TARTU ÜLIKOOL
Arvutiteaduse instituut
Informaatika õppekava

Joosep Heinmets

Riigiasutuse pärandsüsteemi täiendamise juhtumianalüüs

Bakalaureusetöö (9 EAP)

Juhendaja: Vambola Leping

Tartu 2020

Riigiasutuse pärandsüsteemi täiendamise juhtumianalüüs

Lühikokkuvõte:

Käesolev bakalureusetöö annab ülevaate konkreetse näite varal Eesti Vabriigi riigiasutuse aktiivses kasutuses oleva pärandsüsteemi täiendamise protsessist. Töö kirjeldab probleemi äri- ja tehnilist analüüsi, lahenduse tarkvaraarendust, muudatuste testimist ja toodangukeskkonnas kasutusele võtmist.

Võtmesõnad:

Infosüsteem, analüüs, tarkvaraarendus, avalik sektor

CERCS:

P175 Informaatika, süsteemiteooria

Case study: Updating a Legacy System of a State Institution

Abstract:

The aim of this Bachelor's thesis is to give an overview of the process of updating a legacy system of an Estonian state institution through a specific example. The thesis describes the business and technical analysis of the problem, the software development of the solution, testing of the changed system and taking the updated system into use in a live environment.

Keywords:

Information systems, analysis, software development, public sector

CERCS:

P175 Informatics, systems theory

Sisukord

Sissejuhatus

2019. aasta veebruaris otsustas Eesti Vabariigi Riigikogu suurendada riigi poolt pakutava toetusteenuse toetuse määra ehk anda selle toetuse saajatele varasemast rohkem raha. Vastava seadusandluse järgi pidi uus määr kehtima alates 2020. aasta 1. jaanuarist. Toetusteenuse pakkumisega tegeleb riigiasutus ehk selle töö kontekstis Klient.

Klient leidis, et seadusemuudatuse elluviimiseks oli tarvis täiendada infosüsteemi, mille kaudu määratakse ja hallatakse vastavaid toetuseid. Kliendil endal ei olnud majasisest infotehnoloogiliste arenduste võimekust, vaid selle võimekuse hankimisega tegeles teine riigiasutus ehk selle töö kontekstis Vahendaja. Vahendajal oli kõnealuse infosüsteemi arendustöödeks sõlmitud raam- ja hankeleping erasektorist pärit teenusepakkujaga ehk selle töö kontekstis Teostajaga. Arendustele lisas keerukust nüanss, et arendusvajadus oli seotud mitu aastat kasutuses olnud pärandsüsteemiga, mille esialgne looja ei ole Teostaja.

Käesolev kirjatöö kirjeldab Kliendi, Vahendaja ja Teostaja tegevusi eesmärgi ehk toetuste summade suurendamise saavutamiseks. Protsess koosneb kolmest põhietapist. Esiteks on tarvis koostöös kõigi osapooltega analüüsida, mida on infosüsteemis vaja muuta, et eesmärk saavutada. Teiseks on tarvis Teostajal muudatused pärandsüsteemi tarkvaras rakendada. Kolmandaks on tarvis Kliendil muudatused vastu võtta. Selleks tuleb Kliendil valideerida tehtud muudatuste vastavus enda vajadustega, Teostajal parandada Kliendi poolt leitud puudujäägid, Vahendajal muudatused toodangukeskkonda paigaldada ning Kliendil täiendatud süsteem kasutusele võtta.

Käesoleva lõputöö eesmärgiks on läbi kogu protsessi kirjelduse tuua esile kasulikke praktikaid, võimalikke kitsaskohti ja parendusvõimalusi, mis on seotud riigiasutuse pärandsüsteemi arendusega.

Teema valik ja käsitlus tulenes autori harivast kogemusest olla vastutavalt seotud tarkvaraarenduse protsessi kõikide kaasatud etappidega. Käesolev lõputöö omab autori arvates hariduslikku väärtust lugejatele, kuna päriselulist tööprotsessi tarkvaraarenduses. Protsessikirjeldus on ka aktuaalne Eesti e-riigi süsteemide arendamise arutelus, kuna peegeldab erinevusi seadusandluse ja IT-süsteemide muutmises.

Mõisted

Pärandsüsteem (*legacy system*)	Iga vanem süsteem, mis endiselt täidab oma põhifunktsioone ja milles säilitatakse ja mille kaudu tehakse kättesaadavaks peamisi andmeid, aga mille puhul kasutatakse vanemat tehnoloogiat, mille sidumine uuemate süsteemidega on raskendatud ja mille puhul ümberehitust peetakse keerukaks või kalliks. Rangelt võttes võib pärandsüsteemiks pidada iga IT-süsteemi hoolimata selle valmimisaastast, sealhulgas hiljuti loodud süsteeme, mille loomisel ei ole arvestatud uuskasutusvõimalusi ega ühilduvust teiste süsteemidega [1].
modaalaken (dialoog)	Kuva, mis kuvatakse põhikuvast või teisest modaalaknast esile tõstetuna. Aktiivsest modaalaknast tahaplaanile jäävad kuvad muutuvad mitteaktiivseteks ja kuvatakse üldjuhul varjatult või tuhmistununa [2].
feature flag	Tarkvara kasutuselevõtu metoodika, mis eraldab koodikomponendi tarkvaras lisandumise ja selle kasutusele võtmise. Seeläbi toimub uute komponentide aktiveerimine süsteemis läbi tarkvara seadistuse, mitte uue versiooni paigalduse [3].
pipeline (GitLabi kontekstis)	Töökorraldusprotsess, mis koosneb töödest (näiteks kompileerimine, ühiktestide jooksutamine) ja järkudest (näiteks kõigepealt kompileeri, siis jooksuta testid). Ühe järgu töid saab jooksutada asünkroonselt, vähendades sellega kogu protsessi ajakulu [4].

1. Analüüs

Analüüsifaasi eesmärgiks oli ärilise vajaduse põhjal välja selgitada, milliseid tehnilisi tegevusi tuleks teostada, et süsteem vastaks Kliendi ootustele ja vajadustele. Analüüsifaas algas Teostaja jaoks, kui Klient edastas Teostajale ärivajaduse sisendi. Ärivajaduse sisendi põhjal koostas Teostaja Analüütik ärianalüüsi. Ärianalüüsi tulem kooskõlastati Kliendiga. Ärianalüüs sai sisendiks Teostaja Arendajale, kes koostas ärianalüüsile tuginedes tehnilise analüüsi. Pärast tehnilise analüüsi kooskõlastusel ilmnenud puudujääkide eemaldamist valmisid arendusülesanded.

1.1 Olemasolev süsteem

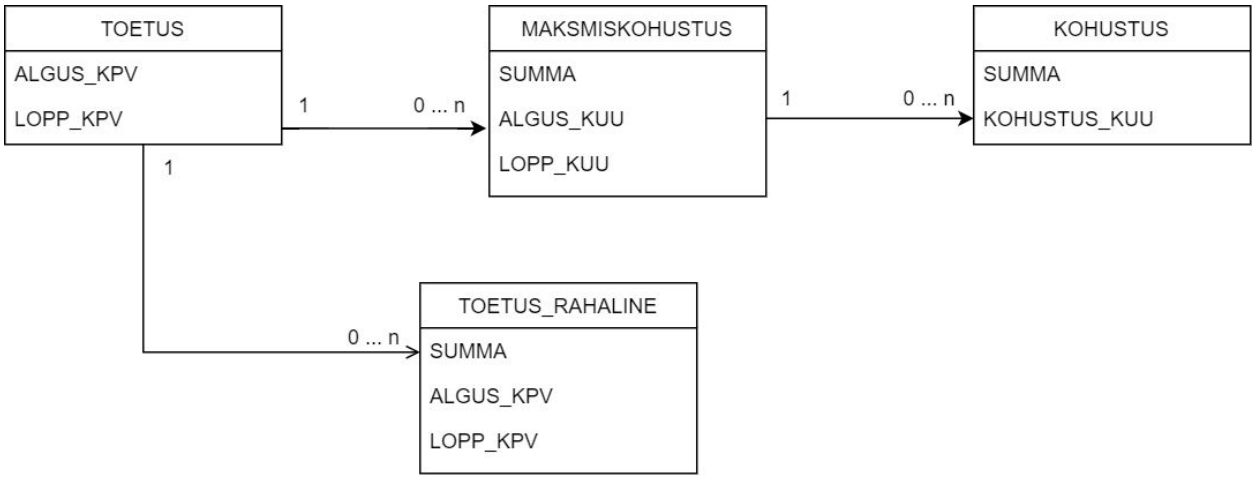
Süsteem, mis täiendusi vajas, oli olnud riigiasutuse poolt aktiivses kasutuses rohkem kui 3 aastat. Süsteem on sisult Springi raamistikule kirjutatud Java veebirakendus, millel on Angularis kirjutatud kasutajaliides ja Oracle andmebaas. Arhitektuuriliselt võib süsteemi lugeda monoliitseks.

Erinevate perioodiliste ja asünkroonsete tööde jaoks on loodud süsteemi osana kodukootud protsessimootor. Protsessimootori eesmärk on erinevatele Oracle AQ järjekordadesse kirjete lisamine ja erinevate tööde käivitusaegade haldamine. Töid on võimalik seadistada perioodiliselt käivituma ja automaatselt uuesti proovima ebaõnnestumise korral, mis võib juhtuda näiteks välise süsteemi vea tõttu. Kirjete lugemisega järjekordadest tegelevad erinevad deemon-protsessid, mis tegutsevad asünkroonselt ja põhisüsteemist sõltumatult. Protsessimootori tööde hulka kuuluvad näiteks X-Tee teenuste perioodiline väljakutsumine ja digiallkirjastamise järjekorra töötlemine.

Süsteem kasutab trükiste genereerimiseks Oracle BI Publisher tarkvara, milles on implementeeritud erinevad mallid ja neid täitvad andmebaasipäringud.

Süsteem kasutab välist raamatupidamistarkvara maksete teostamiseks, sealhulgas pankadega otsesuhtluseks.

