti parendada. Kuna protsessid on tegevuste seeriad, mis aitavad saavutada eesmärke, siis on eriti oluline, et need ka reaalselt töötaks. Kui probleemkohad protsessis lahendatakse, saavad ka eesmärgid kiiremini ja paremini täidetud. Käesoleva magistritöö ajendiks on Rahandusministeeriumi Infotehnoloogikeskuse (edaspidi RMIT) praktiline vajadus kaardistada tarkvaraarenduse projektides toimuv testimise protsess ning tuua välja protsessi kitsaskohad. Kuna arenduste mahud on kasvanud viimaste aastate jooksul, siis on kasvanud ka testimise tegevustega seotud maht ja osakaal organisatsioonis. Tänasel päeval puudub RMITis tarkvaraarendusprojektides toimuvate testimiste protsesse kirjeldav ühtne dokument, testimiste tegevusi kajastavad erinevad korrad killustatult.

Magistritöö uurimisobjektiks on tarkvara testimise protsess ja selle korraldus RMITis. Töö eesmärgiks on kaardistada praeguse tarkvara testimise protsessi probleemkohad ning

luua optimaalne testimise korralduse protsess testimise liikide kaupa. Eesmärgi saavutamiseks on planeeritud järgmised tegevused:

- kaasata ettevõtte seest eksperte tänase olukorra kaardistamiseks seoses tarkvara testimisega;
- määratleda lahendamist vajavad probleemkohad;
- uurida testimise protsessi korraldust teistes sarnastes asutustes;
- kogutud informatsiooni põhjal töötada välja optimaalne testimise korralduse protsess.

RMIT haldusalasse kuuluvad olulised avaliku sektori asutused, kelle tööprotsessides kasutatav tarkvara peab olema kvaliteetne. Avalikkusele osutatavate teenuste kvaliteet sõltub suures osas tarkvara kvaliteedist. Käsitletav teema on oluline, kuna hästi toimiv ning eesmärki täitev testimise protsess on kvaliteetse tarkvara üks tähtsamatest alustest.

Magistritöö koosneb viiest põhiosast – esimeses sissejuhatavas osas on toodud taustinfona ülevaade organisatsioonist ja tarkvara testimisest.

Teine osa käsitleb töö metoodikat – sisaldab uuritava objekti kirjeldust (testimise korraldus organisatsioonis). Lisaks on selles peatükis toodud tegevusprotsess töö eesmärgi saavutamiseks ning ülevaade testimise protsessi hindamiseks kasutatavast mudelist (töös kasutatakse kriitilise testimise protsessi mudelit).

Töö kolmandas osas on välja toodud tänane olukord tarkvara arendusprojektides toimuvate testimistega (fookuses on funktsionaalsete ja mittefunktsionaalsete nõuete testimine sh. koormustestimine, turvatestimine ja koodiaudit). Olukorra kaardistamisel on kasutatud asutuse seest eksperte (erinevate valdkondade IT projektijuhte, administraatoreid, turvaspetsialiste ja ka projektijuhte äri poolelt). Kolmandas osas on toodud ka kokkuvõte IT arendusprojektides toimuvate testimiste kohta nelja sarnase asutuse lõikes.

Neljandas osas on välja toodud ettepanekud testimise protsessi parendamiseks ning parendatud testimise protsessid. Viiendas põhiosas on probleemide analüüs ja järeldused.

Töös on kasutatud 39 allika abi. Töös toodud joonised, tabelid ning lisad illustreerivad käesoleva magistritöö erinevaid osi.

## 1.1 Ülevaade tarkvara testimisest ja tarkvara testimise protsessist

Testimise kontseptisoon, strateegia, tehnikad ja meetmed on võimalik integreerida testimise protsessiks. Testimise protsess toetab testimisega seotud tegevusi ning annab juhised testijale või testijate meeskondadele testimise planeerimisest kuni testimise tulemuste hindamiseni. Tegevused, mida on vajalik teha, et läbi viia erinevat liiki testimisi peavad olema organiseeritud, võtma arvesse ressursse, strateegiat, eesmärke ja konkreetseks testimiseks vajalikke meetmeid. [11, lk 93]

Peamine tegevus, mida tarkvara testimisega seostatakse, on testide käivitamine / testjuhtumite läbimine. Kuid selleks, et olla testimisel tulemuslik ja tõhus on lisaks vajalik testimise tegevusi planeerida, töötada välja sobivad testjuhtumite kirjeldused, teha ettevalmistused testide täitmiseks, viia testid läbi ning ka hinnata saadud tulemusi. [10, ptk 1.4]

Vastavalt rahvusvahelisele tarkvara testimise baastaseme koolitusprogrammile (*ISTQB foundation level syllabus [10]*) koosneb fundamentaalne testimise protsess järgmistest tegevustest:

- testimise planeerimine ja kontroll – testimise planeerimise eesmärgiks on määratleda testimiseks vajalik tegevuskava, muu hulgas tuua välja testimise skoop, vajalikud ressursid, keskkonnad, ajakava. Kontrolli tegevused hõlmavad tegeliku olukorra hindamist ja plaanile vastavuse kontrollimist;
- analüüs ja testide disain – selles etapis kirjeldatakse testijuhtumid vastavalt nõuetele ja testimise eesmärkidele;
- testide käivitamine / testimise läbiviimine – testimisprotseduuride määratlemine ja testjuhtumite läbiviimine, veapõhjuste analüüsimine, vigade paranduste kontroll;
- testimise lõpetamise kriteeriumite hindamine ja raporteerimine – selles etapis hinnatakse, kas nõuded said täidetud ja soovitud eesmärk / oodatav tulemus saavutati või on vajalik täiendav testimine. Toimub testimise tulemuste raporteerimine;
- testimise lõpetamine – selles etapis kontrollitakse, et kõik planeeritud tulemid oleksid testitud, testimisel saadud õppetundide analüüs ja järelduste tegemine. [10, ptk 1.4]

Kuigi ülal toodud etapid on kirjeldatud järjekorras võivad reaalselt nimetatud protsessi tegevused kattuda või toimuda samaaegselt. Testimise kontekstis on tegemist oluliste sammudega, mida ei tohiks jätta arvestamata. [10, ptk 1.4]

Hea tarkvarasüsteemi arendamine on raske ülesanne. Hea tarkvaratoote saamiseks tuleb võtta arvesse mitmeid tarkvara kvaliteedi atribuutide meetmeid. Süsteemi keerukus mängib olulist rolli tarkvarakvaliteedi kontrollimisel. [4, ptk. 1] Testimine kitsamas mõttes on tarkvara täitmine / käivitamine kontrollimaks, kas ta vastab ettenähtud nõuetele ning leidmaks vigu. Laiemas mõttes on testimine tarkvara analüüsi protsess eesmärgiga leida erinevusi olemasolevate ja nõutud tingimuste vahel ning hinnata tarkvara omadusi. Vastutusrikaste süsteemide puhul peavad testid olema süstemaatilised, põhjalikud ja andma hinnangu toote omadustele (näiteks töökindlusele või tootesse jäänud vigade arvule). [6, lk. 24]

Tarkvara süsteemi testimine on otseselt seotud toote kvaliteediga. Toode on kvaliteetne, kui ta rahuldab oma tööga vajadusi, mis motiveerisid ja ajendasid toodet looma. Nõuetele vastavuse kontrolliks on vajalik erinevate testide läbiviimine. Pragmaatiliselt oodatakse tarkvaratootelt usaldusväärsust, et tarkvara funktsioneerib soovitud viisil etteantud tingimustel. Funktsioneerimise viis ja opereerimise tingimused on vaja püstitada toote arenduse esimestel etappidel ning täpsustada kogu arendustsükli käigus. Praktikas on võimatu testida kõikvõimalike sisendparameetrite kombinatsioonidega ning võrrelda kõiki tarkvara väljundeid oodatava väljundiga. Seega on väga oluline valida efektiivne komplekt testandmeid kombineerituna testimise tüüpidega. Kuna tarkvaratoote kvaliteet oleneb suures osas läbiviidavate testide tüübist, tuleb neid hoolikalt valida ja toote tellijaga kokku leppida. [5]

Testimiseks on vajalik, et nõuded oleks püstitatud viisil, et need oleks testitavad. Üheks võimaluseks on nõudeid jaotada funktsionaalseteks ja mittefunktsionaalseteks. Funktsionaalsed nõuded vastavad küsimusele, et mida peab tarkvara tegema ning mittefunktsionaalsed nõuded vastavad küsimusele, et kuidas tarkvara peab vajalikke funktsioone täitma. [6, lk. 14]

Erinevaid omadusi, mille kohta nõudeid defineerida on palju, võimalike omaduste piiritlemiseks ja neis orienteerumiseks on üheks variandiks kasutada kvaliteedikarakteristikuid. Näiteks standardis EVS-ISO/IEC 25010:2011 süsteemide ja

tarkvara kvaliteedinõuded ja kvaliteedi hindamine ning süsteemide ja tarkvara kvaliteedimudelid on välja toodud kvaliteedikarakteristikud (vt. lisa 1), neist tulenevalt saab välja tuua mitmeid testimise liike:

Tabel 1 Kvaliteedikarakteristikud EVS-ISO/IEC 25010:2011 järgi ja testimise liigid [6 lk. 105-108], [7 ptk. 4.2], [9]

| **Kvaliteedikarakteristik** | **Selgitus** | **Testimise liik** |
|---|---|---|
| Funktsionaalne sobivus | Määr, milleni toode või süsteem pakub funktsioone, mis kasutamisel ettemääratud tingimustes rahuldavad sõnastatud ja eeldatavaid vajadusi. | Funktsionaalsuse testimine, regressiooni testimine |
| Soorituse tõhusus | Sooritus ettemääratud tingimustes kasutatava ressursikoguse suhtes. | Koormustestimine, stresstestimine |
| Ühilduvus | Määr, milleni toode, süsteem või komponent saab vahetada teavet teiste toodete, süsteemide või komponentidega ja/või täita oma nõutavaid ülesandeid, kasutades nendega ühist riistvara- või tarkvarakeskkonda. | Integratsiooni testimine, koostalitusvõime testimine |
| Kasutatavus | Määr, milleni ettemääratud kasutajad saavad toodet või süsteemi ettemääratud kasutuskontekstis toimivalt, tõhusalt ja rahuldusega ettemääratud sihtide saavutamiseks kasutada. | Kasutatavuse testimine |
| Töökindlus | Määr, milleni süsteem, toode või komponent ettemääratud tingimustel ja ettemääratud ajavahemikus täidab kindlaid ülesandeid. | Tõrketaluvuse testimine, taastuvuse testimine |
| Turvalisus | Määr, milleni toode või süsteem kaitseb teavet ja andmeid volitamata ligipääsu, muutmise ja avalikustamise eest.<br>Testimine eesmärgiga teha kindlaks, kas objekt (süsteem, komponent, toode, protsess vm) vastab turvanõuetele, ja tuvastada võimalikud nõrkused. | Turvalisuse testimine |

Tabel 1 Kvaliteedikarakteristikud EVS-ISO/IEC 25010:2011 järgi ja testimise liigid [6 lk. 105-108], [7 ptk. 4.2], [9]

| | | |
|---|---|---|
| Hooldatavus | Toimivuse ja tõhususe määr, millega ettemääratud hooldajad saavad toodet või süsteemi muuta. | Skaleeruvuse testimine, hooldatavuse testimine |
| Porditavus | Toimivuse ja tõhususe määr, millega saab süsteemi, toote või komponendi ühest riistvara-, tarkvara- või muust käitus- või kasutuskeskkonnast teise üle viia. | Installeeritavuse testimine, porditavuse testimine |

Kvaliteedikarakteristikute suhteline tähtsus sõltub projekti üldistest sihtidest ja eesmärkidest. Kvaliteedimudelit tuleks enne kasutamist kohandada - piiritleda kõige tähtsamad karakteristikud ja alamkarakteristikud ning jaotada ressursse näitajate eri tüüpide vahel sõltuvalt riskiosaliste sihtidest ja eesmärkidest toote alal. [7, ptk. 3.5]

Testimise liigitusi on erinevaid - tulenevalt testitavast objektist, testide läbiviijatest, testimise viisist või meetodist. Käesolevas töös on fookus eelkõige tarkvara arendustega seonduvate funktsionaalsete ja mittefunktsionaalsete nõuete testimisele (sh. koormus- ja turvatestimisele).

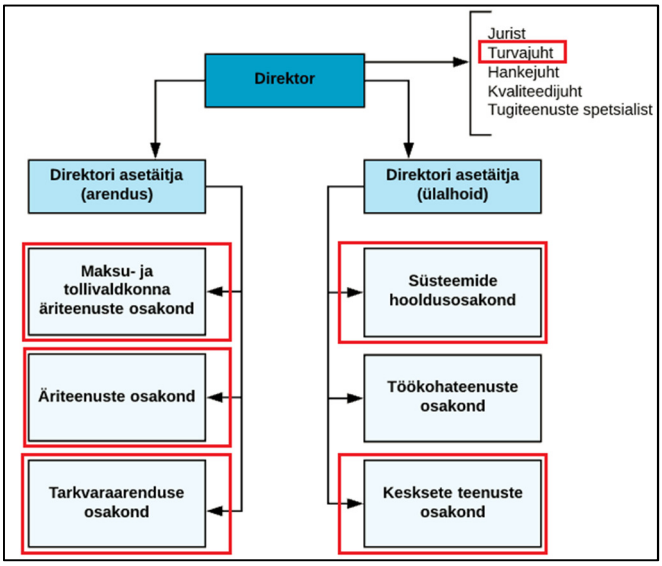
RMIT haldusalas toimuvad ka arendused, mis hõlmavad süsteeme, mis vahetavad andmeid / infot teiste riikidega. Valdavalt on sellised arendused seotud Euroopa Komisjoni poolt tulenevate nõuetega riikidevahelise infovahetuse osas. Testimised sellistes arendusprojektides erinevad mõnevõrra tavapärastest siseriiklikes arendusprojektides toimuvatest testimistest. Sageli selliste projektide puhul on suur osa testimist reglementeeritud Euroopa Komisjoni nõuetega. Komisjoni poolt on ette kirjeldatud testjuhtumid ja nende läbimine toimub üldjuhul vastavates Euroopa Komisjoni poolt hallatavates keskkondades. Testid, mis Komisjoni poolt ette antakse ei kata siseriiklikku funktsionaalsust ja nende testimine tuleb korraldada vastavalt projekti käigus kokkulepitule. Riikide vahelist suhtlust hõlmavates projektides koordineeritakse testimisi Euroopa Komisjoni poolsete kontaktidega ning vahel ka teiste riikidega. Käesolevas töös ei käsitleta Euroopa Komisjoniga seotud IT arendusprojektides toimuvaid testimisi, eelkõige seetõttu et nende projektide testimiste reeglid ja nõuded ei ole ühe riigi poolt lihtsasti muudetavad. Käesolev töö keskendub testimisega

seonduvatele probleemidele, mille osas parendused on võimalikud organisatsiooni kontekstis.

## 1.2 Organisatsioon

RMIT on Rahandusministeeriumi poolt hallatav riigiasutus. RMIT tegevusvaldkonnaks on Rahandusministeeriumi ja selle valitsemisala info- ja kommunikatsioonitehnoloogia (edaspidi IKT) arendamine ja haldamine ning IKT teenuste osutamine ministeeriumi valitsemisala asutustele ning vastavalt kokkuleppele ka teistele riigi-, siht- ja kohalikele omavalitsusasutustele. [3, lk. 3]

Järgneval joonisel on toodud RMIT struktuur, punasega on märgistatud osakonnad, kus töötavad spetsialistid tegelevad IT arendusprojektides tarkvara testimistega:




Joonis 1 RMIT struktuur ja tarkvara testimisega tegelevad osakonnad

Testimiste eest vastutuse kirjeldus rollide lõikes on toodud peatükis 2.1, mis käsitleb detailsemalt testimise korraldust organisatsioonis.

Osakondade põhiülesanded:

1) maksu- ja tollivaldkonna äriteenuste osakonna ülesanneteks on maksu- ja tollivaldkonna äriprotsesse toetavate IKT-lahenduste väljatöötamine ja haldamine;

2) äriteenuste osakonna ülesanneteks on äriprotsesse (v.a. maksu- ja tollivaldkond) toetavate IKT-lahenduste väljatöötamine ja haldamine;

3) tarkvaraarenduse osakonna ülesandeks on infosüsteemide arendustööde teostamine;

4) süsteemide hooldusosakonna ülesanneteks on taristu ja andmeside arendamine ning haldamine ja süsteemide käideldavuse, andmete tervikluse ning konfidentsiaalsuse tagamine;
5) töökohateenuste osakonna ülesanneteks on osutada infosüsteemide kasutajatele vahetut tugiteenust, korraldada ja arendada töökoha IKT teenuseid ning koordineerida keskuse varade haldamist;
6) kesksete teenuste osakonna ülesanneteks on hallata pakutavate teenuste portfelli, arendada ja hallata keskseid teenuseid ning korraldada teenuste kasutuselevõttu. [1, ptk. 3]

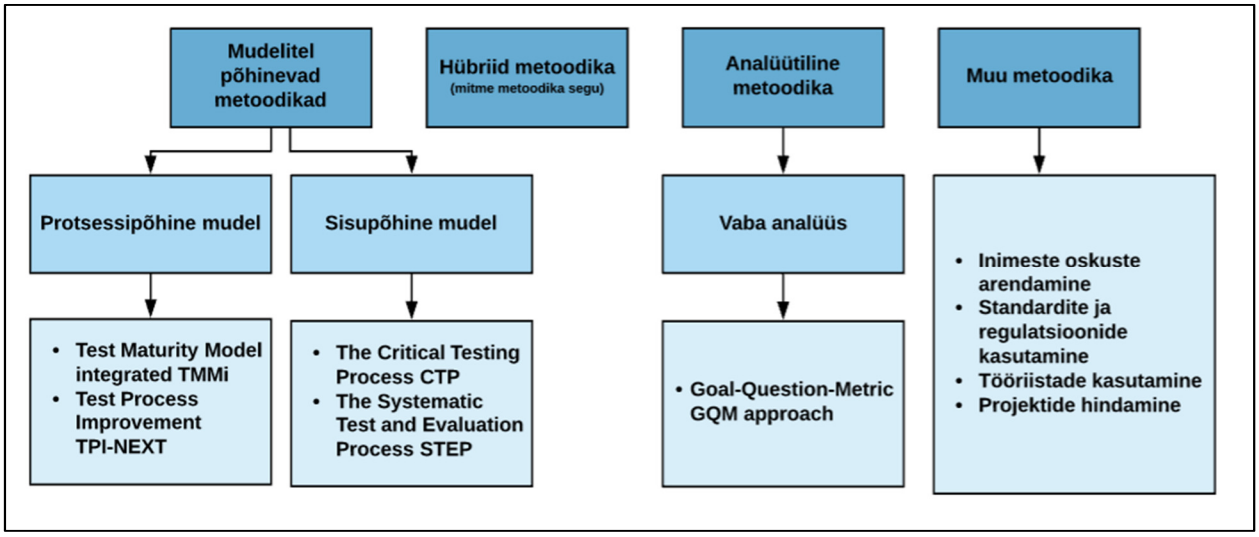
Suurematekс RMIT klientideks on:

- Rahandusministeerium (2017 aastal 13 arendusprojekti);
- Maksu- ja Tolliamet (2017 aastal 166 arendusprojekti);
- Statistikaamet (2017 aastal 19 arendusprojekti);
- Riigi Tugiteenuste Keskus (2017 aastal 3 arendusprojekti);
- Kultuuriministeerium (hetkel arendusprojektid puuduvad). [2]

Asutustele pakutavad teenused ja nende tingimused on reglementeeritud osapoolte vahel sõlmitud teenuslepetega (SLA – *Service Level Agreement*). Neis lepetes on määratletud õigused ja kohustused infotehnoloogia- ja kommunikatsiooniteenuste osutamisel ja tarbimisel ning teenuste tingimused. RMITis töötab kokku veidi üle 130 inimese.

# 2 Kasutatud metoodika ülevaade

Testimise protsessi parendamiseks eksisteerivad erinevad meetodid ja lähenemisviisid. Mitmed kasutamiseks võimalikud meetodid on kirjeldatud rahvusvahelises tarkvara testimise edasijõudnute (*ISTQB advanced level syllabus* [13]) ja ka ekspert taseme koolitusprogrammis *(ISTQB expert level syllabus [12]).* Joonis, mis illustreerib erinevaid testimise protsessi parendamise metoodikaid on toodud järgnevalt:




Joonis 2 Testimise protsessi parendamise metoodikad [15, ptk 2.5]

Näiteks on testimise protsessi hindamisel ja parendamisel võimalik võtta aluseks erinevaid mudelitel põhinevaid metoodikaid - testimisprotsessi küpsusmudel *(Test Maturity Model integrated TMMi®)*, süstematiseeritud testimise hindamise protsess (*The Systematic Test and Evaluation Process STEP*) või kriitiline testimise protsess *(the Critical Testing Process CTP*). Kasutada võib ka analüütilist meetodit, mis sobitub konkreetsete probleemkohtade kindlaks määramiseks näiteks projektipõhiselt. Kõige laialdasemalt kasutatakse testimise protsessi hindamisel ja parendamisel hübriid metoodikaid (mitme metoodika segu). [15, ptk 5.5] Mitte vähem olulisena on ekspert taseme koolitusprogrammis [12] välja toodud ka muud parendamise meetodid, näiteks on testimise protsessi võimalik parendada, ka juhul kui keskenduda ainult üksikutele olulistele aspektidele, milleks võivad olla - testimise parendamine läbi inimeste oskuste arendamise, läbi tööriistade või standardite kasutuselevõtmise või ka testimise parendamine keskendudes spetsiifilistele ressurssidele. [12, ptk 2.5.4]

Teadmised olemasolevast protsessist on oluliseks hindamise ja parendamise aluseks. Mudelitel põhinevaid metoodikaid saab kasutada pigem strateegilise visiooni ja pikema ajaliste parendus eesmärkide seadmiseks. Analüütilise meetodi kasutamine on abiks, et tuvastada probleemkohad, mis vajavad lahendamist lähiajal ja seeläbi sobivad lühema ajaliste parendus eesmärkide seadmiseks. [15, ptk 5.5]

## 2.1 Testimise korraldus organisatsioonis

Testimise vajadused tulenevad eelkõige olemasolevate infosüsteemide muudatustest, parandustest ning uute süsteemide arendustest. Suur osa arendustest hangitakse tarkvaraarendusega tegelevate firmade käest, kuid toimub ka väiksemaid sisearendusi. Arendusi reglementeerib Rahandusministeeriumi valitsemisalas IT arendustööde läbiviimise kord. See kord reguleerib IT arendustööde algatamist, planeerimist, teostamist ja lõpetamist. Lisaks reguleerivad projektides tehtavat tööd RMIT tarkvaraarendusprojektide kodukorra kirjeldus ning projekti alguses, töö käigus ja lepingutes tehtud kokkulepped.

Arendustööde hankimisel lisatakse hankedokumentatsioonile RMIT poolt koostatud mittefunktsionaalsete nõuete[1] ning IT profiili[2] kirjeldus, millega tuleb täitja poolel arendustöö realiseerimisel arvestada. Profiili eesmärk on hoida infosüsteemide haldus-, hooldus- ja koolituskulud optimaalsetena ning minimeerida tehnilist keerukust (näiteks erinevate platvormide arvu). Funktsionaalsete nõuete detailne määratlemine ja täpsustamine toimub tavaliselt projekti analüüsi faasis. Algne nõuete kirjeldus pannakse tavaliselt kokku RMIT ja äripoole koostöös, hanke sisendiks oleva lähteülesande või tehnilise kirjelduse raames. Olulisemad funktsionaalsused, mida kindlasti on vaja testida, määratletakse samuti analüüsi käigus.

