Tallinna Ülikool

Digitehnoloogiate instituut

Infotehnoloogia juhtimine

Äriprotsesside põhine tarkvara kavandamine

Magistritöö

Autor Viljar Komp

Juhendaja Peeter Normak

Autor ..... ……..... 2017

Juhendaja ..... ……..... 2017

Instituudi direktor ..... ……..... 2017

2017 Tallinn

Autorideklaratsioon

Deklareerin, et käesolev magistritöö on minu töö tulemus ja seda ei ole kellegi teise poolt varem kaitsmisele esitatud. Kõik töö koostamisel kasutatud teiste autorite tööd, olulised seisukohad, kirjandusallikatest ja mujalt pärinevad andmed on viidatud.

....

(kuupäev) (autor)

Lihtlitsents lõputöö reprodutseerimiseks ja lõputöö üldsusele kättesaadavaks tegemiseks

Mina _____ (sünnikuupäev _____)

(*autori nimi*)

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

_____

_____

_____

(*lõputöö pealkiri*)

mille juhendaja on _____ ,

(*juhendaja nimi*)

säilitamiseks ja üldsusele kättesaadavaks tegemiseks Tallinna Ülikooli Akadeemilise Raamatukogu repositooriumis.

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

3. Kinnitan, et lihtlitsentsi andmisega ei rikuta teiste isikute intellektuaalomandi ega isikuandmete kaitse seadusest tulenevaid õigusi.

Tallinnas, _____

(digitaalne) allkiri ja kuupäev

Sisukord

Sissejuhatus

Käesolev magistritöö pakub tooteomanikele ühe võimaluse, kuidas kirjeldada kavandatavat tarkvara. Magistritöö vajadus on tekkinud seoses Riigi Kinnisvara AS-i (RKAS) tarkvaraarenduse probleemidega. 2016 aastal toimus RKAS-is suurem põhimõtteline muudatus infotehnoloogia töökorralduses. Muutus täielikult infotehnoloogia osakonna struktuur, kadusid ära arendajate ja disainerite ametikohad. Tarkvaraarendust hakati teenusena sisseostma. Tarkvara hankimise protsess muudeti struktuurimuudatustele vastavaks, lisandus uusi rolle ja muutusid vastutusalad.

Üheks uueks loodud rolliks oli tooteomaniku roll. RKAS-is on tootestatud 25 rakendust ja uusi tooteomanikke kokku on 20. Esimene aasta on näidanud, et tooteomanikel on väga erinev arusaam oma rollist. Kõik tooteomanikud on läbinud agiilsete tarkvaraarenduse metoodikate kursuse, kus tutvustati tooteomaniku rolli. Endiselt saavad tooteomanikud oma rollist väga erinevalt aru. RKAS-i projektijuhtide hinnangul on põhjuseks vähene toetavate materjalide hulk. Väga täpselt on teada, kuidas peavad käituma täitja poole rollid tarkvaraarenduses – süsteemianalüütikud, arendajad ja disainerid. Nende rollide kohta on täpselt kirjeldatud raamistikud ja kuidas neid kasutama peab. Tooteomanike töö kirjeldamiseks sarnaseid materjale ei ole, piirdutakse vastutusalade nimekirjadega. Instruktsioonid vastutusalade töökorralduste täitmiseks puuduvad ja kõik tooteomanikud proovivad tegutseda parema äranägemise järgi. Tulemuseks on, et tarkvaraprojektide tööde korralduse on projektide vahel väga erinev. Tooted peavad kõik olema koosvõimelised, kuna peavad olema ühe põhiprotsessi osad. On tekkinud reaalne vajadus tarkvara funktsionaalsuste kirjeldamiseks, mis on toodete ülesed – keerukate lahenduste kavandamiseks on vaja tarkvaraarenduse protsessid ühtlustada.

Varasemalt on palju põhjalikult uuritud, kuidas tarkvaraarenduse metodoloogiad rakendada ja kuidas jagunevad vastutusalad (Larsson, 2013), (Krisinsdottir, 2014), (Cristian Bogdan, 2016). RKAS on agiilseid metoodikaid tõlgendanud sarnaselt viidatud uurimustöödele ja võib öelda et töökäskude edastamine ja vastu võtmine on defineeritud ja üheselt juurutatud. Töökäskude kavandamine ja loomine aga toimub kõikides projektides erinevalt. Ühtegi uurimustööd, mis kirjeldaks kuidas täpselt peaks toimuma töökäskude kavandamine, pole õnnestunud leida. Viidatakse visiooni ja

skoobi hoidmisele. Uurimustöid mis kirjeldaks kuidas käituda olukorras, kus tooteomanikul puudub selge visioon pole õnnestunud leida.

Magistritöö uurib, kuidas peaks kirjeldama arendusvajadusi ja kuidas saada tagasisidet kavandatavale tarkvarale, millised sammud on vajalik läbida enne, kui hakatakse arenduspartnerit otsima. Eesmärgiks on koostada soovituslik tööprotsessi kirjeldus, mida järgides on tooteomanikul võimalik tekitada kavandatavale tarkvarale visioon ja selge skoop.

Sellise dokumendi koostamiseks on vaja viia läbi intervjuud täitja ja tellija poole esindajatega, kes on pidanud kirjeldama äriprotsessidel põhinevat tarkvara. Intervjuude käigus loodan leida juba olemasolevaid dokumente mis kirjeldavad tarkvara kavandamise protsessi. Täitja poole esindajatelt, kes pakuvad tooteomanike koolitusi, tuleb küsida, kuidas on võimalik koostada visiooni, kuidas nemad seda teeksid. Magistritöö koostamisel oli kolm etappi, intervjuud täitjatega, intervjuud tellijatega ja analüüs koos tööprotsessi kirjelduse koostamisega.

Magistritöö esimeses kahes lõigus tutvustan täpsemini tausta ja valitud metoodikat. Kolmandas lõigus toon välja tarkvaraarenduse olulised põhimõtted, millega peab arvestama. Neljandas ja viiendas lõigus kirjeldan uuringu läbi viimist ja tehtud järeldusi küsimuste haaval. Kuuendas lõigus kogun kokku informatsiooni, mida soovin kasutada magistritöö eesmärgi, tarkvara kavandamise teostamise abimaterjali koostamisel. Neljas viimases lõigus toimub arutelu ja kokkuvõtte tegemistest. Magistritöö tulemusena valminud juhend on lisatud viimase 11-nda lisana.

1. Taust ja probleem

Riigi Kinnisvara AS juhib oma tarkvaraarenduse protsesse agiilseid metoodikaid kasutades, DevOps, Scrum ja KanBan (Wikipedia, 2017). Kõikidest metoodikatest on laenatud elemente, et töökeskkond oleks kõigile meeldiv ja asju tehtaks eesmärgipäraselt. Üheks laenatud elemendiks on retrospektiiv ehk tagasivaatekoosolek. Koosolekute käigus on kõik arendusmeeskonnad toonud välja üheks oluliseks takistavaks teguriks tellijapoolse sisendi – suutmatuse vastata tekkinud jooksvatele küsimustele piisavalt kiiresti ja konkreetselt. Operatiivse suhtluse probleem ei puuduta kõiki arendusmeeskondasid, kuid pigem on väljatoodu reegel kui erand (Berteig, 2011).

Varasemalt on infotehnoloogia osakond palju vajalikke muudatusi, mis erinevatel tagasiside koosolekutel välja on toodud, juba parandanud. Näiteks arenduse ja halduse omavahelise suhtluse parendamine, tellimuste esitamine ja dokumentatsiooni loomise korraldamine. Praktika on näidanud, et kui meeskond teeb korralikult koostööd ja lahendatavatest muredest saadakse ühte moodi aru, pole töökäskude korrektsel esitamisest erilist mõju. Sisend peab olema piisav, mitte täielik (Keith Collyer, 2013), et saada soovitud tulemus. Vahetu suhtlusega meeskonnas piisab kui töökäskudeks on märksõnad ja dokumentatsiooniks on vahekihi kirjeldus, kus hoitakse äriloogikat. Vesteldes erinevate osapooltega tõdetakse fakti, et tellija ei tea mida täpselt vaja on (Püüa, 2017). Tellija oskab viidata probleemile, kuid soovitud lahendust ei osata kirjeldada. Selline mure ei ole midagi uut – sellele viitab ka varasem Mart Rosina magistritöö „Tellija ja teostaja vaheline koostöö tarkvara sisseostmisel“ (Rosin, 2014).

Välja on töötatud palju raamistikke, mis kirjeldavad kuidas tarkvara hankimine ja arendamine peab olema korraldatud. Tuntuimaks on ITIL, mis sobib ka agiilsesse maailma (Sunview, 2017). ITIL kirjeldab ära kõik vajalikud etapid, kuidas tarkvara hankida (IT process maps, 2017), kuid ei pane kaasa instruktsioone kuidas täpselt teenuse disaini sammu täita. Teenuse planeerimise etapis on teenuse disaini meeskonna koordinaatori kohustus tagada, et kõik olulised funktsionaalsused on kaetud. Olukord läheb agiilses maailmas segasemaks, kuna loobutakse analüüsimise etapist ja seda tehakse jooksvalt.

Kui Ettevõtete infotehnoloogia osakonnad on suutnud agiilse tarkvaraarendusega kaasa minna, siis tellija pool ehk äripool, on jäänud kinni vanadesse mudelitesse

(Zalavadia, 2015). Seda kinnitavad ka magistritöö raames teostatud intervjuud, nii täitjate kui ka tellijatega. Kõikides intervjuudes on mainitud agiilseid ja kosk mudeleid, kuid vähesed ettevõtted saavad tunnistada, et enamus nende projekte toimub ainult agiilseid meetodeid kasutades. Tuuakse välja, et kui toote omanikul pole aega või oskusi tarvakara tellimisega tegeleda, on lihtsam kasutada kosk mudelit (Tammik, 2017), alustades detailse spetsifikatsiooniga ja lootes, et lõppkokkuvõttes valminud toode on see, mida tegelikult vaja on. Põhjusi, miks kosk mudel ei tööta, on palju. (Scrum Institute, 2017)

Tänapäeva tarkvaraarenduse eesmärgiks on peamiselt tekkinud probleemide lahendamine (Longstreet, 2010). Teisejärgulisteks projektideks on olemasolevate tööprotsesside digitaliseerimine. Tarkvara tellijate suutmatus piisavalt täpselt ja kiirelt kirjeldada probleeme loob eeldused projekti ebaõnnestumiseks. Kui analüüsi etapis tehakse suurem viga rakenduse kirjeldamisel, siis kosk mudelit järgides on arenduse käigus selle muutmine välistatud. Agiilne tarkvaraarendus üritab probleeme lahendada väikeste sammudena edasi liikudes, pidevalt olukorda analüüsides.

Agiilses tarkvaraarenduses küsib projektimeeskond pidevalt, milline on kõige lihtsam lahendus, kuidas on võimalik kirjeldatud probleem ära lahendada. Kui tarkvara tellitakse nõuete nimekirjana, siis soovitud tulemus võib olla liiga keeruline ja tekitada uusi probleeme. Kui tellitaval tarkvaral peab olema ka kasutajaliides, siis staatiline nõuete nimekirjast ei piisa. Vaja on äriprotsessi või töövoogu ja probleemi kirjeldust, mida üritatakse lahendada. Tarkvaraarenduseks sobiliku äriprotsessi kirjeldamine nõuab tihedat koostööd projektimeeskonna ja tooteomaniku vahel, kuna äriprotsess peab olema ka teostatav läbi kasutajaliidese. Äriprotsesside kirjeldamine koos kõikide oma erisustega on väga keeruline ja tellija loodab, et selle töö teeb ära süsteemianalüütik. Seda tööd peaks tegelikult tegema ettevõtte ärianalüütik. Äriprotsessi kirjeldamist koos visuaalsete abimaterjalidega pakuvad teenusdisaini ettevõtted, kuid nende teenuste eest pole paljud tellijad veel nõus maksma. Vaja on lihtsat ja konkreetset viisi kuidas kavandada tarkvara tellimist. Lähteülesandeks ei pea olema täielik äriprotsessi kirjeldus, vaid piisab minimaalsest kirjeldusest, mis lahendab ära tekkinud probleemi.

1.1. Ülesande püstitus

Käesoleva magistritöö eesmärgiks on kirjeldada tarkvara kavandamisele kaasatud rollide vastutusalad ja käitumine. Kirjeldus peab tooteomaniku rollile andma ülevaate, kuidas toimub uue toote kavandamine.

Lähtuvalt eesmärgist püstitatakse järgmised ülesanded:

- Analüüsida täitja ja tellija arvamusi ideaalsest tarkvaraarenduse protsessist.
- Kirjeldada tööprotsess, mis lihtsustab äriprotsessil põhineva tarkvara kavandamist vastavalt intervjuudelt kogutud materjalidele.

1.2. Töö ülesehitus

Magistritöö on jagatud kolmeks suuremaks osaks. Esimeses osas viiakse läbi intervjuud täitjatega – kuidas nemad näevad ideaalset tööde tellimise protsessi tarkvaraarenduses. Intervjuudes keskendutakse projektidele, mille käigus valmib rakendusele ka visuaalne väljund ehk tekib kasutajaliides. Intervjuude vahel analüüsitakse arutletud teemasid ja kohandatakse järgmise intervjuu eesmärki vastavalt.

Teises osas viiakse läbi intervjuud tellijatega ja uuritakse tarkvaraarenduse projekti tööde tellimise protsessi tellija seisukohast. Intervjuudel keskendutakse projektidele, kus tellitaval rakendusel on olemas ka kasutajaliides. Intervjueeritavateks on tarkvara kavandamisel osalenud projektimeeskonna liikmed, ametinimetust või rolli ei arvestata. Uuritakse, kas tellija on juba praegu ära kirjeldanud tarkvara kavandamise protsessi, mis aitab tooteomanikul oma rolliga harjuda. Intervjuude vahepeal analüüsitakse käsitletud teemasid ja kohandatakse küsimusi vastavalt.

Magistritöö kolmandas osas analüüsitakse intervjuude käigus saadud informatsiooni ja leitakse vastuste ühisosa küsimustest lähtuvalt. Magistritöö tulemuseks on tarkvara kavandamise protsessi kirjeldus tooteomanikule. Lisaks kaardistatakse kuidas jagunevad rollid ja vastutusalad.

2. Metoodika

Magistritöö on kvalitatiivne uurimustöö – kasutatud on intervjuudel põhinevat juhtumianalüüsi meetodit. Töö eesmärgi saavutamiseks kasutatakse lisaks intervjuudele ka tellija poole tööprotsesse ja rolle kirjeldavat dokumentatsiooni. Magistritöö tulemusena pakutakse üks võimalik viis, kuidas tarkvaratellimust kavandada. Magistritöö tugineb paljuski autori senisele kogemusele tarkvaraarendusest avaliku sektori asutustes. Magistritöösse kaasatakse neli täitja poole ettevõtet, keda võiks jagada kahte erinevasse gruppi – traditsioonilised tarkvara arenduse ettevõtted, kes pakuvad „võtmed kätte“ lahendusi ja ettevõtted, kes keskenduvad teenuse disainile. Lisaks intervjueeritakse kuute tellija poole ettevõtet, kes on riigihanke kohuslased. Konfidentsiaalsuse kokkuleppeid intervjuude käigus ei sõlmitud. Intervjuudest saadud arvamusi ja ideid kasutati järgnevates intervjuudes, et saada arvamusi eelnevate ideede kohta. Vahekokkuvõtete kaudu jõuti lõpliku tööprotsessi ja vastutusvaldkondi kirjeldava dokumendini.

Uurimustöös kasutati andmete kogumiseks intervjuusid, kuna puuduvad välja kujunenud standardid, kuidas tarkvaraarendust peaks kavandama. Väga selgelt on välja kujunenud standardid, kuidas käib tarkvaraarendus ja metoodika, kuidas tarkvaraarenduse protsessi on võimalik juhtida. Puuduvad parimad praktikad, kuidas peaks toimuma tarkvara kavandamine ja milles seisneb tooteomaniku olemus. Tarkvaraarenduse kavandamise kohta pole võimalik küsida küsimusi konkreetsete praktikate, standardite kohta ja vastaja hinnanguid neile. Ülevaatlikud intervjuud, mis kaardistavad kehtivat olukorda, on informatsiooni kogumiseks ainuke võimalus. Intervjuu käigus on vaja küsimustele tekitada kontekst, kuna tarkvara kavandamise juures kasutatavad mõisted võib omada erinevaid tähendusi.

Intervjuude suurimaks probleemiks on nende mahukus ja ebamugavus. Intervjuudele vastamine eeldab mõlemalt osapoolelt arvestatavat ajalist investeeringut. Kokku on vaja leppida sobiv aeg ja koht, ning loota et lepitud aja jooksul on võimalik jõuda ammendavate vastusteni.

Valimisse valisin neli täitja poole ettevõtet, kellega mul pole, ega pole olnud, varasemat tööalast suhet. Täitja poole ettevõtted jagasin omakorda kaheks, traditsioonilist tarkvaraarendust pakkuvad ettevõtted ja kasutusmugavuse disaini teenust pakkuvad ettevõtted. Tieto ja CGI kirjeldavad ennast kui täislahendusi

pakkuvaid ettevõtted. Trinidad ja DUX keskenduvad teenuse disainile. Suurimaks riskiks pean olukorda, kus ettevõte peab intervjuu läbiviijat potentsiaalseks kliendiks ja käitub vastavalt.