1.2 Olemasolev andmemudel

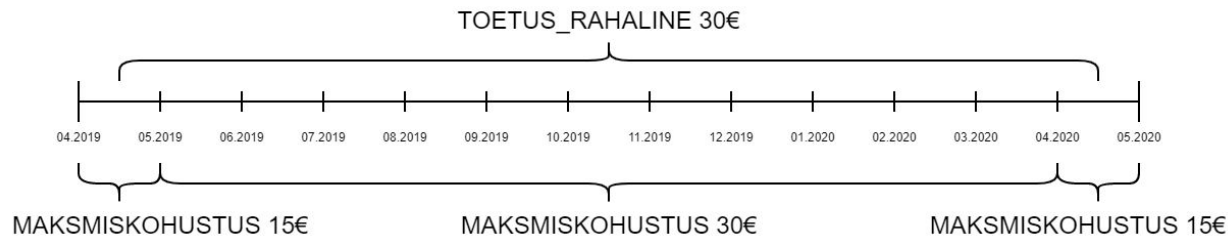


Joonis 1. Toetuse ja rahaliste alamstruktuuride lihtsustatud andmemudel.

Juhtumianalüüsi skoobis mängis keskset rolli toetus ja selle rahalisi osasid kirjeldavad alamstruktuurid. Tasub märkida, et toetusel ei ole alati rahalist osa ehk toetus võib olla mitterahaline. Toetuse rahalisi alamstruktuure on kaks - TOETUS_RAHALINE ja MAKSMISKOHUSTUS. Joonisel 1 on kujutatud toetuse ja rahaliste alamstruktuuride andmemudeli lihtsustatud versioon.

TOETUS_RAHALINE kirjeldab perioode, mille summad on arvutatud samadel alustel. Perioodid on arvestatud päeva täpsusega. Kirjed on informatiivseks kasutuseks ehk kasutajatele kuvamiseks.

MAKSMISKOHUSTUS kirjeldab kuu täpsusega perioode, mille kuumakse suurus on sama. MAKSMISKOHUSTUS kirjete põhjal luuakse iga kuu alguses sellele kuule vastav KOHUSTUS, mille põhjal loob raamatupidamistarkvara makseid.



Joonis 2. Näitetoetuse osad ajateljel.

Näiteks olgu isikule määratud igakuine toetus perioodiks 16.04.2019 - 15.04.2020. Toetuse suurus on määramise alguse hetkel seaduse järgi 30€ kuus. Kui toetust ei maksta täiskuu eest, tuleb toetust maksta proportsionaalselt kuu päevadele.

Nende andmete põhjal peaks tekkima süsteemi:

- üks toetuse kirje, perioodiga 16.04.2019 - 15.04.2020;
- üks TOETUS_RAHALINE kirje, perioodiga 16.04.2019 - 15.04.2020, summaga 30€;
- kolm MAKSMISKOHUSTUS kirjet
    - 04.2019 - 04.2019, summaga 15€
    - 05.2019 - 03.2020, summaga 30€
    - 04.2020 - 04.2020, summaga 15€

Näitetoetuse osad ajateljel on kujutatud joonisel 2.

1.3 Kliendi sisend

Analüüsifaasi sisendiks koostas Klient 14.02.2019 analüüsivajaduse pileti. Piletis sisaldus äriline eelinfo: millised saavad olema toetuste määrade muutused ning mis ajast need kehtima hakkavad. Lisaks oli kirjeldatud eelnevalt teostatud katsetuste tulem. Nenditi, et süsteem ei arvesta tagasiulatuva määramise korral erinevate toetuste määradega. Sõnastati keskne vajadus: “Tarkvaras tuleb teha muudatus, et kehtiksid ka eelmise seaduse määrad.” Klient suunas pileti 28.02.2019 Teostaja Analüütikule.

1.4 Ärianalüüs

Ärianalüüsi käigus esitas Teostaja Analüütik Kliendile täiendavaid küsimusi ning täiendas oluliselt vajaduste nimekirja.

Suurima täiendusena Teostaja Analüütiku poolt sai kirjeldatud olemasolevate toetuste uuendamise äriloogika. Tema nägemuses oleks süsteem pidanud määrade muutuse korral, mis kajastuvad süsteemsete parameetritena, käituma järgnevalt:

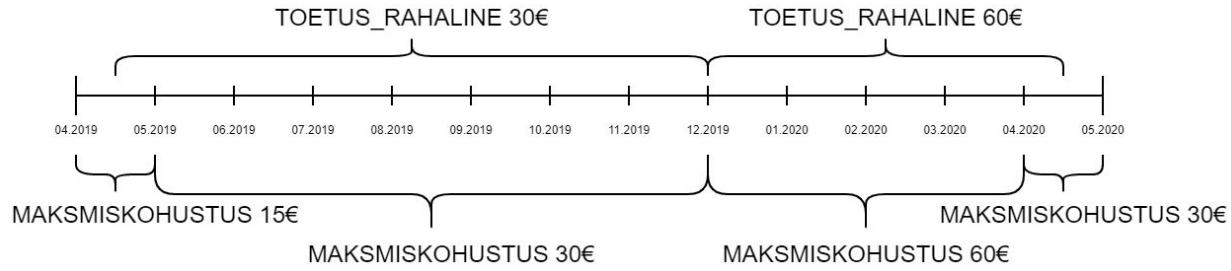
1) Leidma kõik vastava parameetriga seotud kehtivad toetused.
2) Lõpetama toetustega seotud rahalised toetused üks päev enne uue parameetri kehtimist.
3) Lisama uue rahalise toetuse, uue parameetri kehtivusaja algusest alates, mille summa on arvutatud tollel ajahetkel kehtivate väärtustega.
4) Uuendama maksmiskohustuste kirjeid vastavaks rahaliste toetuste infoga.

Teostaja Analüütik täpsustas ka teisi asjaolusid:

- süsteemis peab jääma toimima funktsionaalsus, mis arvestab toetuste proportsionaalsete osadega, kui toetus algab või lõppeb kuu keskel. Näiteks, kui toetuse suurus kuus on X€ ja toetus algab 20.11.2019, peab süsteem tekitama novembri eest maksmiskohustuse summaga X€*11/30 (11 päeva novembrist), alates detsembrist maksmiskohustuse summaga X€ ja nii edasi;
- süsteem peab toetuse määramisel arvestama parameetrite erinevate kehtivusaegadega, ehk lisama rahaliste toetuste ja maksmiskohustuste andmed ajaliselt õigete väärtustega

Teostaja Analüütiku täpsustavate küsimuste käigus sai selgeks, et töö skoopi ei kuulu olukord, kus määr muutub kuu keskel, vaid tuleb arvestada ainult olukorraga, kus määr muutub kuu alguse seisuga. See lihtsustab oluliselt arvutusloogikat. Määrade muutumine ainult kuu alguses tähendab, et kuu lõikes saab olla toetuse suurus kas 0, proportsionaalne toetuse päevade arvu suhtes või määra jagu (täissumma). Määra muutumisel kuu keskel saaks proportsionaalne või täissumma jaotuda veel omakorda proportsionaalselt vastavalt määra muutumise ajale.

Võttes aluseks “Olemasolev andmemudel” peatükis loodud näite, oleks koos määra muutusega näide järgnev.



Joonis 3. Määra muutusega näitetoetuse osad ajateljel.

Olgu isikule määratud igakuine toetus perioodiks 16.04.2019 - 15.04.2020. Toetuse suurus on määramise alguse hetkel seaduse järgi 30€ kuus ja see määr kehtib 2019. aasta 31. detsembrini. Alates 01.01.2020 jõustub toetuse määramise seaduse muudatus ja summa on alates 2020. aasta 1. jaanuarist 60€ kuus. Kui toetust ei maksta täiskuu eest, tuleb toetust maksta proportsionaalselt kuu päevadele.

Nende andmete põhjal peaks tekkima süsteemi:

- Üks toetuse kirje, perioodiga 16.04.2019 - 15.04.2020
- Kaks TOETUS_RAHALINE kirjet
    - 16.04.2019 - 31.12.2019, summaga 30€
    - 01.01.2020 - 15.04.2020, summaga 60€
- Neli MAKSMISKOHUSTUS kirjet
    - 04.2019 - 04.2019, summaga 15€
    - 05.2019 - 12.2019, summaga 30€
    - 01.2020 - 03.2020, summaga 60€
    - 04.2020 - 04.2020, summaga 30€

Määra muutusega näitetoetusega osad ajateljel on kujutatud joonisel 3.

Teostaja Analüütiku tööst jäi õhku küsimus: kuidas hetkel leitakse summade arvutamiseks õiged toetuse määrade parameetrid? Hüpoteesiks oli, et see toimub läbi toetuse liigi, kuid küsimus jäi uurimiseks tehnilise analüüsi protsessile.

1.5 Esmane tehniline analüüs

29. aprillil suunas Teostaja Analüütik pileti Teostaja Arendajale, kes alustas tehnilise analüüsiga. Tehnilise analüüsi eesmärgiks oli kaardistada kõik süsteemis muutmist vajavad komponendid ja kirjeldada, kuidas nende funktsionaalsus peaks toimima Kliendi vajaduste rahuldamiseks.

Tehnilise analüüsi käigus selgus, et Teostaja Analüütiku hüpotees arvutuse parameetrite leidmiseks oli ainult osaliselt õige. Skoobis olevatest toetustest on 8 puhul arvutusloogika otseses vastavuses hüvitise liigiga. Lisaks on skoobis 21 toetust, mille arvutusloogika sõltub peale toetuse liigi ka toetusele menetluse käigus määratud otsuse alusest.

Tehnilise analüüsi käigus tuli ka ilmsiks, et olemasolev süsteem arvutab toetuse summa väärtuse otsuse ettevalmistamisel ning rangelt hetkel kehtivate parameetrite järgi. Arvutatud väärtust kasutatakse edasi läbivalt - nii trükistel, kasutajaliidese kuvadel kui ka otsuse koostamisel.

Sõnastati vajaduse täpsustus. Süsteem peab saama võimekuse ühe toetuse raames tekitada rahaliste toetuste perioodid, arvestades arvutuskäigu parameetrite kehtivusi. Lisaks on vaja tekitada protsess, mis suudab olemasolevad toetused ümber arvutada, arvestades toetuse lisamisest hiljem lisandunud parameetrite väärtusi.

