Tallinna Ülikool
Informaatika Intituut

HCI-metoodikate kasutuslevõtt agiilses arendusprotsessis

Seminaritöö

Autor: Roger Puks
Juhendaja: Zahhar Kirillov, MSc

Tallinn 2014

Sisukord

Sissejuhatus

Tänapäeval on turule tekkimas iga päevaga üha enam tarkvaralahendusi. Väga mitmed rakendused peavad nägema vaeva, saavutamaks konkurentsieelise teiste tarkvaratootjate ees. Pakutav funktsionaalsuse kvaliteet ning toiminguterikkus suudavad kasutajaid meelitada aga teatud piirini – ühel hetkel muutub oluliseks kasutajasõbralikkus, mille puudumisel võivad kaotada osatähtsuse ka muud omadused.

Lisaks tarkvaramaastiku pidevale kasvule on üha enam kanda kinnitamas ka agiilsed arendusmetoodikad. Selle kasvutrendi taga (VERSIONONE Agile Made Easier) on põhiliselt kõnealuste metoodikate üsna vähene ressursinõudlikus projekti algusetappides, võime kiiresti esialgseid "käegakatsutavaid" tulemusi saavutada ning samuti kliendikesksus kogu projekti vältel. See tähendab, et projekti tellijat hoitakse kogu projekti jooksul arenduskäiguga kursis ning ta saab anda jooksvalt tagasisidet, hoides kogu arendusprotsessi ühitatuna oma ärieesmärkidega. Erinevalt klassikalistest arendusmetoodikatest (Waterfall), kus klient näeb töö tulemusi alles mitmekuise projekti lõpus, on agiilsed metoodikad võimelised juba esimeste tööpäevade lõpuks esitlema algseid töötulemusi. Väledaid arendusmetoodikaid on mitmeid - Scrum, Extreme Programming, Kanban - kuid nad kõik jagavad ühtset eesmärki, milleks on kasutaja ärinõudmiste rahuldaine võimalikult väheste ressurssidega.

UX-i hääbuvat rolli arendusmaastikul on raske mitte märgata. Valdava osa UX-disaini tarbijatest moodustavad suurkorporatsioonid, kelle arendusiteratsioonid on pikema kestusega ning pigem traditsioonilised (Gube, 2010). Agiilseid metoodikaid hindavad *start-up* ettevõtted on pigem hakanud aga vaatama UX-disaini kui liigset ajakulu nõudvat protsessi, mille väljund ei ole arendustöö jaoks määrava tähtsusega.

Kasutatavuse disainimisele on võimalik läheneda kahe perspektiivi alt. Esimene neist tähendab erinevate heuristikate rakendamist e. eelnevalt defineeritud üldreeglite rakendamisel, arvestamata konkreetse ülesande, konteksti või sihtrühmaga. Teine lähenemine kujutab endast UX-disaineri poolset pädevust erinevates kasutatavusmetoodikates, tänu millele oskaks ta valida sobivaima ning seda rakendada ka enda projektis. Kuigi tihtipeale on agiilsete ettevõtjate meeskondades esindatud ka kasutatavusdisainerid, põhineb nende töö valdavalt heuristilistel teadmistel. Traditsioonilisemad kasutatavuspraktikad leiavad oma ressursinõudlikuse tõttu pigem harva rakendust.

Interaktsioonidisainerite sobitamine agiilsesse arendusprojekti võib kujuneda üsna keeruliseks. Üldlevinud kasutatavusvaldkonna töömetoodikad võivad olla üsna aeganõudvad, sattudes juba seetõttu vastuollu agiilsete projektide olemusega. Headeks näideteks aeganõudvatest praktikateks on päevikute pidamine (*diary studies* - kasutajatel palutakse päeviku vormis jäädvustada oma emotsioone ning kogemusi rakendustega, mille kasutuskogemus võib aja jooksul muutuda, nt. asukohatundlike mobiilirakenduste puhul (Santafe, 2013)), töötoad kasutajatega (*participatory workshops*, mille raames kogutakse kokku potentsiaalsed kasutajad, et nende arvamuste ja kogemuste abil lahendada erinevaid disainiprobleeme (JISC infoNet, 2012)). Kuid kas see tähendab, et UX tuleks kõnealustest projektidest hoopis välja jätta või oleks otstarbekam leida meetodeid, kuidas kaasata kasutatavusdisainerid arendustöösse ilma liigseid ajalisi ohverdusi tegemata?

Käesolev seminaritöö analüüsib UX-metoodikaid ning nende sobivust agiilsetes tarkvaraarendusprotsessides. Alustuseks esitatakse sissejuhatav tutvustus HCI ning UX valdkondadesse. Seejärel püütakse erinevatele temaatilistele publikatsioonidele tuginedes välja uurida, mis on UX-disaini tahaplaanile jätmise põhjused ning eelnevast lähtudes luuakse raamistik, millele tuginedes on võimalik implementeerida arendustöös nii agiilseid metoodikaid kui ka UX-disaini.