Soovisin valimisse kaasata ka kõik ministeeriumite alluvuses olevad infotehnoloogia kompetentsikeskused, kes korraldavad riigihankeid, mille raames hangitakse raamistike rakendamise kompetentsi. Paraku paljud ametid minu pöördumistele ei vastanud ja asendasin neid valimis jooksvalt sarnaseid hankeid korraldavate asutustega. Suurimaks riskiks pean olukorda, kus intervjueerivat asub kaitsepositsioonile ja avatud suhtlus tarkvaraarenduse võimalustest pole võimalik.

Startup-ide maailm on näidanud, et rakenduste edukuse määrab paljuski selle esteetika (Pun, 2017). Välimuse (kasutajaliidese) kaudu tarkvara kirjeldamine on saanud tarkvaraarenduse juures parimaks praktikaks (Best-Practice Software Engineering, 2013). Sellest tulenevalt olen valinud ka oma küsimused intervjuudele:

- Kuidas te kirjeldaksite ideaalset tööprotsessi täitja ja tellija vahel? – Küsimusega loodan saada vastuse olukorras, kus raha ja aeg on piiramatud.
- Kuidas peaks kirjeldama äriprotsessi? – Küsimusega loodan leida viise, kuidas lihtsustatult kirjeldada äriprotsesse.
- Kui detailne peaks olema prototüüp? Kuidas seda hinnata? – Küsimusega loodan saada vastuse, kuidas objektiivselt mõõta prototüübi küpsust.
- Kas või millistel juhtudel peaks prototüübi kõrval olema ka detailne spetsifikatsioon? – Küsimusega loodan saada vastuse, kuidas peaks kirjeldama dokumentatsiooni taset projekti juures.
- Kuidas peaks prototüüpi hindama? – Küsimusega loodan saada kirjelduse täpsele meetrikale või protseduurile, kuidas peaks prototüüpi hindama.

Kõik intervjuud salvestatakse ja magistritöö lisadesse kirjutatakse ümber intervjueeritava vastused, mis puudutavad uuringu küsimusi. Magistritöö käigus uuritakse iga küsimust eraldi.

3. Olulised tarkvaraarenduse põhimõtted

Agiilse tarkvaraarenduse manifest toob välja neli põhiväärtust: inimesed enne protsesse; töötav tarkvara enne põhjalikku dokumentatsiooni; pidev kommunikatsioon on tähtsam kui lepingud ja muutustele reageerimine on tähtsam kui plaani jälgimine (Manifesto for Agile Software Development, 2001).

Populaarne internetikommuun IT praktikutele CodeProject, võtab eduka tarkvaraprojekti alustalad kokku järgnevalt (Salonek, 2009):

1. Õige meeskond
2. Probleemi defineerimine
3. Efektiivne koostöö
4. Kommunikatsioon
5. Targalt töötamine
6. Protsessi pidev parandamine
7. Selge siht

Õige metoodika valiku ja tarkvaraarenduse sisseostmisega on võimalik ära katta neli punkti seitsmest. Efektiivne koostöö, targalt töötamine, protsessi pidev parandamine ja õige meeskond. Tarkvaraarenduse teenusena sisseostmine annab tellijale paindlikkuse, leida igale tööle sobiv meeskond ja metoodikate valik vastab küsimustele kuidas peaks käima tööprotsess. Arendusmeeskonna sisseostmine aga loob uue probleemi, mida käsitlen järgmises lõigus, kuidas tekib huvide konflikt. Äriprotsessidel põhinevast tarkvara kavandamisest ehk probleemi defineerimisest käsitlen ülejärgmises lõigus. Kuna meeskonna liikmed võivad asuda füüsiliselt erinevates asukohtades, siis tekivad ka probleemid kommunikatsioonis, millest räägib kolmas lõik.

3.1. Tarkvaraarenduse sisseostmine

Riigi Kinnisvara AS otsustas tarkvara arendamist sisse osta teenusena, kuna vajaliku kompetentsi omamine oli muutunud kalliks. Suure arendusmeeskonna omamine ei olnud otstarbekas, kuna nende tööaja täiemahuline sisustamine ei olnud alati kindel. Lisaks tekivad motiveerimise probleemid kui pole piisaval hulgal uusi ja huvitavaid projekte, mille raames töötada (Orient, 2017). Tarkvaraarenduse sisseostmine lahendab mõlemad probleemid – töö ressurss on paindlik ning vajadusel on alati

võimalik projektimeeskonda vahetada, kui on näha, et meeskonnal puudub huvi projekti vastu.

Keskendun oma magistritöös ainult tarkvaraarenduse projektidele, kus lahenduse teostus tellitakse teenusena sisse. Sellises olukorras on kõige keerulisem arendusvajadusi täitjale edastada – töötatakse erinevates keskkondades ja funktsionaalsused, mis tunduvad tellijale elementaarsed, võivad täitjale jääda märkamatuks või arusaamatuks. Need on ka välja toodud kui enamlevinud probleemid, mis tekivad teenuste sisseostmisel IT sektoris (McCray, 2012). Riigi Kinnisvara AS on teenusena ostnud sisse ka muid infotehnoloogia osakonda puudutavaid valdkondi, näiteks serverite omamine ja majutus, töökohtade haldus, litsentside haldus, klienditugi ja võrguteenus ning võib tõdeda, et tulemused on olnud ootuspärased – projektid on õnnestunud. Tarkvaraarendus teenusena vajab veel parandamist.

Tarkvaraarenduse sisseostmine on aga väga kõikuva kvaliteediga. Kvaliteediks pean siinkohal lõppkasutaja hinnangut valminud toodetele. Tarnitud tarkvara vastab kõikidele kirjeldatud nõuetele, kuid kasutaja kommentaar on, et see pole loodud inimesele kasutamiseks. Kasutusmugavuse probleeme on võimalik hiljem parandada, kui selle eest on klient nõus maksma. Magistritöös lähtun seisukohast, et piisava ettevalmistusega on võimalik kasutatavuse probleeme vältida. Probleem iseenesest pole uus, millele viitab ka näiteks metodoloogiast kõneleja Yegor Bugayenko (Bugayenko, 2015). Tema hinnangul on suurimaks probleemiks täitja ja tellija vaheline suhte iseloom. Täitja arvab, et kõige parem klient on tellija, kes maksab kõik kinni, kuna 90% tarkvaraarenduse ettevõtete eelarvest moodustab palk, kuid tellija arvates on parim täitja see, kes on näiliselt kõige odavam.

Kõik tarkvaaraarenduse ettevõtted väidavad, et nad on väärikateks partneriteks tellijatele. Tegelikult ollakse aktiivseteks partneriteks ainult senikaua, kuni tellija on nõus maksma – mis kasumit taotleva ettevõtte seisukohast ongi täiesti õige. Kui edukas on projekt, ei oma täitja seisukohalt mingit tähtsust – täitja ja tellija arusaam edukast projektist on erinev. See on tarkvaaraarenduse sisseostmise probleemide alus. Tellija loodab, et ta ostab sisse meeskonda, kes seisab nende projekti eest, kuid täitja loodab tegelikult leida klienti, kes maksab palju ja õigeaegselt. Loodava tarkvara kvaliteet on alati teisejärguline. Muidugi ei tooda keegi meelega halva kvaliteediga tarkvara, aga

kui tellija pole nõus rakenduse silumise eest maksma, siis täitja seda ei tee. Parima teenuse pakkumine nõuab täitja poolelt märgatavat investeeringut, et kinni maksta uuesti tegemised ja muudatused, mida pole tellija pool kinnitanud.

3.2. Rakenduse kavandamine

Kasutajaliidesega tarkvaraarenduse projektide lahutamatuks osaks on visuaalne disain. Kui rakendus peab olema ka intuitiivne ja kasutajasõbralik, siis luuakse visuaalne kontseptsioon interaktsiooni prototüüpimise käigus. Interaktsioonist lähtuv disain kui väljend, pärineb tegelikult traditsioonilisest tööstusest ja tarkvaraarenduses sellist väljendit väga tihti ei kohta (Seungwoo Maeng, 2012). Tarkvaraarenduses on kasutusel väljend *interacton driven development*, mille all tegelikult mõeldakse viisi, kuidas oma rakenduse koodi organiseerida klasside ja meetodite mõistes. Tarkvaraarenduses, kus loodaval rakendusel on olemas kasutajaliides, lähtutakse peamiselt rakenduse käitumisest ehk kasutatakse *behaviour-driven development* (BDD) metoodikat oma rakenduste mudeldamisel – kõik saab alguse kasutuslugudest.

Käitumise põhine tarkvaraarendus näeb rakendusi staatiliselt ning seda peetaks üheks põhiliseks BDD murekohaks – kui kasutaja avab mingi lehekülje, siis talle kuvatakse ette defineeritud infot, kasutuslood ei kirjelda struktuuri ega viisi, kuidas seda tehakse. Kasutuslood kirjeldatakse ära enne arendustööde alustamist ja neid käsitletakse kui lõplike lahendusi. BDD eeldab, et kõik funktsionaalsused on kirjeldatavad kasutuslugudega – vorm, kuidas lugu kirjeldada, on lahtine. Sellises olukorras jääb tihtilugu asjade visuaalne kirjeldamine tahaplaanile. Asjade visuaalse kujutamise eiramine toob endaga kaasa olukorra, kus samu asju tehakse mitu korda ümber, et leida õige esteetiline väljund. Aja puuduse tõttu, lepitakse tihtilugu "käib kah" lahendustega. Sellist olukorda üritab vältida *raprid prototyping* (Adiseshiah, 2016) metoodika, kus kõik üritatakse töötavate rakendustega ära kirjeldada. Keerukamate rakenduste puhul pole selline lähenemine mõeldav, kuna seda loetakse väga kalliks.

Eestis kasutavad tarkvaraarenduse ettevõtted kohati mitut väljendit, mis kõik kirjeldavad interaktsiooni – kasutajakogemuse disain ja teenuse disain. Kui sinna juurde panna ka lõpliku toote valmistamine, võiks seda kokkuvõtlikult kirjeldada kui interaktsiooni disainist lähtuv tarkvaraarendus. Teenuse pakkujad võtavad protsessi lühidalt kokku järgnevalt. Alguses koostatakse ärianalüüs, selle peale luuakse prototüüp koos oma mitme erineva iteratsiooniga ja kõige lõpus toimub prototüübist

toote arendamine. Äriprotsessi ja interaktsiooni kavandamise kaudu on tarkvaraarenduse projektile võimalik luua selge lähteülesanne ja veenduda, et defineeritud probleem on lahendatav. BDD-st tuntud kasutuslugudena rakenduse kirjeldamine on lihtne ja kiire, kuid jätab arvestamata kasutajakogemuse – mille tulemusena on võimatu ennustada, milline hakkab valminud rakendus välja nägema. Ärianalüüsi ja prototüübi kirjeldamine on keeruline protsess, kuna võimatu on ennustada täpselt kui palju kulub aega ärianalüüsi koostamisele. See sõltub väga palju analüütiku varasemast kogemusest kirjeldatava valdkonnaga ja intervjueeritavate väljendusoskusest.

3.3. Suhtlus täitja ja tellija vahel

Täitja ja tellija eesmärgid lähevad tarkvaraarenduse projektis lahku nagu kirjutas Yegor Bugayenko. Täitja soovib alati kulutada rohkem aega ülesannetele, kui seda peab vajalikuks tellija. Täitjal kulub alati palju aega, et aru saada lähteülesandest, ettevalmistusaega aga ei arvestata reeglina projekti tööülesannete hulka (Usmani, 2013). Seega vastutus projekti kiireks stardiks saab lasuda ainult tellija käes. Kommunikatsiooni projektimeeskonna sees on uuritud varasemalt ja see näitab, et maha kirjutatud dokumentatsioon ei ole väga efektiivne (Agile Practices and Principles Survey Results: July 2008, 2008). Uuringust tuleb välja järgnev tabel (Tabel 1), skaalaks on väärtused -5 kuni 5, ehk väga ebaefektiivne kuni väga efektiivne.

Tabel 1. Suhtlusviis

Suhtlusviis	**Meeskonna sees**	**Koos tellijaga**
Näost näkku	4,25	4,06
Näost näkku koos tahvliga	4,24	3,46
Diagrammid	2,54	1,86
Suhtlustarkvara	2,1	0,15
Dokumentatsioon	1,84	1,86
Telefonikonverents	1,42	1,51

Videokonverents	1,34	1,62
E-post	1,08	1,32
Spetsifikatsioon	-0,34	0,16

Tabelist võiks järeldada, et kõige edukam suhtlusviis tellija ja arendaja vahel on näost näkku vestlus. Praktikas on sellise situatsiooni saavutamine, et tellija saab pidevalt arendajaga koos istuda haruldane. Tarkvaraarendus jälgib endiselt väga tihti tavalist kosk mudelit, kus alguses lepitakse kokku rakenduse spetsifikatsioon, mis seejärel teostatakse. Ülesannete täpsustamine konsultatsioonide näol on pigem erandlikud. Tarkvaraarenduse eelarves puuduvad vahendid konsultatsioonide eest tasumiseks. Lisaks ei soovi tellija tasuda arveid ülevaatlike koosolekute eest.

Isegi kui projektile on varasemalt valmis kirjutatud detailne spetsifikatsioon ja eelarves leiduvad vahendit täiendavaks analüüsiks, siis vahetu suhtlus tellijaga on raskendatud. Riigi Kinnisvara AS praktika näitab, et suuremate osakondade juhtidega näost näkku kohtumise korraldamiseks kulub vähemalt nädal. Jooksvalt vastavalt vajadustele kohtumiste korraldamine on üldjuhul võimatu seda enam, et ka täitja pool ei pruugi asuda samas linnas. Olukorras, kus tooteomanik vastutab korraga väga erinevate ülesannete eest, näiteks vastutab jooksvate tööde planeerimise ja sujumise eest, mõtleb välja uusi töökorraldusi, parandab olemasolevaid korraldusi, lahendab jooksvaid probleeme, vastutab protsesside eest, standardiseerib tegevusi – polegi võimalik, et jooksvalt on võimalik probleemidele vastuseid saada (Mett, 2017). Sellise asünkroonse suhtluse olukorras on spetsifikatsioonide kaudu tellimuste esitamine ainuvõimalik.

Takistuseks on inimeste erinevad arusaamad probleemide kirjeldamisest ja nende tõlgendamisest. Ideaalses olukorras on tellijal olemas analüütik, kes suudab olla hea suhtleja tellija ja täitja vahel, kes kirjeldab probleemi piisavalt detailselt, et see on üheselt mõistetav (Rattur, 2017). Analüütiku puudumisel tuleb kohtumistele määratud aega võimalikult efektiivselt ära kasutada ja selleks on mõlemal osapoolel vaja teha eeltööd. Enne koosoleku algust peab olema teada probleem, mida hakatakse arutama

ja ülevaade tarkvaraarenduse projektist terviklikult. Ülevaade projektist loob probleemi lahendamisele konteksti.

4. Uuringu läbiviimise kirjeldus

Magistritöö intervjuudesse valiti osalema neli ettevõtet täitja poole pealt ja kuus ettevõtet tellija poole pealt. Intervjueeritavatele tutvustati esmase kontakti e-kirjas magistritöö eesmärki – teisi ettevalmistavaid materjale ei olnud. Igale intervjuule kulus umbes üks tund. Intervjuud lindistati, kuid magistritöös on välja toodud ainult küsimused ja vastused, mis on seotud käesoleva magistritööga. Intervjuude käigus ei ole keegi avaldanud soovi jääda anonüümseks.

Keskendutakse kolmele põhiteemale, kuidas piisavalt täpselt kirjeldada, kuidas kavandatav tarkvara lahendab lähteülesande. Kuidas kontrollida, et kavandatav tarkvara on teostatav ja saada kavandatava lahenduse kohta kiiret tagasisidet. Kontrollimise osas keskendun prototüüpimisele ja selle testimisele.

Igast intervjuust on autor võtnud kaasa midagi järgmistele intervjuudele ja esitanud välja pakutud lahenduste kohta küsimusi. Niimoodi tegutsedes oli võimalik kujundada arusaam parimatest praktikatest, mida asutused praegu teevad ja mida peaks tegema. Intervjuudest selgus, et oluliselt erinevalt näevad tellija ja täitja toote visiooni hoidja rolli. Sellest tulenevalt lisasin lõigu visioonist tarkvara kavandamise peatükki.

Metoodika juures kirjeldatud riskid osaliselt täitusid, üks täitja poole esindaja nägi intervjuu läbiviijas potentsiaalset klienti, üks tellija kaitses oma asutuse tööprotsesse ning üks intervjuu tellijaga pidi ootamatult lõppema.

Intervjuu käigus loodi küsimustele kontekst, kuna vastuseid sooviti saada tarkvara kavandamisest lähtuvalt. Uuriti täpsemalt, kuidas peaks tellija ja täitja käituma projekti alguse etappides, millise eeltöö peaks ära tegema tellija, enne kui küsitakse pakkumust. Vaatamata sellele on paljud vastused väga üldistavad kogu tarkvaraarendusele. Prototüübi mõiste jäeti intervjuu läbiviija poolt teadlikult lahtiseks, et vastajad saaksid seda ise tõlgendada.