Lepinguliselt on sätestatud, et täitja poolt tellijale üle antav töö peab olema täitja poolt eelnevalt testitud ja selle tõenduseks täitja esitab tellijale kõik tema poolt läbiviidud testide tulemusena valminud dokumentide koopiad, mis on hankelepingus kokku lepitud. Töö üleandmise kohta koostab täitja akti, milles näidatakse ära üle antavad tööd ning

[1] Dokument, mis määrab kvaliteedi- ja mittefunktsionaalsed nõuded RMIT tellitavatele või hallatavatele infosüsteemidele ning nende dokumentatsioonidele.

[2] Dokument, mis kirjeldab täpsustavaid nõudeid rakenduste haldamise seisukohast.

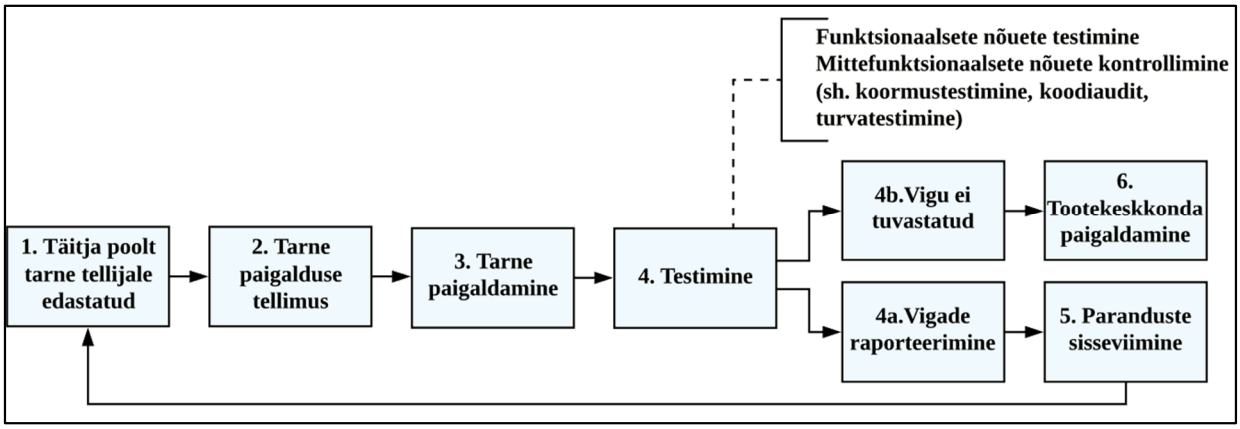
tööde vastuvõtmine tähendab tellija poolt akti allkirjastamist, millega kinnitatakse, et töö vastab kokkulepitud tingimustele ja nõetele.

**Rollide kirjeldus testimise kontekstis:**

- **IT töörühma juht** (RMIT maksu- ja tollivaldkonna äriteenuste osakonna või äriteenuste osakonna peaspetsialist) – tarnete paigalduse organiseerimine testimiseks kokkulepitud keskkonda, koormustestide ja turvatestide läbiviimise korraldamine, mittefunktsionaalsete nõuete kontrollimise organiseerimine ja läbiviimisel osalemine, vajadusel ja kokkuleppel ka muudes testimistes osalemine ning testimise organiseerimise abistamine. Ligipääsude organiseerimine ja tööde koordineerimine IT tööde osas.
- **Süsteemiadministraator** (RMIT süsteemide hooldusosakonna peaspetsialist) – tarnete esmaste ja vajadusel korduspaigalduste tegemine testimiseks kokkulepitud keskkonda (sh. veendumine, et paigaldus- ja tarnejuhised on piisavad ning korrektsed paigalduse läbiviimiseks). Osalemine koormustestimisel ja mittefunktsionaalsete nõuete kontrollimisel. Kaasatud on nii rakendusserveri administraator kui ka andmebaasi administraator.
- **Rakenduse administraator** (RMIT maksu- ja tollivaldkonna äriteenuste osakonna või äriteenuste osakonna peaspetsialist) – tarnete paigaldamine testimiseks kokkulepitud keskkonda, koormustestide käivitamiseks vajalike eeltööde tegemine, koormustestide käivitamine ja läbiviimine, mittefunktsionaalsete nõuete kontrollimisel osalemine. Funktsionaalsete nõuete testimisel osalemine (näiteks liideste testimine, testimiseks vajalike andmebaasi skriptide käivitamine, tehniliste testide läbiviimine).
- **IT arhitekt** (tarkvaraarenduse osakonna peaspetsialist) – viib läbi koodiauditi, osaleb mittefunktsionaalsete nõuete kontrollimisel ning vajadusel ka koormustestimisel (kaasatakse probleemide korral).
- **Turvaspetsialist** (peaspetsialist otsealluvuses direktorile) – viib läbi turvatestimist või korraldab turvatestide tellimise välise partneri käest, osaleb mittefunktsionaalsete nõuete kontrollimisel.
- **Funktsionaalse töörühma juht** (äri poolel töötav ametnik) **–** funktsionaalsust puudutavate testide läbiviimise korraldamine, funktsionaalsete nõuete testimises osalemine.

- **Äripoolne projektijuht** (äri poolel töötav ametnik) **–** projektis äripoolega seonduvaid töid ja tegemisi koordineeriv ja organiseeriv isik. Võib olla ühtlasi ka funktsionaalse töörühma juht.
- **Keksete teenuste osakonna peaspetsialist –** tegeleb tarkvara testimisega kesksete teenuste kontekstis (keskseks teenuseks on näiteks asutuste ülene dokumendihalduse süsteem).

Täitja poolt tellijale üle antava funktsionaalsuse kohta koostatakse tarne, mis edastatakse tellijale kontrollimiseks, testimiseks ja vastuvõtmiseks. Järgnev joonis illustreerib protsessi tarne edastamisest kuni tootekeskkonda jõudmiseni:




Joonis 3 Protsess tarne edastamisest kuni tootekeskkonda jõudmiseni

1. Täitja esitab tarne koos kokkulepitud tulemitega tellijale. Lisaks programmi koodile kaasneb tavaliselt tarnega ka seonduv dokumentatsioon, erinevad juhised ning tarne sisu kirjeldus.
2. Paigalduse tellimuse registreerib ja suunab vastavatesse järjekordadesse tegelemiseks IT töörühma juht.
3. Tarne paigaldamine toimub eelnevalt projekti käigus kokkulepitud testkeskkonda (esmase testkeskkonda paigaldusega tegelevad süsteemiadministraatorid, kes teevad ühtlasi ka vajalikud ettevalmistused, et edasisi korduvaid paigaldusi oleks võimalik teha rakenduse administraatoril). Esmase testkeskkonda paigalduse käigus testitakse ühtlasi ka seda, kas paigaldamine vastavalt edastatud juhistele on võimalik. Tõrgete korral informeeritakse arendajat ning vajadusel edastatakse arendaja poolt tarne parandus või korrastatakse juhendid.
4. Kui tarne saab edukalt paigaldatud, siis informeerib IT töörühma juht sellest äripoole projektijuhti (ja ka täitjat). Äri poolt korraldatakse edasi testjuhtumite jagamine testijatele ning funktsionaalsete nõuete testimise läbiviimine. Vajadusel

kaasatakse funktsionaalsete nõuete testimisele ka RMIT rakenduse administraator või IT projektijuht. Kui funktsionaalsete nõuete kontrollimisel on vajalik kaasata RMIT töötajaid, siis selle korraldamisega tegeleb RMIT IT töörühma juht. Testimise etapis korraldab IT töörühma juht RMIT mittefunktsionaalsete nõuete kontrollimise sh. koodiauditi läbiviimise, koormustestimise ning turvatestimise.

4a. Kõikide testimisel leitud vigade raporteerimine toimub tellija *jira* keskkonda.

4b. Olukord, kus testimise käigus üldse vigu ei tuvastatud on praktiliselt mitte esinev. Testitakse ning kontrollitakse parandusi ja seda, et raporteeritud puudused oleks kõrvaldatud.

5. Täitja poolt tehakse parandused vastavalt raporteeritud vigadele ning edastatakse parandustega tarne RMITle paigaldamiseks. Pärast paigaldust informeeritakse testijaid. Testimised ning uuesti tarnimine toimuvad kuni vead saavad kõrvaldatud.
6. Kui testide tulemusel on veendutud nõuetele vastavuses, siis planeeritakse ja teostatakse tootekeskkonda paigaldamine.

## 2.2 Testimise protsessi hindamiseks kasutatav mudel

Töös kasutatakse testimise protsessi hindamiseks kriitilise testimise protsessi mudelit (*Critical Testing Process CTP*).

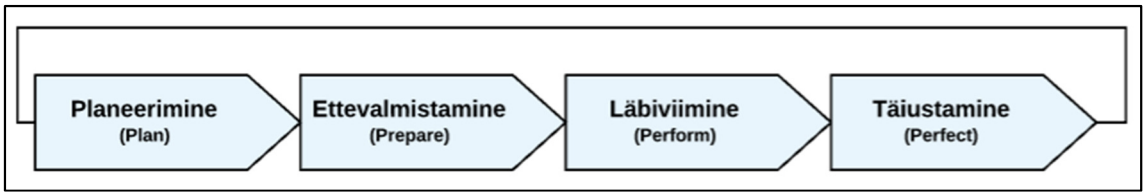
Kriitilise testimise protsess ei ole sõltuvuses tarkvaraarenduse metoodikast. Kriitilise testimise protsessi mudel ei põhine rangelt protsessi küpsuse hindamisel ega määratle kindlat samm-sammulist järjekorda parendustegevusteks. Kriitilise testimise protsessi mudel ei nõua väga struktureeritud lähenemist testimise protsessi küpsuse hindamiseks, mida kasutatakse protsessimudelites nagu *TMMi* ja *TPI NEXT*. Kriitilise testimise protsessi kasutaja saab kombineerida oma kogemused ja teadmised mudeli juhistega, et teha järeldusi testimise protsessi küpsuse kohta ja seeläbi teha ettepanekuid, et millised protsessi osad parendusi vajavad. [15, ptk 3.4] Kriitilise testimise protsessi mudel on valitud seetõttu, et see on paindlik, sellele saab tugineda kui heale praktikale ning seda kombineerida organisatsiooni vajadustega.

Vastavalt kriitilise testimise protsessi mudelile on testimise protsess kriitiline kui:

- Protsessi korratakse sageli - mõjutab meeskonna töö tõhusust;

- Protsess on tihedalt seotud erinevate rollidega – mõjutab meeskonna ühtekuuluvust ja koostööd;
- Protsess on organisatsiooni juhtorganitele jälgitav – mõjutab meeskonna usaldusväärsust;
- Protsess on seotud projekti edukusega – mõjutab meeskonna efektiivsust. [14, lk 1-2]

Kriitilise testimise protsessi mudelis on välja toodud kaksteist kriitilist protsessi, millest esimene on üldine testimise protsess. Üldise testimise protsessi neli peamist sammu on toodud järgneval joonisel:




Joonis 4 Kriitilise testimise protsessi neli peamist sammu [15, ptk. 3.3.5]

Ülejäänud üksteist protsessi jaotuvad joonisel 4 toodud sammude alamprotsessideks järgmiselt:

- **Planeerimine**
    - Testimise konteksti loomine (arvestades nii projekti kui ka organisatsiooni), selle protsessi käigus saadakse selgus testimisega seotud osapoolte ootuste kohta. Selle protsessi juurde kuulub testimise strateegia kirjeldus, siin luuakse alus edasistele testimise protsessi tegevustele;
    - Kvaliteedi riskianalüüs, testimise ja kvaliteediga seotud riskide kindlaks määramine, selle protsessi käigus määratletakse, mida ja kui palju testitakse ja ka see, mida ei testita;
    - Testimise hindamine. Hinnatakse testimise ajalist kestust ja mahtu arvestades projekti eelarvet, ajakava ja riske;
    - Testimise planeerimine. Selle protsessi käigus määratletakse detailne testimisega seotud tegevusplaan ja kava kõikidele testimisega seotud osapooltele;

- **Ettevalmistamine**

  - Testimismeeskonna komplekterimine ja arendamine. Testimine on nii põhjalik ja kvaliteetne kui on meeskond kes seda teeb. Selle protsessi käigus määratletakse, millised on kriitilised oskused, mis on vajalikud testimise meeskonnale;

  - Testimiseks vajaliku alussüsteemi määratlemine ja loomine. Testjuhtumite koostamine, dokumenteerimine, testimiseks vajalike andmemete, skriptide ja keskkondade defineerimine ja ettevalmistamine;

- **Läbiviimine**

  - Testitavate versioonide haldus. See protsess on fokuseeritud tegevustele, et iga järgmine versioon oleks parem kui eelmine ning et oleks selgus milliseid teste ja mida täpselt iga versiooniga on vajalik testida;

  - Testimise läbiviimine. Selle protsessi käigus käivitatakse ja läbitakse testid ning võrreldakse testide tulemusi oodatud tulemustega. Protsessi käigus saadakse info, et mis toimib ja vastab ootustele ning mis mitte, protsess hõlmab palju ressursse;

- **Täiustamine**

  - Vigade raporteerimine, selle protsessi käigus kirjeldatakse detailsemalt funktsionaalsused mis ei tööta, kõrvalekaldumised nõuetest. Läbi vigade parandamise toimub süsteemi nõuetele vastavusse viimine;

  - Tulemuste aruandlus, selle protsessi käigus saab juhtkond info, mis on vajalik projektis otsuste tegemisel. Protsess annab infot projekti eesmärkide täidetuse kohta ja indikatsiooni võimalike vajalike edasiste sammude kohta;

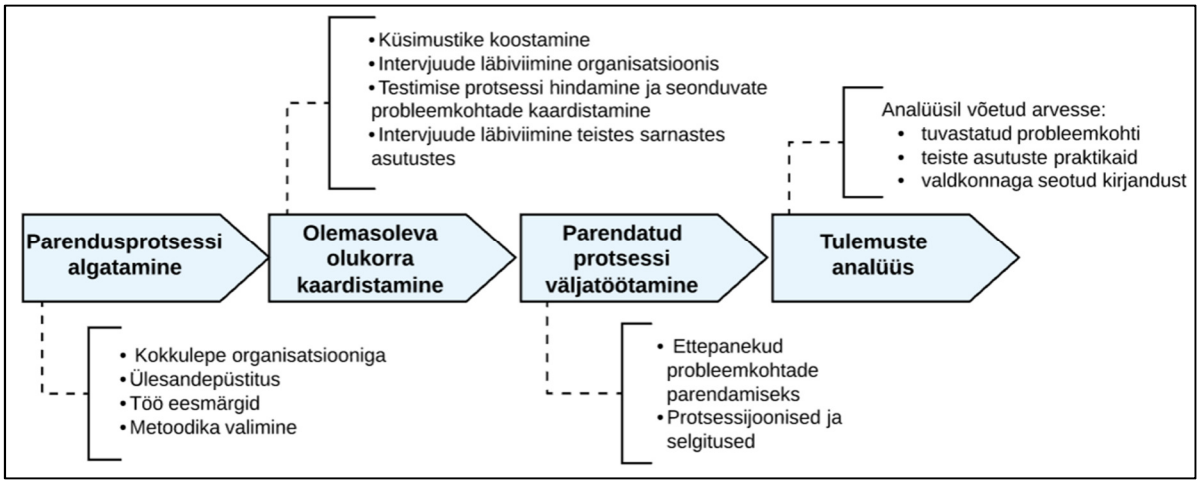
  - Muudatuste juhtimine, selle protsessi käigus tehakse valikud, et milliseid muudatusi realiseerida ning milliseid mitte võttes arvesse muudatusega seotud riske, eelarvet ja ajakava. [14, lk. 2-3]

Kasutades kriitilise testimise protsessi mudelit on esimeseks sammuks eksisteeriva testimise protsessi hindamine. Testimise protsessi hindamiseks RMITis on loodud küsimustikud ja viidud läbi intervjuud, millest kogutud informatsiooni põhjal on tuvastatud, et millised kriitilise testimise mudelis toodud protsessid on probleemsed ja

vajavad käsitlemist ning täiustamist. Vastavalt tuvastatud probleemkohtadele on töö autori poolt tehtud ettepanekud protsessi parendamiseks.

## 2.3 Protsess töö eesmärgi saavutamiseks

Planeeritud tegevusi seoses testimise protsessi uurimise, hindamise ja parendamisega iseloomustab järgnev joonis:




Joonis 5 Töö eesmärgini jõudmise protsess

**Parendusprotsessi algatamine** – selles etapis on saavutatud organisatsiooniga kokkulepe arendusprojektides toimuvate testimise protsesside uurimise ja kaardistamise kohta. Koostatud on ülesandepüstitus ning määratletud töö eesmärgid ja metoodika.

**Olemasoleva olukorra uurimine ja kaardistamine -** selles etapis on koostatud küsimustikud ja viidud läbi asutuse siseselt erinevad intervjuud hindamaks tänast olukorda RMITis testimise liikide kaupa (intervjuudesse on autori poolt kaasatud erinevaid testimisega seotud eksperte asutuse seest). Intervjuude põhjal on kaardistatud tänane olukord – autori poolt on loodud protsessijoonised ja kirjeldused ning toodud välja intervjuudest selgunud probleemkohad. Lisaks on selles etapis töö autori poolt uuritud testimise korraldust ka teistes sarnastes asutustes. Intervjuud on läbiviidud Tervise ja Heaolu Infosüsteemide Keskuses (TEHIK), Registrite ja Infosüsteemide Keskuses (RIK), Siseministeeriumi infotehnoloogia-ja arenduskeskuses (SMIT) ning Keskkonnaministeeriumi Infotehnoloogiakeskuses (KEMIT) (intervjuudes kasutatud küsimustikud on toodud lisades 2 ja 3). Testimise protsessi hindamisel töö autor tugineb kriitilise testimise protsessi mudelile (ptk. 2.2) ja toob välja millised testimise protsessi osad on probleemsed ning vajavad parendamist / täiustamist.

**Parendatud protsessi väljatöötamine –** selles etapis töö autori poolt on tehtud ettepanekud probleemkohtade parendamiseks, toodud välja protsessijoonised koos selgitustega.

**Tulemuste analüüsimine** – Tulemite analüüsil käsitletakse tuvastatud probleemkohti ja töö autori poolt tehtud parendusettepanekuid, võttes arvesse valdkonnaga seotud kirjandust ning teiste asutuste praktikaid.

# 3 Tänane olukord arendusprojektides toimuvate testimistega

Vastavalt asutuse sees läbi viidud intervjuudele on töö autor kaardistanud täna IT arendusprojektides toimuvate testimiste protsessid RMITis ja toonud välja seonduvad probleemkohad. Käesolevas peatükis annab töö autor ülevaate ka testimise korraldusest teistes sarnastes asutustes. Küsimustikud, mida intervjuudes kasutati on toodud lisades 2 ja 3.

## 3.1 Funktsionaalsete nõuete testimine

Funktsionaalsete nõuete testimise eesmärgiks on kontrollida, kas tarnitud arendus vastab vajadustele ning kokkulepitud funktsionaalsetele nõuetele, kas vajalikke tegevusi on võimalik süsteemis korrektselt läbi viia. Testimiseks sobiv keskkond lepitakse kokku projekti käigus. Testimisel on aluseks projekti raames püstitatud ja kirjeldatud funktsionaalsed nõuded. Üldjuhul on testijatele kontrollimisel aluseks dokumenteeritud funktsionaalse spetsifikatsiooni, detailanalüüsi, kasutuslugude või äriprotsessi põhjal kirjeldatud testlood. Testlood on üldjuhul alati üheks lepinguliseks tulemiks ja need edastatakse koos koodi tarnega.

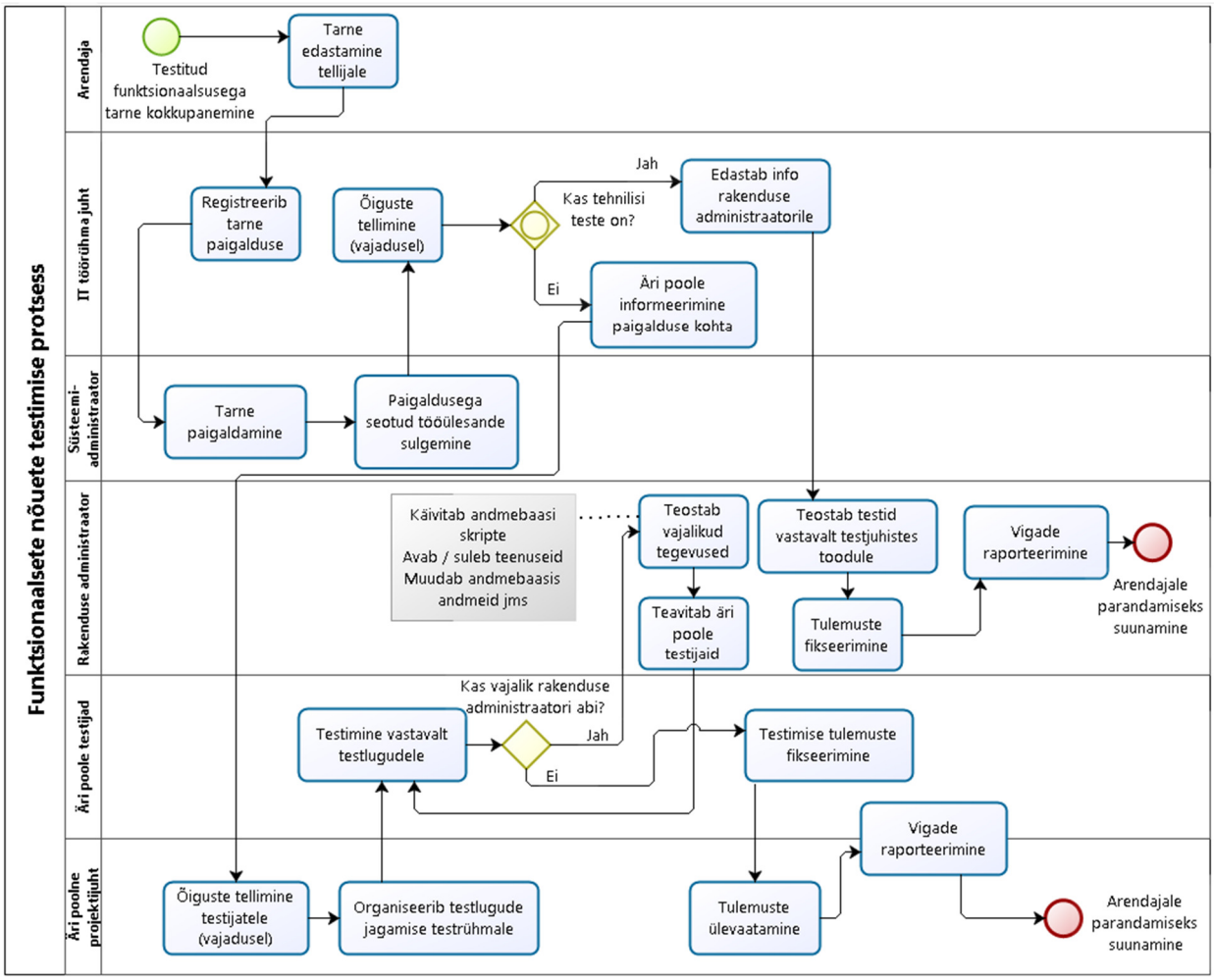
Täpne testlugude formaat lepitakse kokku projekti käigus, üldiselt sisaldavad testlood järgmist infot:

- testi kood – testile omistatud unikaale identifikaator;
- seotud süsteem(id) – testimise tegevustega seotud süsteemide nimetused;
- testandmete kirjeldus / eeldused – kirjeldus millistele tingimustele vastavaid andmeid on vaja või millised eeltegevused peavad olema tehtud;
- tegevuste kirjeldus – sammude kirjeldus, mida on vajalik teha testide läbimiseks;
- oodatav tulem – tegevuste teostamise tagajärjel oodatava tulemuse kirjeldus;
- testimise tulem – info testi tulemuse kohta, kas korras või mitte korras.