Detailsema lahendusena toodi välja:

- Täiendada parameetrite andmebaasist pärimist, eesmärgiga lisada teenusekihti võimekus pärida parameetreid kehtivusvahemike järgi. Süsteemis oli implementeeritud ainult konkreetsel kuupäeval kehtiva väärtuse leidmine.
- Refaktoreerida toetuste arvutamise loogika, muutes selle otsuse ettevalmistamise loogikast eraldiseisvaks. Refaktoreerimine on tarvilik, sest arvutusloogika peab olema ligipääsetav ka arendatavale ümberarvutamise protsessile.
- Otsuse kinnitamisel salvestatakse toetus, selle otsus, rahalised toetused ja maksmiskohustused. Rahaliste hüvitiste ja maksmiskohustuste loomisel peab nüüd arvestama arvutuskomponentidega, mis parameetrite kehtivuse järgi võivad perioodid jaotada alamperioodideks. Iga alamperioodi kohta tuleb tekitada eraldi rahaline toetus ning iga rahalise toetuse kohta maksmisperiood(id). Vana loogika peab esialgu koodibaasi alles jääma ja seda peab olema võimalik tagasi lülitada läbi *feature flag*i.
- Luua protsessimootorist käsitsi käivitamiseks mõeldud protsess, mis:
    - võtab sisendiks toetuse liigi.
    - leiab toetuse liigi alusel kõik kehtivad seotud toetused.
    - arvutab selle rahalise toetuse uuesti.
    - kui tulemus erineb olemasolevast, siis tegema andmestikku parandused.

Tehniline analüüs sisaldas koodiviiteid Java klasside ja meetodite näol, mis teostasid konkreetset puudutatud loogikat.

Tehnilise analüüsi esmane versioon sai analüüsipiletisse lisatud 22.05.2019.

1.6 Kliendi tagasiside

04.06.2019 toimus üle videosilla analüüsikoosolek, kus Teostaja poolelt tutvustas arendaja esmast tehnilise analüüsi tulemit. Lisaks sisulistele täpsustavatele küsimustele avaldas Vahendaja poolt ülevaatust teinud arhitekt soovi, et tehniline analüüs oleks tehniliselt täpsem.

11.06.2019 esitas Teostaja arendaja järgmise versiooni tehnilisest analüüsist, pealkirjaga "Tehniline täpsustus". Võrreldes eelmise versiooniga oli

- välja toodud konkreetsed toetuse liigid (koodid) koos arvutusloogikaga;
- viited konkreetsetele andmebaasitabelitele ja andmemudeli objektidele, mis olid tööga otseselt seotud;
- konkreetsed näited kuupäevade ja summadega erinevate kasutusvoogude tarbeks;
- algoritmiline kirjeldus ümberarvutamise protsessile

Olemasolevate punktide tehniline täpsustus sai heakskiidu 21.06.2019.

18.06.2019 tuli Kliendi poolt kirjalikult täiendavat tagasisidet, millega toodi välja täiendavad arendussoovid. Sooviti, et kõnealune muutuv toetuse määr oleks kajastatud menetlusprotsessi kuval, mis näitab kuude lõikes toetuse väljamakseid. Lisaks sooviti isiku toetuste kuvale uut modaalakent, mis näitaks rahaliste toetuste infot.

21.06.2019 analüüsikoosolekul arutati täiendavaid arendussoove ja tõstatati veel üks murekoht - trükised. Tehnilise analüüsi käigus pakuti välja kaks võimalikku lähenemist - vana Oracle BI Publisheri süsteemi täiendamist ja uue HTML-mallidel põhineva süsteemi juurutamist. HTML-mallidel põhinev süsteem oli kasutusele võetud Teostaja arendatud uute toetusteenuste moodulites, kuid pärandsüsteemis vastavat infrastruktuuri veel polnud.

Täiendavate arendusvajaduste tehniline analüüs esitati ülevaatamiseks 08.07.2019, koos trükiste lahenduse valikuvõimalustega. Vahendaja arhitekti tagasiside oli positiivne ja sisaldas ka kommentaare tagasiühilduvuse tagamise ning tingimuslikest kuvamistest hoidumise kohta. Lisaks tehti otsus, et liigutakse edasi HTML-mallide lahendusega.

Modaalakende täienduste tarbeks lõi Teostaja Analüütik prototüübid, mis olid tehniliselt kuvatõmmiste täiendused tarvilike tekstidega. Pärast Kliendi kooskõlastusringi kiideti need heaks.

12.08.2019 analüüsikoosolekul toodi välja, et süsteemis eksisteerib enammaksete funktsionaalsus, mis arvestab toetuse liigi muutudes olemasolevate maksetega ja korrigeerib uute maksete summasid vastavalt. Põgusa tehnilise analüüsi tulemusena jäi Teostaja Arendajal mulje, et see funktsionaalsus ei vaja täiendavat arendust. Sellegipoolest lepiti kokku, et arenduse ja testimise käigus vaadatakse funktsionaalsus üle ja kontrollitakse selle tööle jäämist. Töö paremaks õnnestumiseks lepiti ka kokku, et Klient esitab ärilised kirjeldused enammaksetega seotud töövoogudest, mille tööle jäämist saaks valideerida.

1.7 Lõplik tehniline analüüs

Lõplik tehniline analüüs anti Kliendile ülevaatamiseks 15.08.2019. Lõplik tekst koosneb sisuliselt eelnevalt teostatud tehnilisest täpsustusest ja täiendavate arendusvajaduste tehnilisest analüüsist. Lahendus jaguneb seitsmeks alapunktiks.

Esimene lahenduspunkt kirjeldab summade arvutamise refaktoreerimist. Eesmärgiks on luua rakendusse uus komponent, mis saab sisendiks toetuse liigi, otsuse aluse ja kuupäeva ning leiab sisendandmete põhjal toetuse summa. Välja on toodud nimekiri 11 toetuse liigi koodist ja nende arvutuskäikudest. 8 toetuse arvutuskäigud sõltuvad ainult toetuse liigist, 3 puhul on lisaks otsuse aluse mõõde 7 erineva võimalusega. Alapunktis on ka välja toodud vajadus uus summade arvutamise loogika kasutusele võtta vajalikes kasutuskohtades. Komponent peab võimaldama väljastada ka infot arvutuskäigus kasutatud parameetrite kehtivusvahemike väljastamiseks.

Teine lahenduspunkt hõlmab toetuse loomise loogika täiendamist, eesmärgiga arvestada kõiki arvutustes kasutatavate osade väärtusi, mis toetuse määramise perioodil kehtivad. Lahenduse kasutus peab olema juhitav *feature flag* kaudu. Töömahu kokku hoidmiseks tuleb vältida laialdast refaktoreerimist ja puutuda ainult otsest muutmist vajavaid komponente. Kirjeldatud on algoritm rahaliste toetuste ja maksmiskohustuste kehtivusvahemike leidmiseks ja loomiseks ning kirjeldus on toetatud näidetega.

Kolmas lahenduspunkt kirjeldab menetluse otsuse koostamisel kuvatava modaalakna täiendamist. Modaalaken näitab toetuste väljamakseid kuu lõikes ja peab uues lahenduses sisaldama staatilise toetuse määra protsendi välja asemel veergu, mis näitab iga kuu juures vastava kuu kohta kehtivat toetuse määra protsenti. Lahenduse kasutus peab olema juhitav *feature flag* kaudu. Komponent, mis arvutab modaalaknale kuu lõikes infot, on kasutusel ka rakenduse teistes komponentides ja need tuleb samuti järgi aidata.

Neljas lahenduspunkt toob välja vajaduse enammaksete funktsionaalsuse tööle jäämiseks ning kirjeldab koodiviidete kaudu, millised komponendid selle funktsionaalsusega seotud on. Eraldi on toonitatud, et enammaksete funktsionaalsus tuleb katta integratsioonitestidega, mis on kirjutatud Kliendi poolt esitatud kasutuslugude põhjal.

Viies lahenduspunkt on tehniline samm-sammult kirjeldus ümberarvutamise protsessi töö kohta. Protsess saab sisendiks toetuse liigi koodi ja leiab selle järgi kõik etteantud liigiga toetused, mis pole hetkekuupäevaks kehtivust lõpetanud ehk kaasab ka tulevikus algavaid toetuseid. Iga leitud toetuse kohta arvutatakse uuesti toetuse küljes oleva kõige hilisema rahalise toetuse osa ning erinevuste korral tehakse nii rahalise toetuse kui maksmiskohustuste andmetes parandused.

Kuues lahenduspunkt hõlmab määratud toetuste kuva täiendust, kus toetuse summale vajutades avaneb nüüd uus modaalaken rahaliste toetuste andmetega. Rahaliste toetuste modaalaken peab endas kajastama ka määra protsendi infot, kui vastav toetus kasutab seda parameetrit oma arvutuskäigus.

Seitsmes lahenduspunkt sõnastab üldiselt vajaduse luua uus komponent, mis genereerib menetluse andmete põhjal HTML-mallide kaudu positiivsete otsuste trükiseid. Töö hõlmab nii uute moodulite eeskujul HTML-mallide kasutamise infrastruktuuri loomist kui ka olemasolevate RTF-formaadis mallide ümber tegemist HTML-formaadis. Mallide sisend ehk RTF-formaadis failid ja nendes olevate muutujate väärtustamise SQL-päringud on lisatud eraldi manusena. Uutes trükistes peab sisalduma ka määra protsendi parameetri erinevate väärtuste info ja erinevad summad, millest kumbki ei olnud vanade trükiste puhul toetatud.

Teostaja Arendaja tõi ka välja, et lahenduspunktid 1-5 on omavahel otseses sõltuvuses ja pole mõistlik paralleelselt arendada. Punktid 6 ja 7 on piisavalt eraldiseisvad, et suurem osa on võimalik teostada ilma punktide 1-5 eelneva valmimiseta, kuid kuni veerand tööst on siiski otseses sõltuvuses.

1.8 Hinnangud

Pärast tehnilise analüüsi kinnitamist tuli Teostajal anda Kliendile tööde mahuhinnangud. Mahuhinnangutest lähtus hiljem Teostaja ka arendusressursi planeerimisel. Hinnangute andmisel tutvustas Arendaja erinevaid lahenduspunkte teistele Teostaja-poolsetele projekti liikmetele ning igale lahenduspunkti osale anti pädevusvaldkondade järgi kollektiivselt mahuhinnang. See tähendab - arendust hindasid arendajad, testimist testijad. Lõplik mahuhinnang koosnes funktsionaalsuse arendusele, automaattestidele ja testija manuaalsele testimisele kulunud ajast, seotud tegevustele kuluvast lisakoefitsendist ning arenduspuhvrist ootamatuste lahendamiseks.