5. Analüüs

Intervjuude käigus arutleti paljude teemade üle, mis ei kuulu käesoleva magistritöö hulka, kuid peamiselt keskenduti kolmele teemale: kuidas kirjeldada äriprotsessi ja kes peaks seda tegema; millal ja kuidas peaks valmima prototüüp; kuidas veenduda, et loodav tarkvara tõesti lahendab ärilist probleemi.

Kuigi käsitletavad teemad olid kõikides intervjuudes ühesed, siis peab vastuste tõlgendamisel tegema teatavaid üldistusi, et saaks vastuseid sisuliselt võrrelda. Kõikide intervjueeritavatega räägiti põhjalikult tarkvara kavandamise protsessist ja kuidas kirjeldada äriprotsesse. Testimine ja prototüüpimine on valdkonnad, mis pole kõigile vastajatele tuttavad, seega ka on vastused nendele küsimustele väga erineva põhjalikkusega.

Magistritöös analüüsin kõiki küsimusi eraldi. Toon välja täitja poole arvamuse, tellija arvamuse ja enda poolt tehtud järelduse. Küsimuse analüüsi käigus võtan arvesse ainult intervjuusid, mis puudutasid püstitatud küsimust. Vastuse puudumist ei tõlgendata. Vastuste andjad on omavahel võrdsed, ühe arvamus ei ole tähtsam kui teine.

5.1. Kuidas kirjeldaksite ideaalset tööprotsessi täitja ja tellija vahel?

Täitja pool soovib läbi intervjuude jõuda selguseni, mida tellija soovib ja pakkuda omapoolse lahenduse (Lember, 2017) – võttes sellega endale tooteomaniku rolli. Täitjal võiks olla vaba voli korraldada nii palju intervjuusid kui on vaja, et jõuda selguseni milline on ideaalne äriprotsess. Täitjal on võimalus välja uurida, kes on projekti sihtgrupid ja neid kaasata lõpliku lahenduse väljatöötamise juurde. Intervjuude tulemusena valmis üks konkreetne lahendus, mida testitakse lõppkasutajate peal. Selleks lahenduseks võib olla detailne prototüüp traatmudelis või visuaalne ja detailne staatiline pilt. Vajadusel seda korrigeeritakse, kui lõppkasutajate testimisel esineb probleeme. Detailne prototüüp koos äriprotsessi kirjeldusega on sisendiks arendajatele, kelle vastutusele jääb lõplik projekti valmimine.

Tellija pool on üheselt nõus, et visiooni kandja rollis peavad nemad olema, kuid puudub ühtne arusaam, kuidas see peaks toimuma. Metoodika valik dikteerib projekti teostamise protsessi – metoodika valitakse vastavalt vajadusele. Nõustutakse, et

kaasamine ja pidev kommunikatsioon on olulised. Lähteülesandena on vaja kirjeldada detailne tööprotsess koos vajaliku probleemi lahendusega. Prototüüpimist ja testimist peetaks heaks tavaks, kuid sellest loobutakse, kui selle jaoks ei leita raha. Teostuse ehk arendustööde tellimise sisendiks peab olema detailne spetsifikatsioon ja nõuete nimekiri, mille vastu on võimalik teostust kontrollida. Välditakse ülesandeid, mille võimalik lahendus ei ole prognoositav. Erandina võiks välja tuua Eesti Energia, kes on tarkvaraarenduse jaganud kaheks. Visuaalne väljund, kaasa arvatud prototüübid luuakse iseenda poolt ja serveri poolse rakenduse arendus tellitakse majast väljast sisse. Sedasi käitudes on võimalik olla kindel lahenduse tulemuses. Tellija pool toob välja mitmeid erinevaid muutujaid, mis mõjutavad tellimisprotsessi valikut: ajalised, rahalised, oskuslikud ning põhimõttelised piirangud (Tammik, 2017).

Ajaline piirang on projektide juures kõige levinum. Piiranguks võib olla seadusest tulenev tähtaeg, eelarveline piirang, et projekt peab valmis saama kindlaks eelarveperioodiks või ka kunstlikult ette seatud tähtaeg. Ükskõik milline põhjus ajalisel piirangul on, leitakse, et sellisel juhul on vaja täpselt kirja panna projekti vastuvõtmise kriteeriumid ja erinevatele tegevustele seada tähtajad – paratamatult hakatakse kasutama koskmudelit oma projekti teostamisel, kuna nii on kõige lihtsam omada ülevaadet projekti valmimise kohta. Prototüüpimine ja lõppkasutaja testimine jäetakse projektist välja, kuna lisaväärtuse mõõtmine on keeruline. Ajaline piirang loob olukorra, kus rakendust tellitakse nõuete nimekirja mitte äriprotsessi täitmise järgi.

Tähtsa piiranguna tuuakse välja raha omaniku küsimus. Arendustööde hinda ei mainita. Kui tellija ise pole raha omanik, siis nähakse, et ainuvõimalikuks viisiks tarkvara tellimisel on nö „võtmed kätte“ lahendused – tarkvara hakatakse tellima nõuete nimekirja järgi. Kui raha on vähe, siis loobutakse esmajoones asjadest, millel puudub otsene kasulik väljund, loobutakse prototüüpimisest, atraktiivsest disainist ja testimisest. Piiratud ressurssidega projekti õnnestumine eeldab, et projektil on väga tugev visioon – kohe alguses on teada, mida täpselt on vaja teha ja kuidas, oluline roll on visiooni hoidjatel. Sellise meeskonna leidmine kõikidele projektidele aga tihti ei õnnestu.

Oskuslikeks piiranguteks loetakse olukorda, mille meeskond on endale ise loonud ehk projekti lahendamise metoodika ja rollid valitakse vastavalt olukorrale. Kui tellija

meeskonnas puudub kompetents äriprotsessi kirjeldamiseks ja fookuse hoidmiseks, siis tellitakse seda täitjalt sisse (Mett, 2017). Ollakse nõus, et alati ei ole lõpptulemuseks see, mida loodeti, aga kuna projekti vastuvõtmine käib nõuete nimekirja järgi, siis lepitakse olukorraga. Projekti õnnestumisse loodetakse panustada järjest kasvavate funktsionaalsete nõuete nimekirjadega.

Põhimõtteline piirang on viis, kuidas üldse projekte alustatakse ja tarkvaraarendust tellitakse. Projekti kirjeldamisel lähtutakse hetkel kehtivatest protsessidest ja tehnoloogilisest paradigmast. Üritatakse digitaliseerida ja ühtlustada seda, mida juba praegu tehakse. Ei osata arvestada võimalustega, mida tehnoloogia võib pakkuda. Projektide kirjeldamine toimub viisil, mis küsib „Mida meil on vaja teha?“, mitte ei lähtuta probleemist „Mis mure on meil vaja lahendada?“. Kehtiva töökorralduse digitaliseerimine seab projektile kindlad piirid ja lõpptulemus on ennustatav. Samal ajal ei arvestata tehnoloogiliste võimalustega ja loodava projekti võimalik väärtus on väike.

Lähenemine, kus täitja poole esindaja on visiooni hoidja, töötab ainult juhtudel kus loodava tarkvaraarenduse projekt ei ole põhiprotsessi osa – see kattub ka tellija poole arvamustega (Püüa, 2017). Põhiprotsesside arendamise korral on nii täitja kui ka tellija põhimõtteliselt ühel meelel, kuidas tarkvara kavandamise tööprotsess peaks välja nägema, kuid vastutusalad jagunevad erinevalt. Kõik osapooled on nõus, et esimene tarkvara kavandamise samm peab olema ärianalüüs. Ärianalüüs peab koostama tellija, kuid selle kirjeldamisel on tal analüütiku abi vaja. Arvamused lähevad lahku, kuidas peaks kirjeldama töökäske. Tellija pool leiab, et kõikidel töökäskudel peab olema konkreetne vastuvõtmise kriteerium, täitja pool leiab, et äriprotsessile viitamine on piisavalt täpne kirjeldus töökäsule. Õige lahendus sõltub ettevõtte töökorraldusest, kui vahetu suhtlus täitjaga on võimaldatus pole töökäsu täpsusel tulemusele olulist mõju.

5.2.Kuidas peaks kirjeldama äriprotsessi?

Täitja poolel on ühine selge arusaam, kuidas äriprotsessi peaks kirjeldama, kuid kirjelduse peaksid looma nemad ise. Ideaalses maailmas kirjeldab tellija pool oma äriprobleemi ja täitja kirjutab sellest valmis äriprotsessi. Kasutades kirjeldamiseks enamlevinuid standardeid ja informatsiooni kogutakse intervjuudega (Sarapuu, 2017). Tellija lõpuks kinnitab valminud tulemuse, mitte ei vali võimalike lahenduste seast parimat.

Tellija pool leiab, et äriprotsesside kirjeldamine enamlevinud standardeid kasutades on keeruline. Äripool ei tunne notatsiooni ja protsesside kirjeldamisel kasutatavaid mõisteid, näiteks kuidas defineerida rolli ja sisendit. Arusaamatuks jäävad abstraktsioonid nagu otsustuskohad ja algus-punktid. Pole kokku lepitud täpset käekirja, kuidas protsesse kirjeldada (Pentjärv, 2010), kardetakse teha vigu ja tulemuseks on, et kõik tooteomanike poolt kirjeldatud äriprotsessi diagrammid on erinevad – neid on raske lugeda. Tellija pool leiab, et aktsepteeritav on olukord, kus äriprotsessid on kirjeldatud ära tekstiliselt – kokku lepitud struktuuri järgi. Olgu selleks kas kasutuslugu või ette antud vorm, tähtis on, et sellega tuleks kaasa täitmise instruktsioonid.

Äriprotsesside kirjeldamine tundub olema valdkond, mille kirjeldamine on tellija poolele kõige segasem. Äriprotsesse ei osata kirja panna kokku lepitud notatsiooni kasutades või ühes äriprotsessis proovitakse kirjeldada kõik võimalikud erijuhud. Taristu projektide puhul tuntakse, et äriprotsess puudub sootuks. Sobiliku lahendusena pakutakse välja, et äriprotsess koostatakse meeskonna poolt, kuhu on kaasatud analüütik, kes tunneb notatsiooni (Rattur, 2017).

5.3.Kui detailne peaks olema prototüüp? Kuidas seda hinnata?

Täitja leiab, et prototüüp peaks ära katma ühe konkreetse põhiprotsessi või alamprotsessi otsast lõpuni (Sarapuu, 2017). Kirjeldused, mida mingi konkreetne nupp teeb ja kuidas ta käitub, peaks olema ära kirjeldatud eraldi dokumentatsioonis, aga ainult vajadusel, kui tellija seda nõuab. Parimat võimaliku viisi, kuidas prototüüp peab olema kirjeldatud, ei oska täitjad välja tuua. Rõhutakse juba välja kujunenud praktikatele, mis on ettevõtetes juba välja kujunenud.

Tellija leiab, et ideaalne prototüüp peaks ära kirjeldama ühe äriprotsessi otsast lõpuni, kõikide oma detailsustega. Kõikidel nuppudel peaks küljes olema täpne kirjeldus, mida ta teeb. Ainult nii on võimalik veenduda, et analüütik pole midagi ära unustanud (Mett, 2017). Kuidas peaks olema detailne prototüüp lahendatud, ei osata vastata, kuna ei tehta vahet, kas tegemist on traatmudeli, staatilise HTML-i või funktsionaalse prototüübiga.

Küsimusele, kuidas peaks hindama prototüübi detailsust, ei osata vastata. Intervjuude põhjal selget vastust ei ilmnenud. Tundub, et detailsus peab olema kokkuleppeline

tellija ja täitja vahel ja sõltub rohkem sellest, mida prototüübiga tulevikus edasi tegema hakatakse. Kui prototüüpi loetakse dokumentatsiooni osaks, siis eeldatakse, et see on väga detailne. Kui tegemist on ainult visuaalse sisendiga projekti teostusesse, siis lühikese aja möödudes vajadus prototüüpi säilitada ja uuendada kaob.

Prototüüpi kasutatakse tagasiside saamiseks, kas mõlemad osapooled on ülesande kirjeldusest ühte moodi aru saanud. Tellija pool ei oska prototüübi testimist väärtustada ja on nõus ükskõik millise lahendusega – niikaua kuni see kinnitab tema arusaama lähteülesandest. Prototüübi vajalikkust on vaja tellija poolele rohkem selgitada. Prototüübi eesmärgiks ei ole tagasiside saamine tellijalt, vaid veendumine, et välja pakutud lahendus tegelikult töötab. Seega peaks prototüübi valimisel lähtuma lihtsusest ja valmimise kiirusest. Võimalikult kiire tagasiside saamine on kõige prioriteetsem.

5.4. Kas või millistel juhtudel peaks prototüübi kõrval olema ka detailne spetsifikatsioon?

Dokumentatsiooni kohapealt puudub ühine arusaam nii täitja kui ka tellija poole peal. Lühidalt on vastus, et vastavalt vajadusele. Ülimalt detailset dokumentatsiooni, mis kirjeldab ära täielikult tarkvara arhitektuuri, peetaks liiga kulukaks. Teisest küljest dokumentatsiooni puudumine loob olukorra, et inimeste vahetumisega kaovad ka teadmised. Võib öelda, et dokumentatsioon peab kirjeldama ära äriloogika, ehk kõik andmete töötlused peavad olema kuskil kirjas. Kuidas seda täpselt kirja panna, sõltub analüütiku enda vabast valikust. Võimalik on see kirja panna juba prototüübi juurde, kuid sellisel juhul on vaja kogu aeg prototüüpi kaasajastada. Levinuimaks praktikaks on äriloogika kirjeldamine kasutuslugudena. Kasutuslood koos prototüübiga on sisendiks tarkvara tellimusele. Õiget vastust dokumentatsiooni taseme hoidmise kohta pole – kõik sõltub analüütiku harjumustest.

5.5. Kuidas peaks prototüüpi hindama?

Täitja poolel on olemas selge visioon, kuidas peaks käima prototüübi testimine. Koostada tuleb testgrupp ja mängida läbi kokku lepitud stsenaarium. Grupp koosneks inimestest, kes on tulevikus tarkvara kasutajad, kuid kes pole otseselt projektiga seotud. Grupil lastakse läbi mängida ette defineeritud stsenaarium ja lastakse kirjeldada enda tegevusi. Mõõdetakse ülesande täitmise kiirust ja tehtud vigade arvu.

Arvamust, kuidas midagi võiks olla lahendatud, ei küsita. Testimise käigus saadud tagasiside põhjal tehakse parandusi prototüübile.

Tellija poolel puudub selge arusaam, kuidas peaks prototüüpi testima. Mitte üheski tellija poole intervjuus ei toodud välja võimalust mõõta testimise tulemust. Prototüübi testimiseks peetakse subjektiivse arvamuse avaldamist. Prototüübi testimist nähakse kui vaheetappi enne arendustööde algust, et veenduda, kas tellitud lahendus on õige.

Testima peab stsenaariumit ja mõõtma peab ülesande soorituse kiirust. Lisaks peab küsima kasutajate arvamust prototüübi kohta, kuna võib juhtuda, et midagi olulist on ära ununenud. Lihtsad asjad, mis otseselt ei aita kaasa ülesande lahendamisele, võivad osutuda tellija silmis väga väärtuslikeks. Lisaks loob arvamuse küsimine tellijas tunde, et teda on kaasatud rakenduse disaini.

6. Tarkvara kavandamine

Intervjuudest võib järeldada, et kõikidel osapooltelt on tarkvaraarenduse protsessist sarnane arusaam. Kõik peab algama lahendatava probleemi sõnastamise ja äriprotsessi kaardistamisega. Äriprotsessi põhjal valmib prototüüp ehk lahenduse kontseptsioon, mida testitakse lõppkasutajate peal. Erinevused on vastutusalades ja detailsuses. Täitja poole ettevõtted soovivad kõike ise teha ja teostatud tööde eest arve esitada. Ideaalne ärianalüüs peab olema mahukas, hõlmama kõiki huvigruppe ja vestelda tuleb mitme esindajaga igast huvigrupist. Optimaalseks ärianalüüsiks loetakse üksikute võtmeisikutega peetud intervjuusid ja selle põhjal arvamuse kujundamist.

Ärianalüüs on ettevõtetes traditsiooniliselt ärianalüütiku vastutusala, kuid paljude ettevõtete struktuurides puudub ärianalüütiku ametikoht. Sarnases olukorras on ka RKAS, ärianalüütiku ametikoht on vakantne juba pikemat aega. Puudub konsensus, kas ärianalüütiku teenuseid on parem sisse osta või hoida seda kompetentsi ettevõtte sees (Hathaway, 2009). Hetkel on ärianalüütiku ülesanded jagatud laiali osakonna juhatajatele. Kehtivast olukorrast tingituna peab olema võimalik ärianalüüsi koostamine ettevõtte sees ja sellest lähtuvalt peab jagama ära ka rollid uue tarkvara kavandamisel.