Lisaks testlugude kirjeldustele on vastavalt vajadustele ja kokkulepetele täitja poolt koostatud ka eraldi testimisjuhiseid (näiteks kui testimised nõuavad mingi täiendava

tarkvara kasutamist, siis kirjeldatakse juhistes ära sammud, mis on vajalikud tegevuste läbiviimiseks vastava tarkvara kontekstis. Näiteks *SOAP UI* kasutamine teenuste testimisel). Testlood ja testimiseks vajalikke juhiseid hoitakse tellija *confluence* keskkonnas (kui projekti käigus pole tehtud teistsuguseid kokkuleppeid). Funktsionaalsete nõuete testid üldjuhul viiakse alati läbi käsitsi.

Järgneval joonisel on toodud funktsonaalsete nõuete testimise protsessiskeem:




Joonis 6 Funktsionaalsete nõuete testimise protsess

**Protsessi kirjeldus:**

1. Arendaja poolt pannakse kokku testitud funktsionaalsusega tarne ning edastatakse tellijale.
2. Tarne paigalduse registreerib ja suunab töösse IT töörühma juht.
3. Süsteemiadministraator paigaldab tarne testimiseks kokkulepitud keskkonda.
4. Pärast õnnestunud paigaldust IT töörühma juht tellib vajadusel testimiseks õigused (tellimine toimub RMIT-i IT abi (*helpdesk*) kaudu).

5. IT töörühma juht informeerib äri poolt paigalduse lõppemisest.
6. Äri poole projektijuht tellib testijatele vajadusel õigused ja organiseerib testlugude jagamise testrühmale (eelnevalt on äri poolt määratletud testijate rühm).
7. Äri poole testijad viivad testid läbi vastavalt testlugudele ja vajadusel kaasavad RMIT rakenduse administraatoreid.
8. Kui tarne sisaldab ka tehnilisi testimisi (näiteks testid, mis nõuavad ainult andmebaasi tasemel tegevusi või eritarkvara kasutamist), siis IT töörühma juht edastab testimiseks vajaliku info (juhised) rakenduse administraatorile, kes viib vajalikud testid omalt poolt läbi.
9. Testimise tulemused fikseeritakse testlugude kirjelduste juurde ja sageli vaatab äripoolne projektijuht testijate raporteeritud vead enne arendajale edastamist üle.
10. Vigade raporteerimine toimub tellija *jira* keskkonda. Iga süsteemist leitud viga registreeritakse eraldi sissekandena ja suunatakse arendajale parandamiseks.
11. Kui arendaja viib sisse parandused, siis pannakse kokku uus tarne ja pärast selle tarne paigaldust korratakse testimist. Paranduste tarnimine ja uuesti testimine toimub kuni vead saavad parandatud.

Kui tarnele on lisatud kaasa ka akt, siis on tellijal funktsionaalsuse vastuvõtmisest keeldumiseks aega 14 kalendripäeva. Kui selle aja jooksul ei ole tellija esitanud vastuväiteid töös esinevate puuduste kirjeldusega, loetakse töö vastuvõetuks.

**Intervjuude käigus selgunud probleemid** (sulgudes on toodud millise kriitilise testimise protsessi osaga on probleem seotud):

- Testplaane projektides tehakse harva – tehakse testjuhtumite kirjeldusi, juhiseid mida testimisel kasutada, testimist planeeritakse projektiplaanis. Korrektne testplaanide koostamine (ja nende ajakohasena hoidmine) projektide käigus aitaks paremini planeerida ressursse ja annaks kõigile projekti osapooltele selguse, et mis on testimise skoop ja metoodika, milliseid teste planeeritakse teha ning millistes keskkondades ja milliseid vahendeid kasutades, kelle poolt ning mis aegadel (**planeerimise protsess** – testimise planeerimine).
- Kuigi testjuhtumite formaat lepitakse projekti käigus kokku, siis testlugude detailsus on sageli pinnapealne, kirjeldus liiga tehniline, sisaldab pigem ainult positiivsete voogude kirjeldusi (**ettevalmistamise protsess** - testimiseks vajaliku alussüsteemi määratlemine ja loomine).

- Testimise eesmärk ei ole testijatele alati arusaadav – testijate gruppides ei ole alati kaasatud isikud, kes igapäevaselt sisuliselt testitava funktsionaalsusega kokku puutuvad ning seetõttu võib testijal jääda osa vigu tähelepanuta või siis ei osata testi tulemust tõlgendada (**läbiviimise protsess** - testimise läbiviimine).
- Sageli toimub testimine ainult täitja poolt etteantud testide kirjelduste ja juhiste põhjal, seetõttu jääb suur vastutus täitja teadlikkusele ja suutlikkusele kirjeldada testjuhtumid nii, et need kataks kõik vajaliku. Siin on oma osa ka projekti ajakaval. Mida vähem on aega, seda vähem jõutakse ise testimise osas laiemalt kaasa mõelda ja keskendutakse sellele, mis on täitja poolt ette antud. Ohuks on see, et täitja kirjeldab testid selliselt, et vead ei tule välja ja testlood ei kata kõiki aspekte (**läbiviimise protsess** - testimise läbiviimine).
- Mobiilivaadete testimisel puudub selgus, mida peab selleks tegema, et saaks mobiilsete seadmetega testida – ei ole selge, milline pöördumine tuleb RMIT IT abile teha (mis info seadmete ja nende platvormide ning versioonide kohta taotlusesse lisada). Kui seadmed saadakse kätte, siis puuduvad juhised, kuidas nendega vajalikesse testkeskkondadesse ligi pääseda. Lisaks erinevate küsitluste käigus toodi mitmeid kordi välja – protsess, et seadmetega testima saaks hakata võtab väga kaua aega (rohkem kui üks nädal). (**ettevalmistamise protsess** - testimiseks vajaliku alussüsteemi määratlemine ja loomine).
- Parandustarnete puhul oleneb testija teadlikkusest, aga sageli ei testita üle varasemalt juba testitud funktsionaalsust. Puudub selgus mida kõike parandused lisaks mõjutada võivad (**läbiviimise protsess** - testimise läbiviimine).
- Testimisel kasutatavate keskkondade uuendamisega kaovad ära testimiseks ettevalmistatud andmed (**ettevalmistamise protsess** - testimiseks vajaliku alussüsteemi määratlemine ja loomine).
- Sisearenduste puhul ei looda alati detailseid testjuhtumeid ning seetõttu on testimine äri poolele keerukas ja võib jääda pinnapealseks / puudulikuks (**ettevalmistamise protsess** - testimiseks vajaliku alussüsteemi määratlemine ja loomine).
- Puudub täpne teadmine milliseid teste täitja teeb enne funktsionaalsuse tarnimist. Testraportid on sageli testlugude kirjeldused, mida tellijale testimiseks antakse (koos infoga testi tulemuse kohta), selgust aitaks tuua näiteks testplaan (**planeerimise protsess** – testimise planeerimine).

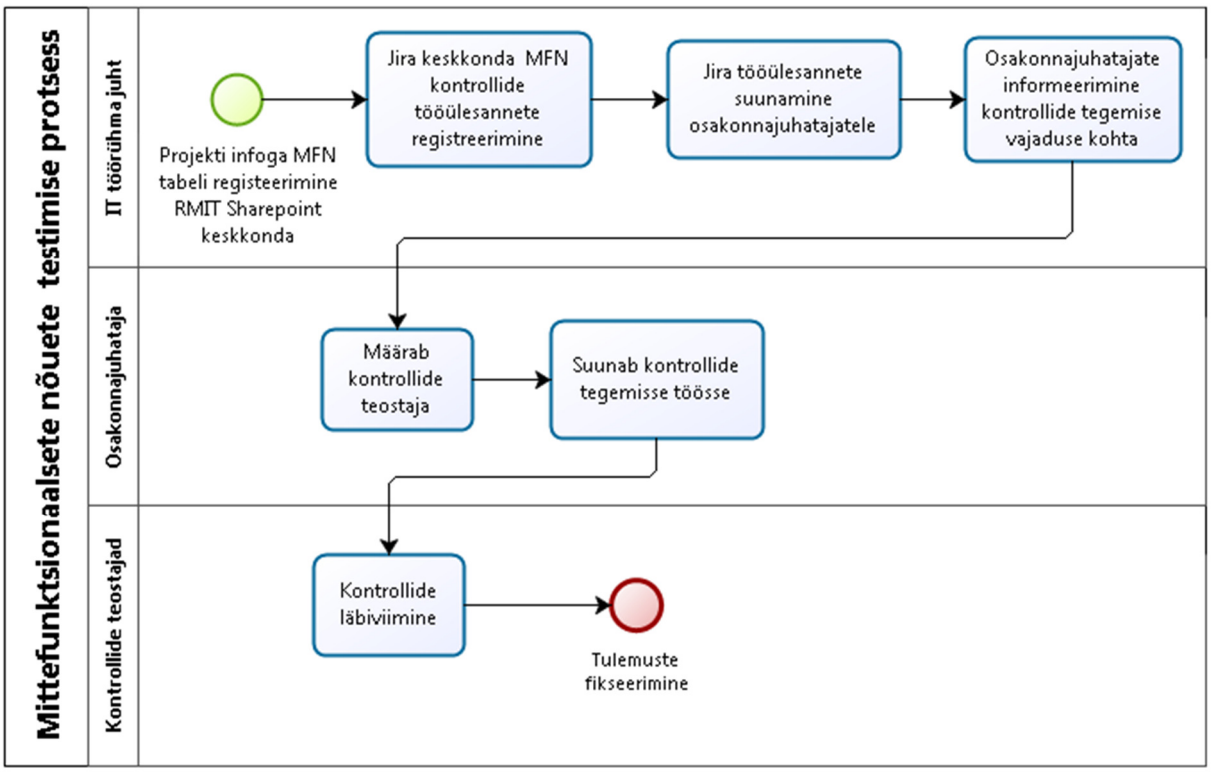
- Testide läbimiseks vajalike õiguste info ei ole sageli lihtsasti leitav ja seda tuleb otsida, täpsustada (vajalikud oleks täpsed õiguste / privileegide nimetused). Eriti oluline on see uute tarnega kaasa tulnud õiguste osas, mille kohta ei ole ka organisatsioonis eelnevat teadmist. See info võiks olla toodud näiteks testplaanis (või testjuhtumite juures). (**ettevalmistamise protsess** - testimiseks vajaliku alussüsteemi määratlemine ja loomine).

## 3.2 Mittefunktsionaalsete nõuete testimine

Mittefunktsionaalsete nõuete testimise eesmärgiks on kontrollida, kas tarnitud tarkvaralahendus vastab RMIT poolt kirjeldatud kvaliteedi- ja mittefunktsionaalsetele nõuete. Arendatava lahenduse realiseerimisel on kohustuslik arvestada RMIT mittefunktsionaalsete nõuete ning selle juurde kuuluva IT profiili kirjeldusega. Arendusprojekti alguses on vajalik veenduda, et täitjal oleks aluseks värskeim versioon mittefunktsionaalsete nõuete dokumendist. Lisaks on vajalik projekti alguses läbi rääkida, et kui mõnda nõuet loodava lahenduse kontekstis ei ole otstarbekas täita, siis tuleb mittetäitmine kooskõlastada tellijaga analüüsi faasis.

Mittefunktsionaalsete nõuete testimise juurde kuuluvad veel koodiaudit, koormustestimine ning turvatestimine. Neid testimisi käsitletakse eraldi peatükkides kuna IT arendusprojektide kontekstis on need eraldi alamprotsessid.

Mittefunktsionaalsete nõuete testimisel osalevad erinevad RMIT spetsialistid (IT projektijuht, rakenduse ja süsteemiadministraatorid, turvaspetsialist, IT arhitekt (või ka RMIT arendaja)). Testimise aluseks on mittefunktsionaalseid nõudeid kirjeldav tabel, kus on toodud nõude number, sisu, nõude kontrollimise eest vastutav osakonnajuhataja, kontrolli teostaja, staatus ning *jira* viide (tuvastatud puudusele). Mittefunktsionaalsete nõuete testimise korraldamine on IT töörühmajuhi ülesanne. Nõuete kontrollimise eelduseks on testkeskkonda paigaldatud põhifunktsionaalsuse olemasolu. Järgneval joonisel on toodud mittefunktsonaalsete nõuete testimise protsessiskeem:




Joonis 7 Mittefunktsionaalsete nõuete testimise protsess

**Protsessi kirjeldus:**

1. IT töörühma juht loob projektipõhise kontrollitabeli RMIT *Sharepoint* keskkonda. Tabelisse on vajalik lisada projekti nimi, lingid seotud dokumentatsioonile (projekti *confluence* ja *jira* viited), kontaktid, kontrollide tähtaeg.
2. Seejärel loob IT töörühma juht *jira* keskkonda tööülesanded koos kontrollide läbiviimiseks vajaliku infoga ning suunab need osakonnajuhtidele. Lisaks tööülesannete suunamisele informeerib IT töörühma juht e-kirja teel kõiki osakonnajuhte kontrollide tegemise vajadusest ning tähtajast.
3. Osakonnajuhid määravad kontrollidele teostaja või teostajad ja suunavad kontrollid töösse.
4. Kontrollide läbiviijad teostavad neile määratud kontrollid ning fikseerivad tulemused projektipõhisesse kontrollitabelisse.
5. Puudused raporteeritakse arendajale *jira* tööülesannetena.

Kontrollidega seotud staatused:

- kontrolli ootel - kontroll ootab teostamist määratud kontrolli teostaja poolt;
- nõue täidetud - kontrolli tulemusel on veendutud, et nõue on täidetud;
- nõue täitmata - kontrolli tulemusel on veendutud, et nõue on täitmata;

- täitmist ei kontrollita - nõude täitmist antud projekti puhul teadlikult ei kontrollita;
- korduvkontrolli ootel - esmase tarne kontroll on läbitud, aga kontroll tuleb läbida uuesti järgmise tarne käigus.

**Intervjuude käigus selgunud probleemid** (sulgudes on toodud millise protsessiga kriitilise testimise protsessi mudelist on probleem seotud):

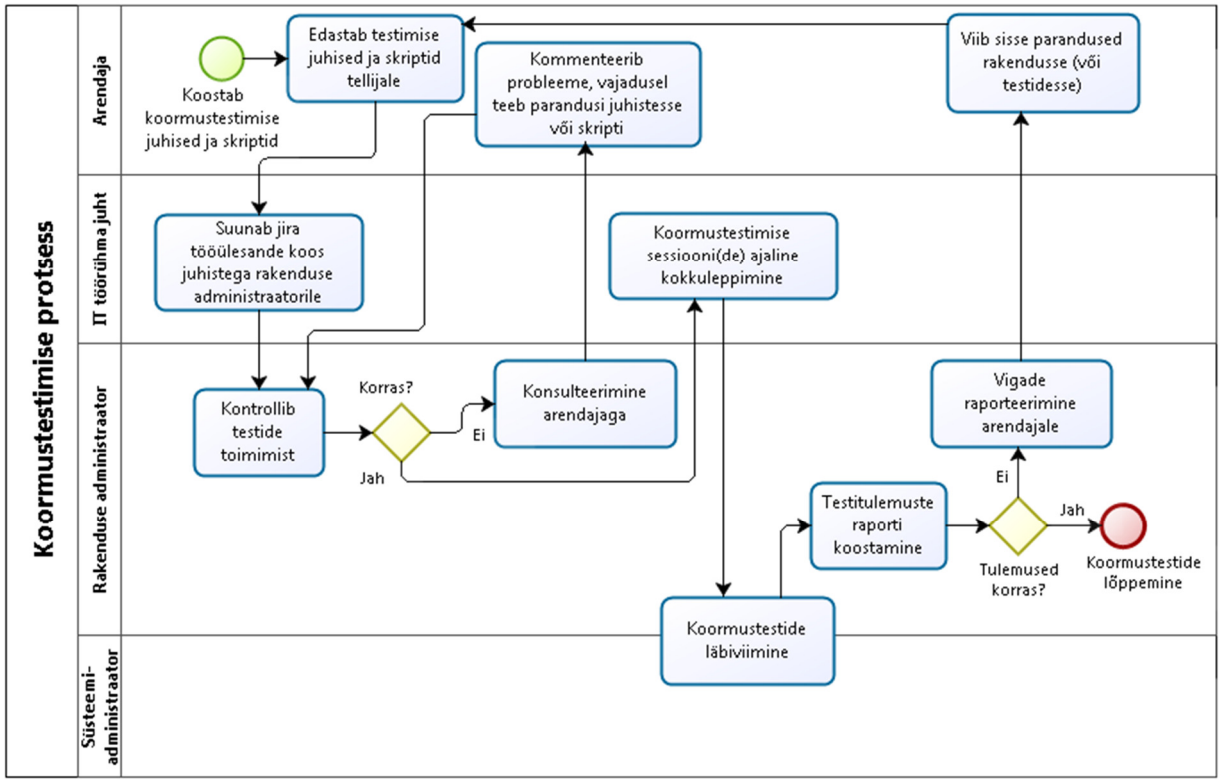
- Mittefunktsionaalsete nõuete kontrollimise organiseerimine on IT töörühmajuhi ülesanne, kuid protsessi kohta puudub kirjeldus (seetõttu uutel IT projektijuhtidel puudub selgus, millised tegevused nende poolt kontrolli korraldamiseks on vajalikud). Kuna protsess ei ole kõigi jaoks üheselt fikseeritud, siis pole kontrolli tegijatel selgust, kas leiud tuleb lihtsalt fikseerida tabelisse, edastada IT projektijuhile, registreerida ise *jira*sse ja suunata arendajale või tuleb kõik tulemused veel täiendavalt sisemiselt üle vaadata enne arendajale suunamist (**planeerimise protsess** – testimise konteksti loomine);
- Kõikide kontrollimiseks kirjeldatud nõuete testimise meetod ei ole kontrollijatele arusaadav (**läbiviimise protsess** - testimise läbiviimine);
- Alati ei jõuta kõiki kontrolle läbi viia enne projekti tootesse paigaldamist (**läbiviimise protsess** - testimise läbiviimine);
- Viimases mittefunktsionaalsete nõuete kontrolltabelis ei ole kontrollide eest vastutajad kooskõlas organisatsioonis toimunud muudatustega (**planeerimise protsess** – testimise planeerimine);
- Kasutatavuse testidele vähene tähelepanu pööramine projektides. Eelkõige on siin mõeldud seda, et kui luuakse teenuseid välistele osapooletele (näiteks ettevõtted), siis reaalsete kasutajate peal kasutatavuse testide tegemine on vähene (**läbiviimise protsess** - testimise läbiviimine).

## 3.3 Koormustestimine

Koormustestimine, on üks mittefunktsionaalsete nõuete testimise osa. Koormustestimise eesmärk on selgitada välja - kas süsteem toimib defineeritud tavakoormusel, kuidas süsteem toimib pideva tavapärasest kõrgema koormuse all ning kuidas ekstreemse koormuse all. Leida parim konfiguratsioon nõuetele vastava jõudluse saamiseks ning kaitsta süsteemi, et suurenenud koormuse tõttu ei kukuks kokku kriitilised süsteemi osad (näiteks andmebaas).

RMIT IT profiilis on määratletud koormustestide läbiviimise töövahendid. Enne koormustestide alustamist on vajalik, et testitavad süsteemi osad / funktsionaalsus oleks realiseeritud ning tarne testimiseks kokkulepitud keskkonda paigaldatud.

Järgneval joonisel on toodud koormustestimise protsessiskeem:




Joonis 8 Koormustestimise protsess

**Protsessi kirjeldus:**

1. Arenduspartneri poolt koostatakse koormustestimise juhised ja skriptid, kasutades kokkulepitud töövahendit ja arvestades projekti käigus ette antud koormuse nõudeid.
2. Koormustestide juhised ja skriptid tarnitakse tellijale ning IT töörühma juht vormistab koormustestide läbiviimiseks *jira* tööülesande rakenduse administraatorile.
3. Rakenduse administraatori poolt on vajalik testide esmane kontroll ja tööle saamine. Kui testide toimimisega on probleemid, kaasatakse arendajat probleemide lahendamisel.
4. Kui testide kontroll on edukas (testid lähevad tööle), siis lepib IT töörühma juht kokku aja(d), millal koormustestid läbi viia. Tuleb leida sobiv aeg kõigile seotud osapooltele – rakenduse administraator, süsteemiadministraatorid (andmebaasi

administraator ja rakendusserveri administraator), arendaja ning ka äri poole projektijuht.

5. Koormustestid viiakse läbi RMIT poolt hallatavas keskkonnas, mis on koormustestideks projekti käigus kokku lepitud. Tavaliselt enne koormustestimise algust lepitakse kokku ka suhtluskanal, kus kõik osapooled saavad jooksvalt seisuga kursis olla ja operatiivselt koormustestimisega seonduvalt infot vahetada.

Koormustestide läbiviimiseks:

- rakenduse administraator käivitab testid vastavalt juhistele, jälgib testitava rakendusega seotud logisid, kasutab rakendust, informeerib probleemide korral süsteemiadministraatorit või arendajat;
- andmebaasi administraator jälgib andmebaasi koormust ja seal toimuvaid päringuid. Testimise lõpus andmebaasi poolse raporti koostamine;
- RMIT rakendusserveri administraator jälgib java serveri koormust, vajadusel muudab java serveri konfiguratsiooni;
- koormustestimisel vajadusel kaasatakse probleemide lahendamisel abistamiseks ka IT arhitekti abi.

6. Pärast testide lõppemist fikseeritakse rakenduse administraatori poolt tulemused. Testimisel tekkinud vead / probleemid raporteeritakse arendajale *jira* tööülesandena.
7. Pärast seda kui arendaja poolt on parandused sisse viidud ja parandustega tarne paigaldatud, siis korratakse koormustestimise sessiooni.