Tabel 1. Arendustegevuste esialgsed mahuhinnangud.

Lahenduse alapunkt	Pealkiri	Arendus	Autotestid	Testimine	Kokku (koos puhvri ja seotud tegevustega)
punkt 1	summade arvutamise refakto	24	16	16	106
punkt 2	hüvitiste loomine õigeteks perioodideks	16	16	16	91
punkt 3	tabeli "Toetuse väljamaksed kuude lõikes" täiendus	16	0	8	49
punkt 4	toetuse ümber arvutamine	8	16	32	108
punkt 5	toetuse korrigeerimise protsess	32	8	16	108
punkt 6	rahalise toetuse modal - eraldi arendatav	24	0	1	50
punkt 7	otsuste PDF-id - eraldi arendatav	64	16	16	223

Mahuhinnagud tööde kaupa on välja toodud tabelis 1. Hinnangute andmisel olid muuhulgas tähtsateks asjaoludeks erinevate lahenduspunktide seotus ja projekti iseärasused.

Projekti iseärasustena tuli nentida, et olemasolevad automaattestid olid antud funktsionaalsuse suhtes puudulikud ja enamikel juhtudel tuli nullist luua. Lisaks polnud olemas kasutajaliidese automaattestimise komponente, mistõttu jäid kasutajaliidese täiendused testidega katmata, sest selliste komponentide loomine poleks enam skoopi mahtunud. Viimaseks oli projekti eelnevate arenduste põhjal teada, et arendajate pakutud hinnangud selles projektis on reeglina liiga optimistlikud ja lisatud puhvrid üldjuhul kasutati täiel määral ära. Põhjuseid võis leida nii projekti tehnilisest kui ärilisest keerukusest ning ka koodi tehnilisest võlast.

2. Teostus

Tehnilise analüüsi järel, pärast Kliendi heakskiitu, sai alustada arendustegevustega. Tarvis oli planeerida arendajate ja testijate ressurss. Tarkvaraarendus on üldjuhul keeruline, mistõttu esineb ootamatusi ja muutusi võrreldes esialgse plaaniga, sealhulgas võrreldes analüüsiga. Kogemus näitab, et tarkvaraarendaja toodab harva täielikult veavaba koodi, kuid puudusi aitab leida manuaalne testimine - eriti, kui seda teostab oma ala spetsialist.

2.1 Arendusressurss

Esialgne arendusplaan lähtus Teostaja Arendaja hinnangust, et tööd saab jagada efektiivselt kahe arendaja vahel. Hindamisel antud ajahinnangute järgi jagati esimene suurem tükk, menetlusprotsessi täiendused ja ümberarvutamise loogika, tehnilise analüüsi kirjutanud Arendajale. Teine suurem tükk, trükiste taasloomine, sai hiljuti projektiga liitunud Teostaja Kaasarendaja põhiülesandeks. Järelejäänud ülesande ehk rahaliste toetuste modaali loomine läks samuti Kaasarendajale, sest esialgsete hinnangute järgi oli Arendajal rohkem tööd ees.

Ajakavas püsimise üle pidas järge Teostaja Analüütik, kes täitis arendusplaani läbiviimisel projektijuhi rolli. Arendajad andsid selle tarbeks sisendit iga tööpäeva hommikul toimunud koosolekul, kus iga osaleja rääkis eelmisel tööpäeval tehtust, eesoleva tööpäeva plaanidest ja vahepeal ilmnenud takistustest.

Arenduse kulgedes tekkis vajadus tööd veelgi arendajate vahel laiali jaotada, kuna tähtajast tulenevate riskide realiseerumine muutus järjest tõenäolisemaks. Ajutiselt lisati veel kaks arendajat konkreetsete ülesannetega. Esimene Abistaja võeti juurde, kui Kaasarendaja oli valmis saanud uue trükiste loomise komponendi ja osaliselt ka eraldiseisvate otsuste trükised. Esimese Abistaja ülesandeks oli tegeleda koondotsuste trükiste ületoomisega. Teine Abistaja lisandus, kui Arendajal oli valminud punktid 1-3 ning tegeles punkti 5 arendamisega. Teise Abistaja ülesandeks oli luua integratsioonitestid punkti 4 katmiseks.

2.2 Arenduspõhimõtted

Projekti arenduspõhimõtted näevad ette, et toetatakse pidevintegratsiooni võimekust. Reaalsuses näeb see ette, et koodirepositoorumis on keskne koodiharu nimega *develop*, mille seisu pealt võib koostada ja saata Kliendile tarnepaki suvalisel ajahetkel. Tarnete tegemine toimub ebaregulaarselt ja vajaduspõhiselt, kuid üldiselt kord nädalas või tihemini. Pidevintegratsioon tingib vajaduse kasutada *feature flag* metoodikat arenduste puhul. Metoodika näeb ette, et enamik uusi arendatavaid funktsionaalsusi on sisse-välja lülitatavad *feature flag*i kaudu. Igale *feature flag*ile oli vastavas andmebaasitabelis rida. *Feature flag* muudatused rakenduvad süsteemis paari sekundiga ning ei vaja rakenduse taaskäivitamist ehk toimivad katkestusteta.

Arendaja vaatepunktist lisab pidevintegratsiooni nõue tema töövoogu täiendavaid tegevusi. Kui tehakse uusi arendusi, tuleb kood kõigepealt lisada muudatustele vastavale *feature*-harru. Tuleb meeles pidada, et arendused peavad olema kaetud *feature flag* metoodikaga. Kui arendus või suurem loogiline osa sellest on arendaja poolt valminud, teostatakse *feature*-harule koodiülevaatus mõne teise arendaja poolt. Pärast koodiülevaatuse protsessi põimitakse arendatud kood *develop*-harru.

Käesoleva arenduse lõpuks võeti kasutusele 3 *feature flag*i:

- `ASSIGNED_ALLOWANCES_SERVICES_AND_PROOFS_USE_MONETARY_ALLOWANCES_MODAL` lahenduspunkti 6 ehk rahaliste toetuste modaalakna näitamise juhtimiseks
- `USE_CALCULATORS_ON_BENEFIT_GENERATION` lahenduspunktide 1-4 ehk uue kalkulaatoripõhise lahenduse kasutamise juhtimiseks
- `USE_FLYING_SAUCER_REPORT_GENERATION` lahenduspunkti 7 ehk uue trükiste lahenduse kasutamise juhtimiseks

Lahenduspunkt 5 ehk toetuse ümberarvutamise protsess ei vajanud *feature flag* metoodikaga peitmist, kuna selle välja kutsumine ei muuda olemasolevaid komponente.

Teostaja poolelt on riistvara ressurss võrdlemisi piiratud ja testkeskkondi on üks, mis sisaldab endas kõige uuemat *develop*-haru seisu. Seega pole *feature*-haru staadiumis võimalik saada manuaalse testimise tagasisidet, vaid see saab tekkida pärast *develop*-harru põimimist, kus testija on *feature flag*i sisse lülitanud ja uue funktsionaalsuse üle vaadanud.

Kuigi projekti esialgne teostaja ei olnud automaattestide kirjutamises järjepidev, on praegune Teostaja võtnud eesmärgiks katta automaattestidega kõik uued arendused ning seeläbi tekitada täiendavat kaetuvust ka vanadele komponentidele. Testid jagunevad kahte kategooriasse - ühiktestid ja integratsioonitestid. Kasutajaliidese automaatteste projektis ei ole vastava komponendi puudumise tõttu.

Ühikteste on lihtne arendada, kuna need testivad kitsaid skoope ning on lihtsalt jooksutatavad, võttes üldjoones aega vähem kui sekundi. Kogu ühiktestide paki jooksutamine, mida tehakse pärast iga *commit*i, võtab aega umbes 2,5 minutit koos rakenduse ehitamisega. Ühiktestide jooksutamisel on kasutusel JUnit ja Mockito raamistikud.

Integratsioonitestid hõlmavad rakenduse osalist initsialiseerimist, mis on aja- ja riistvaramahukas tegevus. Lisaks katavad integratsioonitestid rakenduse keerukamaid osasid ning kohati ka suuremaid andmemahte. Praktikas tähendab see, et integratsioonitestide jooksutamise initsialiseerimine võtab 1-2 minutit aega ja kogu paki jooksutamine koos ehitusega üle 20 minuti. Integratsioonitestide jooksutamiseks on kasutusel Spring-test raamistik.

2.3 Teostatud tehnilised lahendused

2.3.1 Kalkulaatorid

Arendusvajaduse keskseks osaks on põhimõtteline muudatus toetuste summa arvutamisel. Kui varasemalt hõlmas see üldise toetuse summa parameetri ja vastava toetuse koefitsendi hetkeväärtuste omavahelist korrutamist, siis nüüd lisandub valemisse ka ajaline mõõde. Ühe toetuse raames võib olla mitu erinevat kehtivat väärtust ning vajadus on komponendi järgi, mis oskab erinevaid rahalisi perioode koostada ja väljastada.

Selle eesmärgi saavutamiseks loodakse *Template Method Pattern*it järgides `BenefitCalculator` abstraktne klass. *Template Method Pattern* näeb ette, et abstraktses ülemklassis on kirjeldatud algoritmi põhiosa. Alamklassid võimaldavad algoritmi alamosasid erinevalt defineerida, jättes ülemklassis kirjeldatud algoritmi üldosa samaks [5, p. 325].

`BenefitCalculator` sisaldab endas järgnevaid avalikke meetodeid:

- `protected abstract PaParameeterNum getBenefitRateMultiplierParam()`
- `public abstract boolean supports(HyLiik benefitType, OtsusAlus decisionBasis)`
- `public BigDecimal getBenefitRateMultiplier(LocalDate checkDate)`
- `public BigDecimal getMonthlyRate(LocalDate checkDate)`
- `public BigDecimal getBenefitRate(LocalDate checkDate)`
- `public List<CalculationPeriod> getCalculationPeriodsBetweenDates(LocalDate startDate, LocalDate endDate)`
- `public List<MonetaryBenefit> getMonetaryBenefitsBetweenDates(LocalDate startDate, LocalDate endDate)`

`BenefitCalculator` klassi implementeerivad 28 alamklassi, mis implementeerivad `supports(HyLiik benefitType, OtsusAlus decisionBasis)` ja `getBenefitRateMultiplierParam()` meetodeid. Kuna ükski kalkulaator endas olekut ei hoia, luuakse need *Singleton*-objektidena.