6.1. Rollid

Käesolevas lõigus kirjeldatakse kolme osapoole vastutusalad: analüütiku, projektijuhi ja tooteomaniku.

6.1.1. Tooteomaniku vastutusalad

- Teab, mis on tema toote protsessi juures kriitilise tähtsusega. Tooteomanik peab teadma, milline väljund on toote kasutajate jaoks kõige tähtsam ning mis on toote ülesanne kogu ettevõtte tööprotsessis. Tooteomanikul peab olema põhjalik arusaam, kuidas tema toode sobitub ettevõtte ülesesse tööprotsessi.
- Juriidiliste küsimuste lahendamine äriprotsessides (õigusaktid).
- Jälgib oma toote kasutamist, ehk kirjeldab ära meetrika, kuidas oma toote eesmärgi edukat täitmist mõõta.
- Täitja poolsetele küsimustele operatiivne vastamine kahe tööpäeva jooksul.
- Kasutusjuhendite koostamine lõppkasutajale.
- Koolitus: vajalike osapoolte väljaselgitamine, koolitusplaani koostamine.

6.1.2. Projektijuhi vastutusalad

- Arusaam tellija (äripoole) vajadustest, milline on äriline vajadus/probleem, mida tema juhitud projekt parandab.
- Projekti dokumentatsiooni tekitamine, korrashoid, kooskõlastuste korraldamine.
- Aja- ja tegevuskava planeerimine koostöös tellija ja partneritega, selle jälgimine.
- Tootepassi ja SLA haldamine, koostöös tooteomanikuga.
- Tööd takistavatest teguritest teavitamine, võimaluste piires takistavate tegurite kõrvaldamine.
- Projekti lõpetamine ja haldusele üle andmise korraldamine.
- Täitjale sisendi täpsustamine, või selle korraldamine koos toote visiooni hoidjatega.

6.1.3. Analüütiku vastutusalad

- Ärivajaduste kokku kogumine ja nende defineerimine.
- Piirab tellija nõudmisi – kuigi kõikide nõudmiste kirjeldamine on vajalik projekti edukaks lõpetamiseks, peab analüütik piirama nõudeid viisil, et need täidaks äri põhilisi vajadusi, mitte kasutajate isiklikke vajadusi.
- Tõlgib ärilised nõuded tehniliseks dokumentatsiooniks/nõueteks. Analüütik peab kirjeldama ärilised nõuded üldisesse IT arhitektuuri.
- Kontrollib nõuete vajalikkust, korraldades selleks testi, demonstratsiooni, vaatluse või analüüsi.

6.2. Äriprotsessi kirjeldamine

Projekti võimalikult kiireks ja edukaks stardiks, on kõige tähtsamaks osaks äriprotsessi kirjelduse valmimine tooteomaniku poolt. Seda peab kirjeldama tootomanik, et vältida arusaamatusi toote vastuvõtmisel. Funktsionaalsete nõuete nimekiri ei pruugi täpselt edasi anda vajadust, kuidas midagi peab tegema. Funktsionaalsus kirjeldab ainult seda, mida rakendus peab tegema. Äriprotsessi kirjeldusel peab olema kaks dimensiooni: põhiprotsess, mis kirjeldab kogu rakendust läbivat tööprotsessi ja iga etapi alamprotsess või instruktsioonide loetelu. Kirjeldamisel on vaja jälgida järgmiseid põhimõtteid (Young, 2013):

- Defineeritud notatsioon – kui tööprotsessi ei kirjeldata mingi kindla standardi järgi (BPMN, UML,ISO), siis tuleb definitsioonid kirjeldada eraldi protsessi kirjelduse juures.
- Kindlalt määratletud sisendid ja väljundid – minimaalne komplekt andmeid, et oleks võimalik langetada mingi otsus või täpne andmete loetelu, et loogiline algoritm saaks edasi liikuda.
- Kindlalt määratletud tegijad – kõikidel tegevustel peab olema kindel vastutaja, selleks võib olla ka süsteem ise.
- Kindel eesmärk – tervel protsessil või selle etapil peab olema läbi teema, mida üritab süsteem lahendada.
- Täpsemini defineeritud suhtlus protsessi tegijate vahel – vajalik juhul kui protsessis on rohkem kui üks tegija, kellel on erinevad rollid.

Protsessi kirjelduse eesmärgiks on anda ülevaade lugejale, peamiselt arendusmeeskonnale, kuidas arendatav toode toimib. Hea protsessi kirjeldus tutvustab lugejale üldist eesmärki, milleks kirjeldatud toimingud on vajalikud – luuakse kontekst. Suures plaanis jaguneb protsessi kirjeldus kaheks – üldine protsessi kirjeldus, mis loob üldise raamistiku kogu tegevuste jadale ja iga etapi detailne kirjeldus konkreetse eesmärgiga, lisaks võib olla ka kokkuvõte (Transision support, 2013). Lugejatel peab tekkima ühene ja selge arusaam miks ja kuidas midagi tehakse.

Äriprotsesside kirjeldamiseks on välja mõeldud palju erinevaid standardeid: ISO 5807:1985, UML, BPMN. Ilma tooteomaniku koolituseta (Taal, 2017) pole mõistlik nõuda mingi kindla standardi järgi kirjeldatud äriprotsessi kirjeldust. Äriprotsessi kirjeldamine võiks tooteomanikule jätta võimalikult vabad käed, graafilise notatsiooni tundmine ei pea olema kohustuslik. Agiilse tarkvara arendusprojekti üheks oluliseks punktiks on võimalikult kiire esimese versiooni valmimine (Agile Alliance, 2017) – äriprotsessi kirjeldamisele peab lähenema sarnaselt.

Kui äriprotsessis on paralleelselt toimumas rohkem kui üks tegevus, peaks äriprotsessi kirjeldamiseks kasutama diagrammi. Kui paralleelsete tegevuste olukord äriprotsessis puudub, peaks äriprotsess olema kirjeldatud loogiliste tegevuste jadana. Tooteomanik peab olema valmis etapi sisendite ja väljundite kohta andma näiteid analüütikule – projekti täpsema spetsifikatsiooni kirjeldamiseks. Võimalik vorm kuidas protsessi kirjeldada (Tabel 2).

Tabel 2. Protsessi vorm

Protsessi pealkiri	
Lühike tutvustus, miks selline protsess on vajalik ja konkreetse protsessi eesmärk	
Defineerida mõisted, mida protsessis kasutatakse	
Etapp 1	Etapi detailne kirjeldus
Etapp 2	Etapi detailne kirjeldus
Etapp 3	Etapi detailne kirjeldus
Kokkuvõte ja oodatud tulemused (väljund)	

6.3.Prototüübi kirjeldamine

Kui projektile annab skoobi eelmine peatükk, siis järgmiseks sammuks projekti alustamise juures peab olema visandamine ehk *mockup*-de koostamine. Visandamise koosolek peab toimuma näost näkku koosolekul, ning visandajaks võib olla nii projektijuht, analüütik, tooteomanik kui ka täitja poole esindaja. Tähtis on, et visandatakse maha kas terve äriprotsess või selle üks osa – etapp. Visandi peale tuleb kanda kogu informatsioon mis kasutajale on selle protsessi juures oluline. Teadmiseks tuleb võtta, et visandamise juures pole oluliseks mitte niivõrd nende kunstiline väärtus, kui mäng ja eksperimenteerimine. Võimalikult varajases projekti staadiumis on vaja aru saada, kas äriprotsessi kirjeldust on võimalik kasutajaliidese kaudu mõistlikult realiseerida. Teadmiseks peaks võtma järgmised põhimõtted (Shifflett, 2012):

1. Visandeid tuleb võtta kergelt ja huumoriga. Enamus vihikuid kuhu visandatakse näevad küll kohutavad välja, kuid nende väärtus ei ole mitte esteetikas, vaid mõtetes, mida need kritseldused endas kannavad.
2. Katseta erinevaid lahendusi. Visandamise eesmärk on katsetada erinevaid viise, kuidas informatsiooni on võimalik kasutajale kuvada. Põhimõtteliselt on olemas kolme tüüpi ideid: need, mis on suurepärased; need, mis vajavad natuke tööd et olla head; ja need, mis on halvad. Mida rohkem erinevaid visandeid luuakse, seda paremini saadakse aru, mis töötab ja mis ei tööta. Nagu

varasemas punktis on mainitud, siis täitja eesmärgiks ei ole leida parimat lahendust, vaid leida selline lahendus, mille eest on tellija nõus maksma – seega peab katsetamisega tegelema tellija.

3. Visandamise juures tuleb meeles pidada, et kõiki asju viimse detailini välja joonistada ei ole mõtet. Detaile võib visandil väljendada ka kirjalikult.

Meeles tuleb pidada, et korraga on mõistlik visandada ainult ühte tegevust kindla alguse ja lõpuga. Kui on leitud lahendus, millega kõik osapooled on rahul, tuleb välja valitud esmastest visanditest koostada viimane versioon ja koostada puhtand.

6.4.Prototüübi testimine

Töö käigus võib juhtuda, et visanditega ei teki selget favoriiti, milline lahendus oleks parim. Sellisel juhul on vaja kasutatavust testida. Ideaalses maailmas on visandid juba mingil kujul digitaliseeritud, kuid esmase testimise korraldamiseks pole digitaliseerimist tegelikult vaja. Piisab, kui visand on kantud paberile ning vajadusel varustatud tekstiliste selgitusega. Sobib ka olukord, kus tooteomanik võib vajadusel selgitada, mida on paberil kujutatud. Kasutajamugavuse testimine vajab kahte asja: stsenaariumit, mida hakatakse testima ja testis osalejaid, kes pole varem konkreetse projektiga kokku puutunud. Visandite peal testimine nõuab kasutajatelt ka ettekujutusvõimet, ning kõigile osalistele pole loovust antud võrdselt – seega kõik testimised ei pruugi anda oodatud tulemust.

Testimise käigus pannakse laua peale esimese sammu visand, reeglina on selleks rakendusse sisenemise vaade ja antakse kätte ülesanne/stsenaarium, mida kasutaja peab täitma. Iga vaate juures küsitakse testis osalejalt kolme asja:

1. Kirjelda, mida sa visandil näed?
2. Mida peab kasutaja tegema, et liikuda ülesande täitmisele lähemale (täitma lahtrid, vajutama nuppu jne)?
3. Kas siin vaates on midagi olulist puudu, kui jah, siis mis?

Kui testis osaleja vastab esimesele küsimusele õigesti, asetatakse ta ette järgmine vaade. Kui ei, siis seletatakse, mida ta oleks pidanud tegema ja siis asetatakse tema ette järgmine vaade. Arvamust, mis tema arvates võiks teistmoodi olla, ei pea küsima. Edukamaks osutub disain, mille läbimiseks testis osaleja tegi kõige vähem vigu. Testis

osaleja vastused kirjutatakse üles ja neid analüüsitakse hiljem. Vajadusel täiendatakse visandeid või edastatakse saadud informatsioon täitjale.

6.5. Visioon

Tarkvaraarendus vajab visiooni. Kuid visiooni pole võimalik juhendi järgi luua ja kirja panna. Visioon on tunnetus, kuidas vaadeldav olem peaks tulevikus välja nägema. Visioon peab esindama kõikide kasutajate ühist arusaama väärtustest, uskumustest ja eeldustest, milline peaks tegevus välja nägema tulevikus (Fernandes, 2017). Visioon peab olema süsteemne, see peab endas hoidma rolle, reegleid, suhteid ja vastutusalasid. Parimaks viisiks visiooni kujutamiseks on töövood, mis kirjeldavad kõiki nelja komponenti korraga. Visioon peab endas sisaldama hetkel kasutusel olevaid tegevusi ja protsesse, mis on planeeritud tulevikuks. Visioon kannab endas motivatsiooni meeskonnaliikmetele, andes neile selge sihi, kuhu poole ollakse teel. Lisaks annab visioon ka kõikidele asjaosalistele, kes pole otseselt projektimeeskonna osad, arusaama, kuhu poole liigub nendega seotud valdkond. Sedaviisi käitudes, muutub tarkvaraarendus pelgalt reageerivast protsessist, mõtestatud päevast päeva tegevuste jadaks.

Visiooni loomine eeldab, et asjaosalistel on võimaldatud päevast päeva tegeleda visiooni loomise ja ajakohastamisega. See seab ette piirangu, et tooteomanik kellelt eeldatakse visiooni omamist, on peaasjalikult hõivatud toote arendusega – kõik teised kohustused on teisejärgulised (Mett, 2017). Edukas visiooni hoidmine eeldab, et tooteomanik on hingestatult toote arendamise juures. Lisaks eeldab see, et tooteomanik on mingil moel hõivatud valdkonna igapäevaste tegevustega. Visiooni saab luua ainult samm sammult, väikeste muutuste kaupa, et lõhkuda *status quo*-t, mis igapäevaselt valitseb. See eeldab aga seda, et visiooni loojal endal on huvi oma tegevusi parendada ja muuta.

Atraktiivne visioon eeldab, et loodav pilt on realistlik ja usutav. See on saavutatav, kui visiooni looja on tundlik oma hallatava protsessi keerukuse suhtes – ta peab tunnetama oma hallatava protsessi riske ja võimalusi, lisaks peab ta olema teadlik ka oma töökeskkonna võimalustest. Toodete visioonid ei teki tühjast kohast – need hõlmavad endast elemente organisatsioonikultuurist ja tegevustest, mida juba varasemalt on toodud headeks näideteks konkreetse protsessi sees.

Toote visioon ei pea olema teekaart mingite tegevuste kohta, pigem peab see olema nagu kompass, mis arvestab üleüldise organisatsiooni strateegiaga samal ajal näidates suunda, kuhu suunas konkreetne toode peab liikuma. Visioon aitab kasutajatel otsustada, milline lahendus konkreetses situatsioonis on õige.

7. Arutelu

Üldiselt võiks öelda, et kõik osapooled on tarkvara arenduse teenuse sisse ostmisega rahul või on sellega leppinud. Täitjal on võimalik valida endale huvitavaid projekte ja kaubelda välja endale aktsepteeritav hind. Teiselt poolt tellija ei pea tegelema kalli tööjõu motiveerimisega ja saab ressursse kasutada paindlikult. Ainsad, kes tekkinud olukorraga rahul pole, on tooteomanikud, kes pole oma rolli sattunud vabatahtlikult. Üldjuhul määratakse tooteomanikeks inimesed, kes vastutavad eelarve eest. Rahulolematuse põhjuseks on peamiselt teadmatus, mida tooteomanikuks olemine tähendab. Puuduvad välja kujunenud praktikad ja standardid, mis kirjeldaksid tooteomaniku rolli. Ettevõtetel on keeruline hinnata, kes sobiks tooteomaniku rolli kõige paremini. Tooteomaniku sobivus selgub tööde käigus.

Raamat „Think Like a Freak: The Authors of Freakonomics Offer to Retrain Your Brain“ väidab, et inimesed ei tunnista oma vigu, kuna aju registreerib teadmatust samamoodi kui valu, seega lihtsam on teeselda asjade teadmist. (Steven D Levitt, 2014). Kui puudub täielik arusaam, kuidas tööprotsess peaks tulevikus välja nägema, siis on väga raske selle järgi tarnida töötavat tarkvara. Kuidas loodav tööprotsess hakkab uues tehnoloogilises paradigmas välja nägema nõuab visiooni hoidjalt suurt pühendumust projektile. Paraku ei võimalda ettevõtted sellist pühendumust tooteomanikele ja täitjad väga harva suudavad soovitud tasemel visioonihoidja rolli pakkuda. Tooteomaniku roll peaks olema ettevõtetes ärianalüütiku kanda ja see peaks olema struktuuris eraldi ametikoht.

Tooteomaniku roll on selgelt defineeritud ainult tarkvaraarendusteenust vahendavates ettevõtetes, nagu SMIT, TEHIK ja KEMIT. Nende ettevõtete tooteomanikud aga pole ise sisu omanikud, nemad ei defineeri äriprotsesse. Ettevõtetes, kus tooteomaniku roll ja äriprotsessi omaniku roll langevad kokku?, pole selle jaoks aga palgal eraldi nimest. Tooteomaniku roll lasub kohustusena muude rollide kõrval. Kõikidest intervjuudest võib järeldada, et tooteomaniku rollist saadakse erinevalt aru. Tooteomaniku roll vajab selgemat defineerimist ja vaja oleks raamistiku, mille järgi töötada. Tarkvaraarenduse teenuseid pakkuvad ettevõtted on valitsevat olukorda märganud ja pakuvad tooteomanike koolitusi. Ettevõtted, kus pole veel tooteomaniku kompetentsi, vajavad alustamiseks juhiseid, mida pakub käesolev magistritöö. Esimesed rakendamise katsed on näidanud, et liigun õiges suunas.

Tulevikus peaks uurima põhjalikumalt tooteomaniku rolli. Kuidas teha tooteomanikul rolli sisse elamine võimalikult lihtsaks? Kuidas välja selgitada juba uut tooteomanikku valides, kas üks või teine inimene üldse sobib sellesse rolli?