**Intervjuude käigus selgunud probleemid** (sulgudes on toodud millise kriitilise testimise protsessi osaga on probleem seotud):

- Olemasolev koormustestimise tööprotsessi kirjeldus ei kajasta RMIT rakenduse administraatori jaoks olulisi sisemisi testide seadistamiseks vajalikke juhiseid. Puuduvad juhised, et millise serveri peal koormustestid tuleb läbi viia ja millised õigused selleks on vajalikud, mida teha, et skripte saaks käivitada ja kuidas tulemused kätte saada (**ettevalmistamise protsess** - testimiseks vajaliku alussüsteemi määratlemine ja loomine).
- Koormustestid on sageli ebapiisavalt läbimõeldud, mitmeid kordi toodi välja andmete probleemi seoses koormustestidega. Alati ei ole koormustestimiseks eelnevalt olemas piisavalt andmeid (või ka vajalikku andmemahtu) ja

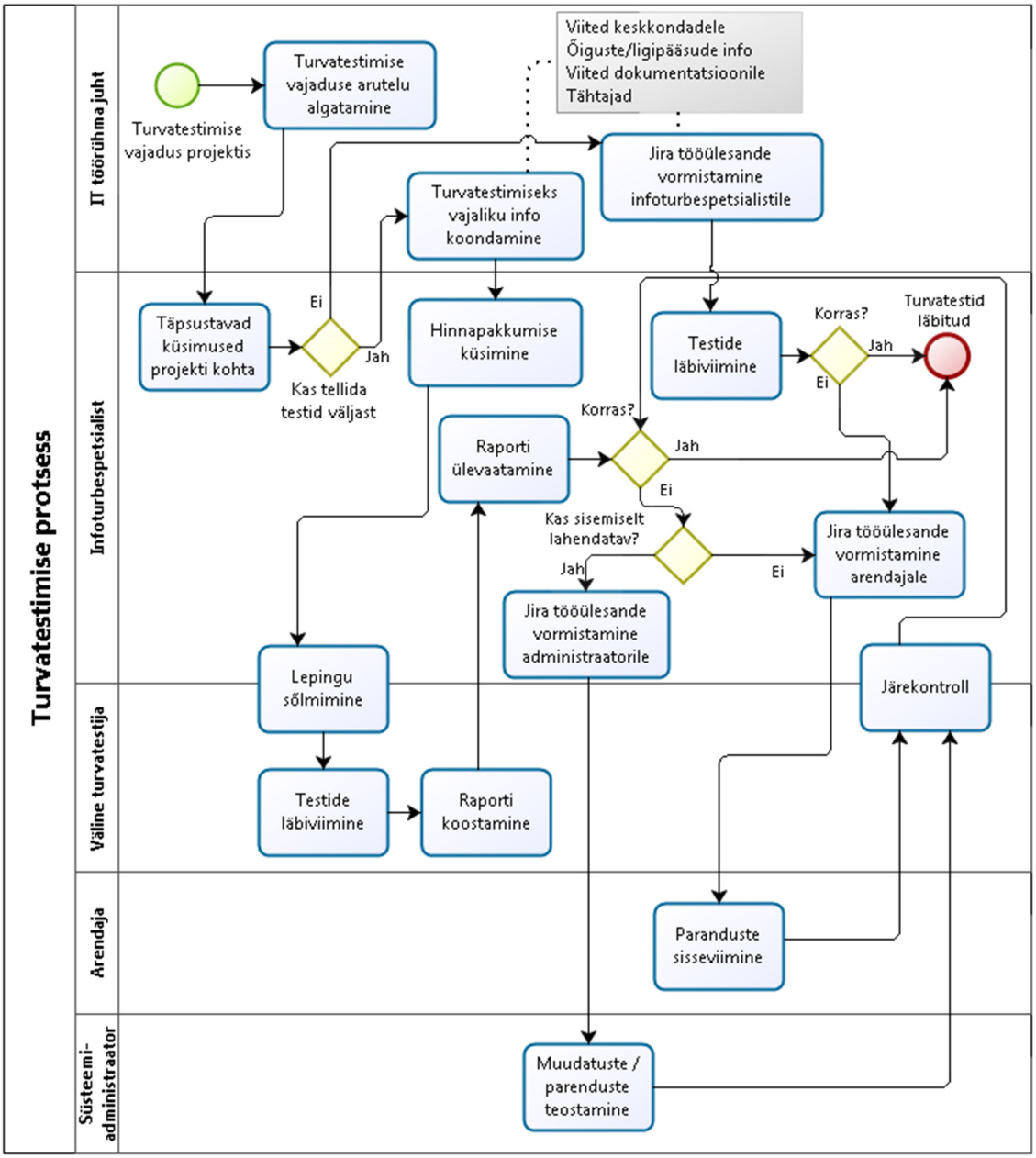
koormustestide juhised harva sisaldavad ka andmete tekitamise juhiseid (**ettevalmistamise protsess** - testimiseks vajaliku alussüsteemi määratlemine ja loomine).

- Pole määratletud, milliseid andmeid koormustestid mõjutavad, koormustestimisel tekitatud andmed tuleks eemaldada, kui need andmed segavad äri poolel testimist. Tegevusi andmete eemaldamiseks sageli ei tehta (puuduvad juhised / info) (**ettevalmistamise protsess** - testimiseks vajaliku alussüsteemi määratlemine ja loomine).
- Arendajate poolt koostatud juhised koormustestide läbiviimiseks RMIT keskkonnas ei ole arusaadavad ja selged. Juhistes jääb ruumi testimise eesmärgi erinevalt tõlgendamiseks ja seetõttu on ka tulemustest järelduste tegemine keeruline (jäädakse hätta tulemuste tõlgendamisega ning selle asemel raportid ja logid saadetakse arendajale kontrollimiseks) (**läbiviimise protsess** - testimise läbiviimine).
- Koormustestide läbiviimisel kasutatakse ühte serverit, erinevate projektide koormustestimiste sessioone ei registreerita ja seega puudub testide planeerimisel teadmine, kas teise projektiga sessioonid kattuvad või mitte (**planeerimise protsess** – testimise planeerimine).
- Koormustestide eelnev kooskõlastamine tellijaga on vähene. (**planeerimise protsess** – testimise konteksti loomine).

## 3.4 Turvatestimine

Turvatestimise eesmärk on teha kindlaks, kas objekt (süsteem, komponent, toode, protsess vm) vastab turvanõuetele, ja tuvastada võimalikud nõrkused. [9] Turvatestimine on üks osa mittefunktsionaalsete nõuete testimisest. Suur osa arendusprojektides läbi viidavatest turvatestimisest tellitakse välise lepingupartneri käest. Enne testimise alustamist on vajalik, et tarne(d) oleks paigaldatud testimiseks kokkulepitud keskkonda ja funktsionaalsus oleks testide läbiviimiseks piisavalt küps.

Järgneval joonisel on toodud turvatestimise protsessiskeem:




Joonis 9 Turvatestimise protsess

**Protsessi kirjeldus:**

1. Protsess saab alguse vajadusest viia arendusprojektis läbi turvatestid, IT töörühma juht algatab turvatestimise tellimise vajaduse arutelu.
2. RMIT infoturbespetsialist koordineerib turvatestide tellimist välise lepingupartneri käest – vastavalt sellele, kas tegemist on väliselt kättesaadava süsteemiga ning süsteemi keerukust ja kriitilisust arvestades tehakse otsus, kas testid viiakse läbi sisemiselt või tellitakse väljast.

3. Juhul, kui testid tehakse sisemiselt, siis IT töörühma juht vormistab infoturbespetsialistile *jira* tööülesande koos vajaliku infoga (viited testitavale rakendusele, dokumentatsioonile, õiguste info ja tähtajad).
4. Vajadusel raporteerib testi tegija arendajale vead, kui vigu ei tuvastata, siis testimine lõppeb. Sisemiselt testimise korral, pärast vigade parandamist teostatakse tavaliselt järelkontroll samuti sisemiselt.
5. Juhul kui turvatestimine tellitakse väljast, siis koondab IT töörühma juht vajaliku info hinnapakkumise jaoks (hinnapakkumise ja lepingu sõlmimisega tegeleb infoturbespetsialist). Hinnapakkumise küsimiseks võib eelnevalt toimuda ka välise partneriga koosoleku vormis infovahetus, et viia lepingupartner paremini kurssi tellitava süsteemi olemusega.
6. IT töörühmajuht annab vajaliku info (viited testitavale süsteemile, õiguste ja ligipääsude info, viited dokumentatsioonile ning projekti tähtajad), õigused ja ligipääsud tellib lepingupartnerile RMIT infoturbespetsialist.
7. Pärast lepingu sõlmimist toimub testide läbiviimine ja raporti koostamine. Testimise käigus vahetatakse infot jooksvalt vastavalt vajadustele ja lahendamist vajavatele küsimustele.
8. Välise testija poolt edastatud raport hinnatakse enne sisemiselt üle ja veendutakse, kas raportis on toodud välja leide, mille korrastamine on RMIT sisemiselt vaja ära lahendada (näiteks süsteemiadministraatorite poolt konfiguratsiooni muutmine).
9. Arendaja vead, mis on tuvastatud kui kõrge ja kriitilise määratlusega suunatakse *jira* tööülesandena arendajale lahendamiseks. Kriitilise määratlusega leiud on üldjuhul takistavaks ka tootekeskkonda paigaldamisel.
10. Pärast arendaja poolt (või sisemiselt) tehtud parandusi ja parandustarne paigaldamist tehakse järelkontroll (RMIT infoturbespetsialist otsustab, kas teeb kordustestid ise või kaasab välise lepingupartneri).
11. Turvatestimine lõppeb kui parandused on tehtud ja kordustestimine edukalt läbitud.

**Intervjuude käigus selgunud probleemid** (sulgudes on toodud millise kriitilise testimise protsessi osaga on probleem seotud):

- Puudulik infovahetus – nii sisemistel kui välistel turvatestijatel ei ole aktuaalselt infot jooksvate parandustarnete paigalduste kohta, projektis toimuvate koormustestimiste sessioonide kohta ega katkestuste kohta, mis mõjutavad

testitavat süsteemi. IT töörühma juhil ei ole alati selgust, et mis kuupäevaks lõplik raport valmib (**planeerimise protsess** – testimise planeerimine).

Intervjuu tulemusena toodi ettepanekuna välja, et turvatestimise tellimine võiks olla täielikult korraldatud IT töörühma juhi poolt. Järelevalve, tulemuste hindamisel osalemine ning tellimise üle otsustamine tuleks RMIT infoturbespetsialistiga kooskõlastada, aga info liikumise hõlbustamiseks toodi välja ettepanek vaadata üle turvatestide väljast tellimise korraldus ja vastutused.

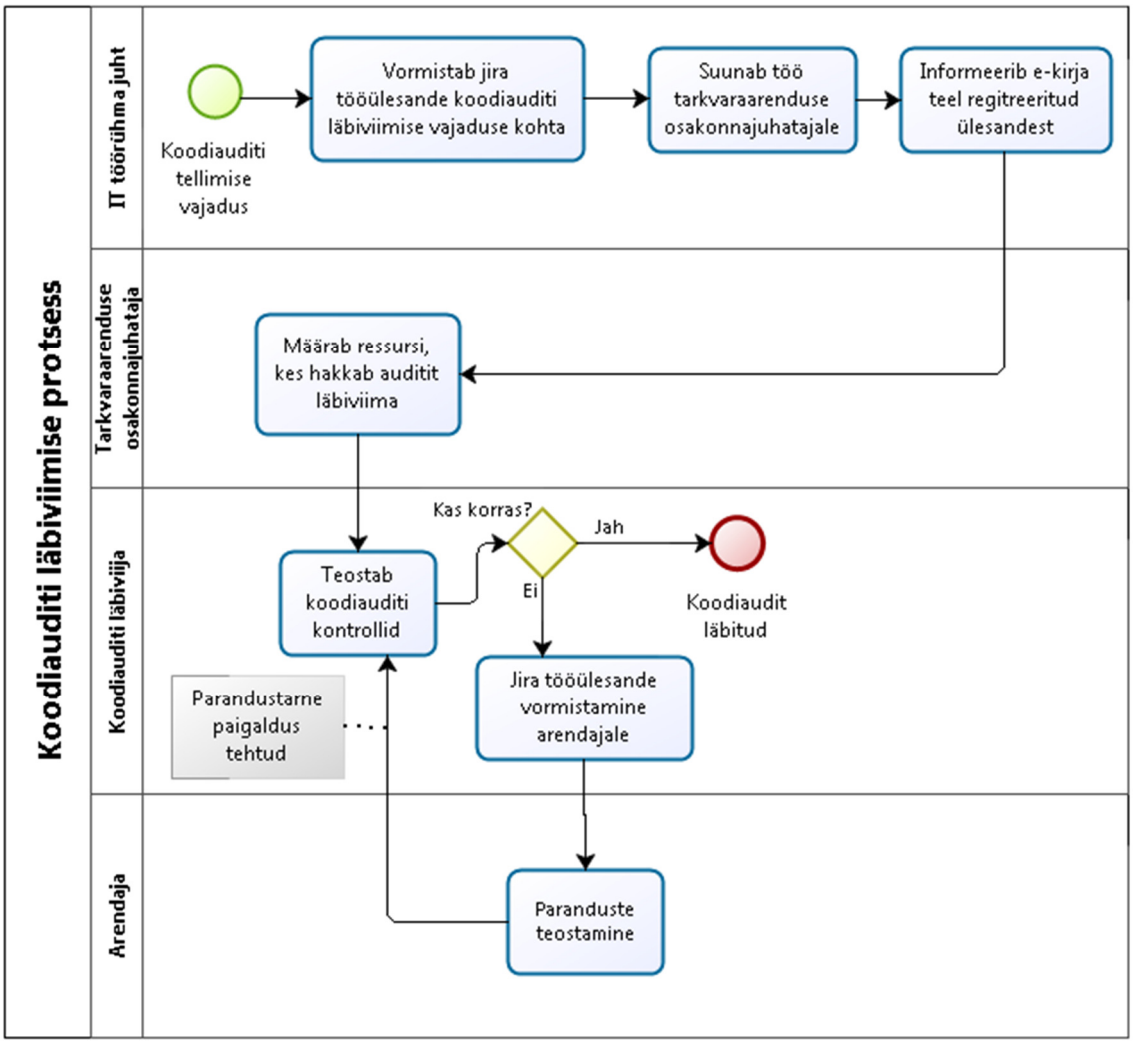
## 3.5 Koodiaudit

Koodiaudit on üks osa mittefunktsionaalsete nõuete kontrollimisest. Selle tegevuse eesmärk on teha kindlaks, kas tarnitud kood vastab RMIT poolt kehtestatud lähtekoodi hindamiskriteeriumitele. Nimetatud kriteeriumite kirjeldus lisatakse tellimisel hankedokumentide juurde ja arendaja peab nendega süsteemi realiseerimisel arvestama.

Koodiauditi käigus kontrollitakse koodi arhitektuuri, disaini ja koodistiili ning ka ehituse ja paigaldusega seotud aspekte. Näiteks veendutakse, kas süsteem on ülesehitatud loogiliste kihtidena, millest igaüks täidab konkreetset eesmärki. Kas koodi projektistruktuur järgib ühtseid lähenemisviise ning toetab modulaarsuse põhimõtteid. Kontrollitakse, et oleks kasutatud „isedokumenteerivat“ kodeerimise stiili ja kommentaaride kasutamist koodis. Tähenduslikku ja modulaarset koodi on tulevikus lihtsam hallata ning ka edasi arendada ja see tähendab tellijale väiksemaid kulutusi. Vaadatakse üle, et kood ei sisaldaks välja kommenteeritud koodi. Ehituse ja paigaldusega seonduvast aspektist hinnatakse ehitusskriptide ülesehitust ning erinevate sihtkeskkondade konfiguratsioonijuhtimise võimalusi. Ühiktestide olemasolu koodi juures on plussiks ja näitab arendajate tööstiili ning vastutust koodikvaliteedi osas.

RMIT poolt koodiülevaatusel rakendatakse staatilise koodianalüsaatori abi, et leida võimalikke koodi ebakorrektsusi ja probleemkohti. Koodiauditi läbiviimisel kasutatakse sageli *FindBugs* nimelist koodianalüsaatorit. Koodianalüsaatori raport on üks osa, mille pealt koostatakse tagasiside arendajale (kriitilised vead raporteeritakse arendajale parandamiseks). Koodiauditi tellimisel tuleks arvestada, et äri pool oleks saanud esmased funktsionaalsuse testid ära teha.

Järgneval joonisel on toodud koodiauditi protsessiskeem:




Joonis 10 Koodiauditi läbiviimise protsess

**Protsessi kirjeldus:**

1. Kui tarnitud funktsionaalsus on piisavalt küps, siis IT töörühma juht tellib koodiauditi läbiviimise registreerides selleks *jira* tööülesande tarkvaraarenduse osakonnajuhile (tööülesandes peab olema välja toodud koodi versioon, mida vaja auditeerida ning koodi repositooriumi(te) info, soovitatavalt ka viited süsteemiga seonduvale dokumentatsioonile).
2. Tarkvaraarenduse osakonnajuht määrab koodiauditi tegija (selleks on tavaliselt RMIT IT arhitekt, aga võib olla ka RMIT poolne arendaja).
3. Koodiauditi läbiviija teostab vajalikud kontrollid ja probleemide korral vormistab *jira* tööülesande arendajale, kes viib sisse parandused vastavalt vea kirjeldusele.
4. Kui parandused saavad paigaldatud, siis viiakse läbi kordus audit (kordus auditi käigus kontrollitakse üle vaid parandustega seonduv).

5. Koodiaudit on läbitud, kui vajalikud kontrollid ja kordustestimised on edukalt läbiviidud.

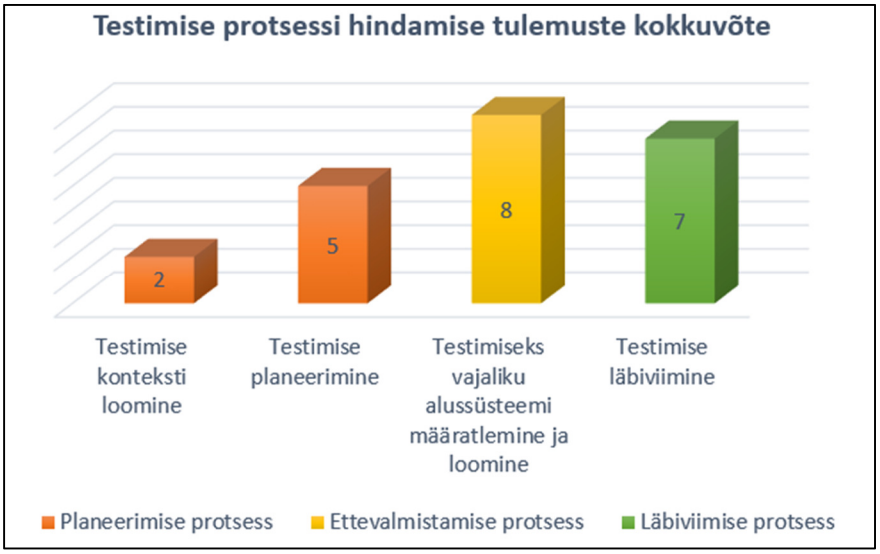
Koodiauditiga ja selle korraldusega seonduvalt ei tuvastatud probleeme, mida käesolevas töös käsitleda, seega ei keskenduta edasises töös selle teema kirjeldamisele.

## 3.6 Testimise protsessi hindamise tulemuste kokkuvõte

Käesolevas töös fikseeritud probleemide kirjeldus ei ole lõplik vaid on läbiviidud intervjuude tulemusena selgunud probleemide määratlus. Hindamise tulemusena saab järeldada, et vastavalt kriitilise testimise protsessidele (selgitatud peatükis 2.2) vajavad täiustamist järgmised protsessi osad:

- **Planeerimise protsess**
  - Testimise konteksti loomine;
  - Testimise planeerimine;
- **Ettevalmistamise protsess**
  - Testimiseks vajaliku alussüsteemi määratlemine ja loomine;
- **Läbiviimise protsess**
  - Testimise läbiviimine.

Tulemusi illustreerib järgmine pilt:




Joonis 11 Testimise protsessi hindamise tulemuste kokkuvõte

Erinevate vestluste käigus saadud info pealt on töö autori poolt tuvastatud kakskümmend kaks probleemi, millest testimise planeerimise ja testimise läbiviimise protsessiga olid

mõlemal juhul seotud seitse probleemi ning kaheksa tuvastatud probleemi olid seotud testimise ettevalmistamise protsessiga.

**Planeerimise protsess** - testimise konteksti loomine osas olid probleemkohad tuvastatud mittefunktsionaalsete nõuete kontrollimise ja koormustestimise protsessi juures. Lisaks töö autor toob välja organisatsiooni testimise strateegia puudumise. Kvaliteedi riskianalüüs ei ole rakendatud IT arendusprojektides. Testimise konteksti loomise osas olid probleemkohad tuvastatud koormustestimise, turvatestimise, funktsionaalsete ja mittefunktsionaalsete nõuete testimise protsessides.

**Ettevalmistamise protsess** - testimiseks vajaliku alussüsteemi määratlemine ja loomine osas olid probleemkohad tuvastatud funktsionaalsete nõuete testimise ja koormustestimise kontekstis.

**Läbiviimise protsess** - testimise läbiviimise osas olid probleemkohad tuvastatud funktsionaalsete nõuete testimise ja mittefunktsionaalsete nõuete kontrollimise kontekstis.

## 3.7 Testimise korraldus sarnastes asutustes

Intervjuud teistes sarnastes asutustes on töö autori poolt läbiviidud eesmärgiga saada infot, kuidas on testimine korraldatud sarnastes organisatsioonides ja käsitleda seda kui teise asutuse parimat praktikat, mille põhjal võimalusel teha RMIT testimise protsessis parendamise ettepanekuid. Intervjuudesse olid kaasatud testjuhi, IT projektijuhi ja tooteomaniku rollis olevad teiste asutuste spetsialistid. Läbi viidud vestluste tulemusena koostatud kokkuvõtted on töö autori poolt kooskõlastatud intervjueeritud isikutega. Intervjuude läbiviimisel kasutatud küsimustik on toodud lisas 3.

**Tervise ja Heaolu Infosüsteemide Keskus (TEHIK):**
Peamised testimised, mida TEHIKus IT arendusprojektides tehakse on funktsionaalsete ja mittefunktsionaalsete nõuete testid, koormustestid, turvatestid. Lisaks toodi välja, et lähitulevikus planeeritakse rohkem rõhku panna regressioonitestimisele[1],

[1] Testimine, mis on suunatud vigade avastamisele varem testitud rakenduses või programmikoodis [24]

suitsutestimisele[1], liideste ja integratsioonitestidele[2]. Projektijuhid tellivad testimisi majasiseste testijate käest. TEHIK poolsed testijad ja lõppkasutajad teostavad testimisi koostöös, testijad on valdkondade kaupa spetsialiseerunud. Testimisplaan tuleb TEHIK testimise meeskonna käest, samuti paljud testimise aluseks olevad testlood koostatakse majasiseste testijate poolt. TEHIK testijad viivad sisse täiendusi ka arenduspartneri poolt loodud testlugudesse (testimisel ei lähtuta ainult täitja poolt etteantud juhistest). Kõikide projektide testimiste (sh ka turvatestimiste) korraldamise eest on vastutav testijuht. Turvatestimine hangitakse väljast, teised testimised tehakse majasiseselt (ka koormustestimise skriptid pannakse kokku TEHIK testijate poolt, koormustestimisel kasutatakse töövahendina *JMeter* programmi ning koormustestide läbiviimisel kaasavad testijad administraatoreid). Sisearendusi ei tehta. Testide läbiviimiseks vajalike õiguste info on kirjeldatud asutuse ligipääsude reeglistikus ning õiguste tellimine käib läbi kasutajatoe (projektijuhid omavad testimiseks vajalike õigustega seonduvat infot).

TEHIK testijate jaoks üheks olulisemaks arenduspartneri poolt koostatud dokumendiks on analüüsidokument, mis on aluseks testjuhtumite kirjeldamisel. Suuremate probleemidena toodi välja - täitja poolt koostatud dokumentatsiooni pinnapealsus ja projektide tihe ajakava. Hästi toimivaks hinnati testimise ettevalmistamist koostöös projektijuhi ja arendajatega. Soovitusena toodi välja, et panna rohkem rõhku sisemiselt testimise ettevalmistusse.

**Registrite ja Infosüsteemide Keskus (RIK):**

RIKis toodi IT arendusprojektides toimuvate peamiste testimise liikidena välja samuti funktsionaalsete ja mittefunktsionaalsete nõuete testid, koormustestid, turvatestid. Arendusi tellitakse välistelt arenduspartnerilt kuid tehakse ka sisemiselt. Testimistel osalevad RIK majasisesed testijad, haldurid, sisu poole inimesed, administraatorid (kaasatud võivad olla ka RIK analüütikud). Majasiseste arenduste puhul testivad alati RIK testijad esmalt lahenduse üle (testimisel on aluseks analüüsidokumentatsioon). Kui tegemist on arenduspartneri käest tellitud arendusega, siis annab täitja tellijale testjuhtumite kirjeldused kaasa (vastavalt vajadusele tehakse ka ise ärianalüüsi põhjal testjuhtumite kirjeldusi). Täpsustatud testimise protsess määratletakse ärianalüüsi etapis.

[1] Pealiskaudne testimine, millega tehakse kindlaks, kas põhifunktsionaalsus töötab. [39]

[2] Testimine leidmaks vigu integreeritud komponentide /moodulite vahelises interaktsioonis. [39]

Paigalduse testi tehakse algselt arenduskeskkonna peal ja sellega tegeleb RIK arendaja, testkeskkonda paigalduse testi teeb RIK administraator. Lõppkasutajatele toimuvad koolitused enne tootekeskkonda paigaldamist. Õigused, mis on vajalikud testimise jaoks on testjuhtumite juures eeldusena kirjeldatud.

