TALLINNA TEHNIKAÜLIKOOL
Infotehnoloogia teaduskond

Annaliisa Romanenkov 192194IAAM

Edumõõdikute kasutuselevõtt featuuride jaoks Telia Eesti AS näitel - jälgimine ja probleemkohad

Magistritöö

Juhendajad: Kristjan-Hans Sillmann
MSc

Nadežda Furs
MBA

Tallinn 2021

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: Annaliisa Romanenkov

20.05.2021

Annotatsioon

Käesoleva magistritöö eesmärgiks on selgitada väljad, millised murekohad takistavad tootejuhtidel featuuridele äriliste eesmärkide püstitamist, kuidas saavutada olukord, kus ärilise väärtuse mõõdikud seatakse korrektselt ja peale featuuri kasutusse minekut jälgitakse eelnevalt seatud mõõdikuid ning tekitada võimalus SMART featuuride osakaalu usaldusväärseks jälgimiseks.

Telia Eesti IT arenduse vaates on SMART featuuride osakaal kõikidest featuuridest üks olulisemaid mõõdikuid, kuid vaatamata sellele on nimetatud mõõdiku tulemused väga madalad. Featuuridele seatakse mõõdikuid, kuid need pole kas mõistlikult seatud või ei jälgita neid piisavalt. Seetõttu on oluline aru saada, miks paljud tootejuhid ei tarbi jätkuvalt oma ärimõõdikuid featuuride tulemuslikkuse eesmärgistamiseks ja jälgimiseks ning mis neid selle juures takistab.

Töö tulemusena koostab autor tervikliku aruande SMART featuuride osakaalu mõõtmiseks ning jälgimiseks, viib tagasiside saamise eesmärgil läbi vestlused tootejuhtidega ning teeb soovitusi parendusteks ja tulevikuplaanideks. Töö tulem võimaldab ettevõttel oma arendusprotsessides parendustegevusi koostada ja seeläbi ka tulemusi parandada ning kuigi lahendus on välja töötatud töös käsitletava ettevõtte vajadusi arvestades, on see kohaldatav ka teistele telekommunikatsiooni valdkonna ettevõtetele.

Lõputöö on kirjutatud eesti keeles ning sisaldab teksti 46 leheküljel, 7 peatükki, 18 joonist, 5 tabelit.

Abstract

Implementation of success indicators for features on the example of Telia Eesti AS - monitoring and problem areas

The aim of this master's thesis is to find out the areas of concern that prevent product managers from setting business goals for features, how to achieve a situation where business metrics are set correctly and those set metrics are monitored, also to provide an opportunity to reliably monitor the percentage of SMART features out of all features.

From the point of view of Telia Estonia's IT development, the share of SMART features is one of the most important indicators, but nevertheless the results of this measure are very low. Metrics are set for features, but they are either not reasonably set or not monitored enough. Therefore, it is important to understand why many product managers still continue to not consume their business metrics to target and monitor the performance of features, and what hinders them from doing so.

As a result of the work, the author compiles a comprehensive report to measure and monitor the share of SMART features, conducts interviews with product managers for feedback, and makes recommendations for improvements and future plans. The result of this work enables the company to prepare improvement activities in its development processes and thereby also improve the results, and although the solution has been developed taking into account the needs of the company discussed in the work, it is also applicable to other telecommunications companies.

The thesis is in Estonian and contains 46 pages of text, 7 chapters, 18 figures, 5 tables.

Lühendite ja mõistete sõnastik

ART	*Agile Release Train*, agiilse toote väljalaske rong
DDD	*Data-driven Development*, andmetel põhinev arendus
DDDM	*Data-driven Decision Making*, andmepõhine otsuste tegemine
Featuur	*Feature*, seatud nõuetele vastav tarkvara funktsionaalsuse tükk, eepiku osa, mis teostatav ühe reliisitsükliga
KPI	*Key Performance Indicator*, tulemuslikkuse põhinäitaja
LSD	*Lean Software Development*, kulusäästlik arendus
MVP	*Minimum Viable Product*, minimaalne elujõuline toode
OKR	*Objectives and Key Results*, eesmärgid ja peamised tulemused
PI	*Program Increment*, tarkvara valmimise protsess
SAFe	*Scaled Agile Framework*, laiaulatuslik agiilse arenduse raamistik
SPC	*SAFe Program Consultant*, SAFe programmi konsultant
TTM	*Time to market*, aeg toote/teenuse arenduse algusest kuni turule jõudmiseni

Sisukord

Jooniste loetelu

Tabelite loetelu

1 Sissejuhatus

Tänapäeval domineerivad agiilsed metoodikad populaarseimate tarkvaraarendusprotsessidena, kuid üldiselt on need kasutuses meeskondade tasemel. Kui tegemist on aga mastaapsete ettevõtetega, siis muutub agiilsete metoodikate rakendamine aina keerukamaks. Selle probleemi lahendamiseks on loodud laiaulatuslik agiilne raamistik (SAFe), mis aitab agiilsust üle terve ettevõtte skaleerida ning tagada, et kõigi osapoolte vajadused on arvesse võetud [1]. Käesolevas töös on uurimisel telekommunikatsioonialane ettevõte Telia Eesti AS, mis on üle tuhande töötajaga üks suurimaid firmasid Eestis. Kui sellises mastaabiga ettevõttes on arendusprojekte, mille puhul on vaja paljude erinevate meeskondade kokku juhtimist ning ühist panust, siis just SAFe aitab tagada parima kvaliteedi, kiired tulemused ning motiveeritud töötajad. Kõnealuse raamistiku juurutamine ettevõttes on pikk protsess, mis nõuab treenimist, juhendamist ning ka juhtide kaasatust protsessi, kuid selle tulemusena on võimalik edukalt saavutada olukord, kus suured ja keerukad arendusprojektid on kõigile osapooltele läbipaistvad, üheselt mõistetavad ning kiirelt ja edukalt lahendatavad.

Magistritöö koosneb seitsmest järgnevast peatükist:

1) Esimeses peatükis kirjeldatakse probleemi ja selle tagamaid ning vaadeldakse töö eesmärki ja skoopi;

2) Teises peatükis antakse ülevaade andmetest, nende põhjal otsuste tegemisest ning vaadeldakse ka erinevaid järgmise peatüki jaoks relevantseid arendusmetoodikaid;

3) Kolmandas peatükis keskendutakse töös käsitletavale ettevõtte Telia Eesti AS, selgitatakse ettevõttes juurutatava laiaulatusliku agiilse raamistiku olemust ja selle kasutuselevõtu põhjuseid ning vaadeldakse täpsemalt selle raamistiku ühte keskset osa - featuure;

4) Neljandas peatükis kirjeldatakse algusest lõpuni tarkade featuuride aruandluse loomist. Ära on kirjeldatud erinevad sammud alates andmete analüüsist ning komplekteerimisest kuni lõpptulemuste visualiseerimise ning järeldusteni;

5) Viiendas peatükis teostatakse tootejuhtidega vestlused, mille käigus selgitatakse välja erinevate meeskondade lähenemine SMART featuuride kasutuselevõtule ja sellega seonduvad probleemkohad ning milles saadakse ka tagasisidet neljandas peatükis koostatud aruandlusest selgunud tulemustele;

6) Kuuendas peatükis käsitletakse tulevikuplaane - ajajoonel vaadeldakse juba toimunud ja ka tulevikuks planeeritavaid tegevusi ning pakutakse välja mõõdikud, mis aitavad SMART featuuride juurutamist jälgida ning parendada;

7) Seitsmendas peatükis tehakse koostatud töö kohta kokkuvõte.

1.1 Probleemipüstitus

SMART featuuride puhul on tegemist äripoole vajadusi täitvate teenustega, millele seatud eesmärgid peavad olema targad - konkreetsed, mõõdetavad, saavutatavad, realistlikud ning ajaliselt piiratud. SMART featuuride suurimaks eeliseks võrreldes tavaliste featuuridega on see, et läbi tarkade eesmärkide seadmise on kõigile osapooltele märkimisväärselt arusaadavam, mida ning kuidas tegema hakatakse.

Telia Eesti IT arenduse vaates on SMART featuuride osakaal kõikidest featuuridest 2021. aasta üks olulisemaid mõõdikuid, kuid selle mõõdiku tulemused on väga madalad. Featuuridele seatakse küll mõõdikuid, kuid need pole kas mõistlikult seatud või ei jälgita neid piisavalt. Seetõttu on oluline aru saada, miks paljud tootejuhid ei tarbi jätkuvalt oma ärimõõdikuid featuuride tulemuslikkuse eesmärgistamiseks ja jälgimiseks ning mis neid selle juures takistab.

1.2 Töö eesmärk

Käesoleva magistritöö eesmärgiks on esmalt uurida, mida tähendab andmetel põhinev arendus ning milliseid erinevaid variante selleks on. Seejärel keskendutakse rohkem ettevõttele Telia Eesti AS ning vaadeldakse sealseid arendusprotsesse. Töö praktilise poole pealt on läbi loodava aruandluse ning intervjuude läbi viimise eesmärgiks välja

selgitada, millised murekohad takistavad tootejuhtidel featuuridele äriliste eesmärkide püstitamist, kuidas saavutada olukord, kus ärilise väärtuse mõõdikud seatakse korrektselt ning peale featuuri kasutusse minekut jälgitakse eelnevalt seatud mõõdikuid, et selgitada välja, kas oodatud tulemus saavutati või oleks vaja muudatusi teha ning tekitada võimalus SMART featuuride osakaalu adekvaatseks jälgimiseks.

1.3 Töös kasutatavad uurimisviisid

Töös kasutatakse probleemi vaatlemiseks ning analüüsimiseks kolme põhilist uurimismetoodikat – teoreetilise informatsiooni uurimine, aruandluse loomine ning tagasiside saamiseks intervjuude läbi viimine. Kõigi kolme meetodi kombineerimine annab tervikliku ning arusaadava tulemuse töös püstitatud probleemist ning selle võimalikest lahendustest.

Teoreetilise informatsiooni uurimine annab põhjalikud teadmised kajastatavast teemast – andmetest ja nendest juhindumisest, erinevatest arendusmetoodikatest ning ettevõttest Telia Eesti AS ja seal kasutatavatest metoodikatest. Saadud teadmised on ka toeks järgnevate uurimismeetodite rakendamisel.

Aruandluse loomise eesmärk on luua kasutatav aruanne SMART featuuride osakaalu mõõtmiseks, mille põhjal on võimalik tuvastada, milline oli algne olukord ning jälgida, kuidas osakaalud ajas muutuvad. Terviklik aruandlus võimaldab ka tulevikus aru saada, kas tehtud parendustegevused aitavad mõõdiku tulemusi tõusvas trendis muuta või oleks vaja meetodeid muuta.

Intervjuude koostamine aitab mõista probleemi, miks tootejuhid ei sea featuuridele ärilisi eesmärke nende tulemuslikkuse mõõtmiseks. Saadud informatsiooni pealt on omakorda võimalik teha järeldusi ning parendussoovitusi tulevikuks.

2 Teoreetiline taust

Antud peatükk uurib, mis on andmed ja kuidas neid äriliste otsuste langetamiseks korrektselt kasutada, mida tähendavad andmetest juhindumine ning andmetel põhinev arendamine. Lisaks vaadatakse levinumaid arendusmetoodikaid, nende tugevusi ja nõrkusi ning mille põhjal erinevad metoodikad omavahel eristuvad.

2.1 Andmed, andmetest juhindumine ja andmetel põhinev arendus

Mõistmaks, kuidas andmeid arendusprotsessides efektiivselt kasutada, on kõigepealt vaja aru saada, mis andmed üldse on ning mida tähendab andmetest juhindudes otsuste tegemine. Antud peatükis on kõigepealt uuritud mõlemat nimetatud teemat ning seejärel selgitatud välja, mida ikkagi tähendab andmetel põhinev arendus.

2.1.1 Andmed

Cambridge Dictionary definitsiooni järgi on andmed „informatsioon, eriti faktid või numbrid, mis on kogutud uurimiseks ja arvesse võtmiseks ning mida kasutatakse otsuste tegemisel, või teave elektroonilises vormis, mida arvuti saab salvestada ja kasutada“ [1]. Andmed jagunevad kvalitatiivseteks ehk kirjeldavaks ja kvantitatiivseteks ehk numbriliseks informatsiooniks. Kvantitatiivsed andmed jaotuvad omakorda veel diskreetseteks ehk mittepidevateks ning pidevateks. Mittepidevad andmed on loendatavad, näiteks täisarvulised numbrid, pidevad andmed aga mõõdetavad, näiteks kaal või pikkus [2].

2.1.2 Andmepõhine otsuste tegemine

Andmetel põhinev lähenemine (*data-driven decision making)* erinevatele protsessidele tähendab seda, et ettevõte teeb strateegilisi otsuseid põhinedes andmetel, nende analüüsil ning tõlgendamisel. See võimaldab ettevõttel läbi andmete uurimise ning organiseerimise klientidele paremat ning ka kliendikesksemat teenust pakkuda. Kui ettevõtte mõistab oma andmete väärtust ning neid efektiivselt ära kasutab, on kõigil töötajatel võimalik andmete põhjal paremaid otsuseid langetada [3].

Andmete täieliku potentsiaali ära kasutamiseks tuleb aga ettevõttes luua sobilik kultuur, mille edukaks saavutamiseks on mõned olulised sammud. Esiteks on vajalik, et soov andmetest juhinduda tuleks ettevõtte tipust. See tähendab, et ettevõtte juhid ning juhtivatel positsioonidel olevad töötajad peavad näitama eeskuju ning tegema otsuseid, mida toetavad vajalikud andmed. Teiseks on oluline kasutada andmeanalüüsi ettevõtte töötajate aitamiseks, mitte vaid klientide ja nende parema kogemuse tagamiseks. Sealhulgas tuleb tähelepanu pöörata ka sellele, et vajalikud andmed oleksid ettevõtte siseselt neid kasutavatele töötajatele lihtsasti kättesaadavad. Üheks oluliseks punktiks on ka õigete ning kasulike mõõdikute välja töötamine ning terviklike andmete kogumine. Selleks on vajalik, et oleks selge, milliseid algandmed on vaja hoiustada ning mida on kõige mõistlikum eesmärgistada ning mõõta, et jälgitavatest tulemustest ka reaalselt kasu oleks. Samuti on tähtis olla paindlik, selgitada erinevate analüütiliste otsuste tagamaid ning olla valmis muudatusteks [4].

Kui ettevõte suudab luua keskkonna, kus andmetest juhindumine pole mitte vaevaks, vaid pigem igapäevane tava, siis on võimeline langetada kiiremaid ning teadlikumaid otsuseid. See omakorda aitab kaasa nii ettevõtte tulemuste parandamisele ning ka töötajate kaasatusele ning koostööle.

2.1.3 Andmetel põhinev arendus

Eelneva info põhjal on võimalik järeldada, et andmetel põhinev arendus (*data-driven development*) on teenuste ja toodete arendamine põhinedes eelnevalt kogutud informatsioonil. See tähendab, et iga arendus peaks olema väärtust loov ning täpselt ja kvantitatiivselt eesmärgistatud.

On mõned olulised suunised, millest juhinduda, et arendusprotsessides edukalt ning efektiivselt andmekasutust rakendada. Kõigepealt peaksid enamus KPI-sid (*Key Performane Indicator*) olema raamatupidamisfunktsiooniga. See tähendab, et enamus töötajaid saavad sama mõõdetava eesmärgi, mille poole kõik ühiselt püüdlevad. Erandid on suurte ettevõtete puhul, kus saab erinevatele üksustele oma mõõdikud seada. Teiseks, tuleks OKR-idega (*Objectives and Key Results*) eksperimenteerida ning vajadusel skaleerida. Tuleks luua tiimidele oma mõõdikud, mida jälgitakse ning regulaarsete ajavahemike tagant üle vaadatakse ning vajadusel kohendatakse. Järgmiseks oluliseks punktiks on arendusmõõdikute jälgimine. Siinkohal on oluline, et arendajatel oleks mõned reaalsed numbrilised eesmärgid, mis aitavad neil oma töö tulemuslikkust mõista,

leida probleemkohti ning arenguvõimalusi. Viimaseks tähtsaks soovituseks on positiivsete käitumismõõdikute jälgimine. Väga oluline on, et töötajad oleksid õnnelikud, kuna seeläbi on ka nende töötulemused kvaliteetsemad. Seetõttu on tähtis meeskonnaliikmete moraali hoidmine ning jälgimine. Üheks lihtsaks mooduseks on näiteks automaatsete positiivsete teadete saatmine ning näiteks läbi küsimustike uuride, milline on töötajate meelestatus [5]. Kõigi eelnevalt mainitud punktide puhul langetatakse otsuseid andmetest juhindudes ning seeläbi muudetakse töökorraldust efektiivsemaks ning töötajaid motiveeritumaks.

2.2 Arendusmetoodikad

Tarkvaraarenduse mudelid erinevad üksteisest sõltudes eesmärgist, ettevõtte suurusest ja muudest piirangutest. Õige arendusmudeli või -metodoloogia valik on väga oluline, et tagada edukas arendus ning loodud toodete ja teenuste rakendamine.