8. Rakendamine RKAS-is

Äriprotsessidel põhinevate arendusvajaduste kirjeldamise kooskõlastamine, nagu seda on kirjeldatud käesolevas magistritöö lisa 11-s tundub esmapilgul lihtne. Kuid selle juurutamine saab olema keeruline. Viie aastaga ei ole RKAS-is suudetud kõiki protsesse piisavalt detailselt kaardistada. Arvestada tuleb organisatsiooni kultuuri ja töötajate oskuseid. Ettevõtte struktuuri muuta ei tuleks, kuna kõik vajalikud ametikohad on juba olemas. Uuendama peaks ametijuhendeid, et need viia kirjeldatud rollidega kooskõlla. Töötajate oskusi on võimalik muuta läbi koolituste või värbamise. Organisatsiooni kultuuri muutmine aga võib aega võtta aastaid (NOBL, 2016).

Esimesed protsessi kirjeldamised on olnud väga erinevate tulemustega. Peamiseks põhjuseks tooksin välja tööprotsesside keerukuste erinevuse. Esimesel juhul puudutas tööprotsess ainult ühte osakonda, ning selle kirjeldamine oli lihtne. Teisel juhul puudutas tööprotsess kahe osakonna omavahelist koostööd ja selle protsessi täielik kirjeldamine veel käib. Kõige rohkem on probleeme tekitanud skoobi hoidmine, mis on analüütiku vastutusala. Äriprotsessi kirjeldamisel üritatakse ühe suure põhiprotsessiga katta ära kõik osakonna vajadused – selle asemel, et kirjeldada mitu erinevat konkreetset probleemi lahendavat protsessi. Visandite loomisel keskendutakse liigselt mugavusfunktsioonidele.

Prototüüpimisele ja testimisele on kõik osapooled reageerinud positiivselt ja sobilike lahendusteni on jõutud kiiresti. Lõplikult kinnitamata tööprotsessi puudumine ei ole seganud prototüüpide loomist. Paari praktilise näite põhjal ei saa veel selgelt välja öelda, kas kokku lepitud kavandamise protsess on õige ja piisav. Kõik osapooled on aru saanud, miks selliseid tegevusi on vaja teha – seega liigume tarkvara kavandamisega õiges suunas.

Kokkuvõte

Riigi Kinnisvara AS, nagu ka paljud teised avaliku sektori ettevõtted on olukorras, kus tarkvaraarendust ostetakse hankeprotsessi kaudu ettevõtte väljast. Protsessijuhtimisest lähtuvalt on tekitatud vajalikud ametikohad, määratud ametikohtadele rollid ja vastutusalad. Rollide vastutusalade kirjeldamise täpsus on erinev asutuste lõikes, kuid läbivalt on kirjeldatud ühe rollina tooteomanikku. Täpsemalt on kirjeldatud vastutusalad ja tööülesandeid, kuid reeglina puuduvad kirjeldused kuidas tööülesandeid täita.

Üldine teadlikkuse tase avalikus sektoris tarkvaraarenduse metoodikatest on kõrge. Viis, kuidas metoodikat on juurutatud, on ettevõtete lõikes väga erinev. Metoodikas kirjeldatud rolle tõlgendatakse väga erinevalt lähtuvalt ettevõtte positsioonist. SMIT ja TEHIK on teenuse vahendaja rollis täitja ja klientide vahel, kuigi nende rolliks põhimääruses on märgitud teenuste arendamine ja haldamine (Siseministeerium, 2015). Nad ei tunneta ennast kui täielikud tooteomanikud, kuna teadmine äriprotsessidest asub ettevõttest väljas. Paremas olukorras on ettevõtted, kes täidavad asutusesiseseid arendusvajadusi, nagu Eesti Energia ja RIA, kaasa arvatud RKAS. Nendes ettevõtetes on tooteomaniku rolli lihtsam täita, kuna korraga on võimalik hallata arendusressursse ja äriprotsessi. Tooteomaniku rolli vastutusalad on kõikides ettevõtetes defineeritud, kas ametijuhendites või eraldi kokkuleppena. Tegevused on ära kirjeldatud protsessides, kuid instruktsioone, kuidas midagi teha, pole. Ühtsete kirjelduste puudumise tõttu on tooteomaniku profiil kõikides ettevõtetes erinev.

Magistritöö tulemusena valmis tarkvara kavandamist kirjeldav dokument RKAS-i tooteomanikule, et projektide alustamine oleks kõikidele osapooltele arusaadavam. Oma intervjuudes erinevate ettevõtete esindajatega olen keskendunud kolmele teemale: äriprotsessi kirjeldamine, prototüüpimine ja dokumenteerimine ning viimase testimine. Intervjuude eesmärgiks on koguda informatsiooni, kuidas teised ettevõtted samasuguseid tegevusi on kirjeldanud. Intervjuudest selgus, et sarnaseid töökorralduslikke dokumente pole, kuid vajadus nende järgi on olemas. Teadmised äriprotsesside kaardistamise kohta on ettevõtetes olemas, kuid testimise ja prototüüpimise kompetents on puudulik. Prototüüp ja testimine on terminid, mida tõlgendatakse väga vabalt. Prototüüp võib tähendada nii disaini, visandit, *mockup*-i, traatmudelit, staatilist HTML-i kui ka valmis rakendust. Suur erinevus on, kuidas

saadakse aru, mis on testimine, mõõtmine ja hindamine. Siinkohal peab tõdema, et täitjate poole esindajatel on selge arusaam, mis on testimine ja kuidas seda teha. Tellija poole peal on suur erinevus – mõõdetakse protsessi, küsitakse arvamust ja hinnatakse disaini. Testimine on tarkvaraarenduse spetsiifika, mis saab liiga vähe tähelepanu tellija poolel.

Kokku sai korraldatud kümme intervjuud, neli täitja poole esindajatega ja kuus tellija poole esindajatega. Esialgne plaan, rääkida ainult mõlema poole projektijuhtidega muutus kiiresti, kuna erinevates asutustes on samadel ametikohtadel erinevad rollid. Eesmärgiks oli intervjueerida inimesi, kes loovad projektidele lähteülesande.

Intervjuud tellija poole esindajatega näitavad, et rahul ei olda praegu tarkvara kavandamise korraldusega. Lähteülesandes peab olema võimalikult vähe ebamäärasust. Nõuete nimekiri on konkreetne, kuid ei anna täitjatele kiiresti selget pilti lahendamist vajavast probleemist. Täitja leiab, et selge ülevaate lahendamist vajavast probleemist annab ärianalüüs, mis tuleks teha enne tarkvara hanke välja kuulutamist. Instruktsioone järgides on tooteomanikud võimelised lihtsa ärianalüüsi ise koostama.

Isikliku hinnanguna arvan, et valminud juhend on esimene samm õiges suunas, kuid juhendit on vaja veel täiendada. RKAS-i struktuuris on tooteomaniku roll jäetud osakonnajuhatajate kanda, kuna nemad vastutavad eelarve eest. Osakondade vahelistest erinevustest tingitult, on tooteomanike kompetentsid väga erinevad. Põhjuseks on tööde iseloom ja protsesside väga erinev keerukus. Keerulisi protsesse ei suudeta väiksemateks tükkideks jagada, loodetakse et seda teeb täitja pooole süsteemianalüütik. Täitja poole esindaja, aga ei suuda piisavalt kiiresti analüüsida äriprotsessi keerukust. Ettevõttele ei piisa ainult vastutusalade kaardistamisest, vajalik on koostada rollidele taustainfot andvad materjalid, mida rolli kandmine eeldab. Vastutusala kirjeldus, et tooteomaniku rolli üheks tegevuseks on visiooni hoidmine ei anna soovitud tulemust. Visiooniks ei saa olla midagi üldsõnalist – rakendus peab tulevikus seda tegevust paremini tegema. Visioon peab olema konkreetne ja teostatav. Tarkvara kavandamine koostatud juhendi järgi annab soovitud tulemuse, kuid nõuab suurt ajalist investeeringut nii projektijuhilt kui ka tooteomanikult.

Business Process Based Software Designing

Master's Thesis

Viljar Komp

Summary

State Real Estate ltd like many other public sector establishments have outsourced their software development. Procurements are managed by clear and defined processes. Establishments structure has necessary positions and roles for software acquisition. Detailed instructions variy for the same role between establishments, but every establishment has a role that could be called product owner. Usually companies have described responsibilities and work task for every role but usually there are no instcutions how to complete mentioned tasks.

Overall knowledge about different software development methodologies is high. The way methodology is applied in different establishments varies greatly. The way establishments interpret methods is determined by their position in the software delevopment process. SMIT and TEHIK act more like service mediators then product owners. They dont position themselves like software owners, because knowledge about specific business processes is held by their clients. Companies like Eesti Energia, RIA and RKAS are in a better situation, because their product owners also manage business processes. Responsibilities are defined in agreements and roles are describled in process management. Usually there are no instructions how to complete the tasks described in the process.

Result from the masters thesis is a document describing how a product owner should plan new software and how to check the viability of the offered solution. Document should make it easier for product owners to understand what is needed from their role. In my interviews with different establishment representatives i concentreted on three different main themes: how to describe a business process, prototyping and testing the prototype. Aim of the interviews was to collect information about current praktices and how their describe product owners role. Interviews revealed that some establishments have described responsibilities in software development process but noone has instructions how to complete the listed tasks. Everybody agreed, that there

is a need for instructional documents for product owners. The general knowledge on client side about describing business processes is average but knowledge about prototyping and testing is poor. There is no common understanding what is a prototype and how to test it. It is understood that idea of prototype testing is to get feedback. On developers side the knowledge about these topics is good but even they dont have a way to describe what a good product owner does. Testing is a field that gets too little attention on clients side.

In total ten interviews were held, four with developers side and six with clients side. Initial plan to interview project managers only changed because establishments describe differently their responsibilities. Interviews were conducted with people who actually lead projects and held the scope of it. Interviews show that clients side is not happy with the way software acquisation is being currently managed. Describing new software as a list of requirements doesnt give enaugh context for developers. Procurements should contrain specific business process it tries to follow and an initial problem it tries to solve. Following a simple instruction product owners should be able to complete a simple business analysis.

Kasutatud kirjandus

(2008). Allikas: Agile Practices and Principles Survey Results: July 2008: http://www.ambysoft.com/surveys/practicesPrinciples2008.html

(04 2017. a.). Allikas: Scrum Institute: http://www.scrum-institute.org/What_Makes_Waterfall_Fail_in_Many_Ways.php

Adiseshiah, E. G. (2016). Allikas: https://www.justinmind.com/blog/rapid-prototyping-isnt-helpful-or-is-it/

Agile Alliance. (2017). Allikas: https://www.agilealliance.org/glossary/continuous-deployment/

Berteig, M. (2011). Allikas: http://www.agileadvice.com/2011/12/05/referenceinformation/24-common-scrum-pitfalls-summarized/

Best-Practice Software Engineering. (2013). Allikas: http://best-practice-software-engineering.ifs.tuwien.ac.at/patterns.html

Bugayenko, Y. (2015). Allikas: http://www.yegor256.com/2015/10/27/outsourcing-doesnt-work.html

Cristian Bogdan, J. G. (2016). *Human-Centered and Error-Resilient Systems Development*. Allikas: https://books.google.ee/books?id=wyTkDAAAQBAJ&pg=PA7&lpg=PA7&dq=responsibilities+product+owner+thesis&source=bl&ots=7yIOrRrU3T&sig=7ttbnoZlmdGRbJQvexcSxQHUV4g&hl=en&sa=X&ved=0ahUKEwi7hKu4pc7TAhXDiywKHb8QDrs4ChDoAQgmMAE#v=onepage&q=responsibilities%20produ

Fernandes, P. (2017). *What Is a Vision Statement?* Allikas: http://www.businessnewsdaily.com/3882-vision-statement.html

Greene, J. (2015). Allikas: https://www.cherwell.com/blog/3-ways-to-stay-agile-with-itil

Hathaway, T. (2009). *Business Analysis Defines the Future. Should You Outsource It?* Allikas: http://businessanalysisexperts.com/outsourcing-business-analysis-outsource/

IT process maps. (2017). *ITIL service design*. Allikas: https://wiki.en.it-processmaps.com/index.php/ITIL_Service_Design

Keith Collyer, J. M. (2013). Allikas: https://www.ibm.com/developerworks/rational/library/compliant-agile-medical-device/

Krisinsdottir, S. (2014). *Responsibilities and Challanges of Product Owners in Practice*. Allikas: http://skemman.is/en/stream/get/1946/19523/42035/1/Sigurhanna_Kristinsdottir_MPM2014_FinalPaper.pdf

Larsson, M. (2013). *Usable Interventions for a Product Owner*. Allikas: https://gupea.ub.gu.se/bitstream/2077/34006/1/gupea_2077_34006_1.pdf

Lember, K. (2017). (V. Komp, Intervjueerija)

Longstreet, D. (2010). *What is the purpose of Software?* Allikas: https://davidlongstreet.wordpress.com/2010/04/30/what-is-the-purpose-of-software/

Manifesto for Agile Software Development. (2001). Allikas: http://agilemanifesto.org/

McCray, S. (2012). Allikas: http://www.ssonetwork.com/business-process-outsourcing/articles/the-top-10-problems-with-outsourcing-implementatio

Mett, H. (2017). (V. Komp, Intervjueerija)

Nindrai, V. (2015). Allikas: http://blog.brightfind.com/web-technology/how-to-be-a-successful-project-manager

NOBL. (2016). *How Long Does It Take to Change Company Culture?* Allikas: https://medium.nobl.io/how-long-does-it-take-to-change-company-culture-deb0aa2021ac

Orient. (2017). Allikas: http://www.orientsoftware.net/software-outsourcing/why-outsourcing

Pentjärv, T. (2010). Allikas: Äriprotsesside modelleerimine Eesti Energia näitel: http://www.cs.tlu.ee/instituut/opilaste_tood/magistri_tood/2010_kevad/tuuli_pentjarv_magistritoo.pdf

Pun, H. (2017). Allikas: Design for Founders: https://www.designforfounders.com/good-design/

Püüa, M. (2017). (V. Komp, Intervjueerija)

Rattur, A. (2017). (V. Komp, Intervjueerija)

Rosin, M. (2014). Allikas: http://www.cs.tlu.ee/teemad/get_file.php?id=291

Salonek, T. (2009). *CodeProject*. Allikas: The 7 Elements of Highly Successful Software Projects: https://www.codeproject.com/Articles/32721/The-Elements-of-Highly-Successful-Software-Proje

Sarapuu, H. (2017). (V. Komp, Intervjueerija)

Seungwoo Maeng, Y.-k. L. (2012). Allikas: Interaction-Driven Design: A New Approach for Interactive Product Development: http://dpl.kaist.ac.kr/wp-content/uploads/DPL_IntConference_10.pdf

Shifflett, O. (2012). *5 Things to Consider Whilst Sketching*. Allikas: https://www.viget.com/articles/5-things-to-consider-whilst-sketching

Siseministeerium. (2015). Allikas: https://www.smit.ee/pdf/SMIT_pohimaarus.pdf

Steven D Levitt, S. J. (2014). *Think Like a Freak: The Authors of Freakonomics Offer to Retrain Your Brain*. Allikas: http://www.amazon.com/Think-Like-Freak-Authors-Freakonomics/dp/0062218336/ref=sr_1_1?ie=UTF8&keywords=think+like+a+freak&qid=1413760378&s=books&sr=1-1

Sunview. (2017). Allikas: https://www.sunviewsoftware.com/learn/blog/5-steps-to-integrate-itil-into-agile-process

Zalavadia, S. (2015). Allikas: https://www.infoq.com/articles/agile-fails-enterprise

Taal, A. (2017). (V. Komp, Intervjueerija)

Tammik, K. (2017). (V. Komp, Intervjueerija)

Transision support. (2013). *Process versus procedures*. Allikas: https://transition-support.com/Process_versus_procedure.htm

Usmani, F. (2013). *Lead Time and Lag Time in Project Scheduling Network Diagram*. Allikas: https://pmstudycircle.com/2013/02/lead-time-and-lag-time-in-project-scheduling-network-diagram/

Wikipedia. (2017). Allikas: https://en.wikipedia.org/wiki/Agile_software_development

Young, E. &. (2013). Allikas: https://www.ria.ee/public/Programm/avaliku_sektori_ariprotsesside_kasiraamat/II_seminari_slaidid_-_Protsesside_kaardistamine.pdf

Lisad

Lisa 1 Tieto Estonia AS, Kaido Heinsalu

K: Kuidas peaks käituma tellija äriprojekti kavandamisel?

Arendajad on tihtilugu väga mugavas olukorras, neil on ees kindlad raamistikud ja sandardid. Välja on kujunenud parimad praktikad, kuidas midagi kirjeldada. Kui uus inimene tuleb organisatsiooni, siis on juba põhimõtteliselt ette teada, kuidas üks või teine roll projekti meeskonnas peaks käituma. Kui tegemist on tellija poole tooteomanikega, on olemas aktuaalne probleem. Mis täpselt on tooteomaniku roll, kuidas peaks tellima tarkvara. See on igas organisatsioonis erinev. Iga organisatsioon kirjeldab protsesse, teenust ja selle komponente isemoodi. Standardeid on palju ja neid kasutatakse erinevalt. Samasugune olukord on ka tarkvara tellimisel, sisend on alati organisatsioonist sõltuv ja läheb palju aega, enne kui projekt tegelikult tööle hakkab. Kui rääkida visiooni hoidmisest, siis ka sellest saavad organisatsioonid erinevalt aru, tihti nähakse, et selleks on arvamusliidriks olemine ja otsuste langetamine. Oskus suunda hoida ja töid jagada pisikesteks tükkideks on puudulik. Oskus määrata projektile konkreetne lõpp on samuti puudulik, seega on võimatu hinnata projekti lõppu ja jääb mulje et projekti lõpp saabub hetkel kui raha otsa saab.