`BenefitCalculator`i kasutamiseks tuleb esmalt vastav kalkulaator leida. Seda võimekust pakub `BenefitCalculatorResolver` klass, mis tagastab sisendina võetud `HyLiik` ehk toetuse liigi ja `OtsusAlus` ehk otsuse aluse objektide järgi sobiva `BenefitCalculator` klassi objekti. Äriloogika, mis puudutab kehtivusvahemike ja summade leidmist, on taotluslikult `BenefitCalculator`i osana kirjutatud, eesmärgiga hoida kalkulaatorite kasutuskohad ühtse äriloogikaga.

2.3.2 Toetuste ümberarvutaja

Rakendusse on vaja luua võimekus käivitada ühekordselt protsess, mis arvutab ümber kõik kehtivad või tulevikus kehtima hakkavad toetused. Võimalusi selle realiseerimiseks on mitmeid - projekti eelnevate näidete najal on ühekordseid jooksutamisvajadusi lahendatud nii uue komponendi ja kasutajaliidese arendamise kui ka täiesti eraldiseisva käsurea-rakenduse näol. Rakenduses on olemas ka eraldiseisev kodukootud protsessimootor, mis tegeleb erinevate perioodiliste ja asünkroonsete ülesannetega. Valik osutus protsessimootorisse uue protsessi ehitamise kasuks. Uue komponendi arendamine ei olnud parim valik, sest teada on, et sisendandmeteks on ainult toetuse liik ja uue komponendi arenduse puhul ei ole nii lihtne vajadus piisav õigustus kasutajaliidese oluliseks täiendamiseks. Eraldiseisev käsurea-rakendus pole samuti eelistatud, sest projekti eelnevad näited lahendavad palju keerulisemat probleemi - mitme-etapilist andmete migratsiooni vanast rakendusest.

Lõpplahenduseks jääb olemasoleva protsessimootorile kahe uue asünkroonse, ainult manuaalselt käivitatava protsessi loomine - `BENEFIT_CREATE_RECALCULATION_QUEUE` ja `BENEFIT_RECALCULATE_MONETARY`.

`BENEFIT_CREATE_RECALCULATION_QUEUE` sisendiks on toetuse liigi kood. Protsess leiab kõik kehtivad või tulevikus kehtima hakkavad toetused, mis vastavad toetuse liigi koodile. Iga leitud toetuse kohta tekitab protsess `BENEFIT_RECALCULATE_MONETARY` protsessi.

`BENEFIT_RECALCULATE_MONETARY` sisendiks on toetuse *id*. Protsess leiab vastava toetuse ja selle kalkulaatori. Protsessi ülesandeks on alates järgmise kuu algusest kuni toetuse lõppkuupäevani arvutada ettenähtud summa. Kui leitud rahalised perioodid ei vasta toetuse küljes olevatele TOETUS_RAHALINE või MAKSMISKOHUSTUS kirjetele, lõpetatakse olemasolevad TOETUS_RAHALINE ja MAKSMISKOHUSTUS kirjed alates järgmise kuu algusest. Seejärel lisatakse toetusele perioodiks järgmise kuu algus kuni toetuse lõpp uued rahalised komponendid.

Ümberarvutamise funktsionaalsus on ennekõike mõeldud kasutamiseks mõne arvutuskomponendiks oleva parameetri muutumisele eelneval kuul. Näiteks, kui on teada, et mingi toetuse liigi summade arvutamisel kasutatav parameeter muutub 01.01.2020, annab parima tulemuse käivitada ümberarvutamine 2019 detsembris. Nõnda käitudes on lõpptulem loogiliselt tükeldatud. Kui ümberarvutaja käivitada samadel alustel 2019 novembris, tekiks ümberarvutaja töö tulemusena rohkem andmemüra ebavajalikult tükeldatud samasummaliste perioodide näol.

2.3.3 Kasutajaliidese täiendused

Menetlusprotsessi lõpus, otsuse koostamise kuval on võimalik avada määratava toetuse detailvaate modaalaken. Modaalaknas kuvati 2 plokki: ühe väärtusega andmed, nagu toetuse periood ja toetuse määr; ning väljamaksete infoga tabel kuude lõikes. Varasemalt oli arvestatud toetuse määra protsent ühe väärtusega andmeväljaks, uute arendustega võib see kuude lõikes muutuda, mistõttu kuvatakse väljamaksete tabeli veeruna. Arenduse mõttes on väljakutseks selle modaalakna andmete genereerimine. Varasemalt töötas loogika, kus väljamaksete iga rea summa oli toetuse määra ja selle protsendi korrutis ja need väärtused olid vastavalt toetusele alati samad, välja arvatud esimese ja viimase kuu piirjuhud. Nüüd tuleb iga kuu kohta kasutada kalkulaatoripõhist lähenemist. Kasutajaliidese enda muudatus on lõpuks triviaalne.

Kliendil on ka soov näha isiku toetuste vaates, iga konkreetse toetuse juures, millistest rahalistest toetustest see toetus koosneb. Kui rahalisel toetusel on seos toetuse määra protsendiga, siis määra protsendi väärtust soovitakse samuti näha. Kliendi vajaduse rahuldamiseks loodakse hüperlink toetuse summa väljale, mis avab uue modaalakna. Modaalaknas on kuvatud vastava toetusega seotud kõikide rahaliste toetuste info. Kuvatakse välju:

- “Liik”, mille väärtus saadakse seotud toetuselt;
- “Summa” ehk rahalise toetuse summa;
- “Protsent määrast” ehk rahalise toetuse perioodi algusajal kehtiv toetuse määra protsent, kui toetuse liigile leidub kalkulaator;
- “Periood”, millel on kindlasti alguskuupäev ja olemasolul lõpukuupäev

2.3.4 Trükised

Varasemalt oli rakenduses trükiste loomiseks läbivalt kasutusel Oracle BI Publisher tarkvara. BI Publisher on väga võimekas ärianalüüsi tööriist, kuid ajaloolistel põhjustel oli selle põhitööks käesolevas projektis trükiste loomine. Kliendil on soov järk-järgult BI Publisher asendada, peamiselt järgnevatel põhjustel:

- trükiste genereerimine läbi BI Publisheri on ajakulukas, võttes tavaliselt mitu sekundit ja halvimatel juhtudel minuteid, olenevalt mallist ja andmetest;
- BI Publisheri litsents oli püsikulu, mida saaks vabavaralise lahendusega vältida;
- BI Publisheri tarkvara on riistvara ressursi mõttes nõudlik;
- trükiste mallide haldamine ja hooldamine oli ebamäärane protsess, osapooltel puudus kompetents

BI Publisheri tarkvarasse on seadistatud erinevad RTF-formaadis mallid ja igale mallide vastavad PL/SQL päringud põhirakenduse andmebaasi vastu. Saades sisendiks näiteks taotluse *id* ja vajaliku malli nime, täidab BI Publisher RTF-formaadis malli andmetega, mis saadakse vastava malli päringute kaudu, konverteerib tulemi PDF-formaati ja tagastab faili põhirakendusele.

Võttes eeskujuks teistele toetustele Teostaja poolt hiljuti arendatud uued moodulid, luuakse uus trükiste loomise lahendus. Esmalt kogutakse kokku tarvilikud andmed rakenduse teenusekihi kaudu. Seejärel kasutatakse andmestikku, et Thymeleaf mallimootori abil täita HTML-formaadis olev mall. Saadud tulem Flying Saucer renderdamisteegi abil PDF-formaati ja salvestatakse rakenduse failihoidlasse.

Selle arenduse skoopi kuulub 7 positiivse otsuse ja 3 koondotsuse üle toomine uuele lahendusele, lisaks vajalikud sisutäiendused rahaliste toetuste kuvamisel. Iga mall tuleb käsitööna ümber kirjutada HTML-formaadis ning välja selgitada, kuidas tõlgenduvad BI Publisheris kirjeldatud päringud teenusekihis kättesaadavateks andmemudeli objektideks. Lisakeerukus tuleneb veel BI Publisheri-spetsiifilistest malli-sisestest loogikaelementidest, sealhulgas *if*- ja *for*-lausetest, mida on võimalik lugeda ainult spetsiifilist Microsoft Wordi pluginat kasutades. Selgus, et nende konkreetsete nõudmiste tõttu pole võimalik Linuxi operatsioonisüsteemis seda infot näha, mis tekitab Linuxit kasutavatele arendajatele täiendavat keerukust. Tagatipuks on BI Publisheri päringutes läbivalt kasutatud erinevaid PL/SQL spetsiifilisi tehnikaid, millest on kerge mööda vaadata või mille täpne käitumine on esmapilgul ebaselge. Parimal juhul väljenduvad need koheselt arendustestimisel tehnilise veana, halvematel juhtudel tulevad välja põnevate andmestike piirjuhtude esinemisel kasutajatestimise etapis.

2.4 Arendustestimine

Kivisse raiutud tähtajad tekitasid olukorra, kus Teostaja-poolset manuaalset testimist tuli alustada enne kogu arenduse valmimist, vastasel juhul ei oleks olnud võimalik arendusi õigeaegselt üle anda. Testija alustas oma tööga oktoobri lõpupäevadel, selleks ajaks olid testitavasse olekusse jõudnud punkt 6 ehk rahaliste toetuste modaalaken ja punktid 1-3 ehk menetluse põhivoo muudatused.

Esimesed päevad möödusid Testijal sisseelamiseks. Kuigi Testija oli rakendusega juba mitu kuud töötanud, siis konkreetse toetusmeetmega polnud ta varem pidanud kokku puutuma. Probleeme ja kitsaskohti selles protsessis oli omajagu.

Kõnealuse toetusmeetme menetlusvoog on võrdlemisi pikk ja kogu voo läbimine on nii aja- kui töömahukas protsess. Ajakulukas on ka katse-eksitusmeetodil menetlusvoo teadmuse tekitamine, sest dokumentatsioon on puudulik ja ühelgi Teostaja töötajal pole täielikku teadmust süsteemist.

Menetlusvoo olemasolevate vigade parandus on projektis olnud väheprioriteetne ning tähtis oli aru saada, kas Testija leitud viga on uute arenduste tõttu tekkinud. Kui viga ei olnud uute arenduste poolt põhjustatud ega mõjutanud seda oluliselt, jäeti vea parandamine tahaplaanile. Vastasel juhul oleks ohtu sattunud tööde tähtajaline üleandmine ja seeläbi ka kasutuselevõtt.