2.2.1 Agiilsed metoodikad

Kuigi erinevaid agiilseid meetodeid kasutati juba palju varasemalt, siis 2001. aastal kohtusid seitsetest tarvakaraarendajat, kelle eesmärk oli leida ühiseid seoseid erinevate metoodikate vahel. Kohtumise tulemusena avaldasid nad enda seniste kogemuste põhjal kirja pandud agiilse arenduse manifesti *„Manifesto for Agile Software Development“*, kus toodi välja agiilse arenduse neli põhiväärtust:

- Inimesed ja suhtlemine on olulisemad kui protsessid ja tööriistad;
- Töötav tarkvara on olulisem kui põhjalik dokumentatsioon;
- Klientidega koostöö on olulisem kui lepinguläbirääkimised;
- Muutustele reageerimine on olulisem kui kindla plaani järgimine [6].

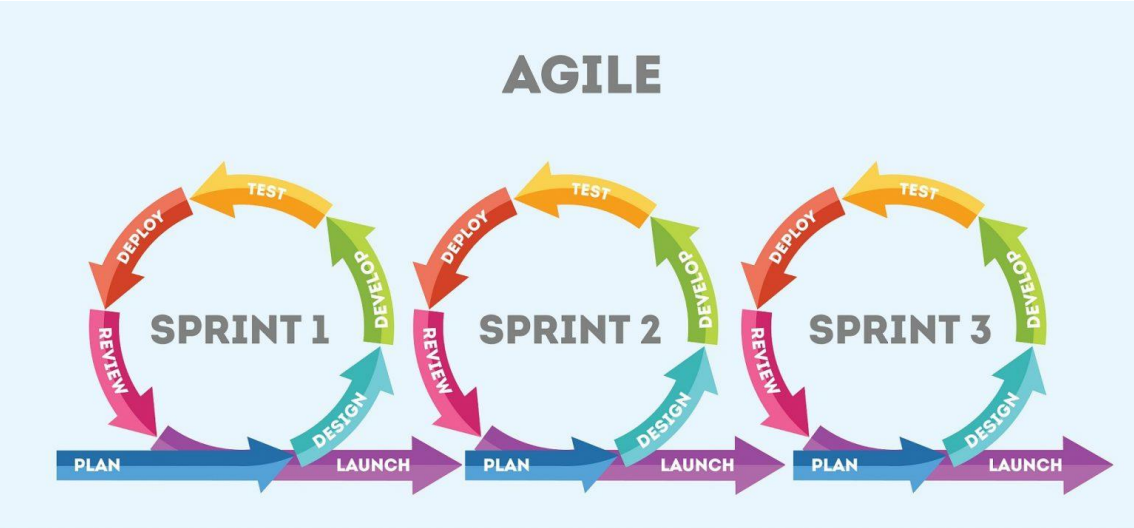
Neid nelja väärtust toetavad kaksteist printsiipi, mis samuti manifesti kirja pandi:

- Kõige tähtsam prioriteet on rahuldada klienti väärtusliku tarkvara varajase ja pideva tarnimisega;
- Tervitada muutuvaid nõudeid, isegi arenduse hilises staadiumis. Väledad protsessid rakendavad muutusi kliendi konkurentsieeliseks;

- Edastada töötavat tarkvara sageli, vahemikus paarist nädalast kuni paari kuuni, eelistades lühemat ajakava;
- Äripool ning arendajad peavad kogu projekti vältel igapäevast koostööd tegema;
- Ehitada projekte motiveeritud inimeste ümber, pakkuda neile vajalikku keskkonda ja tuge ning usaldada, et nad saavad oma tööga hakkama;
- Kõige tõhusam viis arendustiimis ja arendustiimile teabe edastamiseks on näost näkku vestlemine;
- Töötav tarkvara on kõige olulisem arengu mõõdik;
- Agiilsed protsessid soodustavad säästvat arengut. Sponsorid, arendajad ja kasutajad peaksid suutma lõputult ühtlast tempot hoida;
- Järjepidev tähelepanu tehnilisele suurepärasusele ja heale kujundusele suurendab agiilsust;
- Lihtsus, kui tegemata tööde koguse maksimeerimise kunst, on hädavajalik;
- Parimad arhitektuurid, nõuded ja kujundused tulevad isekorraldavatest meeskondadest;
- Regulaarsete ajavahemike järel mõtiskleb meeskond selle üle, kuidas muutuda tõhusamaks ning seejärel häälestab ja kohandab oma käitumist vastavalt. [7]

Nende väärtuste ja printsiipide laialdase kasutuselevõtu tulemusena tekkisid ühtsed arusaamad agiilsetest praktikatest.

Definitsiooni järgi hõlmavad agiilsed (ka väledad) tarkvaraarenduse praktikad endas nõuete avastamist ja lahenduste väljatöötamist iseorganiseeruvate ning ristfunktsionaalsete meeskondade ja nende klientide koostöös. Agiilsed praktikad toetavad adaptiivset planeerimist, evolutsioonilist arendamist, varajast tarkvara edastamist ja pidevat täiustamist ning julgustavad muutustele paindlikult reageerima [8]. Agiilse tarkvaraarenduse põhimõtete järgi toimub tegevus iteratiivselt ehk sama protsessi tehakse läbi korduvalt ja järjepidevalt (Joonis 1).



Joonis 1. Agiilse tarkvaraarenduse elutsükkel [9].

Esimeseks sammuks on planeerimise faas, kus määratakse ära, milline on probleem ning analüüsitakse olukorda, eesmärke ja nõudeid. Peale põhjalikke analüüse on aeg ära kirjelda kujunduse funktsioonid. Sinna alla kuuluvad ka näiteks ärireeglid ja kogu dokumentatsioon. Kujunduse faasi tulemusena peaks olema kõik algselt sisendina saadud nõuded detailselt ja põhjalikult kujunduselementidena kirjeldatud, sealhulgas on tavapäraselt dokumentatsioonis olemas ka erinevad diagrammid ja tabelid, mis aitavad nõudeid arusaadavalt väljendada. Kui kogu funktsionaalsus ja kujundus on põhjalikult ning üheselt kirja pandud, siis on arendajal võimalik nõuetele vastav toode luua. Kuigi teatav arendatava toote testimine toimub juba selle loomise käigus, siis peale arendusfaasi on järg põhjaliku testimise juures. Kuigi kunagi ei ole võimalik absoluutselt kõiki olukord läbi testida, siis on ikkagi äärmiselt oluline, et testimine oleks põhjalik ja detailne, et vältida kliendile vigase toote väljastamist. Järgmisena toimub toote lõplik ülevaatus peale mida saabki selle juba kliendile väljastada. Seejärel algab sama tsükkel kas uue funktsionaalsuse või tootega uuesti peale.

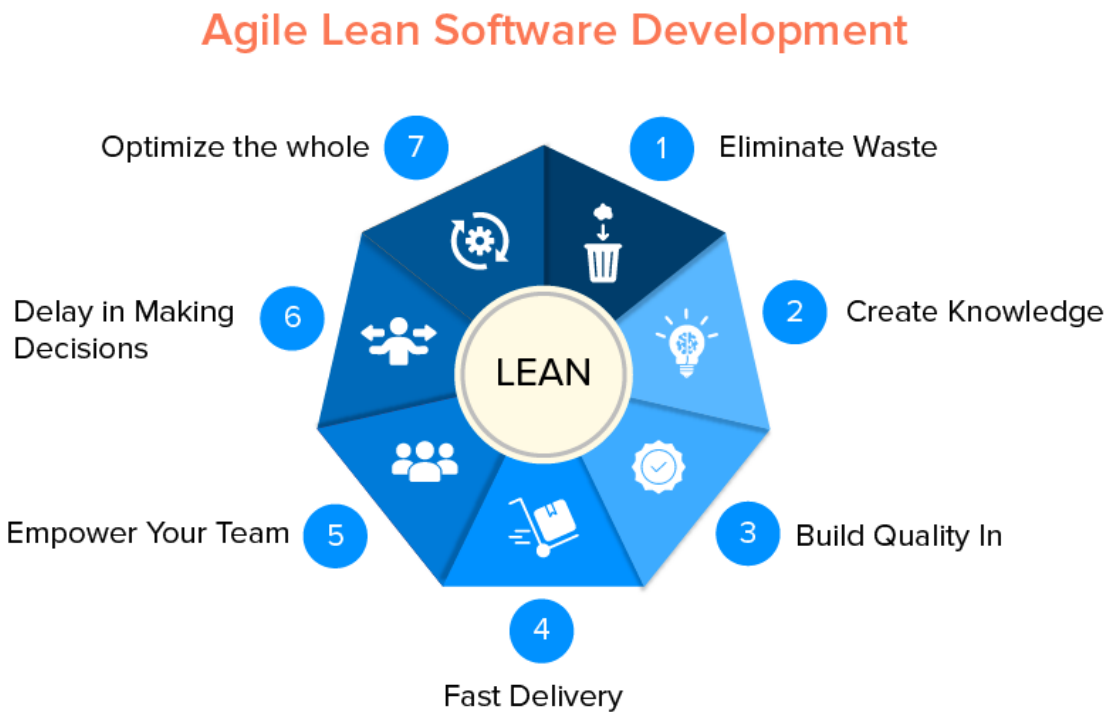
Agiilsel tarkvaraarendusel on mitmeid plusse ja miinuseid. Positiivse poole pealt võib välja tuua näiteks kiirelt kliendile kasuliku ning toimiva tarkvara kätte toimetamise, tiheda ja regulaarse suhtluse äripoole ja arendajate vahel ning võimekuse muutuvate nõuete ja olukordadega edukalt kohaneda. Kuna agiilse arenduse eesmärk on kiirelt toimivaid tooteid kliendile väljastada, siis negatiivse poole pealt on kõige probleemsemateks kohtadeks tihtipeale ebapiisav dokumentatsioon, kuna puudub selle jaoks vajalik aeg ning mõistmine, kui suurt pingutust mingi tükk lõppkokkuvõttes vajab.

Agiilsus toetab küll jooksvaid muudatusi ning nõuete muutumist, aga see omakorda võib tähendada, et projekt kaldub algsest suunast kõrvale.

Agiilse arendusmetoodika alla kuulub mitmeid erinevaid raamistikke, millest on võimalik vastavalt nendevahelistele erinevustele välja valida ettevõtte või tiimi jaoks kõige sobivam. Käesolevas töös vaatame mõne enimtuntuma raamastiku omadusi.

2.2.2 Lean Software Development

Lean tarkvaraarendus on agiilne raamistik, mille põhimõteteks on optimeerida arendusaega ja -ressursse, elimineerida raiskamist ning lõpptulemusena arendada ainult seda, mida toode reaalselt vajab. *Lean* lähenemisele viidatakse tihti ka kui MVP (*Minimum Viable Product*) strateegiale, mille puhul meeskond arendab tootest välja kõige algelisema versiooni, annab selle kasutusse ning seejärel õpib klientide tagasiside pealt, mis neile meeldib, mis mitte ja mida muuta ning teeb selle põhjal parendused [10]. *Lean* arenduse puhul on seitse põhilist printsiipi, millest lähtuda, et arendusprotsessi kiirendada ning kasutajatele väärtust luua (Joonis 2).



Joonis 2. *Lean* arenduse elutsükkel [11].

Esimeseks ning kõige olulisemaks printsiibiks on arendusprotsessist raiskamise kõrvaldamine ehk tuleb vabaneda kõigest, mis ei loo kliendile või lõppkasutajale väärtust. Kui identifitseerida loodava toote väärtus, siis on võimalik tuvastada ka need probleemsed kohad, mis ei aita selle väärtuse loomisele kaasa. Kõige tavalisemad näited raiskamisest on ebaselged eesmärgid, duplitseeritud andmed, ebaefektiivne omavaheline suhtlus ning ebavajalik kood. Teine printsiip *Lean* arenduse puhul on teadmiste loomine ehk informatsiooni dokumenteerimine, koolitustel osalemine, tehtud töö ülevaatamine ja kommenteerimine, töötajate vaheline tarkuse jagamine ja muu sarnane. Järgmine oluline punkt on sisemise kvaliteedi loomine, tänu millele on võimalik vältida vigu, mille parandamine kulutab nii aega kui ka raha. Kogu protsessi vältel kvaliteedi tagamisele aitavad kaasa näiteks paarisprogrammeerimine, mille puhul kahe inimese kombineeritud teadmised ja oskused aitavad efektiivsemaid lahendusi leida, automatiseerimine, mis vähendab inimlike vigade kogust ning testimine, et tagada arendatavate toodete korrektne toimimine. Samuti väga oluliseks osaks *Lean* arenduse juures on kiire toote kätte toimetamine kliendile. See tähendab, et eesmärk on kiirelt MVP valmis saada, see kliendile toimetada ning seejärel tagasiside pealt uusi lisaarendusi teha. Selline toimimine aitab ära hoida ebamõistlikku ajakulutust kõiksugude lisade peale, mida võib-olla tegelikult vaja polegi. Viiendaks printsiibiks on tiimi võimendamine ja austamine – inimestel on vastavalt oma võimetele võimalik ise otsuseid langetada ning omavahelises suhtluses pole keegi kamandav ega üleolev. Töötajatele on oluline, et neil töökeskkond ja üldine suhtlus oleksid meeldivad ning seeläbi on nad ka motiveeritud efektiivselt töötama ning häid tulemusi saavutama. Järgmiseks punktiks *Lean* arenduse elutsüklis on otsuste langetamisega viivitamine. Võib tunduda, et aja kulutamine on vastupidine *Lean* põhimõtetele, kuid antud juhul on oluline tutvuda nõuetega ning jätta ajaline vabadus soovide muutumiseks või murekohtade leidmiseks. Kui olulisi otsuseid langetatakse ilma süvenemata, siis võib selle tulemusena projekt lausa ebaõnnestud, kuna nõuded ning vajadused võivad olla muutunud. Viimaseks printsiibiks on optimeerimine. Siinkohal tuleb vaadata suurt pilti, kogu arendusprotsessi suuremalt, et tuvastada parenduskohti, mis alamprotsessides tähelepanuta võivad jääda [12], [13], [14].

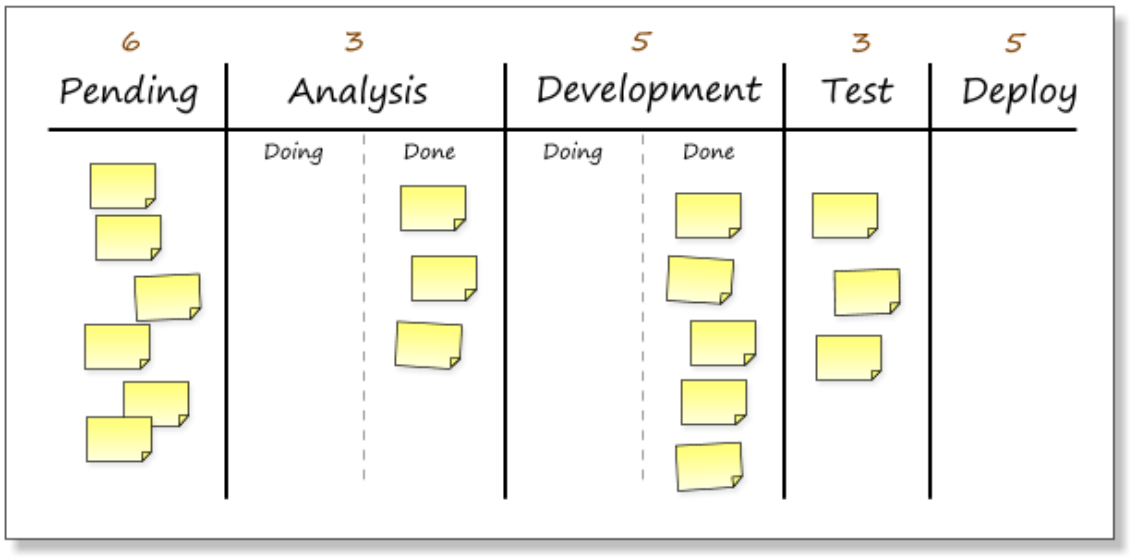
Eelnevate printsiipide põhjal on võimalik järeldada, et põhilisteks *Lean* arenduse tugevusteks on raiskamise vältimine, mis tagab arendusprotsessi efektiivsuse, kaasatud ja motiveeritud meeskond ning kiire ja varajane toote toimetamine kliendile. Negatiivse poole pealt võib aga kogu protsessi kiirus probleeme tekitada. Kui programmeerijad

näiteks nõuetest valesti aru saavad ja kohe nende põhjal toodet arendama hakkavad, siis võib valmida täiesti ebavajalik tulemus. Selle tõttu on samuti oluline, et meeskond oleks kompetentne, kuna jooksvalt juurde õppimiseks jääb vähe aega. Lisaks ei tohi ajalise survega ka liiale minna, kuna niimoodi võivad tekkida pudelikaelad ja vähenenud töötajate ressursi puhul tekivad suured järjekorrad. Seetõttu peaks alati valmis olema ka plaan ootamatute olukordade lahendamiseks.

2.2.3 Kanban

Kanban on raamistik agiilse ja DevOps tarkvaraarenduse rakendamiseks, mis nõuab reaalajalist infovahetust võimekuse kohta ning töö täielikku läbipaistvust. Töö tükid on kanbani tahvlil (ingl.k *Kanban board*) visuaalselt esindatud, võimaldades meeskonnaliikmetel igal ajahetkel näha iga individuaalse töö olekut [15]. Kanban on jaapanikeelne sõna, mille tähendus on otsetõlkes küll „signaalikaart“, kuid tegelikult ei ole kanbani tahvlil olevad sildid mitte signaalideks, vaid nad väljendavad lihtsalt töö erinevaid osasid [16].