Suurtes projektides on murekohaks rollide vastutusalad, kuidas käitub projektijuht ja mida täpselt teeb analüütik.

K: Kuidas peaks valmima projekti dokumentatsioon?

Milline peaks olema projekti dokumenteerimise tase on vastuseta küsimus. Võiks öelda, et see peab olema selline, et kui vahetub projektimeeskond või selle üksikud liikmed, peab teadmine säilima, kus midagi asub. Kuidas see täpselt välja näeb - ma ei tea.

K: Kuidas peaks toimuma projekti testimine?

Testimine on koht projektis, mis peab olema selgelt defineeritud. Kuidas on korraldatud tööde vastu võtmine ehk kuidas käituda olukorras, kui pakutud lahendus ei sobi tellijale. Tööd sellise lahenduse nimel on juba tehtud ja täitjal on õigus selle eest tasu saada. Peab olema täpne kirjeldus, mis määratleb töökäsu eduka täitmise.

Kui hankija ei ole piisavalt kompetentne, et hallata töökäske, siis on lihtne sattuda olukorda, kus raha nii öelda põleb. Kuidas mõõta, et hankija on sattunud sellisesse olukorda? Sellisesse olukorda sattumist peaks ennetama, see aga eeldab selget projekti skoopi ja testimist. Kas liigutakse õiges suunas, selleks on välja mõeldud lõppkasutaja testimine. Selline praktika pole aga klientide juures juurdunud, endiselt ei saa arvele panna eraldi rida UX-i testimise kohta. Sarnane olukord valitses varasemalt projektijuhtimisega, kuid tänaseks on see olukord muutunud.

Lisa 2 CGI EESTI AS, Martti Kebbinau

K: Kuidas te kirjeldaksite ideaalset tööprotsessi täitja ja tellija vahel?

Hankida on mõtet siis, kui reaalselt on motivatsiooni tulemust saada.

Hankija poolelt peab olema pühendunud inimene vastas, kes ei tee seda projekti muu töö kõrvalt, vaid on pühendunud asja edukasse läbiviimisse. Organiseerib tellija poolt õigetelt inimestelt kiired vastused. See on kahepoolne suhe, kus on vajalik ausus ja usaldus.

Usaldus, nagu arsti vastu, kelle otsustes sa ei kahtle, kui ta ütleb et sul on üht või teist analüüsi vaja teha või advokaadi vastu, kui ta ütleb et seadus lubab nii või ütleb teisiti.

Kindlasti peaks teenust hankides jätma sisse ka varu innovatsiooni ja skoobimuudatuste jaoks.

Erinevate vajaduste lahendamiseks sobivad erinevad metoodikad, mistõttu on ideaalset protsessi väga keeruline välja tuua.

Suur vahe, kas tahad uut turgu luua või lahendad ära kohe kokku kukkuva probleemi.

Kui on vaja lahendada konkreetne asi ja on väga täpselt detailselt teada, mida vaja on (või on mingid piirangud ees, näiteks seadusest tulenevad vms.), sobib kosk mudel väga hästi. Kui alles töö käigus mingid asjad selguma ja täpsustuma hakkavad, sobib palju paremini agiilne metoodika.

Praegused probleemid: Täitjal on kohustused ja tähtajad, aga tellijad on inertsiga ja vastavad kui jõuavad muu töö kõrvalt. Täitjale on iga tund ootamist suur kulu.

Maksab kätte see, kui pitsitatakse skoobiga – täitja ei saa pakkuda uut tehnoloogiat ja innovatsiooni, kui pael on tihkelt kaelas ja iga liigutus maksab ja tuleb põhjendada ning vireled kasumlikkuse / kahjumlikkuse ääre peal. Seal ei tule pähe midagi uut ja innovaatilist soovitada. Innovatsiooni tellimine peaks projektis olema võimaldatud. Seda on ka täitjatele ette heidetud, aga ise nööri lõdvemaks ei lasta.

Analüüsi eraldi tellides tehakse valmis väga keeruline süsteem, pärast seda kasutada ei saa, kuna raha sellise asja tellimiseks ei ole. Tasub panna ka analüüsi eesmärk raamidesse.

Tellima peaks terve meeskonna samast majast, mitte arendaja ühest, analüütik teisest ja disain kolmandast kohast. Praegu üritatakse selliselt tellida hägust asja kindla skoobiga, kus asjad ilmselgelt muutuma hakkavad.

K: Kuidas peaks kirjeldama äriprotsessi?

Kasutades kaasaegseid üldlevinud standardeid (Business Process Modelling Notation)

Majasisese asja jaoks oleks hea mingi blokk-diagramm teha ja kõik saavad aru, aga kui välised partnerid peavad ka aru saama, siis globaalne standard oleks kõige parem. Ja kui juba see samm ära teha, et otsustada, et nii teha – see paneb juba detailsuse paika.

Ja tuleb mõelda, kas seda üldse vaja kirjeldada on? See peab olema efektiivne ja et kõik protsessi osad lisaks väärtust ja ei oleks pudelikaelu. Protsessi disain on keeruline.

K: Kui detailne peaks olema prototüüp? Kuidas seda hinnata?

Minimaalne kasutatav asi, mis täidab vähegi arvestatavat osa eesmärgist (Minimum Viable Product).

Lihtne vastus: peab olema suhteliselt detailne ja üpris lõplik, kuna prototüübi mõte on äripoole inimestele, kes ei tarvitse süsteemi dokumentatsiooni lugeda või neist arugi saada, võimalikult varakult ja hästi arusaadavaks teha, milline see lahendus saab olema ja kuidas see töötab.

Minimaalsete kuludega teostatav, võimalikult lihtne lahendus, et täitja saaks aru ja kinnitust, kas ta liigub õiges suunas.

Töö käigus: Prototüüpi on eelkõige vaja, kui on suurem oht, et on erinevaid radu, kuidas midagi lahendada ja tellijal pole üldse ülevaadet (või omab mingit visiooni, aga ei ole sellega ise rahul), kuidas asja võiks lahendada.

Hankides: kui on hulk erinevaid teid, kuidas asja lahendada, aga tellija täpselt ei tea, mis on parim, siis peaks prototüübi tellima, et leida arendajat, kelle mõte talle kõige paremini sobib. Kui ise väga täpselt tead, mida tahad, siis pole prototüübil mõtet.

Ehk et mida hägusem on tellijal teadmine selle kohta, mida ta täpselt tahab, seda detailsem prototüüp peaks olema.

K: Kas või millistel juhtudel peaks prototüübi kõrval olema ka detailne spetsifikatsioon?

Hanke etapis mitte, koormab liialt täitjat ja pakkumisi tuleb selle võrra vähem.

Töö etapis oleks ikkagi vaja nõuda detailset dokumentatsiooni prototüübi kõrvale, et tellijana ka täitjat vahetada saaks ja tehtud töö oleks edaspidi kasutatav. Pealegi, dokumenteerimine on suhteliselt odav. Kui asi on juba analüüsis välja mõeldud (see kallis osa), on normaalne nõuda ka spetsifikatsiooni.

Olenevalt projetki iseloomust ja metodoloogiast võib see olla erineva detailsusega.

K: Kuidas peaks prototüüpi hindama?

Lühike vastus: Vastu nõudeid testima. Kas proto teeb kõike, mida sa oled öelnud.

Lepingu iseloom on ülioluline selle juures.

Kui on *FixPrice* projekt ja nõuded kirjas, siis peaks olema protos kõik detailselt kirjas ja see on nö „piibliks“ tööde teostamisel. Ja kunagi ei tee rohkem, kui protos kirjas on, kuna hinnangud on selle järgi kujundatud.

Kui on lahtised käed ja hägune teadmine, mida vaja on – siis on väga normaalne teha üldisem prototüüp, teeme järjest ja töö käigus selgub, mida kuidas muuta või mis paremini tellijale meeldib.

Et kui lõpuni iga detaili välimust ei tea, ei tähenda kohe et „ei oska tellida“, vaid ongi loogiline kui arenduse käigus asju selgub. Aga selleks peavad need võimalused projekti sisse kirjutatud olema.

Hinnata tuleks protos ka seda, et „kas siin on nüüd kõik olemas, mida me tahtsime“, et hinnata, kas tellija äkki on ise sellega eksinud, mida nad kirja on pannud. Kui proto ikka tuleb eeldatust täiesti teistsugune, tuleks oma kirjeldused üle vaadata.

Et metoodiliselt lähenda, tuleks võtta funktsionaalsed ja mittefunktsionaalsed nõuded ja vaatama, kas see proto siis katab kõiki võimalikke nõudeid.

Aga kui ei ole kirja pandud, näiteks kasutatavuse nõudeid vms, siis ongi küsimus see, kui palju on antud täitjale ruumi teha nii, nagu täitja arvab, et on kaasaegne, äge, mugav, ilus jne… Täitja väga hindab seda ruumi, kuna töötajatele on väga motiveeriv teha uute tehnoloogiatega uusi ägedaid lahendusi. Probleem on lihtsalt selles, et kuidas tellija saaks kontrollida, et täitja lihtsalt oma raha kätte ei ürita saada.

Siin tulebki mängu see ka algul mainitud usaldus-suhe, mis oleks ideaalses protsessis sees.

Lisa 3 Trinidad Consulting OÜ, Hegle Sarapuu

K: Milline võiks olla tööprotsess täitja ja tellija vahel?

Trinidad pakub samuti otsast lõpuni protsessi. Protsess on täitja jaoks oluline ja ühtegi etappi ei ole soovitav sealt vahelt välja jätta. Etappe ei saa vahelt ära jätta, kuna muidu võivad tekkida olukorrad kus midagi peab ümber tegema ja see ei ole mahu hinnangus. Senikaua kuni klient on nõus tunni hinna järgi arveldama on ümber tegemine mõeldav. Kui tegemist on fikseeritud hinnaga, siis protsess on mõeldud üllatuste vältimiseks.

K: Kuidas äripoolele öelda, et kõik pisikesed muutused maksavad midagi?

Kõigepealt tuleb luua rakenduse tervikpilt ja see pisikesteks tükkideks teha. Prototüüpima peab protsesse ja pisikeste tükkide prototüüpimine on väärtusetu. Prototüüpima peab tervet protsessi või selle osa, ja see tuleb läbi mängida. Mis juhtub, kui tekivad pausid protsessi etappide vahel, kuidas toimub salvestamine? Graafiline disain on ainult pisike osa ühe rakenduse arenduse juures. Prototüübi tööprotsessi läbi käimine ei pea olema väga interaktiivne, kui piisab isegi paberi peal asja läbi mängimine – see oleks väga *lean* lähenemine.

K: Kui detailne peaks olema prototüüp?

Sõltub mis saab prototüübist edasi – kui prototüüp ise peab olema dokumentatsioon, siis peab see olema äärmiselt detailne, see peaks olema dünaamiline. Kui lahendus on võimalik valmis teha kuu ajaga ja arendajal on ligipääs prototüübi kirjeldajale, siis prototüüp ei pea olema väga täpne. Prototüübi kirjeldajalt on võimalik alati sisendit saada. Mingil hetkel peab keegi dokumenteerima siiski ära funktsionaalsuse, et paari aasta pärast oleks võimalik aru saada, mida see vaade teeb. Prototüübi funktsioon on kommunikatsioon ja võimalikult varases staadiumis teha kasutatavuse testimist. Pole vaja süsteemi koopiat hoida prototüübis. Kui kasutatakse traditsioonilist kosk mudelit tarkvara arendamisel peab prototüüp olema äärmiselt detailne.

K: Kas prototüübi kõrval peab olema ka detailne spetsifikatsioon?

Sõltub olukorrast, kuna kasutajaliides ei lähe enam klasside ja meetodite paiknemisega ammugi enam kokku. Oluline on, et olukorras kus süsteem ei käitu enam nii nagu me

eeldame, seega hästi tehtud *use case*-d on hindamatud. Otsustada kui detailset dokumentatsiooni on vaja, tuleks vaadata kui kaua kestab terve projekti arendus.

K: Kuidas peaks prototüüpi hindama?

Kõik prototüübid peaksid läbima kasutatavuse testimise. Tihti lähevad lõppkasutaja ja toote omaniku arvamused prototüübist lahku. Kasutaja arvamus on vähetähtis, kasutajale peab ette andma piisavalt detailse prototüübi ja ülesanded, mida ta tulevikus peaks hakkama tegelema, ning vaatame kuidas ta neid täidab. hindame seda kas ta sai aru mis ta tegi või ta oli segaduses. Kas me arvame, et kui ta teeb seda kümme korda päevas, mis ta sellest siis arvab? Kasutajaid peaks olema 3–5. Eristama peaks, kas jalgu jäi prototüüp või kasutatavus.

K: Kuidas te valite testijad?

Kliendi maja seest, need inimesed, kes pole projektiga seotud olnud. Ideaalis peaksid need kasutajad olema võimalikud erinevad.

K: Millal tuleks prototüüpi testida?

Võimalikult varakult, et kohe tuleks välja valed arusaamad. Piisab täiesti sellest, kui ühe eesmärgi tööprotsess on valmis. Kui prototüüpi hakata liiga hilja tegema, siis võib juhtuda et enam pole aega prototüüpi muuta.

K: Millal pole vaja enam prototüüpida?

Kui muudatus on nii väike, et see ei mõjuta enam kogu protsessi või kasutajate efektiivsust. Kuidas hinnata seda, on raske, kuna rakendused lähevad katki samm sammult, mitte ühest konkreetsest hetkest. Võiks öelda et prototüüpima peab siis, kui see puudutab kogu rakenduse üldist protsessi.

K: Kas tarkvara kvaliteeti peaks pidevalt hindama?

Ei oska öelda, kuna vahest võib rakendusel olla väga tubli peakasutaja, kes teeb kasutajate eest kogu töö ära – kasutajad on rahul. Välja tuleb see murekoht alles siis kui inimesed vahetuvad. Tarkvara tuleb kohe alguses luua õigesti ja kasutajasõbralik, et kogu elutsükli jooksul ei peaks seda pidevalt toetama ja arendama. On olemas

kasutatavuse mõõdikud, näiteks kasutajatoe pöördumiste arv, valideerimiste vead, protsessile kuluv aeg jne.

Lisa 4 Design UX OÜ, Kristian Lember

K: Kuidas peaks tellija teie käest tooteid tellima?

Meie teenuseks on väikest ja dünaamilist meeskonda kasutades saada kiiresti aru äripoolest. Mõne päeva jooksul üritatakse ära kaardistada ettevõtte vajadused. Tähtsal kohal on usalduse tekitamine. Et üldse saada asjad liikuma on täitja sunnitud pidevalt pakkuma selgitavat ja juhendavat tausta, et mismoodi tööd tehakse ja kuidas see välja näeb. Kui kaua see aega võtab – ei ole alati teada.

K: Kuidas leida õige tooteomanik üles?

Tihti pole valikut ja reeglina on pandud osakonna juhatajad millegi eest vastutama ja nemad valivad ka täitjad. Sellepärast on vaja ka tekitada usaldust, näidata üles huvi organisatsiooni vastu. Visiooni hoidja roll paratamatult langeb sellistes olukordades täitja õlule. Sellepärast me otsimegi üles probleemid ja defineerime need, kaardistamise protsess on hea tulemuse saamiseks üliloluline. Isegi kui täitjal on visioon, ei pruugi olla see, mida tellijal on tegelikult vaja, tihtilugu selguvad vajadused jutu käigus. Tellija visioonist aga ei tohi väga kaugele liikuda, kuna muidu võib tellija võõranduda kogu projektist, selline olukord pole jällegi hea.

K: Kas tooteomanik peaks korraldama testimise?

Meie samuti puutume kokku olukorraga, et testimine on esimene asi eelarvest alati välja visatakse. Testima peab iseenda toodet mida arendatakse.

K: Kuidas otsustada milline on hea prototüüp?

Kui põhiteenus tuleb prototüübist välja, baas asjad on testitud, toote turule toomine võiks olla jagatud etappideks, mitte üks suur pauk korraga. Pidevalt testid erinevaid toote etappe ja tuleb jupi kaupa oma tootega välja. Sellega lendavad väga paljud *startup*-id õhku, kui üritavad korraga väga keerulise tootega välja tulla. See mis järjekorras jupid valmis teha, sõltub juba projektist.

K: Kas prototüübi testimisel tuleks vanu osasid uuesti läbi mängida?

Kui nad on seotud, siis muidugi, kuid tuleb ära defineerida mis asi on prototüüp. Meie peame prototüübiks *wireframe*-i ja selle peal saame testida seda, kas asi üldiselt töötab.

Paljud inimesed pole võimelised *mockup*-de peal testima ja nad vajavad päris korralike funktsionaalseid prototüüpe, mille peal on juba disain. Kui on võimalik tuleks seda siiski teha *wireframe*-i peal. Kui inimesel pole visuaalset mõtlemist, siis *mockup*-i peal testimine ei õnnestu. Testima peaks *wireframe*-ide erinevaid versioone.