1 Human-Computer Interaction

Inimese-arvuti interaktsiooniks (*Human-Computer Interaction - HCI*) nimetatakse teadust, mis uurib inimese käitumist arvutitega ning erinevate arvutite kasutatavustaset. Tegu on teadusharuga, mille algusajad ulatuvad 1980. aastatesse, kombineerides infotehnoloogia kognitiivpsühholoogiaga (Beale, 2007). Tänaseks päevaks on HCI-st kujunenud välja tootearenduse lahutamatu osa. HCI metoodikate kasutamine erinevates arendusprotsessides on muutumas üha enam standardiks, kuna kehvade disainilahenduste rakendamine võib viia mitmete probleemideni - alustades kasutajate leigest huvist, lõpetades toote kasutuskõlbmatusega. (SO, 2008)

HCI kui teadusharu on olnud pidevas arengus viimased kolm aastakümmet, hoomates tänaseks päevaks mitmeid erinevaid rolle - olulisi rolle mängivad HCI-metoodikates nii infotehnoloogid, sotsiaalteadlased, psühholoogid kui ka disainerid.

Nagu võib välja lugeda terminist endast, koosneb HCI põhiliselt kolmest erinevast osast - kasutajast, arvutist ning nendevahelisest suhtlusest. Kasutaja all peetakse tavaliselt silmas nii individuaalset kasutajat kui ka kasutajate gruppi (Beale, 2007), kes kasutavad uurimise all olevat toodet (olgu tegu füüsilise toote või hoopis tarkvaralahendusega).

Kasutajate puhul tuleb arvesse võtta ka mitmeid erinevaid faktoreid, mis võivad kasutuskeskkonna kujundamist mõjutada - sh. kasutajate kultuurilised/etnilised eripärad, nende eelnev kogemus uurimise all olevate toodetega, nende erinevad sensoorsed võimed jne. HCI praktikates tuleb arvestada eeldusega, mida kasutajad võivad uuritavast tootest oodata ning sellset lähtuvalt luua produkt, mis on kasutajatele atraktiivne, kuid samas ka lihtsasti kasutatav (SO, 2008).

Ka arvuti all võidakse silmas pidada üsna mitmeid erinevaid seadmeid, alustades tavalisest arvutist ning lõpetades suurte arvutisüsteemidega. See tähendab, et sõna “arvuti” taga mõeldakse pigem uuritavat objekti, millega inimene või inimgrupp hakkab suhtlema. Arvutid võivad seega olla nii IT-süsteemid, mobiilid, veebisaidid kui ka muud analoogsed seadmed.

Interaktsioon on kõnealustest kolmest mõistest ilmselt kõige raskemini defineeritav, kuid teisest küljest ka olulisim osa. Interaktsiooni eesmärk on kindlustada, et inimene ning arvuti suhtlevad

omavahel probleemideta, olenemata nendevahelistest erinevustest. Interaktsiooniteguri abil peab tekkima kasutajasõbralik toode või süsteem, sealjuures leides tasakaalu kasutajate nõudmiste ning projekti eelarve vahel (Beale, 2007).