Testija suhtlus Arendajatega oli vahetu, bürokraatiavaba ja efektiivne. Pikemate arutluste korral oli võimalik kõrvaltuppa sammuda ja asjaolud näost näkku selgeks rääkida. Vahetu oli ka vigade raporteerimine - Testija jättis ära veapiletite vormistamise ja andis seotud Arendajale otse teada, kui midagi leidis. Esines ka valepositiivseid olukordi, kus pärast uurimist selgus, et süsteem toimis ootuspäraselt, kuid sellest polnud äriloogika keerukuse tõttu võimalik intuitiivselt aru saada.

Arenduspunkti 5 ehk toetuste ümberarvutaja testimine toimus võrreldes teiste arenduspunktidega erandlikult. Kuigi Testija sai enda loodud andmetega veenduda, et üksikjuhtumite puhul ümberarvutamine töötab, ei andnud see põhjalikku infot toodangukeskkonna andmete ega andmemahtude osas. Selle kindluse loomiseks sai kasutada Vahendaja testkeskkonda koodnimega Murutar, kuhu on kloonitud toodangukeskkonna andmestik hägustatud isikuandmetega. Murutaris testimine tõi kindluse, et ümberarvutamise protsess ei tekita toodanguandmetega ühtegi tehnilist viga, ei ole jäänud katmata piirjuhte ning kogu protsess lõpetab töö mõistliku kiirusega. Kuigi teostatud arendused mõjutasid kõiki 29 toetuse liiki, mõjutas seadusemuudatus konkreetselt 3 neist. Nende 3 mõjutatud liikide toetuste ümberarvutamine võttis Murutaris aega kokku vähem kui 15 minutit, toetusi oli kokku suurusjärgus 10000.

2.5 Vastuvõtukriteeriumid

Projekti erinevate arenduste raames on heaks tavaks iga pileti all sõnastada vastuvõtukriteeriumid, mille vastu valideerides saab lugeda pileti teostatuks. Kui vastuvõtukriteeriumid sõnastada juba analüüsifaasis, saab nendest abi nii Arendaja arendamisprotsessis, Testija testimisel kui ka hiljem Klient vastuvõtutestimisel. Käesolevate arenduste raames sai otsustatud, et vastuvõtukriteeriumid saavad kirja siis, kui Testija on oma töö lõpetanud ja tööd on valmis Kliendile üleandmiseks. Otsus küll ei toetanud arendus- ja testimistööd, aga oli tingitud arenduste suurest mahust. Sellise lähenemisega oli võimalik kirja panna sellised vastuvõtukriteeriumid, mis toetaksid kõige rohkem Kliendi vastuvõtutestimist, vältides ebaolulisi detaile ja keskendudes olulisematele eesmärkidele.

2.5.1 Toetuse summa arvutamise vastuvõtukriteeriumid

Feature flag:

`USE_CALCULATORS_ON_BENEFIT_GENERATION`

- Otsuse koostamise kuval:
    - On võimalik avada modaalaken “Toetuse väljamaksed kuude lõikes”
    - “Toetused väljamaksed kuude lõikes” modaalaknas on nähtav “Määra %” tulp
    - “Määra %” tulp näitab hüvitise liigi %määra parameetri väärtust konkreetses kuus
- Otsuse kinnitamisel luuakse korrektsed finantskohustused
    - Esimese ja viimase kuu finantskohustused on proportsionaalsed vastavasse kuusse jääva päevade arvu järgi.
    - Vaherahade arvutamise funktsionaalsus töötab sama loogika põhjal, nagu varem, kuid lisaks arvestab hüvitise määrade muutusi perioodides.
- Protsessimootori käivitamisel arvutatakse toetus ümber jooksutamise hetkel rakendusse sisestatud toetuste määrade ja määraprotsentidega alates jooksutamisele järgneva kuu algusest:
    - Protsess `BENEFIT_RECALCULATE_MONETARY` puhul on sisendiks toetuse *id*. Antud protsess arvutab ümber ainult konkreetse toetuse.
    - Protsess `BENEFIT_CREATE_RECALCULATION_QUEUE` puhul on sisendiks toetuse liigi kood. Antud protsess arvutab ümber kõik selle toetuseliigiga hetkel või tulevikus aktiivsed toetused.

2.5.2 Rahaliste toetuste modaalakna vastuvõtukriteeriumid

Feature flag:

`ASSIGNED_ALLOWANCES_SERVICES_AND_PROOFS_USE_MONETARY_ALLOWANCES_MODAL`

- Kui feature flag on välja lülitatud, siis ei ole isiku hüvitiste ja teenuste kuval "Määratud hüvitised, teenused ja tõendid" tabelis summa väljal hüperlinki
- Kui feature flag on sisse lülitatud, siis on "Määratud toetused, teenused ja tõendid" tabelis summa väljal hüperlink
- Hüperlink avab "rahalised toetused" modaalakna (näidise kohta on pilt taski manuses)
- Modaalaknas kuvatakse toetuse liik, summa, määra % ja periood, eraldi rida iga perioodi kohta, kui summa ja/või määra % erinevad

2.5.3 Trükiste vastuvõtukriteeriumid

Feature flag:

`USE_FLYING_SAUCER_REPORT_GENERATION`

- Kui toetusel on määramise perioodi jooksul erinev määr või määra protsent, siis on trükisel iga erineva määra või määra protsendiga periood eraldi reana toetuse tabelis
- Trükiste vorming on samaväärne varasemate trükistega
- Positiivse otsuse trükiste sisu on vastav ja korrektne iga toetuse ja otsuse korral

2.6 Ajakulu

Arendus algas 03.09.2019. Arendused anti üle 22.11.2019.

Tabel 2. Arendusülesannete tegelik ajakulu ja hinnatud ajakulu puhvriteta.

Töö	Teostaja	Ajakulu (tundides)	Esialgne hinnang puhvrita (tundides)
Toetuse summa arvutamine (p1-p5)	Arendaja	307	136
Enammaksete funktsionaalsuse integratsioonitestid (p4)	Teine Abistaja	57	16
Rahalise toetuse modaalaken (p6)	Kaasarendaja	54	24
Otsuste trükised (p7)	Kaasarendaja	225	80
Otsuste trükised (p7)	Esimene Abistaja	96	
Testimine	Testija	147	105
Koodiülevaatused	Projekti teised arendajad	12	0
Toetavad tegevused, projektijuhtimine	Analüütik	38	0

Ajakulu eri tegevuste ja teostajate lõikes on välja toodud tabelis 2. Koodiülevaatustel ja toetavatel tegevustel ei ole eraldi hinnangut, sest neid arvestati puhvri osana.

Esialgne mahuhinnang oli kokku 735 tundi. Üleandmise ajaks oli tehtud 936 tundi tööd. Arendajate hinnangute ja tegelikult kulunud aja suurest vahest saab järeldada, et arendajate antud hinnangud olid väga optimistlikud ja puhvrite lisamine mahuhinnangutele oli õigustatud.

Suurim erinevus hinnatud ja tehtud töö vahel on trükiste ületoomise arendusega. Põhjustena võib välja tuua, et trükiste arendusel on seotud arendustest kõige vähem tehnilist analüüsi ja eelnevat kogemust olemasoleva lahendusega. Eelnevalt nimetatud teadmuse tekitamine oleks mingil määral aidanud arenduse väiksema ajakuluga valmis saada, kuid ennekõike andnud parema pildi hinnangute andmisele.

3. Kasutusele võtmine

Selleks, et Teostaja tehtud muudatused toodangukeskkonnas kasutusele võetakse, ei piisa pelgalt muudatuste tegemisest koodibaasis. Muudatused tuleb ka Kliendile tarnida, mis on projektis kujunenud omaette väljakutseks erinevatel ajaloolistel ja tehnilistel põhjustel. Muudatused peavad läbima ka Kliendi testimise ehk vastuvõtutestimise. Peale testimisega leitavate vigade on alati ka oht, et midagi jääb kahe silma vahele, toimub möödarääkimisi või esineb muid ettenägematuid muresid. Siht on aga selge - muudatused on vaja kasutusele võtta. Käesolevate muudatustega on ka selgelt teada, mis ajaks peab kasutusele võtmine toimunud olema - uueks aastaks.

3.1 Tarneprotsess

Selleks, et Klient saaks uusi arendusi vastu võtta, testida ja paigaldada erinevatesse keskkondadesse, tuleb läbida erinevaid tehnilisi ja bürokraatlikke samme.

Teostaja tarne Kliendile algab vajaminevate süsteemi moodulite kaardistamisest. Alati ei piisa konkreetse arenduse raames muudetud mooduli tarnimisest, kuna tarne võib sisaldada muid töid, millel on ka teiste moodulite sõltuvusi. Kirjalikul teel, kas eelnevate teadaannete või projektisisese info jagamise üleskutse näol, kogutakse sõltuvuste infokillud kokku. Järgneb versioneerimine - iga vajamineva mooduli puhul käivitatakse keskse koodiharu tipu pealt GitLabi *pipeline* samm “Release minor version”, mis suurendab versiooninumbrit vastaval moodulil uue *commit*iga. Tulemuseks on ka uus *pipeline*, mille tulem tuleb paigaldada *release-e2e* keskkonda.

Kui kõik soovitud moodulid on uue versiooni külge saanud ja *release-e2e* keskkonda paigaldatud, jooksutatakse keskkonnas *end-to-end* testide kogum, mille eesmärk on rakenduse versiooni regressioontestida. Edukas testide jooksutamine annab tarne teostajale signaali, et *release-e2e* keskkonda paigaldatud seis on tarnevalmis.

Mooduleid seob katuseprojekt *composite*, mis hoiab endas muuhulgas ka versioonide manifesti. Saanud kinnituse, et versioonid on tarnevalmis, muudab tarne teostaja *composite* projekti manifestis moodulite versioonide viited vastavaks. Versioonimuutuste *commit*i peale käivitub *composite* projekti *pipeline*, milles sisalduvad kõikide moodulite paketeerimise ja väljasaatmise sammud, mida tuleb käsitsi käivitada.