K: Kas küsimus, mis sa sellest keskkonnast arvad on õige?

Me üldiselt küsimusi stiili kohta ei küsi – see on liiga subjektiivne, tuleb jälgida kas protsess üldiselt töötab.

K: Kes peaks testima?

Alati võiks olla värske inimene, välja arvatud juhul, kui toimub keskkonna *redesign*. Vahe uue ja vana vahel võib kohati olla väga suur.

K: Kuidas peaks kirjeldama äriprotsessi?

Me alustame kasutajate kirjeldamisest, me üritame kõike teha võimalikult visuaalselt, me üritame kohe hakata asju prototüüpi panema. Ideaalne olukord on siis, kui klient on ära suutnud defineerida probleemi.

K: Kas tellija nõue, et pakkuge meile kolm valikut on põhjendatud?

Ei ole, selline komme on kuskilt vanast ajast pärit, meie pakume ühe lahenduse. Valiku tegemine ei tohiks olla kliendi mure, see näitab täitja tegemata tööd.

Lisa 5 Keskkonnaministeeriumi Infotehnoloogiakeskus, Andres Rattur

K: Kuidas jõuavad teieni tellija tööde kirjeldused?

Üldjuhul ei suuda tellija piisavalt detailidesse minna, hea kui tuleb visioon mis suures plaanis kirjeldab mida rakendus peaks tegema. Kasutuslugude kirjeldamine ei ole üldjuhul võimalik, selleks peab analüütik kohal olema. Vahel ei suudeta isegi kirjeldada kasutuslugusid ja tuleb nimekiri nõuetest millele rakendus peab vastama.

K: Kas kasutate teadlikult mingit metoodikat?

Kasutame segu KANBANist SCRUMist ja LEANist. Kasutame seda mis tundub õige. Keerukamates olukordades kasutame prototüüpe, kuid see võtab ära väärtuslikku ressurssi – kuid jah kasutajad saavad visuaalsest pildist paremini aru. Pigem on funktsionaalne prototüüp erand kui reegel.

K: Millist viisi tarkvara arenduseks te eelistate?

Suuremate projektide puhul peaks olema ees väga detailne analüüs, kuid kõik infosüsteemid mida teeb riik, peaksid lähtuma just pigem kasutusmugavusest. Ideaalselt oleks vaja mõlemat, nii detailset analüüsi, kui ka head prototüüpi. Kui äri poolega maha istuda, siis pannakse igaks juhuks kõik nõuded kirja mis pähe tulevad. Töö käigus paljud asjad visatakse minema.

K: Kas te sõlmite kokkuleppeid äripoolega veel lisaks sellele mis on kirjas hankedokumentides?

Meil on ära jagatud vastutusalad, mis on IT poole projektijuhi ja äripoole projektijuhiga. Kuid ikkagi tekivad arusaamatused tegevustes, mida konkreetselt keegi peab tegema. Me oleme seda dokumenti ka äripoolele tutvustanud. Kindlasti oleks abi, kui nende vastutusalade juures oleks ka konkreetsed tegevuste kirjeldused, mida keegi tegema peab ja kuidas. Kuid on näha, et äri poole koolitamine pole suurt muutust toonud, endiselt on tellimusteks nimekiri nõuetest mida rakendus peab tegema. Õige oleks, et töö oleks konkreetselt kirja pandud, mida tegema peab. See peab olema äripoole huvi, et ta seisaks selle eest mida tal vaja on, selle välja mõtlemine ei ole nii keeruline.

K: Kuidas peaks koostama prototüüpe?

Meie siin kasutamine DrawIO-d selleks, kuid mingitel lihtsamatel juhtudel oleme teinud ka kohe HTML prototüüpe.

K: Kas te olete mõõtnud kuidagi kasutatavust?

Ei. Kuid seda võiks analüüsi staadiumis kirjeldada. Peaks olema võimalik välja mõelda kriteeriumeid, kui kaua mingi protsessi peaks aega võtma, eriti kui seda peab tegema mitu korda päevas.

Lisa 6 Riigi Infosüsteemi Amet, Kairi Tammik, Tuuli Pärenson

K: Kuidas te kirjeldaksite ideaalset tööprotsess täitja ja tellija vahel?

RIA on natuke ebatavaline asutus, meil puudub konkreetne äri tellija. Meie teenuseks on tehnilised platvormid, mida me pakume majast välja. Ilmselgelt kuna klientideks on paljud teised riigiasutused, ei saa me neilt ette konkreetset äritellimust. Küll me küsime neilt tagasisidet, meil on arhitektuuri nõukogud ja kogukonnad, mille kaudu me seda teeme. Arhitektuuri nõukogu otsustab tehniliste küsimuste üle, kogukonnas räägitakse läbi ärivajadused, milleks midagi tehakse. RIA siseselt vastutab asjade eest valdkonna juht, kes oleks kõige lähemal toote omaniku rollile. Tema tükeldab ärivajadused tükkideks. Projekti eest vastutab Projektijuht, kelle esimeseks ülesandeks on ärinõuete kirjeldamine. See on lühike kirjeldus, mis vahest on ainult paar rida pikk.

K: Kuidas peaks kirjeldama äriprotsessi?

Väga palju korrad sisendid varieeruvad, neid on vahest kirjeldatud 70 leheküljeliste esseedena, kuid kohati on ka nimekiri väga detailsetest tehnilistest nõuetest. Seega ei saa öelda täpselt, kuidas peaks ärivajadust kirjeldama. Inimesi kes ärinõudeid kirja panevad on väga vähe, ja neid on võimalik koolitada kuidas ärinõudeid peab kirja panema. Kirja pannakse konkreetne funktsionaalsus või probleem, mida projekt peab lahendama. Sisendid on tihtilugu nii üldsõnalised, et täiesti vabad käed jäetakse konkreetse lahenduse loomiseks.

K: Kui detailne peaks olema prototüüp? Kuidas seda hinnata?

Sõltub projektist, alati tuleb kaaluda kui palju on aega. Kui ees on kindlad tähtajad või rahalised piirangud, siis prototüüpimisest loobutakse väga sageli ja lepitakse olukorraga, et peaasi et midagi on tähtajaks valmis ja hiljem tegeletakse silumisega.

K: Kas või millistel juhtudel peaks prototüübi kõrval olema ka detailne spetsifikatsioon?

Projektijuhi tööks on riskide maandamine, meil ei ole ranget eelistust, kuidas oma projekti sisendit dokumenteerida, see on alati projektijuhi otsus, kuidas mingeid probleeme lahendada. Alati tuleb arvestada inimressursiga, kas meeskonna liikmetel

on aega projektiga tegeleda, kui inimesel on aega, siis prototüüp ega dokumentatsioon ei pea olema väga detailne. Kui inimressurssi ei ole, siis valime projekti lahendamiseks kosk mudeli, ning enne seda tehakse ühe korraga väga detailne dokumentatsioon.

K: Kuidas peaks prototüüpi hindama?

Prototüüpe me ei hinda, ega mõõda. Hindame projektide edukust, kas me püsisime tähtajas ja kui paljud meie projektid lõppevad eduka tootestamisega. RIA-s ei ole väga palju klassikalist lõppkasutaja testimist, kuna kuidas testida näiteks x-tee viimast versiooni. Palju RIA töödest on seotud turvalisusega, mida samuti ei saa kuidagi siduda lõppkasutaja testimisega. Kui rääkida, rakendustest millel on avalik nägu, siis sellistel arendusprojektidel on väga teadlik plaan, kuidas peaks toimuma testimine, ja kõiki vahetulemusi testitakse.

K: Kuidas käib projektide vastu võtmine?

Meil käiakse ikka tellitud funktsionaalsus ikka alati punkt punkti haaval läbi. Tellijast ei sõltu tööde vastu võtmine, kuna puudub konkreetne lõpptarbija. Olukord, kus funktsionaalsus on täidetud lepingu järgi, kuid lõpptulemusega tellija pole rahul, ei ole RIA-s juhtunud, seda arvatavasti RIA omapära tõttu. Tööd võtab vastu projektijuht vastavalt lepingule. RIA-l on partneritega väga hea usalduslik suhe, partneriga kellega on mõistlik suhe kus tegeletakse ühise mure lahendamisega ei teki probleeme, sellise olukorra tekkimine on väga paljuski projektijuhi õlul. RIA-s on väga selgelt defineeritud mis asi on vastuvõtu testimine, see käib täpselt vastu lepingut. Mured, kas toode on nüüd küps ja selle võib tootestada, on eraldi protsessi osa.

Lisa 7 Statistikaamet, Helen Mett

K: Kuidas te kirjeldaksite ideaalset tööprotsess täitja ja tellija vahel?

Iga suurema toote jaoks peaks olema tellijal inimene, kes saabki pühendatult sellega tegeleda. Ma ei mõtle raamatupidaja-projektijuht tüüpi inimest, kes tegeleb paberiga, vaid inimest, kes teabki, mida ja milleks tellitakse, oskab kaasata inimesi, kes hea tulemuse saamiseks suudavad projekti panustada. Ei ole võimalik olla korraga mitmes rollis, kuna kõik muud tegevused röövivad kogu aja, mis tegelikult tuleks projekti õnnetumisse investeerida. Isegi ei saa olla analüütiku, testija ja koolitaja rollis korraga. Ideaalne tellija ongi ainult tellija, mitte midagi muud. Töökorralduslikult, tsentraalne töökäskude süsteem on hea, Jira / Confluence on täitsa kasutatavad. Oluline on, et juba alguses on kokku lepitud, kuidas käib töökäskude edastamine.

Ideaalses maailmas istuksid arendajad meie juures ja teeksid siin tööd. Kuid tegelikult piisab, kui suudetakse korra nädalas kokku saada ja näost näkku infot vahetada.

K: Kuidas peaks kirjeldama äriprotsessi?

Sõltub sellest, mida ma tahan tellida, kas suurt süsteemi või midagi konkreetset. Lihtsamad asjad on võimalik joonistega ära kirjeldada, kõik keerulisemad asjad, tuleb koos andmete liikumisega tekstiliselt ära kirjeldada. See peaks olema tehtud juba enne, kui projektiga pihta hakatakse.

Äriprotsessi kirjeldamise juures on suurimaks mureks tõsiasi, et kirjeldajale on paljud asjad iseenesestmõistetavad. Tellija ei taju detaile, mis süsteemi töötamiseks võivad olla vajalikud. Protsess peab kirjeldama andmete liikumist, et midagi kahesilma vahele ei jääks, tegevused, mis tehakse, lisaks veel vastutavad inimesed.

Kuidas peaks suutma kirjeldada olukorda, kus kaks süsteemi suhtlevad omavahel – ma ei tea.

Äriprotsesside kirjeldamine on väga töömahukas, seda on jällegi raske teha muude töökohustuste kõrvalt. Suure asja puhul peab ilmselt siiski alguses olema analüüsietapp, mis paneb paika nö suure pildi. Hiljem võib tükkidesse süübida detailselt.

K: Kui detailne peaks olema prototüüp? Kuidas seda hinnata?

Prototüüp peab looma kasutajale visuaalse pildi tervikust, ning kõik detailid peavad olema tekstiliselt lahti kirjutatud, miks nad seal on ja mida nad teevad. Näiteks ei tule tellijale kunagi pähe, et kui süsteem teeb midagi, ja kasutaja ootab, peab talle kujutama liivakella. Kas selline asi peaks olema prototüübis?

K: Kas või millistel juhtudel peaks prototüübi kõrval olema ka detailne spetsifikatsioon?

Iga asi ei vaja ilmtingimata prototüüpi ja iga prototüüp ei vaja tehnilist kirjeldust. Kui tegemist on uue asjaga, peab prototüüp olema, kui midagi täiendatakse – võib juhtuda et seda ei vajata. Üldiselt peab sellise asja varem kokku leppima.

K: Kuidas peaks prototüüpi hindama?

Rakse öelda, kogemus näitab, et üldjuhul ei saa asjast ikkagi päriselt aru, enne kui see töötab. Parimaks võiks hinnata prototüüpi, kus kõikidele minu soovitud tegevustele on juurde kirjutatud tekst, mida konkreetne element täpselt teeb, siis ma jäin rahule.

K: Kuidas ei peaks tellimust tegema?

Me hetkel teeme tellimusi kokku lepitud vormi kaudu, kus kirjeldatakse ära eesmärgid, probleemid, mõõdikud, sihtgrupid, piirangud, riskid, ajavaka ja maksumuse. Lisadena esitatakse nõuete nimekirjad ja kasutatavad andmed. Selline lähenemine ei ole mõistlik, kuna see hirmutab ära tellija. Puuduvad elementaarsed näited, kuidas peaks vorme täitma ja seletused kasutatavatele väljenditele. Kui tellimuse esitamine võtab kauem aega, kui kohati lahenduse realiseerimine, siis on ilmselgelt protsessis viga, mitte tellijas. Tellimuse esitamine peab olema võimalikult lihtne ja kiire.

Lisa 8 Siseministeeriumi infotehnoloogia- ja arenduskeskus, Margus Püüa

K: Kuidas te kirjeldaksite ideaalset tööprotsessi täitja ja tellija vahel?

Õiget vastust sellele küsimusele ei ole, on projekte, kuhu sobib tavaline koskmudel, kuna nõuded on väga täpselt paigas - näiteks infrastruktuuri projektid. SMIT-i vaatest töötavad kõige paremini olukorrad, kus SMIT on teenuse pakkuja. SMIT arendab ise oma äranägemise järgi välja toote ja kliendid on teenuse tarbijad. SMIT saab olla täieõiguslik toote omanik. Muudes olukordades, kus SMIT on teenuse vahendaja ja ei suuda olla täielikult toote omanik, tuleb kliendilt küsimas käia, mida see toode peab tegema, siis ideaalses olukorras on meil olemas inimene, kes saab korraga aru nii ärist kui ta infotehnoloogiast. Ta saab tellijalt/kliendilt kas algoritmi, mida on vaja teha, olgu selleks äriprotsess mida rakendus peab tegema või mingi kasutajaliides, SMIT-i tooteomanik suudab selle põhjal valmis teha arendustööde tellimuse ja suunata selle täitmisele. Teisel juhul tuleb kliendilt probleem, mis vajab lahendamist, ning SMIT ise tooteomanikuna saab otsustada, kuidas kõige õigem on probleem lahendada. Paraku on väga väheseid selliseid olukordi, kus me saame väita, et me saame ärist aru. Avalikus sektoris on tarkvaara arenduste tellimine pikk protsess, ning äri või reaalne vajadust muutub kiiremini, kui hetkel tarkvara tellimise olukord seda võimaldab. Äri muutumise kiirus loob olukorra, kus lihtsalt nõuete nimekirja kasutades pole võimalik tarkvara tellida, muutustele on vaja reageerida kiiresti. Seega ideaalses maailmas puudub reaalne väline tellija, vaid SMIT ise on tooteomanik, kes otsustab mida tarkvara tegema peab. Loodud tarkvara on kõik koosvõimeline, et erinevaid mooduleid kombineerides on võimalik luua uusi teenuseid.

K: Kuidas peaks kirjeldama äriprotsessi?

Praegu üritatakse äriprotsessi kirjeldades kajastada hetkeolukorda, mis tuleb vana tehnoloogia paradigmast, räägitakse puudustest ja ideedest. Äriprotsessi kirjeldamisest on tähtsam arusaam, mis on probleem, mida loodav tarkvara peab lahendama. Milliseid andmeid on vaja, et langetada mingisugune otsus. Tuleb vahet teha, kas rakendus lahendab mingit ette kirjutatud algoritmi või tema eesmärgiks on juhtimisotsuste tegemisele kaasa aitamine. Katsetused panna äripool rääkima IT keeles protsessidest, mis on realiseeritavad kasutusel olevaid tehnoloogiaid kasutades ei ole

väga head, üheselt mõistetavad või tulemust andvad. Protsessist tähtsam on arusaam, kontekst probleemile, mida üritatakse lahendada. Tooteomanik, peab suutma ette antud probleemi lõhkuda väikesteks tükkideks, et oleks tõesti võimalik pidevalt midagi tarnida, isegi kui see on väga pisike osa tervest protsessist. Täitja põhiküsimus peaks olema tellijalt, et kas selline lahendus sulle sobib, mitte et mida sul vaja on. Äriprotsessist tähtsam on meeskonnal aru saada äripoole murekohtadest, mis on tema äriidee, kuskohast ta raha kaotab, millised tükid praegu töötavad ja kus need asuvad.

K: Kui detailne peaks olema prototüüp? Kuidas seda hinnata?

Prototüüpimisel ei pruugi olla üldse mõtet, kui meeskond suudab lühikeste tsüklitega kogu aeg midagi tarnida ja küsida tellijalt, et kas see sobib. See aga eeldab et täitja saab aru tellija ärist, ning et ta on nõus võtma riski, et arendustsükli lõpus ei pruugi tellija rahul olla - tema tehtut ei maksta kinni. Tooteomanik võtab vastutuse oma arenduste eest. See kas meeskond mängib midagi prototüübi peal läbi, ei peaks klienti huvitama. Kui nii tellija kui ka täitja suudavad mõlemad töötada agiilselt, siis on prototüüp võimalik tõekriteerium. Kui tellija hindab prototüüpi õigeks, võib arendusega edasi liikuda. Praeguse avaliku sektori rahastuse mudeli juures, kus raha küsitakse palju ja mitmeks aastaks ette, siis võimalik prototüüp oleks kordades keerulisem kui oleks vaja.