Mittefunktsionaalsete nõuete täitmise kohta arenduspartner koostab raporti ja RIK poolsed haldurid ja administraatorid kontrollivad ka omalt poolt nõuete täitmist. Testimiseks on ette antud nõude kirjeldus ja nõudega seotud selgitus. Koormustestimiste puhul arenduspartner koostab raporti, testimine ise viiakse läbi majasiseselt ja juhiseid testimise läbiviimiseks ei tellita (koormustestimisel tööriistana kasutatakse *JMeter* programmi). Turvatestid tellitakse kõik väljast, tellimisega tegelevad RIK turvaspetsialistid. Projektijuhtidele edastatakse info, et millal testimine algab ning millal lõppeb, tulemusena koostatakse testimise raport. Muude testimise liikide kontekstis puhast testimist väljast ei tellita.

Testimise korralduse eest vastutavad projektijuhid. Testimisel on kasutatud ka eraldi testimise haldamise tarkvara *TestRail*. Hästi toimivaks hinnati meeskonnatööd, lisaks toodi välja, et testimisega seotud rollide kaasamine, informeerimine ja asjasse pühendamine peaks toimuma pigem varem kui hiljem ning ressursside juhtide poolt motiveerimine, et testimine on oluline.

**Siseministeeriumi infotehnoloogia-ja arenduskeskus (SMIT):**

SMIT-is viidi intervjuu töö autori poolt läbi ühe valdkonna kontekstis ning järgnevalt toodu ei kajasta SMIT ülest käsitlust arendusprojektides toimuvate testimistega.

SMIT-is varasemalt oli organisatsiooni ülene testimise ja kvaliteedijuhtimine, kuid seoses valdkondlike struktuuriüksuste moodustamisega on mindud üle agiilsele tarkvara arendusmetoodikale *Scrum*. Sellest tulenevalt on vastutus arendusprojekti kvaliteedi osas läinud rohkem üle projektiga seotud meeskonnale. Projekti põhiselt otsustatakse, et milliseid ressursse on meeskonda juurde vaja ning vastavalt sellele hangitakse pakkumuste põhjal erinevate rollide ressursse välistelt arenduspartneritelt. Arendusprojektides kaasatakse pigem alati partnerite poolseid testijaid. Projektides viiakse läbi funktsionaalsete nõuete testimine. Testimise kontekstis lepitakse ühiselt kokku, mida tähendab tehtud (*definition-of-done*). Nõue loetakse täidetuks, kui eelnevalt defineeritud „tehtud“ saab täidetud. Kasutusel on *jira* agiilsed töölauad, mille kaudu

liiguvad *jira* piletitena kirjeldatud ülesanded arendaja käest testijate kätte ja sealt edasi tooteomanikule (või äri poole peakasutajale). Lõplik valideerimine valdavalt toimub tooteomaniku poolt (tooteomanik on SMIT-i pool). Eraldi testjuhtumite kirjeldusi ei ole, *Jira* piletid sisaldavad testimiseks vajalikku infot. Eraldi plaani testimise jaoks ei tehta, lähtutakse agiilsest töölauast, selle kaudu saab info testitud ja veel testimata funktsionaalsuste kohta.

Mittefunktsionaalsed nõuded on pigem toodete spetsiifilised kui asutuse ülesed ning nende testimine sõltub vastava toote eripäradest. Turvatestimine tellitakse väljast ja nende testide tellimine toimub koostöös SMIT infoturbeosakonna ja tooteomanikega. Koormustestimise läbiviimine sõltub projektist ja meeskonna otsusest, konkreetseid ettekirjutusi ei ole. Koormustestimise vajadusel kaasatakse SMIT süsteemi administraatoreid. Testimise ees projektis vastutab meeskond, tooteomanik on juhtiv roll, kes korraldab ja kaasab vajalikke ressursse.

Probleemina toodi välja, et koormustestimiste tegemine on keerukas keskkondade erinevuste, ebapiisavate andmemahtude ning vajalike andmeseoste puudumise ja väliste süsteemidega andmevahetuse tõttu. Hinnati, et liialt keerukas on simuleerida testimisel olukorda, kus suur hulk kasutajaid loovad samaaegselt süsteemis erinevaid seoseid ja toimub andmevahetus liidestatud süsteemidega. Hästi toimivaks hinnati sprintide planeerimist, et ühe sprindi lõikes jõutakse kokkulepitud funktsionaalsus ära testida ja testimine ei jää lõputult venima. Lisaks toodi välja, et ainult arendaja pigem ei suuda enda tööd piisavalt hästi ära testida. Vajalik on testimisse kaasata erinevaid osapooli, iga ressurss teostab testimist oma aspektist lähtuvalt ning see aitab kaasa paremale kvaliteedile ja vigade tuvastamisele.

**Keskkonnaministeeriumi Infotehnoloogiakeskus (KEMIT):**

Peamised testimised, mida KEMIT-is arendusprojektides tehakse on funktsionaalsete ja mittefunktsionaalsete nõuete testid, koormustestid ja turvatestid. Arendused tellitakse arenduspartneritelt, kes teostavad omapoolsed testimised ja esitavad selle kohta raporti. KEMIT poolel testimise etapi alustamise eelduseks on arenduspartneri poolt edastatud tarne. Iga tarnega kaasneb paigaldusjuhend ning tarnitud funktsionaalsus peab vastama KEMITi poolt kehtestatud funktsionaalsetele ja mittefunktsionaalsetele nõuetele ning tehnoloogilisele profiilile (tehnoloogiline profiil on mittefunktsionaalsete nõuete osa).

Tarne valideerimisega tegeleb vastava rakenduse administraator ja vajadusel kaasatakse arhitekt. KEMIT poolne IT projektijuht informeerib äripoolt kui on võimalik testimisega alustada (tarnitud funktsionaalsus on testkeskkonda edukalt paigaldatud). Testimise aluseks olev dokumentatsioon sõltub projekti käigus tehtud kokkulepetest, sageli testitakse lähtuvalt ärianalüüsi nõuetest või kasutus- ja testilugude baasil. Funktsionaalsete nõuete testimise ja selle tulemuste fikseerimise eest vastutab äripoole projektijuht. Suuremahulistel projektidel on alati äripoolelt nimetatud testijuht, kes vastutab testide läbiviimise ja tulemuste fikseerimise eest.

Mittefunktsionaalsed nõuded lisatakse arendusprojektide tellimisel alati kaasa ning nende nõuete valideerimisega tegeleb KEMITi arhitekt. Koormus- ja turvatestid tellitakse väliste partnerite käest. Nende läbiviimise eest vastutab KEMITi projektijuht. Arenduse ühe osana teostab arenduspartner oma arenduskeskkonnas esialgsed koormustestid. Täiendavad koormustestid tellib KEMITi projektijuht. Vastav lähteülesanne koostatakse arhitekti, taristu esindaja ja IT projektijuhi koostöös. Koormustestide teostamiseks on KEMITil raamlepingupartner, kellele tagatakse turvatud ligipääs testkeskkonnale, kus sooritatakse koormustestid. Tulemuseks on koormustestide raport, mille alusel teostab arenduspartner vajadusel muudatused.

Turvatestid tellib KEMITi projektijuht, vastav lähteülesanne koostatakse infoturbejuhi, arhitekti, taristu esindaja ja IT projektijuhi koostöös. Turvatestide teostamiseks on KEMITil samuti raamlepingupartner, kellele tagatakse turvatud ligipääs testkeskkonnale, kus sooritatakse turvatestid. Tulemuseks on turvatestide raport, mille alusel teostab arenduspartner vajadusel muudatused.

Kõikide testimiste puhul vigade raporteerimiseks kasutatakse KEMITi vigade halduse keskkonda *Jira*. Testimised loetakse lõppenuks, kui äripool ja KEMIT kinnitavad, et tarne vastab nõuetele ning kõik mittevastavused on kõrvaldatud.

Sarnaselt RIKile toodi välja, et testijad peaksid olema rohkem motiveeritud, täiendavalt veel juurde, et oluline on julgustada ka äri poole testijaid rohkem küsima.

# 4 Parendatud testimise protsess

Üks peamisi testimise juhtimise väljakutseid on määratleda, et millist hulka funktsionaalsust on oluline rohkem või vähem testida ning millised testid on prioriteetsemad kui teised. Testimise tõhustamiseks on vajalik testitavate tingimuste ja kombinatsioonide hulka piiritleda. Üheks võimaluseks seda lahendada on identifitseerida ja analüüsida riske. Risk on ebasoovitav asjaolu, millel on esinemise tõenäosus ja potentsiaalsest negatiivne tagajärg. [13, ptk. 2.3]

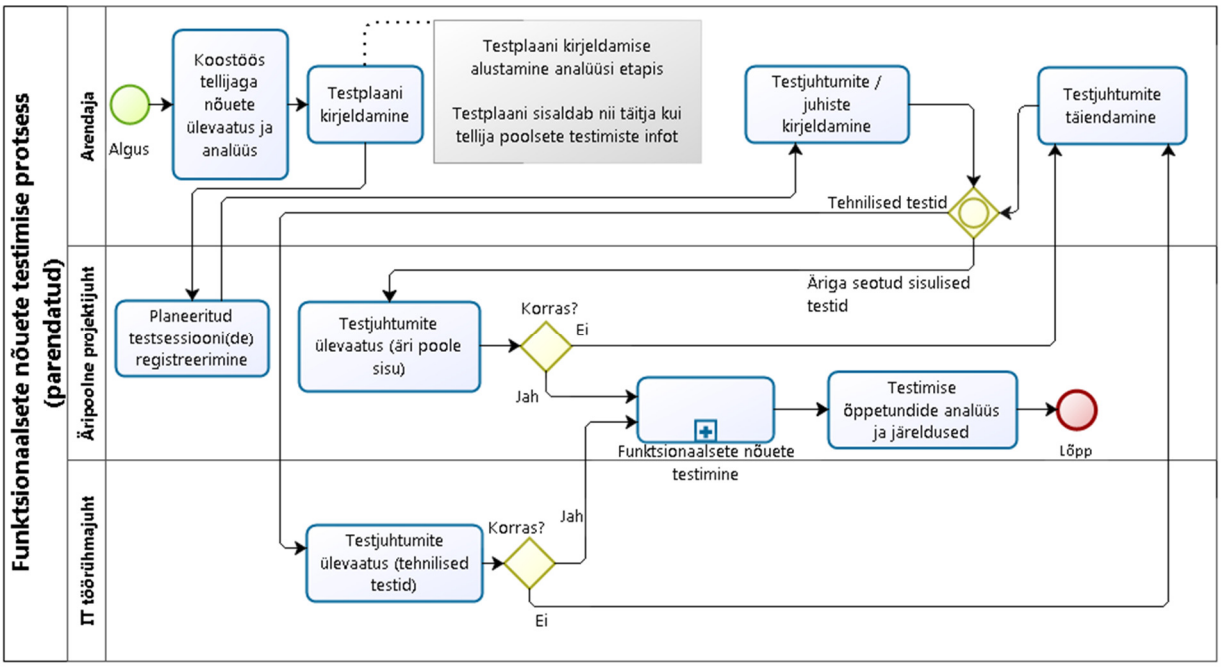
Töö autori hinnangul riskide hindamisele testimise kontekstis ei pöörata piisavalt tähelepanu. Projekti käigus tuleks määratleda toote kvaliteedi riskid (hinnata riski esinemise tõenäosust ja kahjulikku mõju) ning seeläbi teha järeldused, et millised testimised on ühe või teise projekti kontekstis kindlasti olulised läbi viia ja millised vähemolulised või ka üldse mitte vajalikud. Oluline oleks ka organisatsiooni testimise strateegia väljatöötamine, see koondaks testimise põhimõtted ja korralduse.

Vastavalt kriitilise testimise protsessi mudeli kirjeldusele [14] on testimise planeerimise üks oluline komponent testimise ja kvaliteediga seotud riskide kindlaks määramine. Selle protsessi käigus määratletakse, mida ja kui palju testitakse ja ka see, mida ei testita. [14, lk. 2]

Läbiviidud intervjuusid ja tuvastatud probleemkohti arvestades on järgnevalt välja toodud täiendatud testimise protsessid.

## 4.1 Funktsionaalsete nõuete testimine

Töö autor pakub välja parendatud funktsionaalsete nõuete testimise protsessi, mida iseloomustab järgnev joonis:




Joonis 12 Parendatud funktsionaalsete nõuete testimise protsess

Kuna testjuhtumid formuleeritakse peamiselt püstitatud nõuete ja analüüsi faasis loodud kirjelduste ning nõuete täpsustuste pealt, siis on oluline koostöös tellijaga nõuded üle vaadata ja läbi analüüsida. Juba analüüsi etapis on võimalik nõuete valideerimisel vigu tuvastada. Tarkvara testimise aluseid käsitlevas õppematerjalis [23, lk. 18] on välja toodud, et kui leiame vead nõuetes/tarkvara disainis, siis saame anda varakult tagasisidet ja vältida vigade kirjutamist koodi.

Analüüsi etapis tuleb alustada ka testplaani formuleerimisega. Algse testplaani, mis sisaldab infot täitja (arenduspartneri) poolsete testimiste kohta koostab arendusega tegelev meeskond. Edasised täiendused, mis hõlmavad tellijaga seotud keskkondasid ja ressursse on vajalik paika panna täitja ja tellija koostöös.

Testplaani koostamise eesmärgis on määratleda, leppida kokku, fikseerida ja kommunikeerida projekti liikmetele testimise ulatus ja testimisel kasutatavad metoodikad, testplaan võimaldab varakult teha kindlaks testimise jaoks vajalikud ressursid, keskkonnad ja muud nõuded, mis on testimisega seotud. [18, ptk. 7.2.2]

Kui projektipõhine testplaan on kokkulepitud, siis tuleks planeeritud testsessioonid registreerida asutusesiseses testplaanis, selle pealt saaks teadmise, et millised testimised veel samal ajal ja keskkonnas toimuvad.

Testjuhtumite kirjeldusega alustada võimalikult varakult (näiteks, kui mingi süsteemi osa või terviku kohta on analüüs tehtud ja nõuded paigas). Aja jooksul kirjeldusi kogu aeg täiendada. Arenduslepingu sõlmimisel üldjuhul alati ühe tulemina lepitakse kokku, et täitja edastab koos tarnega ka testjuhtumite kirjeldused. Kindlasti tuleks ka tellijal mõelda omalt poolt, et millised testjuhtumid on oluline läbida. Kui projekti käigus on testjuhtumite formaat kokkulepitud ja mingi arendatava / muudetava süsteemi osa kohta testjuhtumite kirjeldused täitja poolt valmis tehtud, siis tuleks need kirjeldused esimesel võimalusel edastada ka tellijale üle vaatamiseks (mitte edastada nö. lõplikke kirjeldusi tarnega koos) vaid nii vara kui võimalik. Sellisel juhul saaks tellija äripoolne projektijuht ja IT töörühma juht enne testsessiooni algust juba aegsasti teha oma ettepanekuid ja esitada küsimusi (ning ka sisemiselt testimisega seotud inimestele loodud kirjeldusi jagada). Kuna testjuhtumite kirjeldused on üheks lepinguliseks tulemiks, siis on joonisel 12 pandud testjuhtumite täiendamine täitja ülesandeks. Suuremate puuduste puhul peaks arendaja kindlasti omapoolsed täiendused tegema, samas oleks vajalik, et ülevaatuse käigus tellija pööraks oma sisu teadmisi arvesse võttes tähelepanu ka muudele võimalikele seotud tegevustele või töövoogudele, mida testjuhised kas skoobist või sisu spetsiifikast tulenevalt ei kata. Kui tellija selliseid aspekte omalt poolt tuvastab, siis võib ka omalt poolt testjuhiseid täiendada ja sinna vajalikud nüansid lisada. Kui testjuhtumid on tellija ja täitja vahel kokkulepitud, siis jätkub funktsionaalsete nõuete testimise protsess mis on kirjeldatud peatükis 3.1. Pärast testimise läbiviimist tuleks analüüsida lõppenud testimisega seotud õppetunde - seda näeb ette ka fundamentaalne testimise protsess (ptk.1.1) ning selle pealt arvestada, mida järgmises projektis teha paremini.

Sisearenduste puhul on testjuhtumite koostamine ja ka testimise planeerimine nõrgalt kaetud. Kuigi sisemised arendusprojektid on pigem mastaabilt väiksemad ei vähenda see testjuhiste olemasolu tähtsust. Sisearenduste puhul on samuti vajalik läheneda analoogselt nagu joonisel 12 kirjeldatud. RMIT ja äri poole koostöös panna paika testimise plaan. Süsteemi spetsiifikat arvestavad testjuhised võiks koostada näiteks IT töörühma juht (või rakenduse administraator) ja äri pool saab neid oma vajadustest tulenevalt täiendada.

Testimise plaani koostamisel näiteks võtta arvesse ISO/IEC/IEEE 29119-3 standardis toodud kriteeriumeid. Vastavalt sellele testimise dokumentatsiooniga seotud standardile sisaldab testplaan:

- testimise konteksti kirjeldust - projekti info, testimise skoop ja testitav funktsionaalsus, eeldused ja piirangud, huvigrupid;
- testimisega seonduv kommunikatsioon;
- riskid;
- testimise strateegia – testimise tulemite määratlus, testimise metoodika, testimise lõpetamise kriteeriumid, nõuded testandmetele ja testkeskkondadele, regressioonitestimine, testimise peatamise ja jätkamise kriteeriumid, kõrvalekalded organisatsiooni üldisest testimise strateegiast;
- testimise tegevused ja ajalised hinnangud;
- testimise meeskond – rollid, nende tegevused ja vastutused, võimalikud välise ressurssi vajadused, koolitus vajadus;
- testimiste ajakava. [19, lk. 52-53]

Testplaani koostamine ei peaks olema rangelt kinni ülal toodud ISO/IEC/IEEE 29119-3 standardis vaid pigem lähtuda sellest kui soovitusest. Arvestades projekti skoopi ja spetsiifikat kirjeldada testplaanis just selle projekti kontekstis olulisemad osad. Kuna punktis 3.1 oli probleemidena välja toodud ka, et testide läbimiseks vajalike õiguste info ei ole sageli lihtsasti leitav, siis üheks võimaluseks on välja tuua testplaanis testimise eeldusena, et millised õigused on testimiseks vajalikud. Teise võimalusena võib õiguste /rollide info lisada ka testjuhtumite juurde eeldusena nagu tehakse ka RIK-is. Õiguste / privileegide info on pigem välja toodud, kui tegemist on arendusprojekti käigus tekkivate uute kasutajaõigustega. Juba varasemalt eksisteerivate / olemasolevate õiguste infot tihti välja ei tooda. Eriti oluline on see juhul, kui testivad uued inimesed, kellel varasemalt õiguseid ei ole. Vajalik on, et testide läbiviimiseks vajalikud õigused oleks fikseeritud testimise kontekstis vähemalt eeldusena testjuhtumite juures, sõltumata sellest, kas tegemist on varasemalt eksisteerinud õigusega või uuega.

Peatükis 3.1 funktsionaalse testimise protsessi juures oli välja toodud probleemina, et mobiilivaadete testimisel puudub selgus, mida peab selleks tegema, et saaks mobiilsete seadmetega testida. Selle lahendamiseks on töö autori poolt läbi viidud vestlus RMIT töökohateenuste osakonna peaspetsialistiga. Vestluse käigus saadud info põhjal on kokku pandud kirjeldus, kus töö autor toob välja aspektid, millega on vaja arvestada mobiilsete seadmete tellimisel. Kokku pandud kirjeldust jagas töö autor organisatsioonis sees, kui

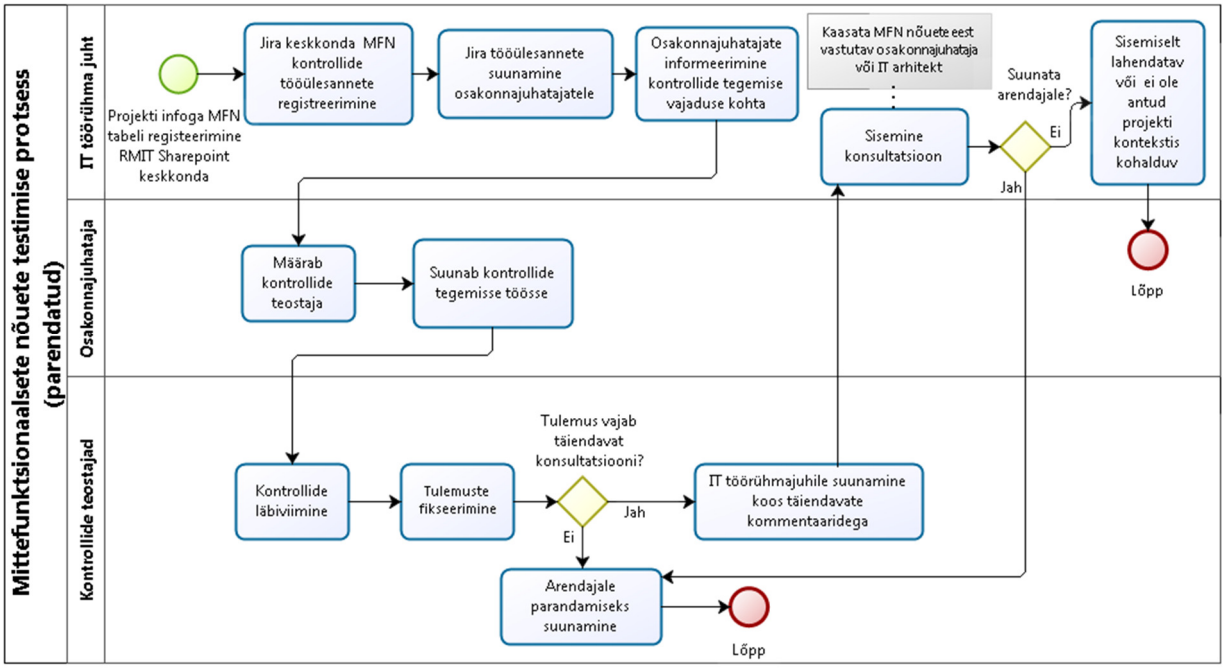
viidet abistavale juhisele, mida vajadusel projektides saab kasutada. Koostatud kirjeldus on toodud lisas 4.

Seonduvalt peatükis 3.1 välja toodud probleemiga, et testimisel kasutatavate keskkondade uuendamisega kaovad ära testimiseks ettevalmistatud andmed, ei ole ühtset lahendust võimalik välja tuua. Osade projektide jaoks on just vajalik värske andmebaasi seis. Autori hinnangul on sellele probleemile võimalik läheneda juhtumipõhiselt, üldiselt kui toimuvad testimiseks kasutatavate keskkondade andmete uuendamised, siis sellest teavitatakse ette mitu nädalat ja kooskõlastatakse sisemiselt. Kui on teada, millal uuendamine tuleb, siis võimalusel püüda jõuda testimine läbi viia enne uuendamiseks kokkulepitud aega. Kui ajakava võimaldab, siis oodata andmete ettevalmistamisega, kuni uuendamine saab tehtud. Kui andmed siiski ära kaovad ja nende ettevalmistamine on olnud suurem töö (või sobivatele kriteeriumitele vastavaid andmeid on raske äri poolel leida), siis üheks võimaluseks on RMIT rakenduse administraatori käest tellida, et ta omalt poolt abistaks ja otsiks välja tingimustele vastavaid andmeid. Variandiks on ka juhtumipõhiselt hinnata - kui tegemist on mingi kriitilisema testimisega, mis keskkonna uuendamise tulemusena kannatada saab, siis tuleks see probleemina üles tuua. Olukorda tuleks hinnata RMIT ja äri poole koostöös vastavalt situatsiooni spetsiifikale. Olukorrast tulenevalt saab teha otsuse, et kas ettevalmistatud andmete tagavara koopia on võimalik (sageli see võimalik ei ole eelkõige seetõttu, et andmete vahelised seosed lähevad katki). Andmetest tagavara koopia tegemine on tõenäolisemalt võimalik, kui tegemist on andmetega, mida tootekeskkonnas veel ei ole (aga ka siin on olulised seosed varasemalt eksisteerinud andmetega). Selle probleemi puhul on oluline osapoolte vaheline kommunikatsioon ja juhtumipõhine hindamine. Otsused saab teha olukorra spetsiifikast tulenevalt ja kui projektide ajakavad võimaldavad, siis variandiks on ka testimiseks kasutatava keskkonna uuendamise kuupäeva muutmine / edasi lükkamine.