Lisaks paki edastamisele on tarvis ka arenduspileteid täiendada, erinevaid dokumente luua ja e-kirju saata. Iga arenduspilet, mis tarnega kaasneb, peab sisaldama vastava tarne märget eraldi *fixversion* väljal. Iga tarne kohta koostatakse Kliendile eraldi veebidokument ehk tarneleht, mis sisaldab:

- *release-e2e* keskkonnas jooksutatud eduka *pipeline* linki;
- *composite* projekti *pipeline* linki, milles toimusid paketeerimise ja väljasaatmise sammud;
- tabelit seotud vea- ja arenduspiletite väljavõttega, mille sisu genereeritakse automaatselt vastava *fixversion* kohta;
- konfiguratsioonimuudatuste detailseid kirjeldusi, kui vastavaid muudatusi on tarvis

Tarne viimaseks sammuks on tarneteade ehk e-mail, mille sisuks on link tarnelehele.

Kogu protsessil on ka erisus pärandrakenduse näol, tulenevalt selle rakenduse ülesehitusest, seotud GitLabi *pipeline* erisustest ja *end-to-end* testide puudumisest. Kui on tarvidus ainult pärandmoodulit tarnida, tuleb ainult *develop*-haru tipus käivitada eraldiseisev paki koostamise ja saatmise samm - jäävad ära versioonimuutused ja automaatne regressioontestimine. Jätkuvalt tuleb arvestada teiste moodulite sõltuvustega. Pärandrakenduse tarnimisel tuleb lisada tarneteatesse eraldi tarnitud *pipeline* link. Olenevalt kehtivast lepingust Teostaja ja Vahendaja vahel võib olla vajalik ka eraldi tarneteate akt üleantud tööde nimekirjaga, mis puudutab pärandrakendust.

Klient kasutab põhiliselt 3 keskkonda - TEST, PRELIVE ja LIVE ehk toodangukeskkond. TESTi paigaldatakse otse läbi *pipeline* funktsionaalsuse, tavaolukorras kedagi teavitama ei pea. PRELIVE keskkonda paigaldusel tehakse paigalduse tellimuse pilet süsteemide haldajale ja PRELIVE keskkonda üldiselt ei paigaldata uusi pakke rohkem kui kord päevas. LIVE keskkonna paigaldused käivad samuti läbi tellimuspiletite, regulaarsed paigaldused toimuvad iga kahe nädala tagant. Tellimuspiletite info lisatakse hiljem ka vastavale tarnelehele.

3.2 Demonstratsioon

Pärast käesolevate arenduste üleandmist Kliendile toimus koosolek, kus Testija demonsteeris muudatusi Kliendile üle videosilla. Koosolek toimus 25.11.2019. Tunniajase koosoleku jooksul käidi läbi üks menetluse positiivne voog, tuues esile muutunud kuvasid, näidates muutunud trükist ja selgitades taustal toimuvaid protsesse. Lisaks näidati, kuidas käivitada ümberarvutamise protsessi ja kuidas selle tulemit valideerida uue modaalakna abil. Selgitati ka seotud *feature flag*e ja nende mõjusid.

Kuigi Testija demonstratsioon oli läbiviimise koha pealt edukas, selgus hiljem, et demonstratsioon oli rohkem formaalsus kui efektiivne infovahetus. Selle tõdemuse andis asjaolu, et demonstreeritud funktsionaalsuse kohta laekusid hiljem küsimused, mille kohta demonstratsioonil juba infot jagati. Teostaja leidis hiljem järele mõeldes, et sellise mahuka arenduse puhul oli ühetunnine demonstratsioon üle videosilla ebapiisav. Puudu jäi esitluse ajast, ehk parem oleks olnud rohkem erinevaid näiteid läbi mängida. Lisandväärtust oleks andnud ka praktiline kogemus Kliendile, ehk ette valmistada kasutusvoog, mida Kliendi spetsialistid kõrvalise abiga saaksid läbida, et paremini tutvuda uute võimalustega.

3.3 Vastuvõtutestimine

Klient koostas vastuvõtutestimise paremaks läbiviimiseks iseseisvalt testimisplaani. Testimisplaanis oli välja toodud 33 erinevat andmekomplekti kirjeldust koos soovitud tulemustega. Testimisplaani jagati arendusfaasi lõpu poole ka Teostajale. Teostaja Testija sai testimisplaani läbi töötades lisamõtteid, milliseid kitsaskohti peaks üle vaatama. Lisaks andis see võimaluse Teostaja poolelt anda tagasisidet omapoolse teadmuse najal, kui testimisplaanis on midagi katmata jäänud. Mõningad Kliendi küsimused ja ebamäärasused said Teostaja poolt täpsustuse, aga üldjoones oli pilt väga positiivne - tundus, et Klient on põhjalik ja asjatundlik.

Pärast arenduste üleandmist Teostaja poolt algas vastuvõtutestimine. Ajalooliselt on olnud probleeme vastuvõtutestimise aegluse või alustamise viivitamisega, aga kuna nende muudatuste toodangukeskkonda jõudmisel oli väga konkreetne ja kiiresti lähenev tähtaeg, alustati tööd kiirelt. Esimestel päevadel ületati üheskoos tehnilisi raskusi, mis olid eelkõige seotud parameetrite kohandamisega testimiseks, *feature flag*ide kasutusega ja protsessimootori töö selgitamisega.

Vastuvõtutestimise raames toimus suhtlus Kliendi ja Teostaja vahel nii üle Skype'i kui ka veapiletite kaudu. Vastuvõtutestimise alguses esines mitmeid olukordi ja veapileteid, kus raporteeriti vigastest kohtadest, mille juurpõhjus ei peitunud arendustes, vaid keskkondlikes eripärades või seadistuses. Siiski jõuti aegsasti arusaamale, et põhifunktsionaalsus töötab ettenähtult. See-eest, mida aeg edasi, seda detailsemalt hakkas Klient vaatama piirjuhte ning sealt koorus välja omajagu õigustatud veapileteid. Tähtajast tingituna said kõik seotud vead väga kriitilise prioriteetsuse astme, olenemata vea reaalsest olemusest - näiteks menetlusvoos õigete summade kuvamine oli sama tähtis kui trükisel otsuse numbri vorminduslik mahtumine ühele reale. Sarnaselt Testijale tõstatas Klient vigu, mis olid küll seotud menetlusvoos, kuid käesolevate arendustega mitteseotud ja seetõttu tuli skoobi hoidmise eesmärgil hilisemaks jätta. Vahel õnnestus bürokraatilist protsessi kiirendada seeläbi, et Klient andis vea olemusest Skype'i teel teada ning pärast Teostaja kinnitust, et viga on tarvilik ja mõistlik kohe ära lahendada, vormistati see piletiks. Sellisel juhul oli Teostajal võimalik veaparandusega alustada varem kui pileti laekumine seda muidu võimaldanud oleks.

Tabel 3. Vastuvõtutestimisel Kliendi poolt raporteeritud veapiletid.

Pilet	Vea olemus	Tulemus
V-323	Otsuse koostamine ebaõnnestub, kui otsuse kuupäev ja toetuse väljamaksmise kuupäev on omavahel võrreldes erinevates kuudes	Tehniline viga parandatud
V-323	Otsuse koostamine ebaõnnestub, kui esmasel menetlusel määrati rahaline toetus ja korduval menetlusel rahalist toetust ei määrata	Tehniline viga parandatud
V-323	Rahalisel toetusel, millel on mitu otsuse alust, ei kuvata rahaliste toetuste modaalaknas info	Tehniline viga parandatud
V-326	Kehtetuks tunnistatud määramise otsus ei tühista seotud määratud toetust, otsuse faili ei õnnestu genereerida	Ei paranda, vastavat funktsionaalsust pole olemas olnud
V-312	Kui rahalise toetuse modaalaknas kuvatakse toetust, mille algusaeg on enne toetuse määra parameetri esimest kehtivuse algust, tekib tehniline viga	Tehniline viga parandatud
V-312	Kui rahalise toetuse modaalaknas kuvatakse toetust, millel on mitu määramise otsust, tekib tehniline viga	Tehniline viga parandatud
V-312	Rahalise toetuse modaalaknas on vale nimetus määra protsendi veerul	Tekst muudetud “Määra %” -> “Protsent määrast”
V-313	Trükistel on vale nimetus määra protsendi veerul	Tekst muudetud “Määra %” -> “Protsent määrast” seotud kuvadel ja trükistel
V-315	[konkreetset liiki] otsuse trükisel on teist liiki otsuse tekstiplokk	Tekst muudetud
V-321	Otsuse numbri lühendi järel ei tohi punkti olla	Tekst muudetud
V-321	Trükisel “Asjaolude muutumine” plokis puudub lauseosa	Tekst muudetud
V-321	Trükisel ei ole arvamuse tekst rööpjoondatud	Vormistatud eraldi piletiks, lahendatud pärast toodangukeskkonnas kasutusele võtmist
V-322	Koondotsuse alamotsused tuleb esitada loendina	Trükis muudetud

V-322	Otsuse kuupäev ja number peavad mahtuma ühele reale	Trükis muudetud
V-327	Kui toetuse saajal on õppimise kirje, millel pole lõpuaega, ebaõnnestub trükise loomine	Trükistel kuvatakse ainult õppimise alguskuupäev
V-327	Kui toetuse saajal on mitu aktiivset õppimise kirjet, ebaõnnestub trükise loomine	Trükisel kuvatakse ainult kõige hilisema algusajaga aktiivne õppimine
V-329	Osade trükiste lausete lõpus on puudu punktid	Puuduvad punktid lisatud
V-331	Trükistel tekib muutuja ja sellele järgneva tähemärgi vahele tühikud	Loodud tehniline lahendus mallimootori täiendusena
V-333	Kui eelnev otsus on koondotsus, ei kuvata trükisel eelneva otsuse numbrit	Loogika täiendatud
V-335	Kui määramine on tekkinud 2017. aasta migreerimisega, ebaõnnestub trükise loomine	Tehniline viga parandatud

Arendajad raporteerisid vastuvõtutestimisel tekitatud veapiletite (tabel 3) raames kokku 97 tundi, Testija 29 tundi. Vahemikus 23.11 kuni 20.12 esines eeltoodud piletitega seotud tarneid 10 korral.

Enamik leitud vigadest olid seotud trükistega. Trükistega seotud tehnilised vead on kriitilised, sest kui trükise genereerimine ebaõnnestub, ei ole võimalik menetlust lõpetada ja otsust koostada. Samas on ka paljud sisulised vead Kliendile väga olulised, kuna otsuseid on võimalik vaidlustada ja vaidlustamise aluseks kasutada eelnevalt väljastatud trükist.