K: Kas või millistel juhtudel peaks prototüübi kõrval olema ka detailne spetsifikatsioon?

Tellijal on vaja kirjeldust millest on võimalik aru saada, mida konkreetne toode teeb. Pole vajadust ehitada kolossaalseid vertikaalseid süsteeme, mis sisuliselt kordavad üksteist. *App*-ide ja *startupp*-ide maailm on näidanud, et pisikesed teenused, mis on koosvõimelised on palju jätkusuutlikumad. Tellija peab saama ise otsustada, kuidas erinevaid tükke kombineerida, et ehitada keerulisemaid süsteeme. Tellija ei pea teadma, millega konkreetselt tegemist on või kuidas ta taustal töötab, tema peab saama lihtsalt otsustada, kas tal seda vaja on või kuidas ta seda ära saab kasutada erinevaid mooduleid kombineerides.

K: Kuidas peaks prototüüpi hindama?

Võtame näiteks Jira või internetipangad – palju kasutajaid ja kõik kiidavad. Jah, internetipankadele on tehtud testimisi, kus kasutajad on lastud uusi funktsionaalsusi testima ja siis nende arvamust küsitud. Ma ei oska öelda, mis selle teadmisega tehtud on, Jira / Confluenc'i süsteem on täis arendusi, mida kellegi arvates kunagi vaja on olnud, tänapäeval keegi vist enam ei tea, mida kõige see süsteem võimaldab teha. Pigem peaks küsima toote omanik, kas tema teab, miks tema toodet vaja on, milleks seda kasutatakse ning nendele muutustele siis kiirelt reageerima. IT arendus peab olema *fun*, kui IT suudab iga kahe nädala tagant midagi tarnida ja pidevalt äri poolele lahendusi pakkuda, siis on kõik õnnelikud. Täitja on vaba ja motiveeritud – äri on nõus maksma.

Lisa 9 Tervise ja Heaolu Infosüsteemide Keskus / Sotsiaalministeerium, Anneli Taal

K: Kuidas te kirjeldaksite ideaalset tööprotsess täitja ja tellija vahel?

Mina töötan tooteomanikuna ministeeriumis umbes aasta, ja sattusin töörühma kus arutatakse digiregistratuuri lähteülesannet. Seal oli juba selline valmis dokument, kus räägiti juba väga konkreetsetest asjadest. Kes mida kuskil täpselt teeb. Seal kõrval oli partnerite töörühm ja seal kõrval ka standardimise töörühm. Mina olin selles sisutöörühmas, kus üritati vastata partnerite töörühma detailsetele küsimustele. Ühel hetkel tekkis küsimus, et mis on üldse selle küsimuse eesmärk ja mis on kogu protsessi eesmärk üldisemalt. Eesmärgiks oli olemasoleva olukorra digitaliseerimine, et kogu protsess muutuks kiiremaks. Õhku jäi küsimus, et kuidas seletada see muutus ära patsientidele ja inimestele kes on sellega seotud, siis alameesmärgiks ei saab olla efektviisus küll, kiud peamiseks eesmärgiks peab olema muuta viisi kuidas patsient üldse süsteemis liigub. Ehk kõigepealt peaks alustama sellest. Alguses oli üksjagu segadust, kuid lõpuks jäädi nõusse. Tegime läheülesande ümber, leidsime et võimalikke lahendusi on mitu ja esitasime leitu juhtrühma. Juhtrühm, küsis et kas teenuse osutajad on sellega kursis. Ei olnud, nemad tuli kaasata ja hakkasime kõike uuesti arutama – lõpuks saime kõik paberile ja kokku lepitud. Nüüd lähme uuesti juhtrühma, et selle tulemusena saaks õigusaktid rakendatud ja IT süsteemid muudetud jne.

Kui sisu pool oli kokku lepitud, mindi lähteülesandega partnerite juurde ning sealt tuli vastus, et kuidas te saate midagi sellist otsustada, et see ei vasta üldse sellele mis varem on kokku lepitud? Me ei osanud arvata, et lähteülesandele tekib mingisugune vastuseis, kuna meil oli olemas dokument, mis kirjeldas töövoogu ja baasäriprotsesse. Plaan oli lihtsalt need protsessid kooskõlastada ja eeldus oli et kui lõpuks tuleb korralduslik dokument, mis kirjeldab suurt pilti, ei tohiks sellele olla suurt vastuseisu.

Probleemiks on alati detailid, et standard, siinkohal kasutatavad saatelehed, on välja töötatud inimeste poolt, kes ei kaasanud sotsiaalministeeriumit, meie soovisime, et patsiendi liikumine süsteemis oleks seotud saatelehtedega ja üheks võimaluseks oleks perearsti võimalus saata patsienti otse EMO-sse, millega polnud EMO-d nõus. Ehk kuna osapooli on palju, siis kinnitamised on alati keerukad.

Seega võiks öelda, et vorm, kuidas sisendit anda pole oluline, kuna mingi osa süsteemist on juba valmis arendatud ja uute asjaolude valguses ei olda nõus süsteemi ümber tegema. Lahenduseks oleks kõikide otsuste fikseerimine juhtrühmas. Kuna partnereid on palju ja kõik ei ole sotsiaalministeeriumite alluvuses (perearstid), siis üritatakse kõike lahendada juba olemasoleva taristu raames. Arvatavasti oleks mure lahenenud iseenesest, kui neid oleks olnud rohkem kaasatud. Hetkel lahendasime olukorra lihtsalt tugevama juhtimisega.

K: Kuidas peaks kirjeldama äriprotsessi?

Meil on kõik protsessid kirjeldatud BPMN-is, need on kirja pannud meie abiline IT poole peal. BPMN iseenesest pole keeruline kuid selle kasutamiseks on kindlasti vaja koolitust.

K: Kui detailne peaks olema prototüüp? Kuidas seda hinnata?

Digiregistratuuri tehes on meid disaini juurde hästi kaasatud. Kõigepealt tehti meiega palju vestluseid, pandi kirja milline võiks protseduur olla ja tehti valmis prototüüp. Siis arutati prototüüp valmis, seda tutvustati teenuseosutajatele, nemad kiitsid selle heaks. Selle tegid meie majasisesed TEHIK-u arendajad. Prototüüp on sisendiks mitte ainult IT-le vaid ka kommunikatsioonile ja õigusaktidele. Kuna selle pealt pannakse kirja täpsed baasärireeglid (alamprotsessid). Kindlat formaati, kuidas me protsesse kirjeldame ei ole, küsime lihtsalt põhilist: mis, miks ja kuidas.

K: Kuidas peaks prototüüpi hindama?

Kaardistasime kõikide erinevate osapoolte arvamused prototüübile ära ja tegime vastavad muudatused. Meil on mõõdikud üldiselt valmis mõeldud alati, ja me mõõdame protsessi ennast küll, kuid mitte prototüübi peal. Digiregistratuuri puhul me mõõdame patsiendi erinevaid ooteaegasi, kui kiiresti ta saab lõpuks eriarsti juurde.

Lisa 10 Eesti Energia AS, Päärn Brauer

K: Kuidas te kirjeldaksite ideaalset tööprotsessi täitja ja tellija vahel?

Meil on õnneks tellija ja täitja ühes majas olemas. Minu, e-kanalite valdkonna juhi, haldusalasse jääb ka sisemiste rakenduste kasutusmugavus. Me õnneks ehitame palju asju nullist. Ajalooliselt on meie rakendused disainitud projektijuhi, analüütiku ja programmeerija poolt. Nüüd me proovime jõuda olukorrast, kus analüütik teeb UX-i, natukene välja, kuna tema seda teha ei saa. UX-i disainer peab tegema rakenduse kirjelduse, analüütik peab ära kaardistama kõik nurgad, mis midagi teevad, aga disainer peab rääkima tellijaga ja välja selgitama kuidas midagi teab töötama. Seda enam et sisemiste kasutajate puhul, keda on umbes 50 kuni paarsada, me teame neid, nendega on võimalik rääkida ja täpselt välja selgitada mida vaja. Prototüüp peaks ära katma 80% ärilistest vajadustest, ja erisused peavad juba seal välja tulema. nii ta ongi, et analüüs, mida me teeme ja kuidas see välja näeb on 80% ja spetsifikatsioon seal juures on väike osa ainult.

Plaan on see, et me kirjeldame ära protsessi, kuidas peaks tarkvara tellimine välja nägema, ma plaanime ära kirjeldada selle protsessi. Teisest küljest me loome stiiliraamatu mis ütleb, kuidas mingi asi peab välja nägema, ning seda saavad kasutada välised partnerid kes selle asja lõpuks valmis teeme. Võiks isegi nii öelda, et majasisese prototüübi peab tegema oma inimene valmis, kuna väljast ei saa seda korralikult kätte.

K: Kuidas peaks kirjeldama äriprotsessi?

Sõltub olukorrast, kui me räägime avalikust veebist, siis me tunneme et meie teame kõige paremini ja siis võiks öelda, et peadisainer on kergelt dikteerija rollis – tema teab kõige paremini. Teistel juhtudel, äripool üritab öelda mis peaks olemas olema ja kuidas see töötama peaks, kuid tegelikult on tihtilugu nemad sama targad kui meie.

Põhjuseks on see, et me üritame kirjeldada asju mida pole olemas, meil on täpselt sama idee milline see peaks välja nägema kui töö tellijal. Kui tegemist on mingi spetsiifilise asjaga, siis analüütik aitab selle ära dokumenteerida ja siis disainer võtab selle üle. Vahepeal käiakse asjad tellijaga üle ja saadakse kinnitust et ollakse õigel teel.

K: Kui detailne peaks olema prototüüp? Kuidas seda hinnata?

Axure'ga tehtud funktsionaalsed prototüübid on tase, mida teeb analüütik. Mina kujutaks ikkagi ette, et disainer teeb selle esimese prototüübi Photoshop'is ja kasvõi mustvalge, kuna värv kipub tihtilugu segama. See annab tellijale kõige parema ülevaate, kuna asi on õiges skaalas ja on näha, kas asjad mahuvad ekraanile ära.

K: Kas või millistel juhtudel peaks prototüübi kõrval olema ka detailne spetsifikatsioon?

Kõigepealt lepime äriga nõuded kokku, seda teeb analüütik. Me paneme kirja andmed mis infot on vaja. Visandame selle *wireframe*-na ära, paneme kirja teenused mida on vaja, et rakendus töötaks. Disainer teeb nendest funktsionaalsed *wireframe*-d ja viiakse läbi esmane testimine.

Meie võtame *back-end* poole ainult majast väljast sisse, ja *front* on pigem meie tehtud. Seega kirjeldamine toimub API tasemel. Prototüübi kõrval on *user story*-d ja *use case*-id, kuid veel täpsemaks me ei lähe. Teenused kirjeldame ära API, andmeedastuse tasemel. Pigem hindame me *front-end*-i arendamisel visuaalset sisendit.

K: Kuidas peaks prototüüpi hindama?

Me oleme küsinud lõppkasutajate arvamust ikka, kui rakendusel on majasisesed lõppkasutajad. Üldjuhul saame paari inimese läbi käimisega aru, mis on valesti ja kuhu peaks edasi vaatama.

Lisa 11 Arendusvajaduste kavandamine

Eduka tarkvaraprojekti aluseks on selge visioon olemasoleva probleemi lahendamisest. Visioon toetab tarkvara projekti kogu tema elutsükli ajal, alustades arendusest ja haldusest, kuni tema mahakandmiseni välja. Visiooni hoiavad projektimeeskonnas ühiselt tooteomaniku, projektijuhi ja analüütiku rollid. Rollid võivad olla määratud ühele või mitmele inimesele. Edukas visiooni kirjeldamine ja sellest töökäskude moodustamine tootekataloogi (*Product backlog*) nõuab endast kolme järgneva sammu läbimist.

1. **Äriprotsessi kirjeldamine**

Pealkiri	
Probleem	
Mõisted	
Etapp 1	Etapi detailne kirjeldus
Etapp 2	Etapi detailne kirjeldus
Etapp 3	Etapi detailne kirjeldus
Väljund	

Pealkiri – projekti nimetus, võiks olla piisavalt lühike, et seda saaks kasutada tarkvara arenduse protsessis ja kanda endas rakenduse olemust.

Probleem – lühike kirjeldus probleemist mida projekti raames üritatakse paremaks muuta/lahendada. Kui tegemist on olemasoleva protsessi digitaliseerimisega või põhiprotsessi liitmisega, siis tuleks seda ka vastavalt kirjeldada.

Mõisted – siia peaks kirjutama kõik mõisted ja lühendid, mis võivad olla võõrad arenduspartnerile.

Etapp 1 kuni Etapp n – kirjeldab kokkuvõtvate pealkirjadena projektis lahendatava töövoo etappe loogiliste tegevuste jadana.

Etapi kirjeldus – kirjeldab iga detailselt iga töövoo tegevuse sisu täpsemalt. Kõikide tegevuste juures peab ära kirjeldama 4 asja: vajalikud andmed ehk sisend; vastutav isik/roll; tegevuse eesmärk/vajadus; millise eduka tegevuse järel liigutakse järgmise sammu juurde.

Väljund – lühidalt kirjeldada tegevuste kokkuvõte, mis on informatsioon mis tekib eelpool kirjeldatud tegevuste jadana. Väljundiks võib olla otsus või erinevate andmete kombineerimisel saadud tuletis.

2. **Prototüüpimine**

Visandamise koosolek tuleb korraldada peale äriprotsessi kirjelduse valmimist. Koosoleku eesmärgiks on kontrollida, kas kirjeldatud äriprotsess on terviklik ja seega ka teostatav – määramatuse osa projektis tuleb viia miinimumi, et tulemus oleks prognoositav. Visandamise koosoleku juures peavad olema esindatud kõik kolm rolli, tooteomanik, projektijuht ja analüütik. Vastavalt vajadusele võib inimesi koosolekule juurde kutsuda.

Tooteomaniku roll koosolekul on äriprotsessi kirjeldamine.

Projektijuhi roll koosolekul on dokumenteerimine, kui toimub muudatus eelpool kirjeldatud protsessis ja hinnangu andmine kas planeeritud tegevused on teostatavad.

Analüütiku roll koosolekul on piirata tooteomaniku äriprotsessi viisil, et see kataks ainult äri põhiprotsessi vajadused. Lisaks aitab analüütik vajadusel projektijuhti kirjelduste sõnastamisel.

NB! Visandeid tuleb võtta kergelt ja huumoriga, need ei oma esteetilist väärtust.

Visandamist alustatakse punktist, kus kasutaja on edukalt avanud rakenduse avakuva, mida ta siis näeb. Üldiselt on esimeseks kuvaks audentimise aken (rakendusse sisenemine). Visandamise käigus tuleb alati vastata kolmele küsimusele: Mida ja kuidas kasutaja ekraanil näeb; Millised on otsused ja tegevused, mida kasutaja peab tegema, et liikuda töövoo järgmise sammu juurde tööprotsessis; Kas ja kui kasutaja peab seda sammu tegema päevas kümneid kordi, siis kuidas seda tegevust mugavamaks teha? Iga uus kuva visandatakse tahvlile ja küsitakse samu küsimusi.

Prototüübi visandeid ühe sammu kohta võib tulla üks või mitu. Kui visandeid tekib rohkem kui üks, on sobiliku valimiseks vaja teha prototüübi testimine.

3. Prototüübi testimine

NB! Puhtand ei pea olema paberile kantud käsitsi, selleks võib kasutada digitaalseid abivahendeid, kasvõi Excelit. Visandite testimine nõuab testijalt natukene ettekujutusvõimet ja kõigil ei pruugi seda olla.

Prototüübi testimise jaoks peab tooteomanik või analüütik välja mõtlema testimise stsenaariumi. See stsenaarium peab endas sisaldama tegevusi, mille lahendamisega saab üks kasutaja roll algusest lõpuni hakkama. Testimise jaoks on prototüüpimise käigus loodud visandid kantud paberile, nö puhtanditeks. Testijaid võiks olla minimaalselt kolm ja nad ei tohiks olla projektiga seotud meeskonna seast.

Testimise alguses asetatakse testija ette esimese kuva visand, milleks tavaliselt on sisse logimise aken ja küsitakse kolme küsimust: kirjelda mida testija visandil näeb; mida peab kasutaja tegema et edukalt liikuda stsenaariumi lahendamisele lähemale; mis võiks siin kuval olla teisiti?

Tooteomanik on testimise juures, kui kasutajal tekib küsimusi visandite kohta. Näiteks mis on seal kujutatud või kuidas seal midagi peaks töötama. Esimese ja kolmanda küsimuse vastust on puhtalt informatiivne. Kui testija vastab teisele küsimusele edukalt, siis asetatakse tema ette järgmise kuva visand. Kui testija vastab küsimusele valesti, siis tooteomanik seletab mis oleks olnud õige tegevus ja asetab ette ikkagi järgmise kuva visandi.