Seonduvalt probleemiga, et parandustarnete puhul sageli ei testita üle juba varasemalt testitud funktsionaalsust, siis siin kohal on autori hinnangul oluline, et tellija vähemalt määratleks ära kriitilisema (või kõige prioriteetsema) töövoo ning arvestaks, et enne uue versiooni tootesse paigaldamist saaks põhitöövoog siiski üle testitud / kontrollitud.

## 4.2 Mittefunktsionaalsete nõuete testimine

Töö autor pakub välja parendatud mittefunktsionaalsete nõuete testimise protsessi, mida iseloomustab järgnev joonis:




Joonis 13 Parendatud mittefunktsionaalsete nõuete testimise protsess

Joonisel 13 on täiendatud peatükis 3.2 kirjeldatud mittefunktsionaalsete nõuete testimise protsessi. Ühe probleemina tänase mittefunktsionaalsete nõuete testimise juures oli välja toodud, et kuna protsess ei ole kõigi jaoks üheselt fikseeritud, siis pole kontrolli tegijatel selgust, kas leiud fikseerida mittefunktsionaalsete kontrollide vastavuse tabelisse, edastada IT projektijuhile, registreerida ise *jira*sse ja suunata arendajale või tuleb kõik tulemused veel täiendavalt sisemiselt üle vaadata enne arendajale suunamist.

Võrreldes peatükis 3.2 toodud protsessi kirjeldusega on joonisel 13 täpsustatud tegevused alates tulemuste fikseerimisest. Juhul kui kontrollide tulemusel tuvastatakse kõrvalekalle, mille osas ei ole vaja täiendavat sisemist konsultatsiooni (tuvastatud mittevastavus on selgelt viga, mis tuleb parandada arendaja poolt), siis tuleks mittefunktsionaalsele nõudele mitte vastavus raporteerida *jirasse* arendajale parandamiseks kontrolli teostaja poolt. Kõrvalekalde registreerimisel kindlasti tuua välja mittefunktsionaalse nõude identifikaator ja nõude kirjeldus ning selgitav kommentaar puuduse kohta. *Jira* kande jälgijaks või teavitatavaks lisada ka IT töörühma juht ning lisada *jira* viide projektiga seotud kontrollide tabelisse. Juhul, kui kontrolli teostaja tunneb, et tuvastatud

mittevastavus oleks vajalik enne arendajale suunamist sisemiselt läbi arutada tuleks kontrolli teostajal teha vastav *jira* kanne ja suunata IT töörühma juhile, kes organiseerib sisemise ülevaatuse või konsultatsiooni. Vajadusel kogub IT töörühma juht kokku mitme kontrollija poolt edastatud küsimused ning organiseerib kas koosoleku vormis või e-kirja teel küsimuse all olevate nõuete ülevaatuse, et saada lõplik otsus. Sisemise konsultatsiooni käigus selgub, kas kontrolli käigus tuvastatud kõrvalekalle tuleb edastada arendajale parandamiseks. Põhjused, miks küsimuse all oleva nõude täitmist arendajale parandamiseks ei suunata, võib olla näiteks:

- on teada, et antud nõue on planeeritud muuta (või eemaldada) järgmises mittefunktsionaalsete nõuete versioonis;
- nõude täitmiseks on vajalikud sisemised tegevused;
- nõue ei kohaldu antud projekti kontekstis.

Sisemise konsultatsiooni osas IT töörühma juht kaasab vastavalt vajadusele mittefunktsionaalsete nõuete eest vastutavat osakonnajuhatajat, IT arhitekti (kontrolli teostajat või RMIT spetsialisti, kelle valdkonnaga nõude täitmine on seotud). Sisemise konsultatsiooni käigus tehtud otsused / kommentaarid fikseeritakse IT töörühma juhi poolt projektiga seotud kontrollide tabelisse.

Arendajale raporteeritud mittefunktsionaalsete nõuete kõrvalekalded on vajalik arenduspartneri poolt parandada ning RMIT-le edastada parandustega korrektselt vormistatud tarne. Kõik parandused on vajalik pärast tarne paigaldust kontrollide tegijatel uuesti üle vaadata - kontrolli teostaja veendub, et viga on kõrvaldatud ning sulgeb kõrvalekaldena raporteeritud *jira* tööülesande vastava kommentaariga.

Seonduvalt peatükis 3.2 välja toodud probleemiga, et osade mittefunktsionaalsete nõuete kontrollimeetod ei ole üheselt arusaadav, toob töö autor vajadusena välja, et sellised nõuded tuleks ära fikseerida ning kirjeldusi täiendada testimist selgitava infoga.

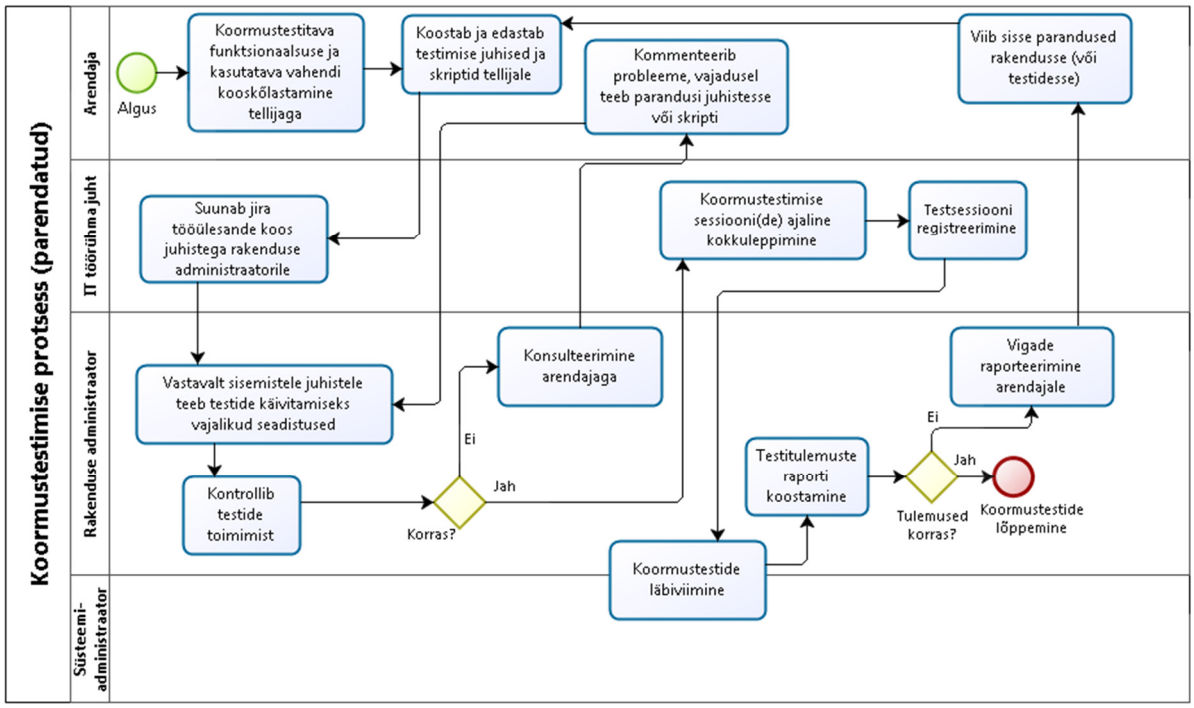
Oluline oleks jõuda mittefunktsionaalsete nõuete kontrollid tehtud enne tootesse paigaldamist, seda tuleks tähtsustada ka osakonnajuhtide poolt. Tänases protsessis vajab mittefunktsionaalsete nõuete testimiste lõpule viimine väga palju IT töörühma juhi poolset meelde tuletamist.

Kasutatavuse testimise aspekt on arendusprojektides töö autori hinnangul nõrgalt kaetud ja siinkohal oleks vajalik ära määratleda üldpõhimõtted, et milliste projektide puhul

kindlasti kasutavusega seonduvaid teste tuleb läbi viia, mis meetodeid ja vahendeid selleks testimiseks kasutada ning vastutused seoses kasutatavuse testimistega.

## 4.3 Koormustestimine

Töö autor pakub välja parendatud koormustestimise protsessi, mida iseloomustab järgnev joonis:




Joonis 14 Parendatud koormustestimise protsess

Joonisel 14 on toodud täiendused peatükis 3.3 kirjeldatud koormustestimise protsessile. Ühe probleemina seoses koormustestimistega oli välja toodud, et koormustestide eelnev kooskõlastamine tellijaga on vähene. Kuigi kooskõlastamise näeb ette ka tänane koormustestimise läbiviimise protsess, ei ole see aspekt pigem kaetud. Protsessi alguses on vajalik, et koormustestitav funktsionaalsus (kui ka testimisel kasutatav vahend) saaks osapoolte vahel kokkulepitud.

Pärast testjuhiste ja testimisel kasutatavate skriptide koostamist arendaja poolt suunab IT töörühma juht *jira* tööülesande RMIT rakenduse administraatorile. Kuna olemasolev koormustestimise tööprotsessi kirjeldus ei kajasta RMIT rakenduse administraatori jaoks olulisi sisemisi (testide seadistamiseks vajalikke) juhiseid, siis on joonisel 14 toodud välja, et rakenduse administraator saab võtta testide toimimise kontrollimiseks ja

seadistuste tegemiseks aluseks kirjeldatud sisemised juhised. Rakenduse administraator teeb esmased testide kontrollid ja vajadusel konsulteerib arendajaga, kes puuduste korral teeb omalt poolt korrektuure või annab täiendavaid selgitusi. Kui testid on korras, siis organiseerib IT töörühma juht koormustestimise sessiooni kokkuleppimise (kaasates sinna RMIT rakendusserveri administraatorit, andmebaasi administraatorit, rakenduse administraatorit, arendajat). Lisaks informeerib koormustestimise sessioonist äripoole projektijuhti ja ka süsteemide hoolduse osakonnajuhatajat. Kui ajaline kokkulepe on tehtud, siis IT töörühma juht registreerib koormustestimise sessiooni. Üheks variandiks seda teha on lisada kokkulepitud ajale kalendrikutse seotud osapooltele (seda on igal juhul soovituslik teha). Koormustestide läbiviimisel kasutatakse ühte serverit ja seega oleks hea, kui planeeritavad testsessioonid saaks registreeritud näiteks testsessioonide kalendrisse (või peatükis 4.1 välja toodud asutusesiseses testplaanis). Edasi toimub koormustestide läbiviimine kokkulepitud ajal. Selles osas on protsessi kirjeldus sarnane peatükis 3.3 tooduga.

Seonduvalt probleemiga, et koormustestimiseks ei ole eelnevalt olemas piisavalt andmeid või vajalikku andmemahtu, siis see vajab alati hindamist juhtumipõhiselt. Eelkõige võib see teema olla probleemiks, kui testitav objekt on uus komponent, mille poolt kasutatavad andmetabelid on tühjad. Sellisel juhul tuleks alati mõelda juba koormustestide planeerimisel andmete tekitamise võimalustele. Samuti tuleb juhtumipõhiselt hinnata ka seda, et kas koormustestimise tulemusena saavad mõjutatud äri poole testimiseks vajalikud andmed või tekivad testimise tulemusel poolikud andmed, mis oluliselt segavad edasist testimist. Kui selline probleem realiseerub, siis tuleb otsused teha vastavalt olukorrale. Vajadusel kaaluda andmete taastamist koormustestide eelsesse seisu.

Kuna arenduspartneri poolt koos koormustestide skriptidega tulevad kaasa ka juhised ja töö autori poolt RMIT rakenduse administraatoritega läbiviidud intervjuu tulemusena selgus, et juhised on sageli ebaselged, siis on vajalik ära määratleda, milline on nõuetele vastav juhis. Ilmselt projektide põhiselt on erinevaid nüansse ja ühtselt malli, mis kõikidesse projektidesse alati sobib ei ole. Aga siiski töö autori hinnangul peab olema võimalik paika panna üldised kriteeriumid, mida tuleb arvestada (näiteks fikseeritud koormusnõuded ja tehnilised piirid, testimise eeldused ja kasutatavad vahendid, millist funktsionaalsust testitakse ja mis on testide eesmärk, juhised testide käivitamiseks, viited testimiseks vajalikele skriptidele).

## 4.4 Turvatestimine

Tänane turvatestimise protsessi kirjeldus on toodud joonisel 9. Turvatestimise teemal läbiviidud intervjuu tulemusena selgus, et probleemiks turvatestimise juures on puudulik info vahetus. Turvatestimise protsessi kirjeldust siiani ei ole eksisteerinud, autori hinnangul võib jääda joonisel 9 toodud protsess kehtima, kuid olenemata sellest, kas turvatestimine tellitakse väljast või otsustatakse teha sisemiselt – on vajalik osapoolte vahel enne testimist kokku leppida kanal jooksvaks infovahetuseks. Informeerida sinna kanalisse tegevustest, mis võivad mõjutada turvatestimise läbiviimist (näiteks jooksvate parandustarnete paigalduste info (sh. info paranduste sisu ja muudatuste kohta), projektis toimuvate või projekti mõjutavate koormustestimiste sessioonide ajad ning info katkestuste kohta, mis toimuvad turvatestimisega seotud keskkonnas). Üldiselt tekib ka turvatestijatel töö käigus jooksvaid küsimusi, seega infovahetus kogu turvatestimise perioodi vältel peaks osapoolte vahel toimuma võimalikult operatiivselt.

Lisaks sellele, et turvatestimisega tegelejad saaksid vajaliku info on vajalik ka, et IT töörühma juhil oleks selgus, et mis ajal valmib lõplik tulem (turvatestimise raport). Tulemi(te) üleandmise kuupäev määratletakse tavaliselt osapoolte vahel sõlmitavas lepingus ja see info tuleks IT töörühma juhile kindlasti edasi anda (sh info selle kohta kui tulemi üleandmine mingil põhjusel jääb viibima).

# 5 Tulemuste analüüs

Töö tulemusena on kaardistatud RMITis tarkvara arendusprojektides toimuvate testimiste protsessid. Asutuse sees läbiviidud intervjuude tulemusena on toodud välja erinevad probleemkohad ning töötatud välja parendustega testimise protsess. Peamisi erinevusi tänase ja parendatud protsessi vahel iseloomustab järgnev tabel:

Tabel 2 Erinevused tänase ja parendatud testimise protsessi vahel

| **Funktsionaalsete nõuete testimine** | |
|---|---|
| **Tänane protsess** | **Parendatud protsess** |
| Protsessi kirjeldus puudu. | Protsessi kirjeldus loodud. |
| Testimise planeerimine pinnapealne. Testimise planeerimine projektiplaanis. | Projekti põhise testplaani koostamine (tellija ja täitja koostöös). Testplaan sisaldab nii tellija kui täitja poolset testimiste infot. Planeerimisel võtta aluseks standard ISO/IEC/IEEE 29119-3 ja kohandada vastavalt vajadustele. Asutusesisese testplaani pidamine, mis annab teadmise, et millised teiste projektide testimised veel samal ajal ja keskkonnas toimuvad. |
| Testimisel põhinemine arendaja poolt koostatud juhistel. | Tellija poolne testjuhtumite ülevaatus, hindamine (ka täiendamine). |
| Sisearenduste puhul testjuhtumeid ei tehta. | Sisearenduste puhul testjuhtumite koostamine. |
| Mobiilivaadete testimisel ebaselgus, mida peab selleks tegema, et saaks seadmetega testida. | Koostatud juhis, mida saab mobiilsete seadmete tellimisel võtta aluseks. |
| Testimiseks vajalike õiguste info ei ole lihtsasti leitav. | Testimiseks vajalike õiguste info lisamine eeldusena testplaani ja / või testjuhtumite juurde. |

Tabel 2 Erinevused tänase ja parendatud testimise protsessi vahel

| | |
|---|---|
| Testimise lõppemisel ei tehta tagasivaatavat hindamist. | Testimise lõppemisel õppetundide analüüs. |
| **Mittefunktsionaalsete nõuete testimine** | |
| **Tänane protsess** | **Parendatud protsess** |
| Protsessi kirjeldus puudu. | Protsessi kirjeldus loodud. |
| Kõrvalekallete raporteerimine ei ole ühtselt kõigile selge. | Kõrvalekallete arendajale raporteerimine protsessis fikseeritud. |
| Osade nõuete puhul ei ole testimise meetod selge. | Täiendada nõuete selgitusi „kuidas testida“ infoga. |
| Kasutatavuse testimine vähene. | Põhimõtete väljatöötamine kasutatavuse testimise kontekstis. |
| **Koormustestimine** | |
| **Tänane protsess** | **Parendatud protsess** |
| Testide kooskõlastamine tellijaga vähene või ei tehta. | Koormustestitava funktsionaalsuse ja testide kooskõlastamine tellijaga. |
| Sisemiste juhiste olemasolu rakenduse administraatoritele puudub. | Rakenduse administraatorid põhinevad sisemisel juhisel testimise ettevalmistamisel. |
| Koormustestimise sessioone ei registreerita. | Koormustestimise sessiooni registreerimine testplaani. |
| Tellijale koormustestimise juhise kokkupanemiseks nõuded puuduvad (juhised on pigem ebaselged). | Arendaja võtab aluseks tellija nõuded ja koostab selle alusel koormustestide juhised tellijale. |
| **Turvatestimine** | |
| **Tänane protsess** | **Parendatud protsess** |
| Protsessi kirjeldus puudu. | Protsessi kirjeldus loodud. |
| Info jagamine turvatestijatele testsessiooni ajal vähene või ei tehta. | Info jagamine turvatestijatele paigalduste, katkestuste, koormustestimise sessioonide jm kohta, mis nende tööd mõjutab. |

Lisaks on peatükis 4 „Parendatud testimise protsess“ välja toodud ka kaks ettepanekut, mida võtta pikemaajaliseks eesmärgiks testimise kontekstis. Toote kvaliteedi riskide hindamine projektides ning organisatsioon testimise strateegia väljatöötamine.

## 5.1 Funktsionaalsete nõuete testimine

Funktsionaalsete nõuete testimise aspektist on peatükis 3.1 toodud välja, et testimise planeerimisele ei pöörata piisavalt rõhku, seega on funktsionaalsete nõuete testimise parendatud protsessis toodud välja testplaani kirjeldamise vajadus. Testplaani täiendamine ja ajakohasena hoidmine projekti käigus on oluline, vastasel juhul ei ole plaani koostamisel mõtet. Töö autori hinnangul testplaanide koostamine (ja nende ajakohasena hoidmine) projektide käigus aitab paremini planeerida ressursse ja annab kõigile projekti osapooltele selguse, et mis on testimise skoop ja metoodika, milliseid teste planeeritakse teha ning millistes keskkondades, kelle poolt ning mis aegadel. Mis on piirangud testimisel ning milliseid vahendeid planeeritakse kasutada. Peatükis 4 on töö autori poolt välja toodud soovitus testimise planeerimisel võtta aluseks ISO/IEC/IEEE 29119-3 standardis toodud kriteeriumeid ning kohandada (või täiendada) neid vastavalt projekti vajadustele.

Testplaani koostamisel tasuks arvestada ka kriitilise testimise protsessi käsitlevas raamatus toodud aspektiga – standardid ja mallid on kasulikud, kuid neid kasutades ei tohi unustada olulist projekti kontekstis. Testplaani malli jälgimisel võib tekkida olukord, kus selle täitmisse suhtutakse kui vajadusse täita dokumendis lüngad. Mallile tuginemine võib olla abistav kuid see ei asenda mõtlemist testimise planeerimisel. [21, ptk. 7]

Lisaks on kriitilise testimise protsessi raamatus toodud ka, et lihtsalt testplaani kirjutamisest ei ole kasu. Kirjeldatud plaanist on kasu vaid juhul, kui projekti meeskond ka sellest lähtub, selle järgi tegutseb ja seda arvestab. Kõik testimisega seotud osapooled peavad olema teadlikud ja nõustuma tegema seda, mis on plaani järgi nende vastutus ja kohustus. [21, ptk. 7]

Eduka tarkvara testimise aluseks on testimise planeerimine. [17, lk. 54] Testimise planeerimise eesmärgiks on võtta arvesse testimise strateegiat, ressursse, vastutusi, riske ning prioriteete. [16, ptk. 3.5] Autori hinnangul on testimise planeerimisele suurema rõhu

panemine oluline, kuna see annab kõigile testimisega seotud osapooltele selguse testimise ajakava ja tegevuste osas.

Funktsionaalsete nõuete testimisega seonduvalt oli probleemkohaks ka see, et sageli toimub testimine ainult täitja poolt etteantud testide kirjelduste ja juhiste põhjal, seetõttu jääb suur vastutus täitja teadlikkusele ja suutlikkusele kirjeldada testjuhtumid nii, et need kataks kõik vajaliku. Testjuhtumitega seonduva probleemina oli peatükis 3.1 toodud välja, et testjuhtumid on pigem liialt pinnapealsed, seega võib järeldada, et ainult täitja teadlikkusele ja põhjalikkusele lootma jääda ei ole mõistlik. Parendatud funktsionaalsete nõuete testimise protsessis on töö autori poolt välja toodud, et täitja peaks testjuhtumid aegsasti tellijale üle vaatamiseks edastama. Tellija poolse ülevaatuse käigus on oluline pöörata sisu teadmisi arvesse võttes tähelepanu ka muudele võimalikele seotud tegevustele või töövoogudele. Puuduvate aspektide katmiseks vajadusel viima omalt poolt sisse täiendusi / märkusi, tegema ettepanekuid test stsenaariumite parendamiseks või arvestama kirjeldamata tegevuste läbiviimisega testimise käigus.

Testjuhtumite kirjeldused peamiselt põhinevad analüüsi faasis määratletud ja kirjeldatud nõuetel ning testimise käigus kontrollitakse vastavust nõuetele. Autori arvates kui arendaja loob uue süsteemi või täiendab olemasolevat, siis arendaja poolsed testjuhtumite kirjeldused pigem fokuseerivad süsteemi käitumisele ja sellele, et süsteem vastaks nõuetele.

Äri poole tegevust toetava infosüsteemi väljatöötamiseks ja tarnimiseks ei ole ainult oluline verifitseerida, et süsteem vastaks nõuetele vaid ka valideerida, et süsteem pakuks äri poolele väärtust, mida kasutajad tegelikult ootavad. [20, lk. 1] Seetõttu on ka oluline, et testimine ei piirduks ainult arendaja poolt kirjeldatud juhistel. Nagu ka testplaaniga ei ole kindlasti mõtet testjuhtumeid toota lihtsalt seetõttu, et midagi oleks kirjeldatud. Töö autori hinnangul on halb, kui testjuhtumid üldse puuduvad - eelkõige uute süsteemide puhul sageli ei suuda vastuvõtu testijad sel juhul kõiki süsteemi aspekte läbi testida. Sisearenduste puhul on vajalik projekti käigus kokku leppida ja fikseerida stsenaariumid / kirjeldused, mida testimisel aluseks võetakse. Autori hinnangul on halb ka see, kui testjuhtumid on üle detailiseeritud. Kuna ajas tuleb sageli teha täiendusi ja mida granulaarsemad on kirjeldused, seda kauem võtab aega nende täiendamine või muutmine.