Kanban *board* (Joonis 3) on tööriist, mis aitab tööd visualiseerida, poolelioleva töö kogust piirata ning efektiivsust maksimeerida. Selline tahvel aitab nii agiilsetel kui ka DevOps tiimidel igapäevases töös korda luua [17].



Joonis 3. Kanban *board* [18].

Kanban tahvel koosneb sildikestest ning tulpadest, mille vahel neid silte liigutatakse. Tahvli edukaks kasutamiseks jaotab tiim kõigepealt kogu oma töö juppideks, kus iga silt väljendabki ühte konkreetset töö tükki (nt. kasutajalugu) ning peale seda pannakse sildid

tahvlile. Tahvel on omakorda jaotatud tulpadeks, kus iga tulp on üks töövoo osa. Kanban tahvli eesmärgiks on jooksvalt töö käigus sildikeste liigutamine erinevate olekute vahel ning jälgida, et kõik töö tükid oleksid alati õiges tulbas ning et ükski silt ei jääks liiga pikaks ajaks ühele kohale seisma. Põhiline omadus, mis kanban tahvlit teistest visuaalsetest tahvlitest eristab, on iga tulba kohal olev maksimaalsete lubatud tööde arv. Igas staadiumis olevate tööde arvu piiramine aitab keskenduda konkreetsete tööde lahendamisele ja seeläbi vältida üle tootmist ning samuti tulevad selle abil välja probleemkohad, kus töö seisma jääb. Näiteks, kui arenduse staadiumis on juba maksimaalne lubatud arv töid, siis eelnevatest tulpadest ei ole enam võimalik töid edasi liigutada ning tekivad pudelikaelad. Seeläbi tulevad väga kiiresti välja probleemkohad kogu arendusprotsessis ning on võimalik nende parendamisega tegelema hakata [18].

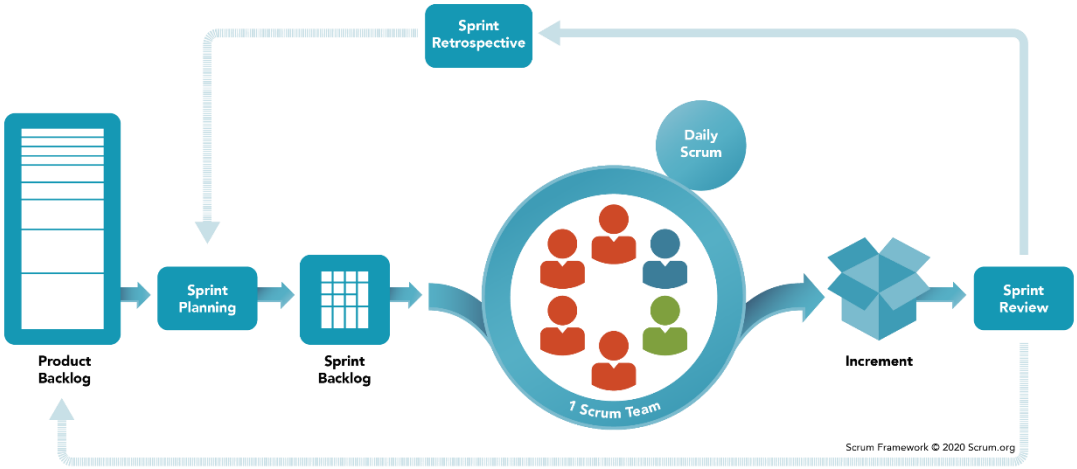
Oma 2010. aastal välja antud raamatus „*Kanban: Successful Evolutionary Change for Your Technology Business*“ kirjeldab David J. Anderson kanban meetodi arengut ning toob välja ka viis põhilist omadust, mis on vajalikud, et edukalt kanban meetodit rakendada:

1. Töövoo visualiseerimine;
2. Poolelioleva töö koguse piiramine;
3. Töövoolu mõõtmine ja haldamine;
4. Protsessipoliitika selgesõnaliseks muutmine;
5. Mudelite kasutamine parendusvõimaluste tuvastamiseks [16].

Kokkuvõtvalt on põhilisteks kanban meetodi eelisteks pudelikaelade kiire tuvastamine ning paindlikkus tahvlil olevaid töötükke vastavalt muutuvatele vajadustele ümber järjestada. Samuti on kanban tahvlit kasutades kõik meeskonnaliikmed töödest ning nende olekutest samal arusaamal. Vastupidiselt negatiivse poole pealt peavad aga kanbani kasutades kõik jälgima, et nad tahvlil olevaid töötükke alati õigeaegselt staatuste vahel liigutaksid, kuna vastasel juhul ei kajasta tahvel enam korrektset seisu, mille tulemusena võivad tiimiliikmetel tekkida erinevad arusaamad tööde olekutest. Samuti on kanbani miinuseks ajagraafikute puudumine, mis raskendab ajalist planeerimist ning samuti ei ole seetõttu teada, millal mingisugune töö tükk järgmisesse tulpa võiks liikuda [19].

2.2.4 Scrum

Scrum on agiilne raamistik, mis aitab ristfunktsionaalsetel meeskondadel keerukatele probleemidele läbi kohandatud lahenduste väärtust luua. Meeskonda kuuluvad antud raamistiku puhul üks *Scrum Master*, üks tootejuht ning arendajad. Kõige olulisemal kohal Scrum meeskonnas on *Scrum Master*, kes vastutab kogu meeskonna efektiivsuse eest, ning kelle ülesandeks on võimaldada tiimil areneda. Tooteomanik vastutab meeskonna poolt loodava toote väärtuse maksimeerimise eest ning arendajad on need, eks läbi läbi toote loomise arendusprotsessis reaalset väärtust loovad [20]. Scrum meeskonna töö toimub sprintides, mis on kindla pikkusega (üldiselt 1-4 nädala pikkused) järkjärgulised iteratiivsed tööjärjestused (Joonis 4).



Joonis 4. Scrum raamistiku sprint [21].

Regulaarse sprintide tsükli edukaks toimimiseks peab kogu aeg eksisteerima tootejuhi poolt tellitavate tööde järjekord (*Product Backlog*). Iga sprindi alguses toimub planeerimine (*Sprint Planning*), mille jooksul meeskond lisab sellest nimekirjast kõige olulisemad tööd sprindi tööde järjekorda (*Sprint Backlog*). Seejärel toimub arendusprotsess, mille puhul kasutatakse tüüpiliselt tööde üle järje pidamiseks Kanban *board*'iga analoogset tahvlit – iga töö jaoks on oma sedel, mida seejärel staatuste vahel liigutatakse. Scrum raamistiku üheks oluliseks osaks on ka kogu sprindi vältel toimuvad igapäevased kohtumised, kus iga meeskonnaliige annab teada, mis tal tehtud on, mis plaanis on ning millised probleemid töös esinevad. Sprindi jooksul tehtud tööde tulemuseks on kliendile uue väärtuse loomine, näiteks tarkvarast uue ja parema versiooni või mine olulise lisa näol. Viimasteks sammudeks sprindis on kokkuvõte (*Sprint Review*)

ning tagasivaade (*Sprint Retrospective*), mille jooksul saab meeskond vaadata, mis tehtud sai, kuidas läks ning mida saaks muuta või parandada.

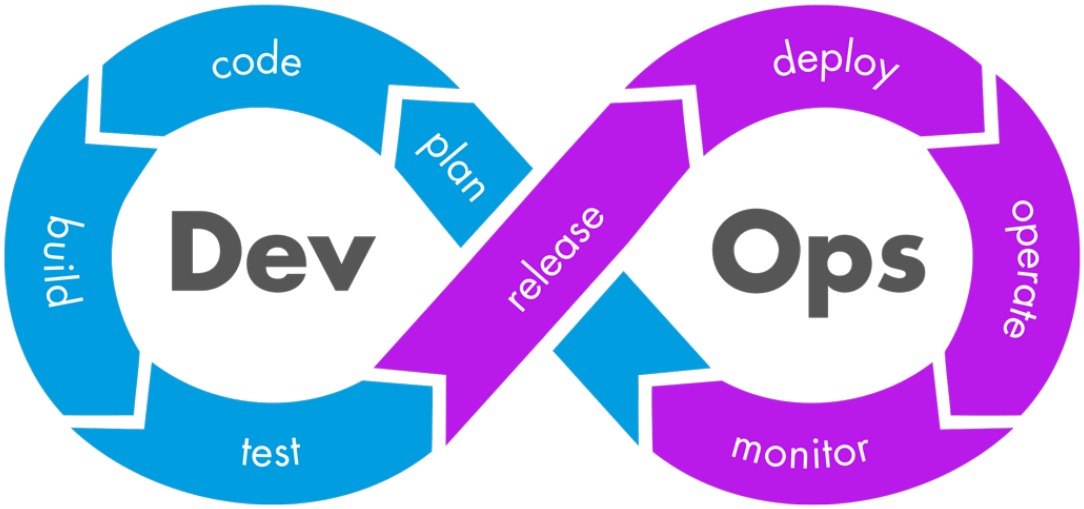
Scrum metoodika edukaks toimimiseks peab seega *Scrum Master* arendama ning säilitama keskkonda, kus:

- Tooteomanik tellib keeruka probleemi lahenduseks vajaliku töö tööde järjekoda;
- Meeskond loob sprindi jooksul valitud tööde kaudu lisaväärtust;
- Meeskond ja huvitatud osapooled uurivad töö tulemusi ning häälestuvad järgmiseks sprindiks;
- Tsükkel kordub [20].

Scrumi põhilisteks tugevusteks on seega kindel ajagraafik, mille lõpuks kliendile ka väärtust luuakse, regulaarsed meeskonnaliikmete kohtumised, mis aitavad tööde seisust väga head ülevaadet hoida ning kokkuvõtted ja tagasivaated sprindi lõpus, tänu millele on meeskonnal võimalik areneda ning parendusi teha. Üheks põhiliseks mureks võib antud raamistiku puhul saada *Scrum Master*'i roll – kui *Scrum Master* ei toeta ega abista tiimi piisavalt või ei oma vajalikku kompetentsi, siis võib kogu protsessis palju probleeme tekkida ning meeskond ei pruugi saavutada soovitud tulemusi. Puudusteks võivad saada ka koosolekute rohkus, mille arvelt võib reaalse arendustöö maht kannatada ning kogu meeskonna kompetents – kuna Scrum tiimid on üldiselt pigem väiksemapoolsed, siis on oluline, et iga meeskonnaliige omaks igaks sprindi staadiumiks vajaminevaid teadmisi ning oskaks neid sobivalt rakendada.

2.2.5 DevOps

DevOps on praktikate kogum, mille eesmärgiks on arendus- ja IT tiimide vaheliste protsesside ühendamine ja automatiseerimine, et nad oleksid võimelised tarkvara kiiremini ning usaldusväärsemalt arendama, testima ning kasutusse andma [22]. Termin DevOps moodustub mõlema eelnevalt mainitud osapoole – arendus (*development*) ja käitus (*operations*) – nimetuste kombineerimisest. Kuna DevOps põhimõtete järgi töötades on arendus- ning IT tiimid tihedas suhtluses ning koostöös (Joonis 5), siis paraneb seeläbi ka tarkvaraarenduse kiirus ning kvaliteet.



Joonis 5. DevOps ülesehitus [23].

DevOps töövoog koosneb kuuest põhilisest faasist Esimeseks sammuks on planeerimine, mille jooksul koostatakse toote arendamise järgmise iteratsiooni ajakava ning tagatakse, et kogu meeskond mida ning millises ajaplaanis tehakse. Järgmisteks faasideks on toote arendamine, testimine ning seejärel ka töökeskkonda paigaldamine. Kui toode on juba kliendi poolt kasutusse võetud, siis on olulised ka kliendile toote uuenduste tarnimine ning tarkvara jõudluse jälgimine ning logimine, et tagada toote stabiilne toimimine ning kliendirahulolu. Viimaseks faasiks on seejärel kliendilt tagasiside kogumine, et selle põhjal parendusi ning muudatusi teha [24].

DevOps parimad praktikad on:

- Järjepidev integratsioon;
- Järjepidev kättetoimetamine;
- Mikroteenused;
- Infrastruktuur koodina;
- Jälgimine ja logimine;
- Suhtlus ja koostöö [25].

Eelneva kirjelduse põhjal on üheks põhiliseks DevOps meetodi tugevuseks meeskonna efektiivsus, kuna toimub tihe suhtlus tiimiliikmete vahel, kes kõik tehtava töö eest

võrdselt vastutust jagavad. See omakorda loob meeskonnas produktiivse ning ühtse keskkonna. Samuti on tänu järjepidevale integratsioonile ja automatiseerimise tööriistade kasutamisele ka toote turule jõudmise aeg lühem. Vastupidise poole pealt on üheks DevOps nõrkuseks riskantsemad toote kasutuskeskkonda paigaldamised, kuna arendus ja paigaldus toimub väga kiire protsessina. Teiseks probleemiks on kallid tööriistad – et DevOps saaks efektiivselt toimida, on vaja kasutada erinevaid tarkvara testimise ning paigaldamise tööriistu, samuti monitooringu ning logimise süsteeme [26], [27].

3 Hetkeolukorra kirjeldus ettevõttes Telia Eesti AS

Käesolevas peatükis kirjeldatakse ettevõtte Telia Eesti AS tausta ning uuritakse ettevõttes kasutuselolevaid arendusmetoodikaid. Arendusmetoodikate peatükis vaadeldakse täpsemalt SAFe metoodika juurutamist, featuure ning mis teeb featuurid targaks.

3.1 Ettevõtte kirjeldus

Telia Eesti AS on üks Eesti suurimaid telekommunikatsioonifirmasid, mis asutati 1993. aastal ning kuulub ettevõttele Telia Company AB. Telia Eesti AS nimetust kannab ettevõtte alates 2016. aastast, varasemalt on tegutsetud ka nime AS Eesti Telekom all, mille tütarettevõteteks olid Elion Ettevõtted AS ja AS EMT. [28] 2020. aasta lõpu seisuga töötas ettevõttes umbes 1600 töötajat [29].

Ettevõtte poolt peamisteks pakutavateks teenusteks on TV, internet ja mobiilsed teenused ning lisaks on võimalik kliendil ka mitmeid erinevaid lisateenuseid tellida. Nimetatud teenuste müümisega on ettevõte jõudnud kolme Eesti suurima telekommunikatsioonieettevõtte hulka. Telia Eesti AS loobki klientidele väärtust läbi

Telia Eesti AS ise positsioneerib ennast turul kvaliteediliidriks [30], mida on tõestanud ka 2020. aastal toimunud rahvusvahelise konsultatsioonifirma Omnitele mõõtmised, mille käigus selgus, et kohalike operaatorite võrkudes kõneside kvaliteeti mõõtes saavutati parimad tulemused Telia 4G võrgus tehtavate kõnede puhul [31].

Lisaks kvaliteetsete teenuste pakkumisele on Telia Eesti eesmärgiks ka vastutustundlik äritegevus, mida tõestab asjaolu, et ettevõttele väljastati Vastutustundliku Ettevõtluse Foorumi poolt ka kõrgeima ehk kuldtaseme kvaliteedimärgis. Antud tunnustus määratakse ettevõtetele tunnustuseks ausa, teadliku ja keskkonda säästva tegevuse eest, kes peavad oluliseks ettevõtte jätkusuutlikku arengut ja panustavad strateegiliselt sotsiaalse- ja looduskeskkonna arengusse [32].

Telia Eesti missiooniks on panustada ka Eesti ühiskonna arengusse – näiteks avas Telia esimese Eesti mobiilioperaatorina oma klientidele 5G võrgu, millega kaasnevad eelised aitavad luua aina edukamat digiühiskonda ning ühtlasi suurendada konkurentsivõimet globaalsel turul [33]. Ettevõte müüb ja rendib klientidele kasutatud seadmeid, kuna uue asemel kasutatud nutiseadme ostmine hoiab ära süsihappegaasi teket. Ökoloogilise

jalajälje vähendamiseks on Telia Eesti oma autoparki elektriautode vastu välja vahetamas ning päikseenergiale toetuvaid andmekeskuseid asustamas. Lisaks alustati koostöös tehnoloogiaettevõttega Cleveron iseteeninduses induste testimist ning korraldatakse digitaalset koristuspäeva, et vähendada digitaalset jalajälge [34].

3.2 Arendusmetoodikad ettevõttes Telia Eesti AS

Antud peatükis kirjeldatakse, milliseid arendusmetoodikad on ettevõttes Telia Eesti AS varasemalt rakendatud ja mida kasutatakse praegu. Lisaks vaadatakse, millest on sellised valikud tingitud olnud ning mida tehakse, et valitud metoodikaid ka edukalt rakendada.

Üldiselt kasutatakse ettevõttes Telia Eesti AS Scrum arendusmetoodikat, mis omakorda tarbib DevOps loogikat. See tähendab, et kõik haldusega seotu on toodud küll arendusele lähedale ning toimub tihe koostöö, aga halduse pool ei ole otseselt arendustiimide sisse pandud. Selle tulemusena on tekkinud olukord, kus tiimid teenindavad end ise – nad paigaldavad näiteks uusi arendusi test- või *live* keskkondadesse, teostavad iseendale monitooringut jne.