Vastuvõtutestimine lõppes 18.12.2019 ja testimisplaanide tulemuste kohta raporteeriti piletile kommentaariumites.

3.4 Toodangukeskkonnas kasutusele võtmine

Vastuvõtutestimise lõpp oli ka otseselt seotud toodangukeskkonnas kasutuselevõtuga, mis toimus 18.12.2019 õhtul. Kliendi jaoks oli see mingis mõttes viimane hetk muudatuste paigaldamiseks, sest tegu oli 2019. aasta viimase täispikkuses töönädalaga - järgnesid jõulupühad ja puhkused.

19.12.2019 hommikul lülitati sisse *feature flag*id, mis aktiveerisid muudatused. Kliendi poolelt tuli Teostajale sõnum, et nad soovivad nüüd määrade parameetrid ära muuta, aga Teostaja poolelt anti mõista, et see oleks pidanud koos teiste Teostaja muudatustega juba tehtud olema. Sellegipoolest sai Teostaja poolt toodangukeskkonna seis üle vaadatud ja tuli välja, et detsembri alguses oli ainult toodangukeskkonnas juba uusi parameetri väärtuseid käsitsi lisatud. Teostaja toonitas, et tegu on takistava seadistusveaga ja rakendus oleks kasutamisel hakanud palju tehnilisi vigu saama. Klient koostöös Vahendajaga tegi parameetrite seadistuse poole tunniga käsitsi korda.

Järgmise murena tuli Kliendi suitsutestimise raames välja, et menetlusvoos ei ole enam võimalik väljamaksekanalina uut pangakontot lisada. Selgus, et toodangukeskkonda oli jõudnud ebaküps muudatus, mis suunas kasutaja uue pangakonto lisamiseks eraldiseisvale uue mooduli kuvale. Muudatus ei olnud *feature flag*iga kaetud, kuid oleks pidanud olema, sest polnud veel Kliendile üle antud ja kasutuselevõtt oleks hõlmanud teisi vajalikke ettevalmistusi. Olukord põhjustas takistusi menetlejate töös ja vajas kiiret lahendust. Teostaja poolt sai vastava *feature flag* loogika lisamise muudatus 3 tunniga *develop*-harru. Toodangukeskkonda jõudis muudatus sama päeva õhtul, töövälisel ajal.

20.12.2019 hommikul raporteeris Vahendaja, et toetuste ümberarvutamise protsessid käivitati edukalt ja ühtegi tehnilist viga nendega seoses rakenduse logidesse ei jõudnud. Klient teostas pistelist kontrolli ümberarvutamise tulemustele ja ei leidnud vigu.

Tabel 4. Kliendi poolt raporteeritud veapiletid pärast toodangukeskkonnas kasutusele võtmist.

Pilet	Vea olemus	Tulemus
H-1625	Ei ole võimalik [menetluse käigus lisatud] tekstile reavahesid lisada	Loodud tehniline lahendus mallimootori täiendusena
H-1627	Trükisel näitab eksperdi isa nime	Nime kuvamise loogika muudetud
H-1624	Trükisel näitab vallavalitsuse nime ees ‘null’	Nime kuvamise loogika muudetud
H-1628	Trükisel kuvatakse valed alused	Tekst muudetud

Toodangukeskkonnas kasutusele võtmise järel loodi 4 veapiletit, mis on välja toodud tabelis 4. H-1624, H-1625, H-1627 vormistati piletitena 19.12.2019 õhtul, parandati 20.12.2019 hommikul, tarniti 20.12.2019 pärastlõunaks ning jõudsid toodangukeskkonda enne 20.12.2019 tööpäeva lõppu. H-1628 ei jõudnud pärastlõunase tarne ajaks testitud ning seetõttu ei läinud tarnega kaasa. Tegu polnud Kliendi jaoks kriitilise veaga ja sel põhjusel tarniti parandus pärast puhkuseperioodi 10.01.2020 ning paigaldati toodangukeskkonda 23.01.2020.

Uue aasta alguses tuli Teostajale Kliendi poolt mitmeid positiivseid sõnumeid töö hästi õnnestumise kohta. Paradoksaalselt oli üks suuremaid tunnustusi vähene meediakajastus. Kui oleks realiseerunud mõni negatiivsem stsenaarium, näiteks ümberarvutamise ebaõnnestumine või menetlusprotsesside pikem seisak, oleks Klient saanud palju negatiivset tähelepanu nii mõjutatud toetuste saajate kui meedia poolt. Arendus oli edukas ja toetuse määra muutumist kajastati muuhulgas Aktuaalses Kaameras paarisekundilise mainimisega.

18.02.2020 kirjutas Vahendaja esialgse p1-p5 arenduspileti alla: “17.02.2020 koosoleku otsusega sulgen pileti. Testimine on lõpetatud, töö on lives kasutusel.”

Kokkuvõte

Käesoleva töö raames on kirjeldatud riigiasutuse pärandsüsteemi täiendamise protsessi, hõlmates nii analüüsi, arendust kui muudatuste kasutuselevõttu.

Analüüsietapis selgitati välja olemasolev funktsionaalsus ning selle puudujäägid. Ärisisendist ehk seadusandluse muudatuse kirjeldusest koorus välja oluliselt mahukam muudatusvajaduste komplekt. Kliendi, Vahendaja ja Teostaja koostööl loodi tehniline analüüs muudatusvajaduste teostamiseks. Teostaja andis muudatuste teostamisele esialgse ajahinnangu.

Arendusetapis planeeris Teostaja tarvilikud tegevused vastavalt muudatuste sisule, saadaval olevale ressursile ja tähtaegadele. Probleemide ennetamiseks ja aegsasti lahendamiseks kasutati agiilsete tööprotsesside poolt pakutavaid vahendeid. Tehniliste lahenduste teostamisel kasutati levinud ja praktikas tõestatud arenduspraktikaid - *feature flag*id, vabavaralised abiteegid, koodi disainimustrid, automaattestid. Manuaalse arendustestimise tulemusena lahendati operatiivselt ja põhjalikult veaolukordi ning sõnastati vastuvõtukriteeriumid. Selgus, et analüüsietapis antud ajahinnangud ei olnud pädevad, kuid see risk oli projektijuhtimise tasandil hallatud ja ei mõjutanud projekti edukust.

Kasutuselevõtu etapis tarnis Teostaja Kliendile muudatused ja mitmel korral ka vajalikke parandusi. Teostaja tutvustas Kliendile muudatusi demonstratsioon-esitluse vormis. Teostaja toetas Kliendi tarvilikke tegevusi jooksvalt ja võimalikult vähese ajakuluga, et tagada projekti õigeaegne õnnestumine. Klient viis läbi vastuvõtutestimist tuginedes oma põhjalikule testimisplaanile ja Teostaja nõuannetele. Toodangukeskkonnas muudatuste kasutusele võtmine oli edukas, eesmärgid saavutati õigeaegselt ja takistavate vigadeta.

Töös kirjeldatud tegevused, tehtud valikud ja otsused ei pruugi olla kõikides tarkvaraarenduse olukordades parimad või isegi kasulikud. Tuleb arvestada olemasoleva süsteemi võimalusi, arenduseks kasutatavaid ressursse, tähtaegu ja kohati isegi ajaloolisi tagamaid. On mõistetav, et infosüsteemi loomisel ei ole tihtilugu võimalik ette näha, milliseid vajadusi peab süsteem tulevikus rahuldama. Küll aga tuleb tõdeda, et parimate praktikate järgi kirjutatud kood on hiljem paremini hooldatav. Riigiasutuste ärinõuded ja seeläbi nende infosüsteemid on keerulised ning nende arendus ajamahukas.

Viidatud kirjandus

[1] Euroopa Komisjon. Euroopa koosvõimeraamistik Euroopa avalike teenuste osutamiseks. Lisa 2. Komisjoni teatis Euroopa Parlamendile, nõukogule, Euroopa Majandus- ja Sotsiaalkomiteele ning Regioonide Komiteele. 2010. https://ec.europa.eu/transparency/regdoc/rep/1/2010/ET/1-2010-744-ET-F1-1-ANNEX-2.PDF (27.04.2020)

[2] World Wide Web Consortium. WAI-ARIA Authoring Practices 1.1. 2019. https://www.w3.org/TR/wai-aria-practices-1.1/ (27.04.2020)

[3] Aijaz, Adil; Echagüe, Pato. "Managing Feature Flags". O'Reilly. ISBN 9781492028598. 2018.

[4] GitLab. "CI/CD pipelines". GitLab Docs. https://docs.gitlab.com/ee/ci/pipelines/ (27.04.2020)

[5] Gamma, Erich; Helm, Richard; Johnson, Ralph; Vlissides, John. "Template Method". Design Patterns. Addison-Wesley. ISBN 0-201-63361-2. 1994.

Litsents

Lihtlitsents lõputöö reprodutseerimiseks ja üldsusele kättesaadavaks tegemiseks

Mina, Joosep Heinmets,

(autori nimi)

1. annan Tartu Ülikoolile tasuta loa (lihtlitsentsi) minu loodud teose
   Riigiasutuse pärandsüsteemi täiendamise juhtumianalüüs,
   (lõputöö pealkiri)
   mille juhendaja on Vambola Leping,
   (juhendaja nimi)
   reprodutseerimiseks eesmärgiga seda säilitada, sealhulgas lisada digitaalarhiivi DSpace kuni autoriõiguse kehtivuse lõppemiseni.
2. Annan Tartu Ülikoolile loa teha punktis 1 nimetatud teos üldsusele kättesaadavaks Tartu Ülikooli veebikeskkonna, sealhulgas digitaalarhiivi DSpace kaudu Creative Commonsi litsentsiga CC BY NC ND 3.0, mis lubab autorile viidates teost reprodutseerida, levitada ja üldsusele suunata ning keelab luua tuletatud teost ja kasutada teost ärieesmärgil, kuni autoriõiguse kehtivuse lõppemiseni.
3. Olen teadlik, et punktides 1 ja 2 nimetatud õigused jäävad alles ka autorile.
4. Kinnitan, et lihtlitsentsi andmisega ei riku ma teiste isikute intellektuaalomandi ega isikuandmete kaitse õigusaktidest tulenevaid õigusi.

Joosep Heinmets

27.04.2020