Teiste asutustega läbi viidud intervjuudest selgus, et TEHIK ja RIK pigem ei keskendu testimisel põhiosas arendaja poolt antud juhistele vaid teevad omalt poolt täiendavaid testimisi, TEHIK koostab palju testjuhtumeid ise. KEMITis sõltub projekti käigus tehtud kokkulepetest, aga sageli kasutatakse testimise aluseks ärianalüüsi või kasutusjuhtumeid. Töö autori arvates ettekirjutatud testjuhtumitele lisaks täiendavate testimise tegevuste läbiviimine aitab suurema tõenäosusega tuvastada rohkem vigu.

Ametnikud, keda kaasatakse testrühma teevad testimisi oma tavapäraste töökohustuste kõrvalt (üheks testgruppi kaasamise kriteeriumiks võib olla fakt, et testivad need, kellel on rohkem aega). Kui osa testimisi teevad isikud, kes igapäevaselt testitava funktsionaalsusega kokku ei puutu, siis võib juhtuda, et testija ei oska tulemust tõlgendada või kõrval mõju märgata. Efektiivseks testimiseks ei piisa, kui fokuseerida positiivsele stsenaariumile, etteantud kirjeldustele või ainult süsteemile. On vaja teada nõudeid ja protsesse, mis omakorda tulenevad - ülesande püstitusest, organisatsioonist, seadusandlusest, standarditest, riskianalüüsist, headest tavadest, kasutusviisist jne. [8, lk. 28] Mida rohkem testija saab aru äriprotsessidest, omab tausta infot äri poole sisu ning eesmärkidega seonduvalt, seda paremini ta suudab leida vigu ning seda tulemuslikum on testimine.

Testimise baastaseme koolitusprogrammis (*ISTQB foundation level syllabus [10]*) on öeldud, et süsteemis esinevate vigade tuvastamiseks on testijale vajalikeks omadusteks – uudishimu, kriitiline meel, detailidele tähelepanu, hea suhtlusoskus ja kogemus valdkonnas. [10, ptk. 1.5] Uurimuses, mis kirjeldab kõrge sooritusvõimega testijate omadusi [22] on välja toodud, et olulised on valdkonna teadmised ja ka erialased tehnilised oskused. Vähem tähtsam pole ka kogemus arendatava või muudetava tootega ja suurema pildi omamine (näiteks teadmised seotud süsteemidest ja andmete liikumise ahelast).

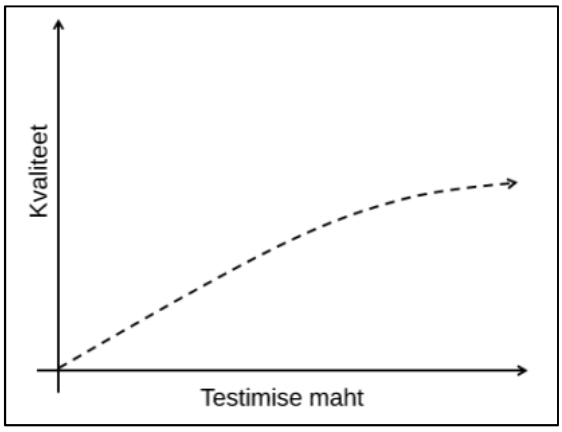
Üheks veaks, mida autori hinnangul sageli tehakse on see, et vastuvõtu testimisega tegelejaid kaasatakse tihti pigem hiljem kui varem projekti infovälja. Vastuvõtu testijal sageli ei ole enam oluliselt aega tausta informatsiooni uurida, kui testjuhised juba täitmiseks edastatakse. Ka RIKga vestluse tulemusena on välja toodud, et testimisega seotud rollide kaasamine, informeerimine ja asjasse pühendamine peaks toimuma pigem varem kui hiljem ning samuti on oluline ka ressursside juhtide poolt motiveerimine, et testimine on oluline.

Reaalsed süsteemi kasutajad on väga olulised testijad, testgruppi tuleks neid kindlasti kaasata. Soovitatavalt tuleks neid kaasata aegsasti. KEMITiga vestluses toodi ka välja, et oluline on julgustada äri poole testijaid rohkem küsima.

Seoses probleemiga, et parandustarnete puhul sageli ei testita üle varasemalt juba testitud funktsionaalsust on peatükis 4 töö autori poolt välja toodud, et oluline on tellija poolt määratleda ära kriitiline töövoog ning arvestada, et enne uue versiooni tootesse paigaldamist saaks see põhitöövoog siiski üle kontrollitud. Testimist, mis on suunatud vigade avastamisele varem testitud rakenduses või programmikoodis nimetatakse regressioonitestimiseks [24]. Autori hinnangul ei ole kõikide funktsionaalsuste ja olukordade üle testimine mõistlik, kuid tuleb saada kindlus, et prioriteetne funktsionaalsus toimiks.

Eelnevalt toodud hinnang toetub ühele seitsmest testimise baastaseme koolitusprogrammis (*ISTQB foundation level syllabus [10]*) välja toodud testimise põhiprintsiibile - kõikehõlmav testimine ei ole võimalik ja pigem on oluline testimisel arvesse võtta riske ja prioriteete. [10, ptk. 1.3] Kindlasti tuleks regressioonitestimisel arvestada ka seda, et suure tõenäosusega tekib nö. pestitsiidi paradoks, see on samuti üheks testimisega seotud põhiprintsiibiks – kui samasuguseid teste pidevalt korratakse, siis ei leia need lõpuks enam ühtegi viga. Pestitsiidi paradoksi vältimiseks on vajalik pidevalt korratavaid testjuhtumeid regulaarselt üle vaadata ning täiendada, et leida potentsiaalseid uusi vigu. [10, ptk. 1.3]

Artiklis, mis käsitleb manuaalse regressioonitestimisega üle pingutamist [25] on toodud , et üldlevinud arvamus on, mida rohkem testida, seda parem on kvaliteet, kuid teatud hetkel rohkem testimine ei anna enam nii palju kvaliteedile juurde kui võiks arvata. Sõltuvalt arendatavast tootest ja organisatsioonist võib määratlus piisavast testimisest olla erinev, kuid testimise ja kvaliteedi vahelise seost iseloomustab hästi all toodud pilt [25, ptk. 2.2]:




Joonis 15 Testimise ja kvaliteedi vaheline seos [25]

Mingi ajahetkeni me testide täiendamise ja uute tegevuste lisamise (ja ka kirjeldamisega) suurendame kvaliteeti, kuid ühel hetkel jõutakse kriitilise piirini, kus testjuhtumeid on üleliia palju või testplaan on muutunud nii mahukaks, et nende haldamine ei ole enam lihtne. Sellisel juhul võidakse hakata loodud kirjeldusi ignoreerima, dokumentatsioon kipub ajast maha jääma ning keegi ei suuda seda enam hallata ja jälgida. Selles punktis võib juhtuda, et kvaliteet hakkab hoopis langema. [25, ptk. 2.2]

Autori hinnangul tuleb nii testplaani koostamisel, testjuhtumite kirjeldamisel ja testide läbiviimisel arvestada mõistlikkuse printsiipi - nii palju kui vajalik ja nii vähe kui võimalik, võttes arvesse konkreetse projekti eripärasid, prioriteete ja riske. Regressioonitestimise aspektist töö autori hinnangul tasuks kaaluda automatiseerimise võimalusi. Automatiseerimise vajalikkus ja ka võimalikkust tuleks eraldi läbi analüüsida. Uurimuses, mis käsitleb testimise automatiseerimise piiranguid ja kasu [26] on toodud, et kuigi automatiseeritud testid hoiavad kokku testimisele kuluvat aega, siis peab kogu automatiseerimine olema hästi läbimõeldud, automaatse testimise skriptid vajavad hooldust ja uuendamist ning ka vastavat tehnilist kompetentsi. Testimise automatiseerimine ei pruugi suurendada vigade tuvastamise tõenäosust. Automaatsed testid koostatakse testijate poolt ning palju sõltub testimismeeskonna oskustest ja võimekusest implementeerida reaalset kasu andev automaattestimise lahendus. [26, lk. 39-40]

Ühe soovitusena on töö autor peatükis 4 toonud välja, et arendusprojektides oleks vajalik määratleda toote kvaliteedi riskid (hinnata riski esinemise tõenäosust ja kahjulikku mõju) ning seeläbi teha järeldused, et mida on kõige olulisem testida.

Riskide hindamine võib olla aeganõudev tegevus ja aega tavaliselt arendusprojektides ei ole. Seetõttu võivad meeskonnad võtta lähenemise „testime nii palju kui võimalik“ või „testime seda mida oskame“. Kumbki mainitud lähenemine ei ole hea. [31, lk. 3] Töö autori hinnangul on vajalik täpsemalt analüüsida, milline oleks IT projektide jaoks sobivaim riskide hindamise metoodika. Kindlasti ei peaks testimisel lähtuma vaid riskidel põhineval testimisel, tuleks kasutada ka teisi testimise viise, näiteks ekspertteadmiste põhine testimine, koormustestimine, turvatestimine. Konkreetseid testimise viise, mida kasutada, tuleks kaaluda vastavalt projekti spetsiifikale ja eesmärkidele.

## 5.2 Mittefunktsionaalsete nõuete testimine

Mittefunktsionaalsete nõuete kontrollimise organiseerimine on IT töörühmajuhi ülesanne, kuid protsessi kohta puudus kirjeldus. Käesoleva töö peatükis 4 on töö autori poolt toodud mittefunktsionaalsete nõuete testimisega seotud parendatud protsessi kirjeldus. Autori hinnangul on see abistavaks näiteks uutele RMITi IT projektijuhtidele, kes saavad väljatoodud kirjelduse alusel selgema arusaamise, et kuidas mittefunktsionaalsete nõuete kontrollimine ja selle organiseerimine täpsemalt tuleb lahendada.

Töö autori poolt läbiviidud intervjuude käigus selgus, et kontrolli tegijatel pole täpselt selge, kas leiud tuleb lihtsalt fikseerida tabelisse, edastada IT projektijuhile, registreerida ise *jira*sse ja suunata arendajale või tuleb kõik tulemused veel täiendavalt sisemiselt üle vaadata. Sellest tulenevalt teeb töö autor ettepaneku, et kui tuvastatud mittevastavus on selgelt viga, siis tuleks see kontrolli teostaja poolt ise arendajale parandamiseks suunata. Selliste kõrvalekallete puhul ei ole vajalik seda edastada IT töörühma juhile ega lihtsalt kontrollide aluseks olevasse tabelisse märkida, vea kohene registreerimine *jirasse* hoiab kokku aega ja nii ei ununе probleem kontrollimise aluseks olevasse tabelisse seisma (välisel arendajal puudub ligipääs nimetatud tabelile).

Kuigi toimuvad regulaarsed mittefunktsionaalsete nõuete korrigeerimised ja täiendamised, siis töö autori arvates oleks vajalik üle vaadata ka selgitused, mida nõuete juures kuvatakse. Tuleks koguda tagasisidet selliste nõuete kohta, mille kontrollimise meetodid ei ole lõpuni selged. Nõuete juurde kuuluvaid selgitusi tuleks kriitilisemalt üle vaadata ja vajadusel täiendada „kuidas testida“ vaatest. Analoogne aspekt toodi välja ka RIKga vestluses, et teatud mittefunktsionaalsete nõuete juures oleks hea, kui nõudega

seotud selgitusi täpsemaks kirjeldada. Lisaks regulaarsetel ülevaatustel pidada silmas ka organisatsioonis või personalis toimunud muudatusi ja hoida nõuete eest vastutajad ajakohasena.

Seonduvalt probleemiga, et alati ei jõuta kõiki mittefunktsionaalsete nõuete kontrolle läbi viia enne projekti tootesse paigaldamist, siis töö autori hinnangul ei peaks IT projektijuhi poolne sage ja pidev meeldetuletamine olema ainus meetod, mis aitab enne tootesse paigaldamist saada kontrollimise tehtud. Töö autori kogemuse põhjal - mida hiljem kõrvalekalded nõuete osas välja tulevad, seda aeglasemalt parandused tellijani jõuavad. Projektiga seotud meeskonna ühine eesmärk peaks olema jõuda mittefunktsionaalsete nõuete kontrollid läbi viia enne toote keskkonda paigaldamist.

Kasutatavuse testide osas on peatükis 4 töö autor toonud välja, et vajalik on luua üldised nõuded kasutavuse testimise kontekstis. Kasutatavus on oluline nii väliskliendile kui ka sisemisele ning töö autori hinnangul selleks, et kasutatavusele hinnagut anda tuleb kaasata hindamisse reaalseid lõppkasutajaid (pigem ei piisa ainult projekti kaasatud kasutatavuse eksperdist).

Oktoobris 2016 Riigikontrolli poolt läbiviidud uuringus „Avalike e-teenuste kasutatavus“ (kuhu oli kaasatud ka Rahandusministeeriumi valitsemisala teenuseid) on välja toodud, et riik peab senisest enam pöörama tähelepanu avalike e-teenuste kasutatavusele, sest riigi pakutavad e-teenused ei ole ühtlaselt lihtsad, kasutajasõbralikud ega lisandväärtust pakkuvad. Selleks et kõik riigi e-teenused oleksid hea kasutatavusega, tuleks seada avalikele e-teenustele kohustuslikud kvaliteedinõuded ning e-teenuseid pakkuvad asutused peaks e-teenuste kasutatavust hindama ja parandama regulaarselt. [27]

Kasutatavuse kontekstis on oluline ka WCAG 2.0 nõuete aspekt – sageli jäävad need nõuded piisava tähelepanuta. Kuigi RMIT mittefunktsionaalsetes nõuetes on öeldud, et rakenduse kasutajaliides peab vastama vähemalt WCAG 2.0 tasemele AA (keskmine vastavustase, rakendatud kõik A taseme edukriteeriumid ning 13 AA taseme edukriteeriumit [35]), siis nende nõuete testimine on projektides ebaselge ja halvasti kaetud. Vajalik määrata nende nõuete testimise prioriteetsus ja testimise metoodika.

## 5.3 Koormustestimine

Tänases koormustestimise protsessis ei ole ära fikseeritud, et millised tegevused (ja ka õigused) koormustestide toimimise kontrollimiseks ja tulemuste vaatamiseks on rakenduse administraatorile vajalikud. Kuna koormustestimine on arendusprojektides valdavalt kohustuslik, siis ei ole otstarbeks, et iga rakenduse administraator, kes varem koormusteste asutuses läbi pole viinud või kes teeb seda harva, peab iga kord uuesti uurima vajalike õiguste, võrguligipääsude ja seadistamise tegevuste kohta. Peatükis 4 on töö autor toonud välja, et RMIT rakenduse administraator seadistab koormustestid vastavalt sisemistele juhistele (sh selleks, et viia läbi testide toimivuse kontroll). Koormustestimisega seonduva sisemise juhise väljatöötamisega on vajalik organisatsiooni siseselt tegeleda. Lisaks on RMIT sisemiselt vajalik ära fikseerida, milliste nõuetega peab arendaja arvestama kui koostab koormustestide läbiviimise juhist / kirjeldust. Peatükis 4 on töö autori poolt välja toodud mõned näited, mida võiks arendaja poolt üle antav kirjeldus sisaldada, aga kindlasti vajaks see sisemiselt veel täiendavat läbimõtlemist ja hindamist.

Koormustestide eelnev kooskõlastamine tellijaga on vähene. RMIT mittefunktsionaalsetes nõuetes on toodud, et jõudlustestide täpne kirjeldus ja kasutatavad töövahendid tuleb kokku leppida detailanalüüsi käigus ja siin kohal oleks töö autori hinnangul vajalik suurem initsiatiiv IT töörühma juhi ja ka rakenduse administraatori poolt, et see samm ei jääks tegemata. Selleks, et testimine õnnestuks on vajalik, et kõik testimisega seotud osapooled, saaksid ühte moodi aru testimise eesmärgist ja vajalikest tegevustest - seeläbi osatakse ka tulemustest täpsemaid järeldusi teha.

TEHIK puhul koormustestimised tehti alguses lõpuni sisemiselt nende oma testijate poolt. Kuna RMIT kontekstis viivad koormustestimisi läbi rakenduse administraatorid, kelle tööülesandeks on muu hulgas ka erinevate tootekeskkonnas olevate rakenduste probleemide lahendamine, siis rakenduse administraatori ressursi kokkuhoiu eesmärgil ei pruugi TEHIKu lahendus RMIT jaoks olla sobiv. Samas kuna arendusprojektides on lisaks koormustestimistele veel ka teisi erinevaid tehnilisi testimisi (näiteks masin-masin liideste testimine, andmete migreerimise testimine), siis tasuks hinnata tehnilise testija rolli loomist või mõelda olemasolevate ressursside kontekstis, et kes võiks RMITis seda funktsiooni täita või sellele valdkonnale spetsialiseeruda. Töö autori hinnangul on organisatsioonis vajadus analüüsida ka testimiste automatiseerimise võimalusi.

KEMIT puhul viis koormustestid läbi arenduspartner (KEMIT poolt hallatavas keskkonnas). Töö autori arvamusel infoturbe aspektide tõttu RMIT jaoks see ei ole potentsiaalselt sobilik. Praegu RMITis kasutatava töökorralduse puhul jäävad alles koormustestimise läbiviimiseks vajalikud juhised, mis on abistavaks juhul kui peaks tekkima majasiseselt vajadus ise korrata tulevikus eelnevalt läbiviidud koormustestimisi.

SMITis hinnati ressursside vajadusi projektide põhiselt ning vastavalt sellele hangiti pakkumuste põhjal erinevate rollide täitjaid välistelt arenduspartneritelt. Kuna RMIT klientide arenduste mahud arvuliselt ja rahaliselt on viimase paari aasta jooksul oluliselt kasvanud, siis on kasvanud ka vajadus testimise ressurssi järele. Nii RMIT kui ka äri poolt teostatavad testimise tegevused viivad ametnikud läbi oma põhi töökohustuste kõrvalt. Üha enam kasvavate testimiste mahtude katmiseks ei pruugi olemasolevast ressursist piisata. Testimise tööjõu täiendava vajaduse üheks lahendamise võimaluseks on hankida täiendavat testimise ressurssi arenduspartneritelt. Sellise ressursi hankimise vajadus peaks olema hinnatud juba projekti alguses, kuna hankimise tegevus võib võtta aega. Testijaid on soovituslik kaasata arendusprotsessi võimalikult varakult. Testija jaoks on oluline, et ta saaks olla kursis nii süsteemi ja selle lahenduse detailidega kui ka äri poole nõuetega. Töö autori hinnangul on ühest küljest hea, kui on võimalik täiendavat majavälist testimise ressursi vajadusel kaasata, kuid teisest küljest on testimine tegevus, mille käigus ka õpitakse tundma süsteemi ja selle käitumist ning eripärasid. Seega ressursi hankimisel väljast tuleb arvestada, et meile jääb seeläbi ka omale teatud kompetentsi sisemiselt vähem. Ressursi hankimine väljast tähendab täiendavat kulu, aga kui sisemiselt ei jõuta testimist ära katta, siis mitte testimine võib lõppkokkuvõttes olla veel kallim.

## 5.4 Testimise kasu äri aspektist

IT süsteemide suutlikkus ja kvaliteet omavad efekti äri tulemustele, kuna mõjutavad nii lõppkasutaja kogemust kui ka äritegevust. Lõppkasutajate ootused e-teenuste ja lahenduste kättesaadavusele on kõrged ning seetõttu on vajalik fokuseerida kliendile väärtuse loomisele ja kasutaja kogemusele. Halvasti toimiv IT lahendus mõjub kahjustavalt ettevõtte mainele näiteks seeläbi, et lõppkasutajad levitavad oma rahulolematust läbi sotsiaalmeedia. [28, lk. 8]

Aastatel 2015 ja 2017 läbiviidud uurimustes [28], [29], kuhu oli kaasatud üle 1500 ettevõtte juhi (sh ka testjuhid) 32 erinevast riigist on välja toodud, et testimisele ja

kvaliteedile fokuseerimise eesmärk on ettevõtte maine kaitsmine ning tarkvara turvalisuse tõhustamine. Vajadust panna testimisele ja kvaliteedile rõhku, et kaitsta ettevõtte mainet, hinnati skaalal ühest seitsmeni (7 tähendas „väga oluline“) ja hindamise tulemuseks saadi 6,1. [28, lk. 7] 2017 aastal läbiviidud uurimuses 65% vastanutest hindasid, et testimine ja kvaliteedile panustamise eesmärgiks on IT lahenduste turvalisuse tõstmine. [29, lk. 7]

Töö autori hinnangul on testimise puhul oluline hinnata riske, keerukas on määratleda - mis maksab, kui üldse ei testita. Suure tõenäosusega ebakvaliteetne IT lahendus mõjub kahjulikult nii äri kui IT organisatsiooni mainele. Lisaks võivad kahju kannatada ka lõppkasutajad (nii juriidilised kui füüsilised isikud). Kui projekti käigus on läbiviidud kvaliteedi riskide hindamine ja kõige kallimad või oluliselt kahjuliku mõjuga riskid on läbi testimiste maandatud, siis seeläbi on võimalik suurendada kindlust, et arendatav toode on piisavalt töökindel. Testimine aitab minimeerida riskide realiseerumist.

Testimise kasuna võib välja tuua:

- varajane vigade avastamine aitab vältida / kokku hoida kulusid;
- testimine tagab parema toote kvaliteedi;
- madalam kogu omamiskulu (*Total Cost of Ownership*);
- vähesem halduskoormus. [30, lk. 24]

Lisaks on testimine oluline, kuna tootekeskkonnas leiduvate vigade parandamine on mitmeid korda kallim. Seda väidet illustreerib järgnev tabel:

Tabel 3 Vigade parandamise keskmine maksumus [32, ptk. 3]

| | | **Vea tuvastamise faas** | | | | |
|---|---|---|---|---|---|---|
| | | Nõuete analüüs | Arhitektuuri määratlemine | Arendamine | Testimine | Tootes kasutamine |
| **Vea tegemise faas** | Nõuete analüüs | 1 | 3 | 5 - 10 | 10 | 10 - 100 |
| | Arhitektuuri määratlemine | - | 1 | 10 | 15 | 25 - 100 |
| | Arendamine | - | - | 1 | 10 | 10 - 25 |

Vigade avastamine varakult on oluline, kuna mida kauem viga on tuvastamata seda suurema tõenäosusega see kandub edasistesse etappidesse ja võib seeläbi tekitada

täiendavat kahju. Näiteks nõuete määratlemisel tehtud viga, mille parandamise hind nõuete valideerimise faasis on 100 eurot võib maksta süsteemi testimise etapis 1000 eurot. [32, ptk. 3] Täiendav kahju tekib sellest, et ajas süsteemi keerukus kasvab, samuti kasvab ka seoste ja sõltuvuste hulk. Vea parandamine hilisemas etapis suure tõenäosusega tähendab sõltuvustest tulenevalt paranduse tegemist mitmetes kohtades ja seeläbi on parandusest mõjutatud rohkem funktsionaalsust. See jälle omakorda tähendab ka suuremat ajakulu paranduse sisseviimisele ja testimisele.

Kui viia testimise tegevused läbi paralleelselt projekti arendustega, siis vead mis tuvastatakse saavad kergemini ja kiiremini parandatud. Selle põhjuseks on fakt, et arendajal, kes aktiivselt on tegelemas just konkreetselt aktuaalse projektiga on lihtsam orienteeruda teemas, ta ei pea hakkama meenutama ja otsima infot. Seeläbi samuti väheneb vea parandamisele kuluv aeg, mis omakorda tähendab, et viga saab parandatud väiksema rahalise kuluga. [33]

## 5.5 Järeldused ja soovitused

Kuna RMITis arendusprojektides toimuvate testimiste tegevusi kajastavad erinevad korrad killustatult või ainult osaliselt, siis tasuks kaaluda testimise strateegia väljatöötamist - see koondaks testimise põhimõtted ja korralduse organisatsioonis. Leida sobiv kvaliteedi riskide hindamise metoodika ning panustada tellija poolt ise rohkem kaasa mõtlema testimiste tegevuste läbiviimisel. Üheks võimaluseks testimistega seonduvate protsesside juhtimiseks ja testimise teemade haldamiseks oleks näiteks testjuhi ametikoha loomine, kuid seal juures tuleks hoolikalt läbi mõelda selle rolli vastused ja kohustused.