3.2.1 SAFe ning selle rakendamine

Ettevõttes Telia Eesti AS on alates 2018. aastast juurutamisel ka laiaulatuslik agiilne raamistik, mis on suurtele ettevõtetele suunatud versioon agiilsest arendusmetoodikast – SAFe (*Scaled Agile Framework*), versioon 5. SAFe kombineerib *Lean*, agiilse ja DevOps arenduse metoodikad arusaadavaks süsteemiks, mis aitab ettevõtetel kasvada, pakkudes innovaatilisi tooteid ja teenuseid kiiremini, parema prognoositavusega ning kvaliteetsemalt [35].

SAFe vundament koosneb kuuest põhilisest elemendist:

- *Lean*-agiilsed juhid – kuna juhtkond on see, kes vastutab äritulemuste saavutamise eest, siis on oluline, et juhid oleksid kursis ning teadlikud SAFe põhimõtetest ning agiilsetest arendusmetoodikatest;
- Põhiväärtused – neljaks SAFe põhiväärtuseks, mis defineerivad SAFe veendumuste- ja väärtussüsteemi, on kohandamine, sisseehitatud kvaliteet, läbipaistvus ning programmi täitmine;

- *Lean*-agiilne mõtteviis – see on kombinatsioon SAFe metoodikaga tegutsevate inimeste agiilsetel ning *Lean* printsiipidel põhinevatest uskumustest, arvamustest, suhtumistest ning tegudest;

- SAFe printsiibid – SAFe tavad põhinevad printsiipidel, mis kombineerivad agiilseid meetodeid, *Lean* tarkvaraarendust, DevOps-i, süsteemset mõtlemist ning pikaaegseid praktilisi kogemusi;

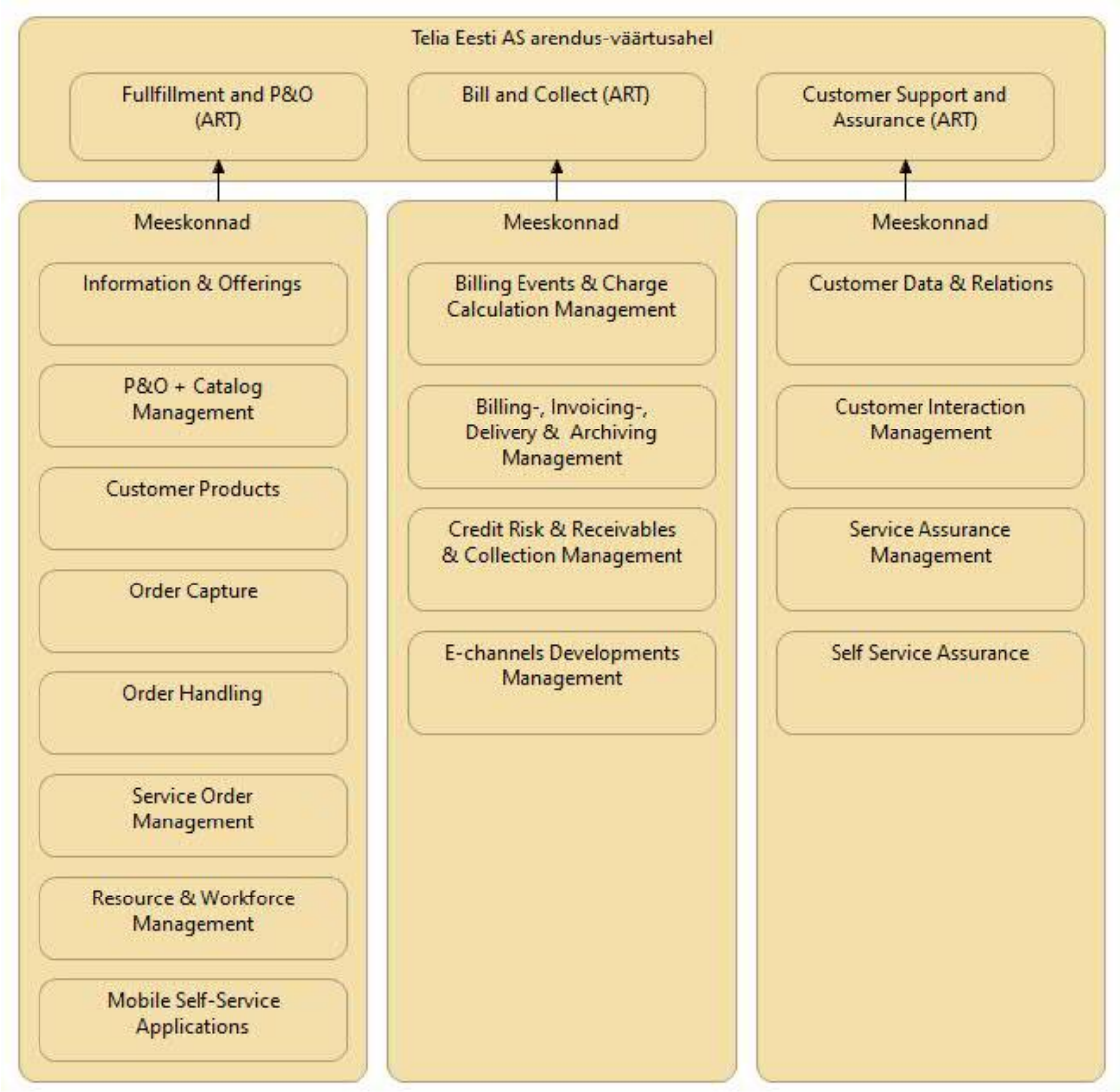
- Implementeerimise plaan – SAFe pakub implementeerimise juhiseid, et abistada ning juhendada ettevõtteid suurte muudatuste tegemisel, mis on vajalikud, et saavutada *Lean*-agiilsed tehnoloogiad;

- SPC-d (*SAFe Program Consultant*) – SAFe programmi konsultantide näol on tegemist isikutega, kes kombineerivad oma tehnilised teadmised loomuomase motivatsiooniga parendada ettevõtte tarkvara- ning süsteemiarenduse protsesse [36].

SAFe juurutamise vajadus tekkis ettevõttes Telia Eesti AS olukorras, kui üritati hakata vananenud müügi ja tarne platvormi uue vastu välja vahetama. See tähendas, et uue süsteemi loomiseks oli vaja paljude erinevate meeskondade panust. SAFe raamistik ongi mõeldud selliste olukorda jaoks, kus sõltuvused on mitmete eri tiimidega ning neil kõigil on vaja omavahelist koostööd teha. SAFe rakendamine eesmärgiks oli ka leevendada varasemalt esinenud probleemi, et polnud aru saada, kui kaugel meeskonnad enda arendustega on ning kaua võiks veel aega minna, kuna planeerimised ning jooksev progress muutuksid äripoole jaoks palju nähtavamaks ja arusaadavamaks. SAFe juurutamise initsiatiiv oli IT-poolne, kuna äripoolele polnud senine progress sobiv ning IT otsustas end ümber organiseerida eesmärgiga olla ärile paremini ennustatavamad, nähtavamad ning läbipaistvamad.

Esialgu moodustati kolm ART-i (*Agile Release Train*) – *Fulfillment and P&O* (Müügi ja tarne juhtimine ning toote ja pakkumise elutsükli juhtimine), *Bill and Collect* (Arveldus ja võlahaldus) ning *Customer Support and Assurance* (Kliendikontaktide juhtimine) – eesmärgiga katta ära klientidele nähtavad tegevused (Joonis 6). Üks ART koosneb kõigist vajalike oskuste ning teadmistega inimestest, et tarkvara või riistvara edukalt implementeerida, testida, paigaldada ning kasutusse anda. Iga ART on omamoodi virtuaalne organisatsioon, mis planeerib, arendab ning paigaldab tööd üheskoos [37].

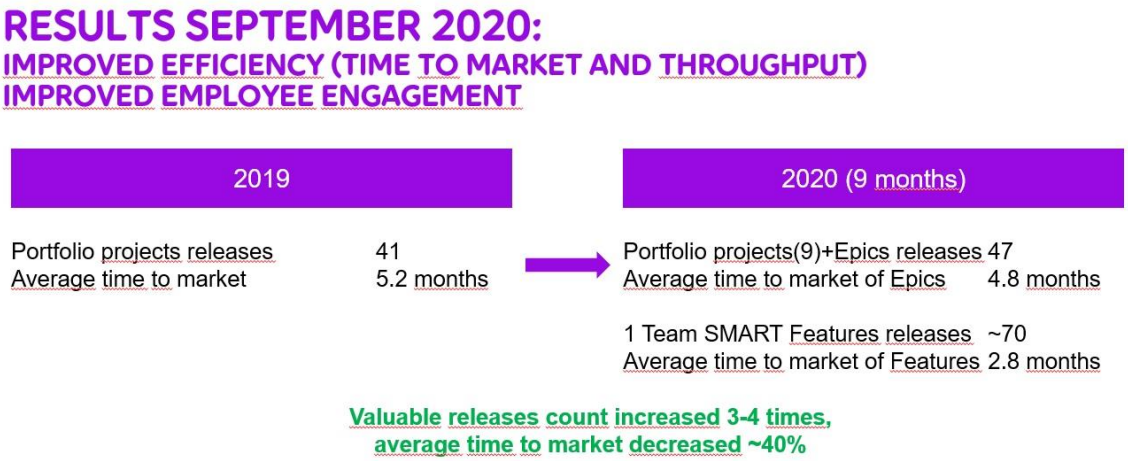
Kõik ülejäänud ehk ettevõtte sisemist tööd toetavad tegevused, nagu näiteks raamatupidamine ning tehniliste telekommunikatsiooni teenustega seonduv, kus tegemist on tooteplatvormi meeskondadega, jäeti teadlikult esmasest skoobist välja, et oleks kaasatud vaid kliendile iseteeninduses nähtav tegevuste jada. Kuna SAFe lahendab keerukuste ja sõltuvuste probleemi, siis võeti fookusesse just need kohad, kus äripoole/tellija jaoks oli probleem kõige suurem – pidevalt on vaja suures koguses asju teha, jõudlust pole piisavalt ning seda on vaja optimeerida. Kõigil fookusesse võetud meeskondadel on omavahelised kokkupuutepunktid ning ükski neist ei saaks ise üksikuna tervikut valmis.



Joonis 6. Telia Eesti AS ART-id ja meeskonnad (autori koostatud).

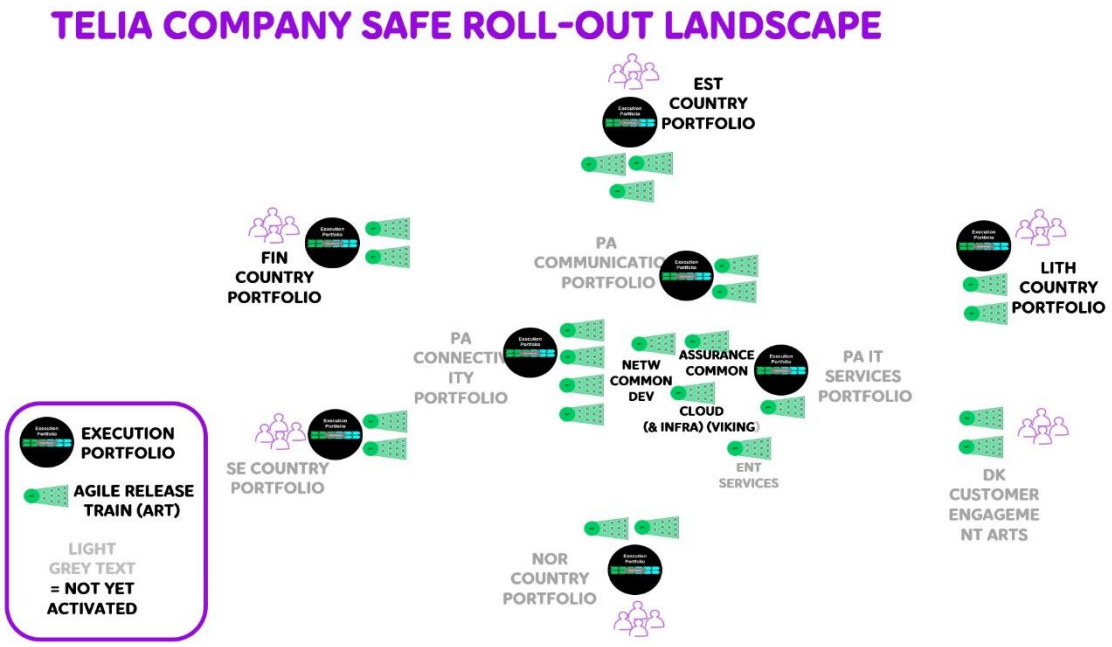
Praeguseks on selge, et SAFe raamistiku rakendamine ettevõttes on olnud õige suund ning tänu sellele on ka arendusprotsesside efektiivsus paranenud. Lühema ajaperioodi

jooksul tehakse ära rohkem suuri projekte ning lisaks veel ka väiksemaid tükke, mille tulemusena on arendusaeg vähenenud ning väärtuslike toodetud tööjuppide arv suurenenud (Joonis 7).



Joonis 7. SAFe raamistiku juurutamisega kaasnenud positiivsed tulemid [38].

Kuna ka Telia Company grupp liigub ART-i põhise arenduse suunas, siis saab olla kindel, et SAFe raamistiku rakendamine jätkub ettevõttes ka tulevikus. Käesolevast kolmest ART-ist edasi ka teistele valdkondadele laienemise katab ära just nimelt grupp. On loodud eraldi üle grupis olevate riikide ettevõtte teenused (*Enterprise services*), mis katab näiteks omaette ART-iga ära siseteenused. Lisaks on veel loomisel üle riikide tootevaldkondade ART-id, kuhu kuuluvad erinevad tooteplatvormid. Telia Company eesmärgiks on luua kõik ART-id üle grupis olevate maade (Joonis 8).



Joonis 8. *Telia Company SAFe roll-out landscape* [38].

Selline lähenemine aitab ühtlustada arendusprotsesse ning tagada, et kõik ühes grupis olevad tütarettevõtted püüdlevad ühtse eesmärgi nimel.

3.2.2 SMART featuurid

Üheks keskseks SAFe raamistiku osaks on featuurid (*Features*), mille puhul on tegemist teenustega, mis täidavad äripoole vajadusi. Iga featuur koosneb hüpoteetilisest kasutulemist ja vastuvõtu kriteeriumist ning tükeldatakse sobivasse suurusesse, et featuur valmiks ühe ART-i poolt ühe PI (*Product Increment*) jooksul. Võimekus (*Capability*) on kõrgema taseme lahendus, mis tüüpiliselt ulatub üle mitme ART-i. Võimekused tükeldatakse omakorda mitmeks featuuriks, et hõlbustada nende implementatsiooni ühe PI jooksul [39].

Featuurid jagunevad suures plaanis kaheks:

- Ärilised featuurid (*Business features*)
- Võimaldavad featuurid (*Enabler features*)

Ärilised featuurid loovad otsest väärtust ärile või kliendile, seevastu *enabler* featuurid ei loo küll otsest ärilist väärtust, aga täidavad tööülesandeid, mis võimaldavad seejärel vähemalt ühe ärilise featuuri edukat implementeerimist. Kuna tihtipeale võivad featuuride kirjeldused ning eesmärgid jääda väga ebamääraseks või arusaamatuks, siis selle vastukaaluks on loodud tarkade (SMART) eesmärkide seadmine. SMART on lühend

viiest sõnast: *Specific*, *Measurable*, *Achievable*, *Realistic* ning *Time-bound* [40]. See tähendab, et seatud eesmärgid peavad olema konkreetsed, mõõdetavad, saavutatavad, realistlikud ning ajaliselt piiratud ehk neid peab olema võimalik saavutada ühe PI jooksul. Näiteks, ebatäpne eesmärk oleks „jõudluse parendamine“ – sellise eesmärgi puhul puuduvad konkreetsed kriteeriumid, mida saavutada. Tark eesmärk oleks aga näiteks „veebilehe avanemise kiiruse tõstmine 5% võrra“.

Tabel 1. Mitte-SMART ning SMART featuuride võrdlus (autori koostatud).

Mitte-SMART featuur	**SMART featuur**
Loodetav tulem on kirjeldatud mitmeti mõistetavalt	Loodetav tulem on täpselt kirjeldatud
Seatud eesmärgid on ebamäärased ning ebatäpsed	On seatud täpsed eesmärgid, mille saavutamist on võimalik mõõta
Eesmärkide täitmine võib olla meeskonna ulatusest väljas	Meeskonna võimuses on seatud eesmärgid täita
Väliste faktoritega ei ole arvestatud, eeldatakse parima lahenduskäigu realiseerumist	On arvestatud väliste segavate faktoritega, mis võivad plaani häirida
Lahendusaeg võib olla vaba ajavahemik	Lahendusaeg on ajaliselt konkreetselt piiratud

Featuuride puhul on väga tähtsaks osaks nende õigesse suurusesse tükeldamine, et leida just need tükid, mis kasutajale väärtust loovad. On olemas nii häid kui ka halbu tavasid, mis featuuride korrektsele tükeldamise kasuks või vastupidi kahjuks tulevad. Kõige olulisem on, et tükeldamine toimuks vertikaalselt mitte horisontaalselt. Horisontaalse tükeldamise puhul pole peale iga individuaalse faasi (nt. analüüs või testimine) lõppu kliendile midagi konkreetset näidata, mille pealt oleks võimalik ka tagasisidet saada. Vertikaalse tükeldamise puhul keskendutakse väiksemale osale kogu lahendusest ja tuuakse need osad miniatuursete kosemeetodi sammudena tootmisesse. Nii jõuab iga tükiga kliendini ka reaalne kasu ning tal on võimalik adekvaatset tagasisidet anda (Joonis 9).

Joonis 9. *Slicing Features horizontally vs. slicing them vertically* [41].

Mõned head tavad featuuride tükeldamiseks on järgnevad:

- Lihtsa disaini printsiip – eesmärgiks peaks olema lihtsaim võimalik featuuri teostamine, mille saab anda kasutusse mittefunktsionaalseid nõudeid (nt. süsteemi jõudlus) ohustamata, kuna kõiksuguseid lisasid saab alati hiljem juurde toota;
- Valikuliste käitumisviiside edasilükkamine – kui featuur hõlmab endas palju valikulisi funktsionaalsuseid, siis tasuks kaaluda nende jaoks eraldi featuuride loomist, kuna esmajärgus on oluline põhifunktsioonide rakendamine ning võib selguda, et kõik algselt kirjeldatud lisavajadused ei olegi hiljem enam pädevad;
- Ärivariatsioonide eraldamine – kui featuuri on võimalik järkjärguliselt erinevatele äri valdkondadele välja anda, siis tuleks alustada kõige lihtsama ärivariandiga, et genereerida kiiret tagasisidet;
- Erinevate kanalite eraldamine – kui featuuri on võimalik järkjärguliselt erinevate kanalite või tehnoloogiate kaudu välja anda, siis on võimalik iga kanali implementatsiooni tükeldada eraldi featuuridesse;
- Erinevate kasutajagruppide eraldi adresseerimine – kuna erinevad kasutajagrupid võivad soovida erinevaid funktsionaalsuseid, siis featuuri tükeldamine funktsionaalsuste alamrühmade kaupa aitab olulisematele gruppidele esmajärgus väärtust pakkuda;
- Andmete järkjärgulise hankimise kaalumine – kui mingisugust väärtust saab luua ka ilma kõiki andmeid kohe omamata, siis on võimalik andmeid kas järkjärguliselt tarbida või hankida juba olemasolevatest allikatest;

- Erivariatsioonide isoleerimine – kõigepealt tuleks keskenduda suuremahulistele ja olulistele juhtumitele ning seejärel lisada erijuhtumid lisafeatuuridena, läbi mille võib avastada, et need lisajuhud ei loo tegelikult suurt väärtust ning nende implementeerimine ei pruugigi mõistlik olla;
- Ühiste võimaldajate leidmine – ärilised featuurid tuginevad tihtipeale samadele süsteemsetele käitumistele, mille tõttu võib esimese featuuride komplekti realiseerimine tunduda suure ja keeruka tööna, aga nende tükeldamine ärilisteks ning võimaldavateks featuurideks aitab vähendada riski ning ka featuuride suurust;
- Kasutuslugude grupi leidmine – featuuride puhul kehtib Pareto printsiip ehk 80% ärikasust tuleb tõenäoliselt 20%-st kasutuslugudest ehk tuleks leida just need kasutuslood ning kohelda neid kui omaette featuuri;
- Uurimisvõimaldajate kasutamine – kui featuuri kohta pole piisavalt informatsiooni, et oleks võimalik seda korrektselt tükeldada, siis tuleks kasutada uurimist võimaldavaid kasutuslugusid (*exploration Enabler Story*), mis esindavad tegevusi nagu näiteks taustauuring, disain, prototüüpimine jne, et leida vajalikud teadmised [42].

Teisest küljest on oluline mõista ka, millised valed featuuride tükeldamise praktikad võivad lahenduse hoopis ohtu seada:

- Mittefunktsionaalsete nõuete osas järeleandmiste tegemine – tuleks vältida selliste featuuride tootmist, millel on küll toimivad ärifunktsionaalsused, kuid mitteaktsepteeritav kvaliteet või jõudlus;
- Featuuride võla tekitamine – kuigi on olukordi, kus featuuri piiratud välja andmine väiksemale kasutajagrupile vähendab rakendatavate mittefunktsionaalsuste nõuete kogust, siis tuleks kindlasti tagada, et kui featuur hiljem kõigi kasutajateni jõuab, ei jääks algselt puudu olnud funktsionaalsused lisamata;
- Liiga vara tükeldamine – featuuri tuleks tükeldada ainult siis, kui see on lähitulevikus vajalik ning liiga suur, et seda oleks võimalik kohe ühes tükis lahendada;
- Üle-tükeldamine – tuleks vältida ühe korraga featuuri tükeldamist paljudeks väikesteks osadeks ning pigem leida kõigepealt üks-kaks põhilist tükki ja

lükata ülejäänud funktsionaalsuse tükeldamine edasi, kuni esmaste osade kohta on tagasisidet saadud;

- Toimingu või töövoo sammu järgi tükeldamine – kui tükeldada featuure konkreetsete toimingute või töövoo sammude järgi, siis võib tekkida olukord, kus vaid ühe sammu lõpuni viimine ei loo mingisugust ärilist väärtust;
- Komponendi järgi tükeldamine – arhitektuurilise komponendi või arhitektuurikihi alamsüsteemi järgi tükeldamise puhul tekivad samasugused probleemid nagu ka tomingu või töövoo sammu järgi tükeldamise puhul;
- Featuuri tasemel testimise unustamine – kui featuur tükeldatakse liiga väikesteks osadeks, siis ei tohiks unustada, et on oluline testida nii igat tükki eraldi kui ka kontrollida kogu algse suure featuuri lahendust [42].

Kõigi eelnevalt välja toodud heade ja halbade tavade teadmine ning rakendamine tagab õige featuuride tükeldamise ning seeläbi ka edukalt toimiva arendusprotsessi.

4 Aruandluse loomine

Selles peatükis selgitatakse SMART featuuride aruandluse loomise protsess lahti sammhaaval. Samuti vaadatakse, millised probleemid erinevates staadiumites tekkisid, kuidas need lahendati ning millised olid tulemused. Lisaks uuritakse, kuidas loodud aruandlust efektiivselt kasutada ning milliseid otsuseid välja töötatud mõõdikute pealt teha saab. Andmete töötlemiseks ning aruandluse loomiseks kasutati Tableau analüüsi- ning visualiseerimistarkvara.

4.1 Andmete analüüs ja komplekteerimine

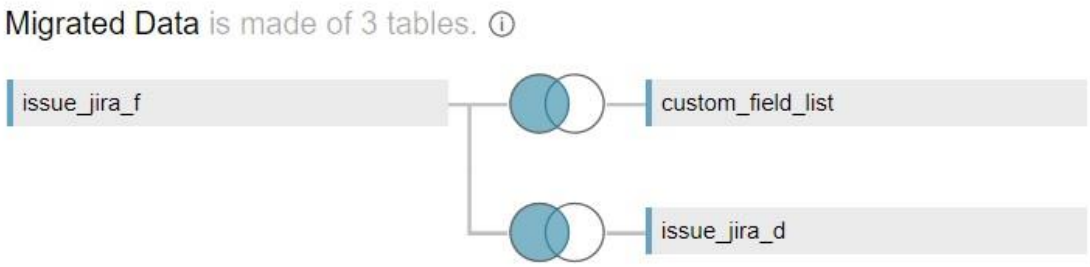
Esimese sammuna oli oluline välja selgitada, millised andmed on aruande loomiseks vajalikud. Äritellijaga koostöös sai kokku lepitud, et aruandes on vaja näha arendustiimide kaupa, kui palju featuure nad konkreetse PI jooksul valmis said ning kui paljud neist olid SMART featuurid. Lisaks oli oluline, et arenduse algus- ning lõppaeg oleks aruandes sees konfigureeritavad.

Aruande looja oli vajalike andmetega juba varasemalt kokku puutunud, kuid siiski oli esimeseks sammuks ettevõttesisese Jira andmete dokumentatsiooni üle vaatamine, kuna Jira andmed asusid kuueteistkümnes erinevas andmetabelis. Seetõttu oli väga tähtis aru saada, milliseid neist on vaja reaalselt kasutada ning millised võib kohe kõrvale jätta. Aruandluse loomiseks tuli kõigepealt Tableau tarkvaras luua ühendus ettevõtte andmeaidaga ning luua vajalikest tabelitest koosnev andmekomplekt. Esimese sammuna oli vaja leida kõik featuurid ning vastavad PI nimetused, mille lõpuks nad valmis said. Andmeallikas oli kaks põhilist tabelit issue_jira_f ning issue_jira_d, milles olid kõik Jira tööd ridade kaupa – ühes tabelis pigem kuupäevadega seotud väljad ning teises tööga seotud isikute väljad, lisaks ka muid detailseid välju. Nendest tabelitest oli võimalik kätte saada featuuri kood ning vastav arendusmeeskond. Kuna aruande puhul oli tähtis vaadata vaid neid featuure, mis reaalselt valmis said, siis aruande äritellijaga kokkuleppes sai otsustatud, et õigeks väljaks, millel on märgitud vajalik PI nimetus, on Jiras „*Feature actual stop*“. Tegemist on kohandatud väljaga (*Custom field*), mida saab Jiras luua ja redigeerida administraatori õigustega kasutaja [43]. Kohandatud väljad võimaldavad sisestada informatsiooni konkreetse meeskonna või projekti vajadustest lähtuvalt. Kõik kohandatud väljad asusid eraldi andmetabelis nimega custom_field_list, kus olid kõikide

Jira tööde kõigi kohandatud väljade nimetused ning neile vastavad väärtused. Kuna loodavas aruandes oli esialgu vaja ainult ühte kohandatud välja, siis eesmärgiga minimeerida andmeallika suurust, ei lisatud allikasse mitte kogu tabelit, vaid päriti järgneva SQL päringuga vaid vajaminev osa:

```
select * from custom_field_list
where issue_jira_cf_nm='Feature actual stop'
```

Läbi eelnevalt nimetatud andmete sidumise sai loodud esmane andmeallikas (Joonis 10), kus tabelid olid omavahel seotud välja issue_jira_host_cd kaudu, mille puhul oli tegemist featuuri koodiga.



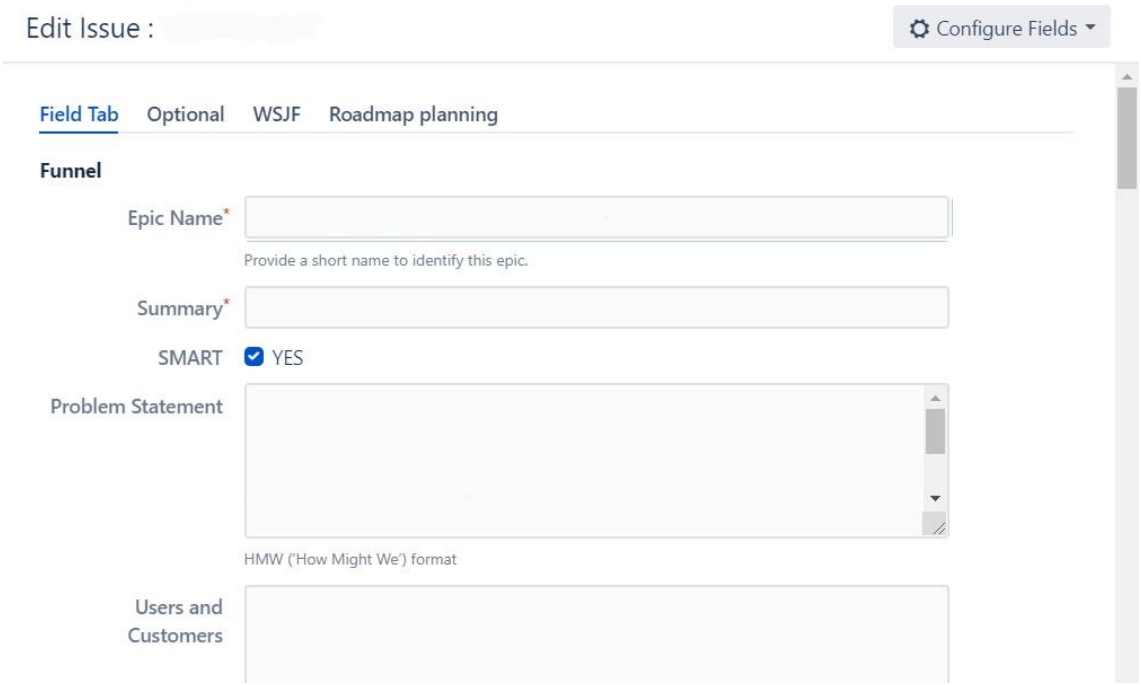
Joonis 10. SMART featuuride aruande esmane andmeallikas (autori koostatud).

Selle andmeallika pealt oli omakorda võimalik andmete algseks kontrollimiseks luua ka esialgne aruande vaade (Joonis 11), kus on arendustiimide põhiselt näha, kui palju featuure neil igas PI-s oli.

		Feature actual stop				
Development team (Level 1)	Development team (Level 2)	PI-06	PI-07	PI-08	PI20-Q4	PI21-Q1
Arveldus		10	15	9	10	5
		7	1	4	4	5
		3	3	4	3	3
		3	3	2	6	4
		5	4	2	3	5
		1	3	5	4	2
Assurance		20	11	8	8	4
		6	8	6	10	6
				1		
		1	1		2	
			2	1	1	2
		6	7	7	4	4
		4	5	6	3	
		6	5	1		
			1			
POF						1
			8	10	10	10
			5	2	4	4
			8	3	7	7
			3	3	3	9
			5	5	3	4
			5	2	7	9
			1	2		
			7	3	2	7
			8	2	5	5
				3	9	11
Grand Total		72	119	91	108	107

Joonis 11. SMART featuuride aruande esmane kuju (meeskondade nimed peidetud) (autori koostatud).

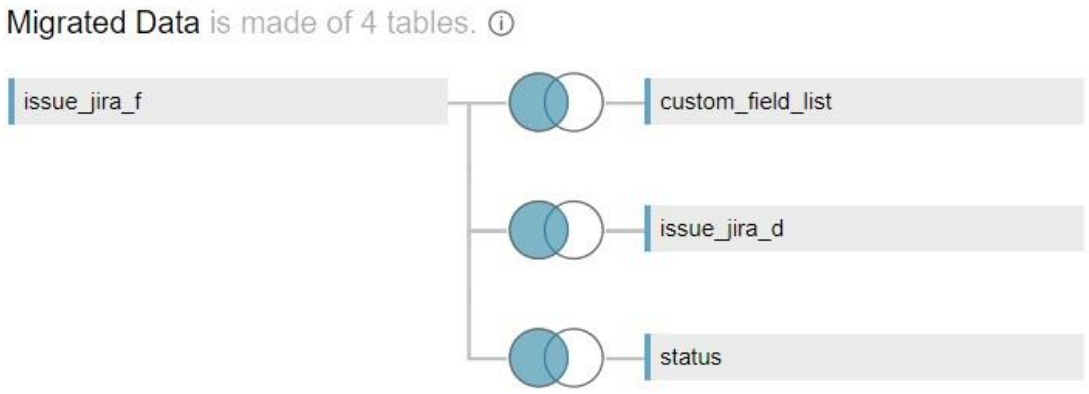
Peale tulemuste õigsuse kontrolli sai liikuda edasi järgmiste vajaminevate andmete lisamise juurde. Aruande eesmärgi täitmiseks oluline väli oli SMART featuure eristav linnuke (Joonis 12), mille puhul oli samuti tegu kohandatud väljaga.



Joonis 12. SMART valik Jiras (autori koostatud).

Andmeallikasse oli vaja lisada ka erinevad staatused, kuna vajalik oli nii alg- kui ka lõppstaatuste muutmine ning SAFe tüübi valik. Osad featuuri staatused oli võimalik kätte saada kohandatud väljadelt, osad aga Jira muudatuste logidest.

Eelnevate muudatuste tulemusena koosnes lõplik andmeallikas neljast tabelist (Joonis 13).



Joonis 13. SMART featuuride aruande lõplik andmeallikas (autori koostatud).