Kokkuvõtvalt töö autor soovitab:

- korrata testimisega seotud tegevuste ja protsesside hindamist ning seonduvate probleemide kaardistamist teatud regulaarsusega;
- korrastada ja luua puuduvad testimise tegevustega seotud juhised. Koondada testimise põhimõtted ja korraldus - määratleda testimise strateegia;
- määrata ning hinnata projektides kvaliteedi riske ning panna rohkem rõhku testimise planeerimisele;
- kaaluda tehnilise testija rolli loomist;

- rohkem pöörata rõhku testimiseks vajaliku ressurssi ja ka seonduvate kompetentside hindamisele;
- testimisega alustada nii vara kui võimalik;
- testimise lõppedes viia läbi hindamine ja õppetundide analüüs;
- rõhutada testimise olulisust IT arendusprojektides ning motiveerida testijaid;
- arvestada, et testimise jaoks vajalike tegevuste selgus ja kõigile osapooltele arusaadav vastutuse jaotus on testimise kontekstis oluline.

Testimistega seotud protsesse ja ka vajalikke ressursse tuleks asutuses üle vaadata regulaarselt. Protsessi parendamiseks on vajalik kaasata erinevaid spetsialiste, et mitmete erinevate rollide vaatest tuvastada probleemkohti ning fokuseerida nende parendamisele / puudujääkide korrigeerimisele. Seeläbi saame arendada ja soodustada testimisega seotud tööprotsesside sujuvamat laabumist IT arendusprojektides.

# 6 Kokkuvõte

Käesolev magistritöö on kirjutatud teemal „Testimise protsessi väljatöötamine Rahandusministeeriumi Infotehnoloogiakeskuse näitel“, töö ajendiks oli vajadus uurida tänast olukorda IT arendusprojektides toimuvate testimistega - täna kajastavad erinevad korrad ainult osaliselt tarkvara testimise organiseerimist ja läbiviimist ning mitmete testimiste kontekstis protsesside kirjeldused puuduvad. Magistritöö uurimisobjektiks oli valitud tarkvara testimise protsess ja selle korraldus RMITis.

Töö eesmärgiks oli kaardistada tänane olukord Rahandusministeeriumi Infotehnoloogiakeskuses IT arendusprojektides toimuvate testimistega, tuua välja testimise protsessiga seonduvad kitsaskohad ning määratleda optimaalne testimise korralduse protsess. Seatud eesmärk on töö autori poolt täidetud, peamisi tulemeid kirjeldavad peatükid 3 „Tänane olukord arendusprojektides toimuvate testimistega “ ja 4 „Parendatud testimise protsess“.

Töö tulemusena on:

- viidud asutuse siseselt läbi intervjuud seoses tarkvaraarendusprojektides toimuvate testimistega;
- kaardistatud tänased arendusprojektides toimuvate testimiste protsessid;
- toodud välja tänased testimise protsessidega seotud probleemid (testimise liikide kaupa);
- viidud läbi intervjuud neljas sarnases IT asutuses ja lisatud ülevaade nendes asutustes kasutatavate IT arenduste testimiste kohta;
- võttes arvesse kaardistatud probleeme on loodud parendatud testimise protsessid;
- võttes arvesse teiste asutuste praktikaid ja valdkonna kirjandust on analüüsitud töö tulemeid ning toodud välja testimise kasu äri aspektist.

Testimise protsessi hindamiseks kasutati intervjuusid ja kriitilise testimise protsessi mudelit. Parendatud protsesside loomisel võeti arvesse kaardistatud kitsaskohti ja valdkonna praktikaid.

Hindamise tulemusena ja tänase olukorra kaardistamisel tuvastas töö autor probleemkohti testimise planeerimise, ettevalmistamise ja testimise läbiviimise protsessides ning tõi

välja omapoolsed parendusettepanekud. Tulemuste analüüsi käigus on võetud arvesse teiste asutuste soovitusi kui ka valdkonnaga seonduvat kirjandust.

Töö tulemusena autor järeldab, et testimine on oluline, et tagada kvaliteetsem, turvalisem ja ka tellijale sobivam lahendus. Testimine on kulukas, kuid mitte testimine võib olla veel kulukam (mitte testimise tagajärjel võib näiteks tekkida mainekahju, turvaintsidendid tootekeskkonnas ja suureneda tarkvara tootega seotud halduskoormus). Autor näeb vajadust organisatsioonis koondada tarkvara testimise põhimõtted ja korraldus näiteks testimise strateegia näol. Oluline on korrata regulaarselt testimisega seotud tegevuste ja protsesside hindamist ning parendamist ja seeläbi panustada kvaliteetsema tulemi saamisele IT arendusprojektides.

# Kasutatud kirjandus


[1] Rahandusministeeriumi Infotehnoloogiakeskuse põhimäärus. [WWW] https://www.rmit.ee/sites/default/files/rmit-pohimaarus.pdf (22.01.2018)

[2] Rahandusministeeriumi Infotehnoloogiakeskuse veebilehekülg. [WWW] https://www.rmit.ee/ (22.01.2018)

[3] Rahandusministeeriumi infotehnoloogiakeskuse arengukava 2018-2022. [WWW] https://www.rmit.ee/sites/default/files/rmit-arengukava-2018-2022-kinnitatud.pdf (22.01.2018)

[4] Nayyar Iqbal, M. Rizwan Jameel Qureshi.
Improvement of Key Problems of Software Testing in Quality Assurance. 2009 [WWW] https://arxiv.org/ftp/arxiv/papers/1202/1202.2506.pdf (26.01.2018)

[5] Tarkvara ja süsteemi testimine. [WWW] http://opiobjektid.tptlive.ee/B1/b14_tarkvara_ja_ssteemi_testimine.html (26.01.2018)

[6] Tepandi, J. Tarkvara protsessid, kvaliteet ja standardid. 2017 [WWW] http://tepandi.ee/tks-loeng.pdf (26.01.2018)

[7] Süsteemi- ja tarkvaratehnika. Süsteemide ja tarkvara kvaliteedinõuded ja kvaliteedi hindamine. Süsteemide ja tarkvara kvaliteedimudelid standard : EVS-ISO/IEC 25010:2011. Eesti Standardikeskus, 2012 (01.02.2018)

[8] Tšukrejeva, J. Tarkvarasüsteemi kvaliteet ja testimine. 2015 [WWW] http://maurus.ttu.ee/sts/wp-content/uploads/2015/10/IDK0071-Loeng-7-Kvaliteet-ja-testimine.pdf (02.02.2018)

[9] Turvatestimine. [WWW] http://akit.cyber.ee/term/1161-turvatestimine (02.02.2018)

[10] International Software Testing Qualifications Board. Certified Tester Foundation Level Syllabus. 2011 [WWW]

https://www.istqb.org/downloads/send/2-foundation-level-documents/3-foundation-level-syllabus-2011.html4 (02.02.2018)

[11] Bourque, P., Fairley, R. Guide to the Software Engineering Body of Knowledge (SWEBOK 3.0) - *IEEE Computer Society*. 2014 [WWW] http://www.computer.org/portal/web/swebok/v3guide) (02.02.2018)

[12] International Software Testing Qualifications Board. Certified Tester Expert Level Syllabus (Improving the testing Process). 2011 [WWW] https://www.istqb.org/downloads/send/12-expert-level-documents/75-expert-level-syllabus-improving-the-testing-process-2011.html (02.02.2018)

[13] International Software Testing Qualifications Board. Certified Tester Advanced Level Syllabus (Test Manager). 2012 [WWW] https://www.istqb.org/downloads/send/10-advanced-level-syllabus-2012/54-advanced-level-syllabus-2012-test-manager.html (02.02.2018)

[14] Critical Testing Processes: An Open Source, Business Driven Framework for Improving the Testing Proces [WWW] https://rbcs-us.com/site/assets/files/1170/critical_testing_processes.pdf (05.03.2018)

[15] Bath, G., Veenendaal, E. Improving the Test Process: Implementing Improvement and Change - A Study Guide for the ISTQB Expert Level Module. USA : Gerhard Rossbach, 2014 (05.03.2018)

[16] Afzal, W. Metrics in Software Test Planning and Test Design Processes : magistritöö. Sweden, School of Engineering Blekinge Institute of Technology [WWW] http://www.diva-portal.org/smash/record.jsf?pid=diva2%3A833623&dswid=711 (05.03.2018)

[17] R. D. Craig., S. P. Jaskiel. Systematic Software Testing. London : Artech House Publishers, 2002 (05.03.2018)

[18] Software and systems engineering - Software testing - Part 2: Test processes : ISO/IEC/IEEE 29119-2. *IEEE Computer Society,* 2013 [WWW] https://ieeexplore.ieee.org/document/6588543/ (10.03.2018)

[19] Software and systems engineering - Software testing - Part 3: Test documentation : ISO/IEC/IEEE 29119-3. *IEEE Computer Society,* 2013 [WWW] http://homes.ieu.edu.tr/userhat/courses/SE344_Fall_2015-2016/ISO_IEC_IEEE_29119_Part3.pdf (10.03.2018)

[20] Aoyama, M., Kikushima, Y. Business-Driven Acceptance Testing Methodology and its Practice for e-Government Software Systems, 2013 [WWW] http://ieeexplore.ieee.org/document/6754361/ (10.03.2018)

[21] Black, R. Critical Testing Processes: Plan, Prepare, Perform, Perfect. Boston : Addison-Wesley Professional, 2003. (10.03.2018)

[22] Iivonen, J., Mäntylä, M., Itkonen, J. Characteristics of high performing testers: a case study, 2010 [WWW] https://dl.acm.org/citation.cfm?id=1852862 (16.03.2018)

[23] Loog, K. Tarkvara testimine, 2009 [WWW] http://maurus.ttu.ee/sts/wp-content/uploads/2011/11/06_Tarkvara-testimine_sissejuhatus_TUT.pdf (16.03.2018)

[24] Regressioonitestimine. [WWW] https://et.wikipedia.org/wiki/Regressioonitestimine (16.03.2018)

[25] Shepard, L. Avoiding Overkill in Manual Regression Testing, 2012 [WWW] http://www.uploads.pnsqc.org/2012/papers/t-46_Shepard_paper.pdf (16.03.2018)

[26] Rafi, D., Reddy, K., Mäntylä, M. Benefits and Limitations of Automated Software Testing: Systematic Literature Review and Practitioner Survey, 2012 [WWW] https://ieeexplore.ieee.org/document/6228988/ (17.03.2018)

[27] Avalike e-teenuste kasutamismugavusele tuleb pöörata rohkem tähelepanu. 2016 [WWW]

http://www.riigikontroll.ee/Suhtedavalikkusega/Pressiteated/tabid/168/ItemId/921/View/Text/amid/557/language/et-EE/Default.aspx (17.03.2018)

[28] Capgemini Group. Capgemini's World Quality Report 2015-16 [WWW] https://www.uk.sogeti.com/globalassets/uk/reports/sogetiuk-wqr-2015-16-full-report.pdf (24.03.2018)

[29] Capgemini Group. Capgemini's World Quality Report 2016-17 [WWW] https://www.capgemini.com/wp-content/uploads/2017/08/world_quality_report_2016-17_secure.pdf (24.03.2018)

[30] Hunt, B., Abolfotouh, G., Carpenter, J. Software Test Costs and Return on Investment (ROI) Issues. 2014 [WWW] http://www.iceaaonline.com/ready/wp-content/uploads/2014/03/Software-Test-Cost-and-ROI-Galorath-Feb-14-Hunt.pdf (24.03.2018)

[31] Philipp, I. Simplifying Risk-Based Testing. 2016 [WWW] https://www.tricentis.com/wp-content/uploads/2016/07/Simplifying-Risk-Based-Testing_FINAL.pdf (24.03.2018)

[32] McConnell, S. Code Complete, Second Edition. Washington : Microsoft Press, 2004 (24.03.2018)

[33] BBC Academy. Why do you need to test software? [WWW] http://www.bbc.co.uk/academy/technology/article/art20150223103348419 (24.03.2018)

[34] About ISTQB [WWW] https://www.istqb.org/about-as.html (31.03.2018)

[35] WCAG 2.0 rakendusjuhised [WWW] https://www.mkm.ee/et/wcag-20-rakendusjuhised (31.03.2018)

[36] Arendusprojektide haldussüsteemi Atlassian JIRA põhikursus [WWW] https://www.tark.ee/arendusprojektide-haldussusteemi-atlassian-jira-pohikursus (20.04.2018)

[37] What is Confluence? [WWW] https://confluence.atlassian.com/confeval/confluence-evaluator-resources/confluence-what-is-confluence (20.04.2018)

[38] Kogu omamiskulu [WWW] https://et.wikipedia.org/wiki/Kogu_omamiskulu (20.04.2018)

[39] ISTQB Glossary all terms 3.1 [WWW] https://www.istqb.org/downloads/send/20-istqb-glossary/186-glossary-all-terms.html (20.04.2018)

# Lisa 1 – Tootekvaliteedi mudel EVS-ISO/IEC 25010:2011 järgi [7]

| (Alam)karakteristik |
|---|
| **Funktsionaalne sobivus** |
| funktsionaalne täielikkus |
| funktsionaalne õigsus |
| funktsionaalne kohasus |
| **Soorituse tõhusus** |
| ajaline käitumine |
| ressursikasutus |
| suutvus |
| **Ühilduvus** |
| koosoluvõime |
| koostalitlusvõime |
| **Kasutatavus** |
| kohasuse mõistetavus |
| õpitavus |
| käsitsetavus |
| eksitusekindlus |
| kasutajaliidese esteetika |
| hõlpsus |

| Töökindlus |
|---|
| küpsus |
| käideldavus |
| tõrketaluvus |
| taastuvus |
| **Turvalisus** |
| konfidentsiaalsus |
| terviklus |
| salgamatus |
| jälitatavus |
| autentsus |
| **Hooldatavus** |
| modulaarsus |
| taaskasutatavus |
| analüüsitavus |
| modifitseeritavus |
| testitavus |
| **Porditavus** |
| sobitatavus |
| installeeritavus |
| asendatavus |

# Lisa 2 – Tänase olukorra kaardistamisel kasutatud küsimustik

**Funktsionaalsete nõuete testimine:**

1. Milline näeb täna välja funktsionaalsete nõuete testimise protsess?
2. Millised on testplaanid (on need formuleeritud) nii sisemiste arenduste kui tellitud arenduste kontekstis?
3. Kuidas sisemiste arenduste puhul tekib testimiseks vajalik sisend (kuidas ja kelle poolt koostatakse)?
4. Mille alusel toimub testimine?
5. Kas testimise aluseks olev dokumentatsioon / juhised on piisavad ja arusaadavad testimise läbiviimiseks?
6. Kas testimise eesmärk on arusaadav?
7. Kas testimisel piirdutakse / lähtutakse ainult testimise sisendiks antud dokumentatsioonist?
8. Testimise protsessi jälgimine (kas testimise seisust on võimalik saada ülevaadet)?
9. Kuidas / mille abil hindade süsteemi kaetust testidega?
10. Kes on testijad?
11. Kas vastutuse jaotus on selge ja arusaadav?
12. Kas funktsionaalsete nõuete testimisel kasutatakse mingeid vahendeid?
13. Mobiili vaadete testimine kuidas toimub? Kas on teada mida selleks tegema peab?
14. Raporteeritud vigade prioritiseerimine (kelle poolt ja kuidas toimub)?
15. Kas tarkvara arendusprojekti käigus täitja esitab tellijale raporti täitja poolt läbiviidud testimiste kohta?
16. Regressioonitestid – kas (parandus)tarnete puhul testitakse üle ka varasemalt tarnitud funktsionaalsust?
17. Kas erinevatest projektides on sarnane lähenemine funktsionaalsete nõuete testimisele või mitte?
18. Kuidas on korraldatud testimine, mis hõlmab kas välist kasutajat? Kelle vastutada on juhiste väljatöötamine välisele kasutajale (ettevõtted)?
19. Kui palju teate milliseid teste täpselt teeb tarkvara arendaja arendusprojektis?
20. Kas näete tänase funktsionaalse testimise protsessiga probleeme? Kui jah, siis milliseid?

**Mittefunktsionaalsete nõuete testimine:**

1. Milline näeb täna välja mittefunktsionaalsete nõuete testimise protsess?
2. Kas kõikide nõuete puhul, mis on kontrollimiseks kirjeldatud on testimise meetod arusaadav ja selge?
3. Kas pärast seda kui projekt on lõppenud tehakse tegemata kontrolle ka tagantjärgi?
4. Kas arendusprojekti käigus esitab täitja tellijale mittefunktsionaalsete nõuete vastavustabeli?
5. Kas kasutatavuse testimine on piisav?

6. Kas näete tänase mittefunktsionaalsete nõuete testimise protsessiga probleeme? Kui jah, siis milliseid? Kas mittefunktsionaalsete nõuete testimisel on midagi, mida peaks tegema teistmoodi või paremini. Kas on midagi mida näiteks ei tehta, aga võiks või peaks tegema veel lisaks?

**Koormustestimine:**

1. Milline näeb täna välja koormustestimise protsess?
2. Kelle käest tulevad koormuse nõuded?
3. Kas koormustestide juhised sisaldavad ka andmete tekitamise juhiseid (juhul kui vajalikke andmeid ei ole)?
4. Kas arendajate poolsed koormustestide juhised on arusaadavad ja selged?
5. Lisaks arendaja poolsele juhisele on ka mingid sisemised tegevused, mis on vajalikud testimise läbiviimiseks, mis on need täpsed tegevused?
6. Kas need tegevused on kusagil ka kirjalikult kirjas?
7. Kas rakenduse testide jooksutamise ajal kasutatakse rakendust?
8. Kas on ülevaade sellest, millised teste samal ajal veel tehakse?
9. Kas koormustestide tulemustest järelduste tegemine on alati selge ja arusaadav / Kas testimise eesmärk on arusaadav?
10. Kas tulemused fikseeritakse kusagile? Kui jah, siis millisel kujul kas on ühtne lähenemine või mitte?
11. Kas koormustestide planeerimise / läbiviimise juures saaks teha midagi paremini? Kas on midagi mida näiteks ei tehta, aga võiks või peaks tegema veel lisaks? Kas näete tänase koormustestimise protsessiga probleeme? Kui jah, siis milliseid?

**Turvatestimine:**

1. Milline näeb täna välja turvatestimise protsess?
2. Kes hindab turvatestimise vajadusi? Millal kindlasti tellime väljast ja millal teeme sisemiselt?
3. Andmete tekitamine turvatestide poolt, kas teate milliseid probleeme tekitab? Kas on ülevaade, milliseid andmeid testimine mõjutab?
4. Mis hetkel projekti käigus tuleb tellida turvatestid?
5. Kas on ülevaadet, kui palju turvateste on vajalik planeerida? Kui palju saate ette teada testimise vajadusi?
6. Kas projektidest turvatestimisse antav sisend on piisav?
7. Kuidas kordusteste tehakse – millal teeme ise ja millal tellime?
8. Kas näete tänase turvatestimise protsessiga probleeme? Kui jah, siis milliseid?

**Koodiaudit:**

1. Kuidas tehakse koodiauditit? Milliseid tegevusi see sisaldab?
2. Mis info on oluline koodiauditi tellimisel?
3. Mis tingimustele peab vastama kood?
4. Kas *Junit* testid on meie jaoks olulised, kas neid tehakse? Kas me jooksutame neid kui väline arendaja on teinud?
5. Kas on midagi seoses koodiauditiga välja tuua, näiteks probleemina üles tõstatada?
6. Kas on midagi mida sooviks uurida selles kontekstis teistelt sarnastelt asutustelt?

# Lisa 3 – Teistes asutuses kasutatud küsimustik

1. Milliseid testimisi teie asutuses tehakse arendusprojektides? Milline on üldine testimise protsess vastavalt testimise liigile?
2. Mitu % testimisest hangitakse väljast ning kui palju testitakse sisemiselt?
3. Kes vastutab testimise korraldamise eest arendusprojektides?
4. Kes viivad läbi testimisi (kes on testijad)? Millised rollid on testimisega seotud? Kas testijate gruppi on kaasatud reaalsed süsteemi kasutajad?
5. Mida näete omalt poolt suuremate probleemidena seoses arendusprojektides toimuvate testimistega?
6. Mis aspektid seoses testimisega arvate, et on teie asutuses hästi ja toimivalt korraldatud?

# Lisa 4 – Millega on vaja arvestada mobiilsete seadmete tellimisel

Mobiilsete seadmetega testimise vajadusel on vajalik arvestada järgnevaga:

- Arendusprojektides on mobiilivaadete testimisel võimalik kasutada mobiilseid seadmeid – testimiseks on võimalik RMIT help kaudu tellida *Android* ja *iOS* platvormidel põhinevaid tahvleid ja mobiiltelefone, mida hangitakse juurde vastavalt vajadustele ja võimalustele;
- Seadmete tellimiseks on vajalik teha taotlus kas RMIT help e-maili aadressile või kasutada taotluse tegemiseks RMIT kasutajatoe portaali;
- Help taotluses peab olema lisatud järgmine info:
    - Mis platvormidel põhinevaid seadmeid soovitakse (ja kas soovitakse ainult telefone, ainult tahvleid või mõlemaid);
    - Välja tuua, et on vajadus kasutada *Intune* litsentsi (1 litsents annab kasutajale kuni viie mobiilse seadme kasutamise õiguse);
    - Viide keskkonnale või keskkondadele, kus soovitakse testimisi teha;
    - Mis projekti raames seadmeid vaja on.
- *Intune* annab võimaluse saata serverilt mobiilsele seadmele vajalikud seadistused ja sertifikaadid, lisaks on *Intune* vajalik ka selleks, et VPN ühendus mobiilses seadmes töötaks;

- Tuleb arvestada, et seadmete ettevalmistamine võtab aega. Kui on teada, et projektis on vajalik mobiilsete seadmetega testimine, siis seadmed tuleks tellida aegsasti (ettevalmistused võivad võtta kuni üks nädal). Uue seadme (või seadmete komplekti) soetamise vajadusel läheb aega rohkem. Ajaliselt läheb kauem ka juhul, kui testida on vaja uusi rakendusi, mida VPN profiilides ei ole (sellisel juhul kooskõlastused võtavad aega)
- Kui RMIT poolt on vajalikud eelseadistused tehtud, siis tuuakse seadmed tellijale kohale ja abistatakse tööle panemisel. Seadme ekraaniluku kood öeldakse edasi seadistaja poolt (tavaliselt on seadistajaks RMIT töökohateenuste osakonna peaspetsialist);
- Selleks, et seadmega testkeskkonda minna on vaja mobiilses seadmes tööle panna *Cisco AnyConnect* VPN, see tegevus näidatakse kasutajale ette kui seadmed kohale tuuakse. VPN ühendus võib küsida domeeni kasutajanime ja parooli. Kui VPN ühendus on loodud, siis testkeskkonda pöördumiseks avada mobiilses seadmes veebilehitseja ning sisestada sinna testkeskkonna URL;
- Kuna testimiseks kasutatavad keskkonnad üldjuhul nõuavad mobiil ID-ga autentimist, siis tuleb arvestada, et kasutajal peab testimiseks olema mobiil ID kasutamise võimalus;
- Kui seadmeid testimiseks enam vaja ei lähe, siis tuleb sellest RMIT helpi teavitada.