Andmeallikas kasutati kahte terviklikku tabelit (*issue_jira_f* ja *issue_jira_d*) ning kahte SQL päringuga loodud tabelit, kus päringu *custom_field_field* sisuks oli:

```
select * from trinity.issue_jira_custom_field_list_f
where issue_jira_cf_nm='Feature actual stop' or
issue_jira_cf_nm='Feature actual start' or
issue_jira_cf_nm='Available for Customer' or
issue_jira_cf_nm='SMART' or
issue_jira_cf_nm='SAFe type'
```

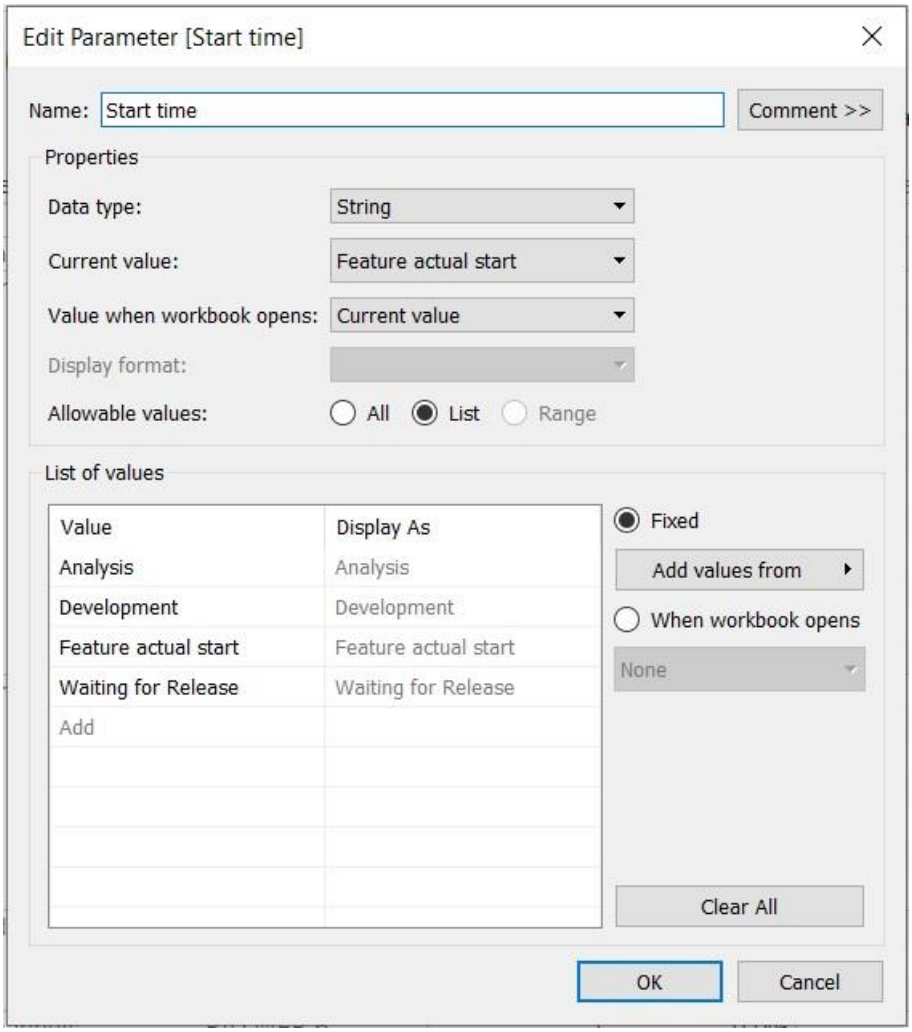
Päringu *status* sisuks oli:

```
select * from trinity.issue_jira_changelog_f
where issue_jira_change_field_nm='status'
```

4.2 Kalkulatsioonid ning visualiseerimine

Peale andmeallika valmimist oli järgmiseks ülesandeks teha aruandes vajalike väljade arvutused. Keerukust tekitas asjaolu, et nii arenduse algus- kui ka lõppaeg oli vaja luua filtrist konfigureeritavana. Parimaks lahenduseks oli Tableau tarkvara puhul parameetri loomine, kuhu oli võimalik sisestada kõik vajalikud staatused, et neid hiljem arvutusvalemisse lisada. Algusaegade (*Start time*) valikuks olid eelnevalt kokku lepitud

Analysis, *Development*, *Feature actual start* ning *Waiting for Release* staatused (Joonis 14).



Joonis 14. Featuuri arenduse algusaegade valikud (autori koostatud).

Lõppaegade (*Stop time*) valikus olid analoogselt eraldi loodud parameetrina *Waiting for Release*, *Done* ja *Available for Customer* staatused.

Kuna eesmärgiks oli välja kuvada vaid need featuurid, millel on kasutaja valitud algus- ja lõppaeg olemas, siis oli vaja vaatele rakendada vastav filter. Selleks sai loodud väli nimega *TTM* (*Time to market*), mis väljendab featuuri arenduse valmimiseks kulunud aega. Selle välja arvutuseks oli arenduse lõppaeg miinus algusaeg (*Stop time – Start time*). Varasemalt said küll kirjeldatud algus- ning lõppaja parameetrid, kuid nende valemites kasutamiseks oli vaja ka vastavaid kalkulatsioone.

Algusaeg oli vaja leida selliselt, et kui parameetrist on valitud teatav väärtus, siis arvutatakse algusajaks vastava staatuse aeg:

```
IF [Parameters].[Start time]='Analysis' THEN [Analysis time]
ELSEIF [Parameters].[Start time]='Development' THEN [Development time]
ELSEIF [Parameters].[Start time]='Feature actual start' THEN [Feature actual
start]
ELSEIF [Parameters].[Start time]='Waiting for Release' THEN [Waiting for
Release time]
END
```

Analysis time arvutuseks võeti fikseeritult iga featuuri kohta *status* päringust esimene *Analysis* olekusse muutmise aeg:

```
{ FIXED [Issue code] : MIN(IF [issue_jira_field_new_ds]="Analysis" THEN
[issue_jira_change_ts] END) }
```

Development time arvutuseks võeti iga featuuri kohta *issue_jira_f* tabelist olemasolevalt esimest korda arenduse staatusesse muutmise väljalt aeg:

```
{ FIXED [Issue code] : MIN([first_development_status_ts]) }
```

Feature actual start arvutuseks võeti iga featuuri kohta *custom_field_list* päringust esimene *Feature actual start* välja väärtus:

```
{ FIXED [Issue code] : MIN(IF [issue_jira_cf_nm]="Feature actual start" THEN
[issue_jira_cf_datevalue_ts] END) }
```

Waiting for Release time arvutuseks võeti iga featuuri kohta *status* päringust esimene *Waiting for Release* olekusse muutmise aeg:

```
{ FIXED [Issue code] : MIN(IF [issue_jira_field_new_ds]="Waiting for Release"
THEN [issue_jira_change_ts] END) }
```

Analoogselt algusaja arvutuse loogikale tuli välja arvutada ka lõppaeg:

```
IF [Parameters].[Stop time]='Waiting for Release' THEN [Waiting for Release
time]
ELSEIF [Parameters].[Stop time]='Done' THEN [Done time]
ELSEIF [Parameters].[Stop time]='Available for Customer' THEN [Available for
Customer time]
END
```

Waiting for Release time arvutus oli sama, mis algusaja puhul, kuna sama staatust oli soov kasutada ka lõppajana. Done time arvutuseks võeti iga featuuri kohta *status* päringust esimene *Done* olekusse muutmise aeg:

```
{ FIXED [Issue code] : MIN(IF [issue_jira_field_new_ds]="Done" THEN
[issue_jira_change_ts] END) }
```

Available for Customer time arvutuseks võeti iga featuuri kohta *custom_field_list* päringust esimene *Available for Customer* välja väärtus:

```
{ FIXED [Issue code] : MIN(IF [issue_jira_cf_nm]="Available for Customer"
THEN [issue_jira_cf_datevalue_ts] END) }
```

Loodud valemite aruandes rakendamiseks sai arvutatud väli *TTM* vaatele filtrina rakendatud sellisena, et aruandes näidataks ainult olemasolevaid väärtusi. Ehk kui lõpuaja filtrist on valitud *Done*, siis tänu eelnevalt rakendatud filtrile kuvatakse välja ainult sellised featuurid, millel on sellise staatuse aeg reaalselt ka olemas.

Lisaks parameetritele oli vaja aruandele lisada ka SAFe tüübi (*SAFe type*) filter, mille abil on kasutajal võimalik valida, kas välja kuvatakse ainult neid featuure, mille tüüp on *Business*, *Enabler*, tüüp puudub või kõiki korraga. Kõigi eelnevalt koostatud arvutuste ning filtrite põhjal loodi SMART featuuride aruanne (Joonis 15), kus kuvati välja PI-de lõikes iga ART kohta kogu featuuride arv ning protsentuaalne SMART featuuride osakaal.

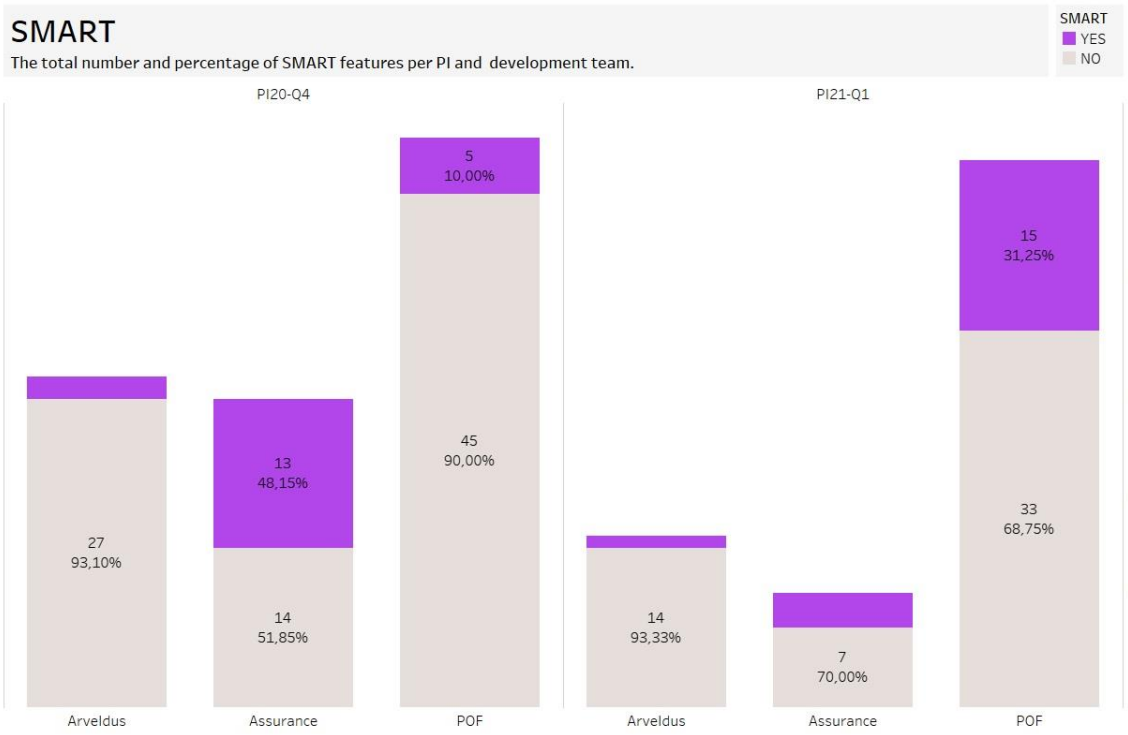
SMART

The total number and percentage of SMART features per PI and development team.

		Feature actual stop					
		PI20-Q4		PI21-Q1		Grand Total	
Development ..	Development team ..	# of Features	SMART %	# of Features	SMART %	# of Features	SMART %
Grand Total		106	18,9%	73	26,0%	179	21,8%
Arveldus	Total	29	6,9%	15	6,7%	44	6,8%
		10	20,0%	4	0,0%	14	14,3%
		3	0,0%	4	0,0%	7	0,0%
		3	0,0%	1	0,0%	4	0,0%
		6	0,0%	3	33,3%	9	11,1%
		3	0,0%	1	0,0%	4	0,0%
		4	0,0%	2	0,0%	6	0,0%
Assurance	Total	27	48,1%	10	30,0%	37	43,2%
		8	62,5%	4	25,0%	12	50,0%
		10	30,0%	4	25,0%	14	28,6%
		2	0,0%			2	0,0%
		1	100,0%	1	0,0%	2	50,0%
		4	75,0%	1	100,0%	5	80,0%
		2	50,0%			2	50,0%
POF	Total	50	10,0%	48	31,3%	98	20,4%
		10	30,0%	10	70,0%	20	50,0%
		4	0,0%	4	0,0%	8	0,0%
		7	0,0%	7	42,9%	14	21,4%
		3	0,0%	5	0,0%	8	0,0%
		3	66,7%	1	100,0%	4	75,0%
		7	0,0%	3	0,0%	10	0,0%
		2	0,0%	5	0,0%	7	0,0%
		5	0,0%	3	0,0%	8	0,0%
		9	0,0%	10	40,0%	19	21,1%

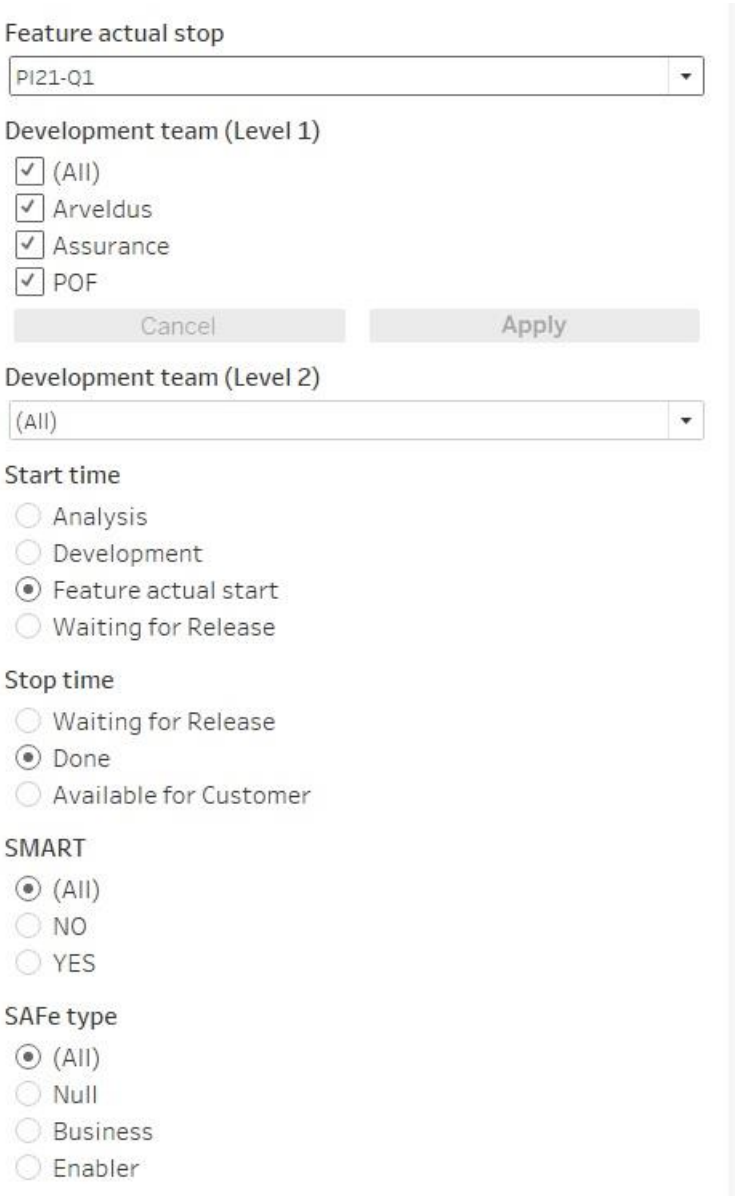
Joonis 15. SMART featuuride aruanne tabeli kujul (meeskondade nimed peidetud) (autori koostatud).

Kuigi tabeli formaadis aruanne on detailne ning informatiivne, siis visuaalne graafik on kergemini hoomatav ning seetõttu sai aruandele juurde loodud ka graafiline vaade. Kuna vaadeldavate andmete puhul oli tegemist numbriliste tulemustega ning nende omavahelise võrdlusega, siis oli kõige mõistlikum neid väljendada tulpdiagrammil. Loodud graafikul (Joonis 16) kujutab tulpade kõrgus seda, kui palju oli igal meeskonnal PI raames featuure kokku ning värvid demonstreerivad, kui paljud neist olid SMART ning kui paljud mitte. Lisaks on tulpade peal välja toodud ka numbrid ning SMART osakaalud, et graafiku mõistmist lihtsustada.



Joonis 16. SMART featuuride aruanne graafiku kujul (autori koostatud).

Mõlema vaate puhul on kasutajal võimalik filtrite nimekirjast (Joonis 17) konfigureerida aruandes kuvatavad mõõdikud just enda vajaduste järgi sobivaks.



Joonis 17. SMART featuuride aruandes kasutajate poolt konfigureeritavad filtrid (autori koostatud).

Valitavatakse filtriteks on featuuri reaalne lõppaeg (PI kujul), esimese taseme arendusmeeskonna ehk ART-i valik, teise taseme meeskonna ehk individuaalsete tiimide valik, arenduse algusaeg (staatuse kujul), arenduse lõppaeg (staatuse kujul), SMART olek (jah/ei) ning featuuri tüüp. Erinevate filtrite konfigureerimise võimaluse tõttu on loodud aruanne universaalne ning rahuldab mitmesuguste kasutajate vajadusi.

4.3 Aruande kasulikkus ning järeldused

Kuna SMART featuuride osakaalu jälgimine ning tõstmine on ettevõtte Telia Eesti AS IT arenduse üks 2021. aasta tähtsamaid fookuseid, siis aitab loodud aruandlus just selle eesmärgi täitmisele kaasa. Aruandes on võimalik näha nii algseisu, kui jälgida ka ajas

muutuvaid trende. Loodud vaadetest on võimalik kasu saada erinevatel osapooltel – tootejuhid ja meeskonnaliikmed näevad, millised on nende tulemused erinevate PI-de lõpuks ning saavad nähtu pealt enda tiimile kõrgemaid eesmärke seada, agiilsed juhendajad saavad omakorda informatsiooni selle kohta, kui hästi on SMART featuuride kasutuselevõtt toiminud ning saavad vastavalt SMART featuuride osakaalu muutumisele kas positiivses või negatiivses suunas ka sobilikke parendusplaane teha.

Töö koostamise hetkeks oli võimalik võrrelda vaid kahe PI tulemusi – PI20-Q4 puhul oli tegemist 2020. aasta viimase kvartaliga ning PI21-Q1 puhul oli tegemist 2021. aasta esimese kvartaliga. Kuna aasta vahetumisega muutus ka töökorraldus ning seati eesmärgiks SMART featuuride osakaalu tõstmine, siis juba 2021. aasta esimese PI puhul on võrreldes eelmise PI-ga näha väikest SMART osakaalu tõusu valminud featuuride osas. Vaatamata mõningasele tõusule üldpildis on siiski paljudel meeskondadel SMART featuuride osakaal kas täiesti nullis või väga väikene.

Tekkinud olukorra juurpõhjuste leidmiseks oli vaja koguda tagasisidet konkreetsete meeskondade liikmetelt. Selleks koostati järgneva sammuna intervjuud erinevate võimekuste eest vastutavate tootejuhtidega.

5 ART-ide tagasiside

Teiseks uurimismeetodiks oli antud magistritöös tootejuhtidega tagasiside saamise eesmärgil vestluste läbi viimine. Intervjuud toimusid vabas vormis ning nende eesmärgiks oli tuvastada, millised probleemid on tootejuhtidel featuuridele äriliste mõõdikute seadmisel ning mida teha, et olukorda parandada.

5.1 Tagasiside vestluste kirjeldus

Põhiliseks probleemi valideerimise ning murekohtade välja selgitamise meetodiks oli antud töös intervjuude läbiviimine. Intervjuuks nimetatakse süsteemset lähenemist, mille tulemusena saadakse inimeselt või inimeste grupilt vestlemise, asjakohaste küsimuste küsimise ning vastuste dokumenteerimise abil ärianalüüsiks vajaliku informatsiooni [44]. Intervjueeritavaks rolliks oli tootejuht, kelle ülesandeks on SAFe raamistiku puhul suhelda klientidega, määratleda loodavate toodete nõuded ja skoop ning seejärel edastada kõik vajalik info tooteomanikule. Kõik intervjueeritavad said intervjuudele eelnevalt informatsiooni, mis teemal intervjuu toimub ning millised ootused intervjueeritavatele on. Intervjuud toimusid struktureerimata vormis – iga intervjueeritav sai kõige pealt ise enda valdkonnast ning hetkelisest seisuskorrast rääkida ning seejärel esitas intervjueerija saadud informatsiooni põhjal sobivaid küsimusi. Algne intervjuueritavate valim koosnes viiest tootejuhist ning valimi eesmärgiks oli, et kõik kolm ART-i oleks kaetud (Tabel 2).

Tabel 2. Esimene valim tootejuhte intervjuudeks (autori koostatud).

Intervjueeritav	**ART**	**Võimekus**	**Intervjuu pikkus**
Intervjueeritav 1	*Customer Support and Assurance*	*Service Assurance Management*	19 minutit
Intervjueeritav 2	*Fulfillment and P&O*	*Resource & Workforce Management*	31 minutit
Intervjueeritav 3	*Customer Support and Assurance*	*Self Service Assurance*	11 minutit
Intervjueeritav 4	*Bill & Collect*	*Billing-, Invoicing-, Delivery & Archiving Management*	18 minutit
Intervjueeritav 5	*Fulfillment and P&O*	*Customer Products*	24 minutit

Kuigi enamikul esimeses valimis olnud tootejuhtidest ei esinenud intervjuude jooksul saadud informatsiooni põhjal töös uuritavat probleemi, ilmnes paari vestluse käigus siiski esimene murekoht – andmekvaliteet. Nimelt selgitas üks intervjueeritutest, et tema meeskonna featuuridel on küll küljes ärimõõdik, kuid seda ei mõõdeta regulaarselt, kuna vajalikud andmed on kättesaadavad vaid otse süsteemist. Sellisel juhul tehakse maksimaalselt paaril korral süsteemipäring vajaliku arvu kohta ning võrreldakse seda seatud eesmärgi vastu. See tähendab aga, et ei jälgita trende ega tulemuste muutumisi ajas, vaid vaadatakse ühekordselt, kas seatud eesmärk realiseerus või mitte.

Kuigi esimese viie intervjuu põhjal oli võimalik juba mõningaid järeldusi teha, siis oli siiski ilmselge, et valim oli liiga väike ning ei kajastanud täielikult reaalseid probleeme. Põhjalikuma probleemianalüüsi eesmärgil otsustati teha järgmine voor intervjuusid, mille tarbeks loodi neljast eelnevalt intervjueerimata tootejuhist koosnev teine valim (Tabel 3).

Tabel 3. Teine valim tootejuhte intervjuudeks (autori koostatud).

Intervjueeritav	**ART**	**Võimekus**	**Intervjuu pikkus**
Intervjueeritav 6	*Bill & Collect*	*E-channels Developments Management*	16 minutit
Intervjueeritav 7	*Bill & Collect*	*Credit Risk & Receivables & Collection Management*	10 minutit
Intervjueeritav 8	*Customer Support and Assurance*	*Customer Interaction Management*	21 minutit
Intervjueeritav 9	*Fulfillment and P&O*	*Order Handling and Service Order Management*	14 minutit

5.2 Tagasiside vestluste tulemused

Kuna arendusmeeskonnad on erinevad – mõned loovad otseselt kliendile ärilist väärtust, teised aga loovad esmajärguliselt kasu hoopis mõnele muule meeskonnale, siis peale teise intervjuueritavate valimiga suhtlemist kujunes välja selgem pilt tiimide murekohtadest. Eelnevalt ilmnenud probleemi andmete kättesaadavusega kinnistasid ka teises voorus intervjueeritud. Selgus, et tihtipeale tundub automaatse ning adekvaatse aruandluse

loomine suurema vaevana, kui lihtsalt ühekordse süsteemsete andmete pärimine. Samuti on aruande loomine märkimisväärselt ajakulukam. Kui vajalikke andmeid juba ettevõtte andmeaidas olemas ei ole, siis tuleb need kõigepealt sinna tellida. Esimese sammuna tuleb analüüsida, millisest süsteemist andmeid vaja on, millised väljad on vajalikud, kui kaua neid andmeid säilitada ning millised muud kriteeriumid andmetele rakenduvad. Seejärel tuleb teha tellimus nende andmete andmeaita laadimiseks. See protsess võtab omakorda vähemalt kaks nädalat, kuna selline on andmete laadimisega tegeleva meeskonna sprindi pikkus, aga tihtipeale ka kauem, kui teised prioriteetsemad tööd ees on. Peale andmete andmeaita laadimist jõuab tööjärg vastava teema eest vastutavad analüütikuni, kes saab analüüsitarkvaras vajalikud kalkulatsioonid ning visualiseerimised teha. Kogu eelnevalt kirjeldatud protsess võib aga võtta liiga palju aega või ressursse, et sellest lõpuks konkreetse featuuri tulemuslikkuse mõõtmiseks kasu oleks. Võib tekkida olukord, kus vajaminev aruandlus valmib alles peale featuuri kasutusse minekut, kuigi vajadus selleks oleks olnud juba palju varem, et selgitada välja näiteks algpunkt, millega võrreldes tulemusi parendada tahetakse.

Vajalike andmete olemasolul oleks aruandluse loomise protsess märkimisväärselt kiirem, kuna jääks välja algandmete analüüsimise ning andmeaita laadimise sammud. Kui adekvaatne aruandlus on olemas kohe featuuriga tegelemise alguses, siis lisaks sellele, et on võimalik jälgida, milline on hetkeseis, on võimalik ka kogu aeg jooksvalt tulemuste muutumist jälgida. Kui teha peale featuuri kasutusele võttu ühekordne väljavõte andmetest, võib saada täiesti vale ettekujutuse arenduse tulemustest. Näiteks, kui eesmärk on tõsta konkreetse veebilehe külastuste arvu ning tehakse kaks väljavõtet, esimene enne parenduste kasutusse minekut ning teine kuu aega hiljem, siis nende päringute ajal võivad kehtida erinevad külastatavust mõjutavad kriteeriumid, mille tulemusena võivad numbrid olla moonutatud. Seevastu andmete pidev jälgimine aitab paremini tuvastada muutusi ning pikemaajalisi trende.

Teise suure probleemina ilmnes intervjuude käigus asjaolu, et mitmed tootejuhid ei ole päris kindlad, kuidas ja millistele featuuridele tuleks ärilisi eesmärke seada. Kuna SMART featuuride kasutust hakati ettevõttes Telia Eesti AS alles hiljuti juurutama, siis on nendega seotud teadmised ning kasutuskogemus samuti veel vähene. Lisaks on ka olukordi, kus kirjeldatakse featuuri juures ära küll ärilised eesmärgid ja mõõdikud ehk featuur on küll ametlikult SMART, kuid tegelikult kliendile sisulist väärtust ei loo.

Sellel murekohal on kaks põhilist juurpõhjust – nagu eelnevalt mainitud, on SMART eesmärkide seadmine ettevõttes alles uus praktika ning lisaks ei peakski vaid tootejuht vastutama featuurile eesmärkide seadmise eest. Kuna intervjuudest ilmnes, et keerukust valmistavad just tehnilised ja *enabler* tüüpi featuurid, siis nende puhul vajab tootejuht kogu meeskonna abi. Tiimis olev tehniline inimene (näiteks arhitekt või analüütik) on see, kes oskab öelda, miks seda featuuri tehakse ning millist kasu ta loob. Kuna ka äriliste featuuride eeldustöid on võimalik teha SMART-ina, siis tuleks näidata välja, mis kasu sellest laiemalt on – sellisel juhul pole otsene kasu siis mitte kliendile, vaid näiteks ettevõttele – ning IT on see, kes peaks seda kasu nägema ning väljendama.

Kuna mitmes intervjuus väitsid tootejuhid, et nende võimekuse vaates ongi palju tehnilisi featuure ning seetõttu neid ei eesmärgistata, siis see tõestas, et on väga palju featuure, mille puhul pole täpselt teada, mida ning milleks tehakse. Seda kinnitab ka eelnevalt loodud aruandlus, kust on väga selgesti näha, et paljudes meeskondades ei valminud PI-de jooksul mitte ühtegi tarka featuuri.

Kui vaadata SMART featuuride osakaalu ka intervjueeritud tootejuhtide võimekuste vaates, siis ka nende puhul on näha, et enamustel ei ole osakaal varasema PI-ga võrreldes tõusnud, vaid on pigem hoopis langenud (Tabel 4).

Tabel 4. SMART featuuride osakaal võimekuste ja kvartalite lõikes (autori koostatud).

ART	**Võimekus**	**PI20-Q4 SMART %**	**PI21-Q1 SMART %**
Bill & Collect	*Billing-, Invoicing-, Delivery & Archiving Management*	25%	0%
	Credit Risk & Receivables & Collection Management	0%	-
	E-channels Developments Management	0%	33%
Customer Support and Assurance	*Customer Interaction Management*	63%	25%
	Self Service Assurance	57%	50%
	Service Assurance Management	50%	-
Fulfillment and P&O	*Customer Products*	67%	100%
	Order Handling and Service Order Management	0%	0%
	Resource & Workforce Management	0%	0%

Selle tulemusena on võimalik järeldada, et kuigi eelnevalt loodud aruandluse põhjal oli suures pildis näha SMART featuuride osakaalu tõusu, siis meeskondadesse sisse vaadates ei ole enamuse tulemus aga paranenud. Probleemide lahendamiseks on järgmises peatükis välja toodud parendusvõimalused, mis aitaksid meetrikat positiivses suunas liigutada.

5.3 Parenduskohad vestluste põhjal

Intervjuudest kujunes välja kaks põhilist murekohta – probleemid andmekvaliteedi ja andmete kättesaadavusega ning tootejuhtide vähene kogemus SMART featuuridega ja neile ärimõõdikute seadmisel, mistõttu ei olda alati kindlad kas või millistele featuuridele üldse ärilisi mõõdikuid seada saaks või tuleks. Mõlema leitud tulemi lahendamiseks on erinevaid võimalusi.

Vajalike andmete puudumise näol on tegemist ettevõtteülese probleemkohaga. Selle lahendamiseks on vaja põhjalikku arusaama juba olemasolevatest andmetest, mida laetakse andmeaida tabelitesse ning kas need andmed on piisavalt terviklikud või mitte. Seejärel peaksid algandmetega kursis olevad isikud ära kaardistama, mida oleks potentsiaalselt vaja juurde laadida, millises mahus ning millise regulaarsusega. Kui ärimõõdikute seadmisega seotud andmed oleksid alati lihtsalt ning kiirelt kättesaadavad, oleks nende põhjal võimalik märkimisväärse aja ning ressursi kokkuhoiuga mõõdikute jälgimist soodustav aruandlus luua. Sellises olukorras oleks ka tootejuhtidel endil võimalik aruandeid ehitada, kuna Tableau tarkvara võimaldab näiteks analüütiku poolt valmis tehtud andmeallikaid kasutada ka neil, kes andmeaidaga ehk nii tuttavad pole. Niimoodi jääks aruandluse loomise ressurss täielikult meeskondadesse sisse ning puuduks põhjus teiste tiimide järel ootamiseks. Andmete puudulikkuse probleemi leevendaks ka täiesti uue tootejuhi ametikoha loomine, kelle ülesanneteks oleks juhtida kokku ärilised vajadused just arendustegevustes, tagada andmete kasutatavus analüüsi ja andmeteaduse vaatenurgast ning lisaks garanteerida, et arenduste tulemust oleks võimalik kasutada andmetel põhinevate äriotsuste tegemisel. See tähendab, et sellel ametikohal olev inimene aitaks ettevõtte üleselt ühtlustada erinevate arenduste poolt loodavaid andmeid ning kindlustaks, et neid andmeid oleks võimalik omavahel võimalikult hästi kokku viia (nt. ühiste ID koodide järgi). Lisaks peaks see tootejuht olema piisavalt kursis kõigi eelnevalt mainitud teemadega, et informeerida äripoolt ka tulevastest võimalustest,

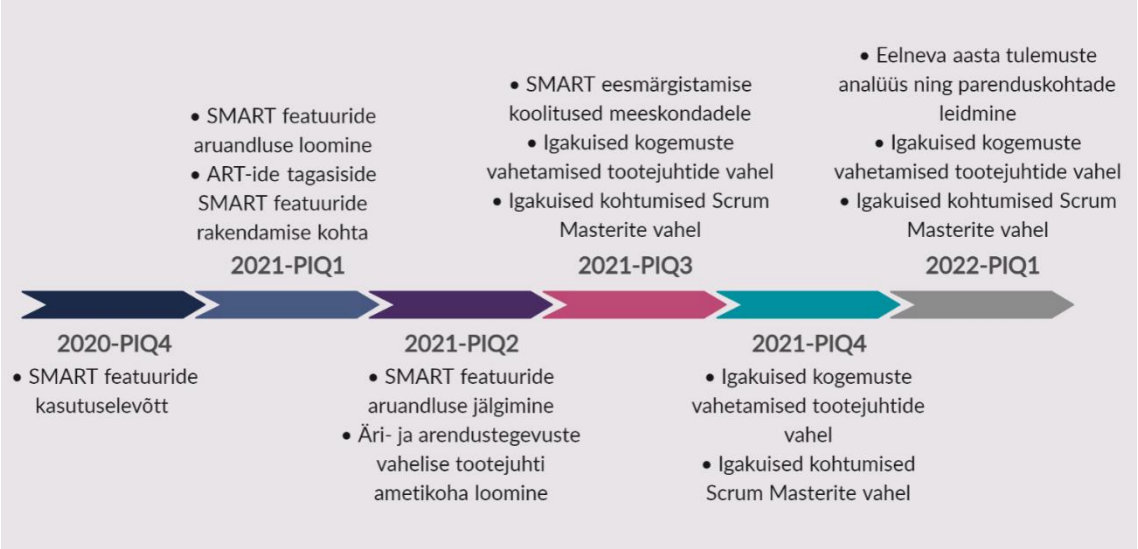
mida teenuste andmekvaliteedi tõstmine luua võib. Kokkuvõtvalt tagab see ametikoht, et ettevõttes on olemas isik, kes suudab edukalt kõigi osapoolte vajadusi kokku juhtida ning aidata ettevõttel jõuda punkti, kus on võimalik äriotsuste tegemiseks kasutada terviklikke andmeid.

Teise probleemi – vähene teadlikkus SMART eesmärkide seadmisest – leevendamiseks on kõige tähtsam meeskondade ning tootejuhtide informeerimine ning toetamine mõõdikute leidmisel ning seadmisel. Kuna intervjuude põhjal selgus, et tiimidel on väga palju tehnilisi featuure, mida seni eesmärgistatud pole, siis järgmise sammuna tulekski meeskondade põhiselt vaadata, miks neil sellised featuure nii palju on ning mida saaks teha, et nad ka väärtust looks. Üheks võimaluseks on ka featuuride teistmoodi tükeldamine, et leida just need kohad, mis oleks väärtust loovad. Selleks tuleks koos meeskonnaliikmetega vaadata üle featuuride tükeldamise parimad tavad ning kindlustada, et neid ka jälgitakse. Lisaks võib tihtipeale tunduda, et featuurist pole kliendile otsest ärilist väärtust ning seetõttu polegi vaja eesmärke seada, kuid tegelikult saab ka teisele arendusmeeskonnale väärtust loov featuur SMART olla, kui tänu sellele näiteks tehniline teenus paremini vastu peab. Sellisel juhul ei näe klient küll kasu, aga ettevõttele on sellest väga suur kasu olemas.

Loomulikult ei ole õige oodata või eeldada, et kõik meeskonnad peaksid jõudma sajaprotsendilise SMART featuuride osakaaluni. On mitmeid eri põhjuseid, miks kõike ei saa tarkade eesmärkidega lahendada. Esimene eesmärk on aru saada, milline on alguspunkt ning seejärel tulemusi tõsta. Oluline on tuvastada, kui palju oleks võimalik lahendada üldiste informatiivse koolitustega – meeskonnaliikmete teadlikkust tõstes paranevad ka nende oskused efektiivseid eesmärke seada. Loomulikult on väga suur osa ka iga tiimiga eraldi töötades – meeskonnad on väga individuaalsed ning ka nende featuuride sisu erineb suuresti, seega ei ole võimalik kõiki ühtse koolitusega samamoodi abistada. Kui meeskond jõuab sinnamaani, et saab loetleda põhjuseid, miks neil pole enam võimalik SAMRT featuuride osakaalu kõrgemaks saada, siis neist põhjustest kujunevad omakorda välja järgmised parenduskohad. Selliste parenduskohtade leidmine ongi just SMART featuuride osakaalu mõõtmise oluline osa.

6 Tulevikuplaanid

Eelnevas peatükis tehtud järeldustest ning parendusvõimalustest lähtuvalt on paika pandud teekaart ajajoonel PI-de lõikes (Joonis 18). Ajajoon on koostatud alates SMART featuuride kasutuselevõtust kuni 2022. aasta alguseni.



Joonis 18. Teekaart ajajoonel PI-de järgi (autori koostatud).

2020. aasta viimase PI jaoks võeti esmakordselt kasutusele SMART featuurid. Järgneval PI-l toimus SMART featuuride aruandluse loomine, et oleks võimalik ajas muutuvalt jälgida tarkade featuuride osakaalu kõigist. Lisaks viidi läbi vestlused tootejuhtidega, et saada tagasisidet SMART featuuride kasutuselevõtu ning rakenduse kohta. 2021. aasta teise PI jooksul toimub eelnevalt loodud aruandluse jälgimine ning loodi ka uus tootejuhti ametikoht, kes on vahelüliks äri- ning arendustegevuste vahel, tagamaks andmete terviklikkuse ning kättesaadavuse. 2021. aasta teisel poolaastal on kõigepealt planeeritud meeskondade koolitused tarkade eesmärkide seadmise kohta ning samuti on oluline igakuiselt hakata tootejuhtide vahelisi kogemuste jagamisi ning Scrum Masterite kohtumise korraldama. 2022. aasta alguses jätkuvad eelneval aastal alustatud tegevused ning samuti tehakse kokkuvõtte eelmise aasta tulemuste kohta, et leida probleemkohti ning neid seejärel ka lahendada.

Samuti on oluline ka tulevikus jälgida SMART featuuride juurutamisega saavutatud tulemusi arendusprotsessis meeskondade üleselt ning vaadata, kas on jõutud positiivse tulemuseni. Selleks on välja pakutud kolm põhilist KPI-d, mida jälgida – SMART

featuuride osakaal, kogu edukalt kliendi toimetatud featuuride arv ning featuuride keskmine arendusaeg (Tabel 5).

Tabel 5. Mõõdikud ja nende kirjeldused (autori koostatud).

KPI	**Kirjeldus**	**Alguspunkt**	**Eesmärk**
SMART %	SMART featuuride osakaal kõigist featuuridest	26 %	Tõsta SMART featuuride osakaalu 50 % peale
Featuuride arv	Edukalt kliendini toimetatud featuuride arv	73	Tõsta ühe PI jooksul edukalt kliendini toimetatud featuuride arv 100 peale
Featuuride keskmine TTM	Keskmine aeg, mis kulus featuuri arenduse algusest kuni kliendini toimetamiseni	2,5 kuud	Vähendada aega, mis kulus featuuri arenduse algusest kuni kliendini toimetamiseni, 2 kuu peale

Mõõdikute algpunktid on võetud 2021. esimese kvartali seisuga ning eesmärgid on seatud jäädes realistlikuks. Kuna SMART featuuride osakaal on iga meeskonna puhul väga individuaalne ning ühtlasi nõuab veel ka juurutamist, siis pole mõistlik seada kohe maksimaalset ning saavutamatut eesmärki. SMART featuuride juurutamisel saavad featuurid meeskondade jaoks arusaadavamaks, konkreetsemaks ning piisavalt väikesteks tükkideks, et ka ühe PI jooksul edukalt valminud featuuride arv peaks selle tulemusel kasvama ning keskmine arendusaeg võiks minna lühemaks.

KPI-de jälgimine on tähtis, kuna selle tulemusena saadakse tagasisidet meeskondade ning ka ettevõtte suutlikkuse ning toimimise kohta [45]. Samuti on võimalik selgelt näha, kus asutakse konkreetsel ajahetkel võrreldes seatud eesmärkide ja ootustega. Seega on oluline kõiki välja toodud mõõdikuid regulaarselt nii PI-de kui ka meeskondade lõikes jälgida, et kiirelt kõik murekohad tuvastada ning lahendada.

7 Kokkuvõte

Käesoleva magistritöö eesmärgiks oli välja selgitada, millised murekohad takistavad tootejuhtidel featuuridele äriliste eesmärkide püstitamist, kuidas saavutada olukord, kus ärilise väärtuse mõõdikud seatakse korrektselt ja peale featuuri kasutusse minekut jälgitakse eelnevalt seatud mõõdikuid ning luua adekvaatne aruandlus SMART featuuride osakaalu muutumise jälgimiseks.

Töö eesmärgi saavutamiseks uuriti kõigepealt kirjandusest tausta andmetel põhineva arenduse ning erinevate arendusmetoodikate kohta. Seejärel kaardistati töös kajastava ettevõtte Telia Eesti AS hetkeolukord – millega ettevõtte tegeleb, millised on ettevõtte väärtused ning millised arendusmetoodikad on kasutusel. Järgmise sammuna loodi Jira andmete põhjal SMART featuuride osakaalu kajastav aruandlus, mille abil on võimalik jälgida SMART featuuride osakaalu algseisu ning muutust erinevate PI-de lõpuks. Aruandest kujunes välja, et SMART featuuride osakaal on peale kasutuselevõttu küll kõikide meeskondade üleselt tõusnud 18,9 protsendi pealt 26 protsendi peale, mis tähendab 7,1 protsendilist tõusu, kuid täpsemalt sisse vaadates selgus, et mitmel individuaalsel meeskonnal oli tulemus hoopis langenud. Loodud aruande tulemuste selgitamiseks viidi järgmiseks erinevate tootejuhtidega läbi tagasiside saamise intervjuud, et mõista, millest on tingitud konkreetsete meeskondade SMART featuuride madal osakaal kõikidest PI jooksul valmis saadud featuuridest. Aruandlusest ning tagasidest saadud informatsiooni põhjal tehti töös ka erinevaid parendussoovitusi ning tulevikuplaane, mis abistaksid ning toetaksid meeskondi featuuridele äriliselt mõõdetavate eesmärkide seadmisel.

Magistritöö tulem on kasulikuks sisendiks ettevõtte arendusprotsessides parendustegevuste koostamiseks ning tulemuste parandamiseks.

Kasutatud kirjandus

[1] „SAFe: framework for scaling Agile,“ Scaled Agile, [Võrgumaterjal]. Available: https://www.scaledagile.com/enterprise-solutions/what-is-safe/.

[2] „DATA | meaning in the Cambridge English Dictionary,“ [Võrgumaterjal]. Available: https://dictionary.cambridge.org/dictionary/english/data.

[3] „What is Data?,“ MathsIsFun, [Võrgumaterjal]. Available: https://www.mathsisfun.com/data/data.html.

[4] „Data-driven decision making: Succeed in the digital era,“ Tableau, [Võrgumaterjal]. Available: https://www.tableau.com/learn/articles/data-driven-decision-making.

[5] D. Waller, „10 Steps to Creating a Data-Driven Culture,“ 6 February 2020. [Võrgumaterjal]. Available: https://hbr.org/2020/02/10-steps-to-creating-a-data-driven-culture.

[6] J. Fries, „Data-driven software engineering: How to avoid common problems,“ [Võrgumaterjal]. Available: https://techbeacon.com/app-dev-testing/data-driven-software-engineering-how-avoid-common-problems.

[7] „Manifesto for Agile Software Development,“ [Võrgumaterjal]. Available: https://agilemanifesto.org/.

[8] „Principles behind the Agile Manifesto,“ [Võrgumaterjal]. Available: https://agilemanifesto.org/principles.html.

[9] „Agile software development,“ Wikipedia, [Võrgumaterjal]. Available: https://en.wikipedia.org/wiki/Agile_software_development.

[10] „Is Agile always the best solution for software development projects?,“ [Võrgumaterjal]. Available: https://www.soldevelo.com/blog/is-agile-always-the-best-solution-for-software-development-projects/.

[11] „Lean Software Development,“ [Võrgumaterjal]. Available: https://www.productplan.com/glossary/lean-software-development/.

[12] „How We Align Lean Principles in Our Software Development Process,“ [Võrgumaterjal]. Available: https://appinventiv.com/blog/how-we-integrate-lean-principles-in-software-development/.

[13] „How Your Business Should Benefit of Lean Software Development,“ [Võrgumaterjal]. Available: https://perfectial.com/blog/lean-software-development/.

[14] F. Malas, „The 7 Lean Principles To Help Your Software Development,“ 5 September 2019. [Võrgumaterjal]. Available: https://medium.com/@faisal_81902/the-7-lean-principles-to-help-your-software-development-e81884dca8fa.

[15] S. Srivastava, „How We Align Lean Principles in Our Software Development Process,“ 23 July 2020. [Võrgumaterjal]. Available:

https://appinventiv.com/blog/how-we-integrate-lean-principles-in-software-development/#2.

[16] D. Radigan, „Kanban - A brief introduction,“ Atlassian, [Võrgumaterjal]. Available: https://www.atlassian.com/agile/kanban.

[17] D. J. Anderson, Kanban: Successful Evolutionary Change for Your Technology Business, Blue Hole Press Inc, 2010.

[18] M. Rehkopf, „What is a Kanban Board?,“ Atlassian, [Võrgumaterjal]. Available: https://www.atlassian.com/agile/kanban/boards.

[19] D. Peterson, „What is Kanban?,“ [Võrgumaterjal]. Available: https://kanbanblog.com/explained/.

[20] „Kanban vs. Scrum,“ ProductPlan, [Võrgumaterjal]. Available: https://www.productplan.com/learn/kanban-scrum/.

[21] K. Schwaber ja J. Sutherland, „Scrum Guide,“ Scrum Guides, 2020. [Võrgumaterjal]. Available: https://scrumguides.org/scrum-guide.html.

[22] „What is Scrum?,“ [Võrgumaterjal]. Available: https://www.scrum.org/resources/what-is-scrum.

[23] „What is DevOps?,“ Atlassian, [Võrgumaterjal]. Available: https://www.atlassian.com/devops.

[24] „What is DevOps?,“ Netmind, [Võrgumaterjal]. Available: https://netmind.net/en/devops-best-practice/.

[25] „What is DevOps and How Does It Work?,“ Synopsys, [Võrgumaterjal]. Available: https://www.synopsys.com/glossary/what-is-devops.html.

[26] „What is DevOps?,“ AWS, [Võrgumaterjal]. Available: https://aws.amazon.com/devops/what-is-devops/.

[27] M. Aleksandrova, „Benefits and Challenges of Taking the DevOps Route,“ DZone, 21 January 2019. [Võrgumaterjal]. Available: https://dzone.com/articles/benefits-and-challenges-of-taking-the-devops-route.

[28] M. Courtemanche, E. Mell ja A. S. Gillis, „What is DevOps? The ultimate guide,“ TechTarget, September 2020. [Võrgumaterjal]. Available: https://searchitoperations.techtarget.com/definition/DevOps.

[29] „Telia Eesti,“ Vikipeedia, [Võrgumaterjal]. Available: https://et.wikipedia.org/wiki/Telia_Eesti.

[30] Inforegister, „TELIA EESTI AS,“ [Võrgumaterjal]. Available: https://www.inforegister.ee/10234957-TELIA-EESTI-AS.

[31] Digitark, „Telia juht: valida saab kas odava või kvaliteetse teenuse vahel,“ [Võrgumaterjal]. Available: https://digitark.ee/telia-juht-valida-saab-kas-odava-voi-kvaliteetse-teenuse-vahel/.

[32] „Telia 4G võrgu keskmine allalaadimiskiirus ulatub ligi 60 Mbit/s,“ [Võrgumaterjal]. Available: https://www.telia.ee/uudised/telia-4g-vorgu-keskmine-allalaadimiskiirus-ulatub-ligi-60-mbit-s.

[33] „Üldinfo,“ [Võrgumaterjal]. Available: https://www.telia.ee/ettevottest/uldinfo/.

[34] „Telia avab oma klientidele uue põlvkonna 5G võrgu,“ [Võrgumaterjal]. Available: https://www.telia.ee/uudised/telia-avab-oma-klientidele-uue-polvkonna-5g-vorgu.

[35] „Telia ühiskonnas,“ [Võrgumaterjal]. Available: https://www.telia.ee/ettevottest/telia-uhiskonnas/.

[36] „SAFe for Lean Enterprises,“ [Võrgumaterjal]. Available: https://www.scaledagileframework.com/safe-for-lean-enterprises/.

[37] „SAFe 5 for Lean Enterprises,“ Scaled Agile, Inc., 10 February 2021. [Võrgumaterjal]. Available: https://www.scaledagileframework.com/safe-for-lean-enterprises/.

[38] E. Peterson, „What are Agile Release Trains?,“ Planview, [Võrgumaterjal]. Available: https://www.planview.com/resources/guide/what-is-agile-program-management/agile-release-trains/.

[39] K.-H. Sillmann, „Capabilities,“ IT wiki , 23 March 2021. [Võrgumaterjal]. Available: Telia Eesti AS ettevõttesisene Confluence.

[40] „Features and Capabilities,“ Scaled Agile, Inc., 10 February 2021. [Võrgumaterjal]. Available: https://www.scaledagileframework.com/features-and-capabilities/.

[41] „PI Objectives,“ Scaled Agile, Inc., 10 February 2021. [Võrgumaterjal]. Available: https://www.scaledagileframework.com/pi-objectives/.

[42] M. Steffen, „Slicing Features—vertically,“ Medium, 23 September 2019. [Võrgumaterjal]. Available: https://medium.com/product-manager-tools/slicing-features-vertically-b06bd22efe9b.

[43] I. Spence, „Right-Sizing Features for SAFe Program Increments,“ Scaled Agile, Inc., [Võrgumaterjal]. Available: https://www.scaledagileframework.com/right-sizing-features-for-safe-program-increments/.

[44] „Configure issue custom fields,“ Atlassian Support, [Võrgumaterjal]. Available: https://support.atlassian.com/jira-cloud-administration/docs/configure-issue-custom-fields/.

[45] International Institute of Business Analysis, A Guide to the Business Analysis Body of Knowledge, 2015.

[46] L. Mosca, „Key Performance Indicators 101 & Why They're Important,“ Forbes, 18 June 2019. [Võrgumaterjal]. Available: https://www.forbes.com/sites/louismosca/2019/06/18/key-performance-indicators-101-why-theyre-important/?sh=4e7cb7e72652.

[47] „SAFe 5 for Lean Enterprises,“ Scaled Agile, Inc, 10 February 2021. [Võrgumaterjal]. Available: https://www.scaledagileframework.com/safe-for-lean-enterprises/.

Lisa 1 – Lihtlitsents lõputöö reprodutseerimiseks ja lõputöö üldsusele kättesaadavaks tegemiseks[1]

Mina, Annaliisa Romanenkov

1. Annan Tallinna Tehnikaülikoolile tasuta loa (lihtlitsentsi) enda loodud teose „Edumõõdikute kasutuselevõtt featuuride jaoks Telia Eesti AS näitel - jälgimine ja probleemkohad“, mille juhendajad on Kristjan-Hans Sillmann ja Nadežda Furs-Nižnikova.
   1.1. reprodutseerimiseks lõputöö säilitamise ja elektroonse avaldamise eesmärgil, sh Tallinna Tehnikaülikooli raamatukogu digikogusse lisamise eesmärgil kuni autoriõiguse kehtivuse tähtaja lõppemiseni;
   1.2. üldsusele kättesaadavaks tegemiseks Tallinna Tehnikaülikooli veebikeskkonna kaudu, sealhulgas Tallinna Tehnikaülikooli raamatukogu digikogu kaudu kuni autoriõiguse kehtivuse tähtaja lõppemiseni.
2. Olen teadlik, et käesoleva lihtlitsentsi punktis 1 nimetatud õigused jäävad alles ka autorile.
3. Kinnitan, et lihtlitsentsi andmisega ei rikuta teiste isikute intellektuaalomandi ega isikuandmete kaitse seadusest ning muudest õigusaktidest tulenevaid õigusi.

20.05.2021

1 Lihtlitsents ei kehti juurdepääsupiirangu kehtivuse ajal vastavalt üliõpilase taotlusele lõputööle juurdepääsupiirangu kehtestamiseks, mis on allkirjastatud teaduskonna dekaani poolt, välja arvatud ülikooli õigus lõputööd reprodutseerida üksnes säilitamise eesmärgil. Kui lõputöö on loonud kaks või enam isikut oma ühise loomingulise tegevusega ning lõputöö kaas- või ühisautor(id) ei ole andnud lõputööd kaitsvale üliõpilasele kindlaksmääratud tähtajaks nõusolekut lõputöö reprodutseerimiseks ja avalikustamiseks vastavalt lihtlitsentsi punktidele 1.1. ja 1.2, siis lihtlitsents nimetatud tähtaja jooksul ei kehti.