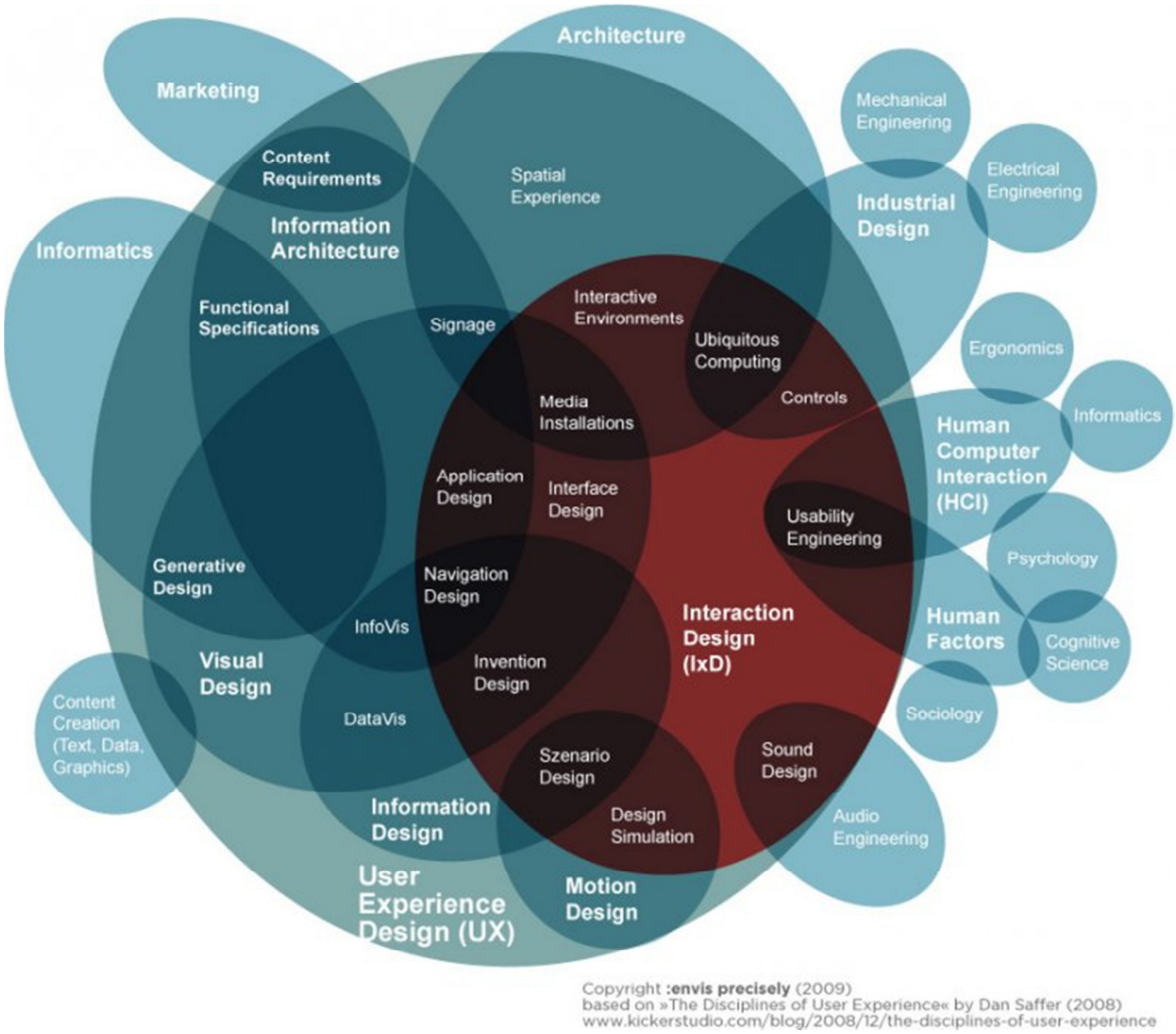
1.1 Ajalugu

Kuni 1980. aastate alguseni oli arvuti pigem nišitoode, mis polnud massitarbimisse veel levinud. See kõik hakkas muutuma aga personaalarvutite turulesaabumisega. Personaalarvutid muutsid arvutikasutuse võimalikuks kõigile - levima hakkasid nii tavatarkvara (tekstitöötlusprogrammid, arvutimängud) kui ka arvutiplatvormid (sh programmeerimiskeeled ja operatsioonisüsteemid). See pani aluse tõsiasjale, et kompuutrid pidid muutuma kasutajasõbralikuks ka üldtarbijatele.

HCI esialgne ning praegune põhifookus asub mõistel "kasutatavus" (*ing k Usability*), mis vastavalt ISO 9241 standartile defineeritakse järgnevalt: ulatus, milleni teatud toodet saab kasutada, et saavutada teatud eesmärke efektiivsuse, tõhususe ja rahuloluga (Bevan, 2006). Kui HCI algusaegadel nähti kasutatavuse all toote muutmist lihtsast õpitavaks ning kasutatavaks, siis tänaseks päevaks on sellest kasvanud välja üsnagi lai ning probleemiderohke teadusvaldkond, mis näitab HCI olulisust tootearenduses. Kasutatavusvaldkond tegeleb lisaks tavapärasele kasutuslihtsusele ka selliste tooteomadustega nagu efektiivsus, sujuvus, arenguvõime, tõhusus jpm.

HCI algas kui rangelt arvutivaldkonda puudutav teadusharu, kuid on oma arenguaastate jooksul kasvatanud oma haaret ka lugematutesse muudesse valkondadesse, olles nüüd õpetatud enamustes õppekavades, mis puudutavad kas otseselt või kaudselt infotehnoloogiat ja/või tootearendust - HCI kombineerib kõikide nende õppekavade teadusharud.

Selle kasvu paratamatuks kaasmõjuks on olnud ka erinevate HCI-valdkondade spetsialistide teke. Kui inimese-arvuti interaktsiooni algusaegadel oli keerulisem HCI-spetsialistide seas erinevaid rolle välja tuua ja võrrelda, siis tänapäeval võib eristada kasutatavusdisainereid, interaktsioonidisainereid, kasutajaliidese spetsialiste jpm.



Pilt 1 Visualisatsioon HCI valdkondade varieeruvusest ning kattuvusest

1.2 HCI metoodikad

Inimese-arvuti interaktsiooni arenguaastate jooksul on välja kujunenud mitmeid erinevaid metodoloogiaid. Osad neist kujutavad endast lihtsalt mudelit, milles kirjeldatakse kasutajate, disainerite/arendajate ning süsteemide omavahelist suhtlust, kuid eksisteerib ka põhjalikumaid mudeleid, mis kombineerivad endas erinevaid meetodeid, mida rakendada arendus- või disainiprotsessi jooksul. Metoodikate eesmärgid on läbinud aastakümnete jooksul ka mõningaseid muutuseid. Kui algsed metoodikad soovitasid (Carroll) disaineritel lähtuda kognitiivpsühholoogias paikapandud fundamentaalsetest heuristikatest, võimaldades mingi

tasemeni aimata kasutaja eelistusi toote või keskkonna suhtes, siis tänapäevased metoodikad soovitavad pigem keskenduda emotsiooidele ning kogemustele, mida kasutajad peaksid omandama toodet kasutades. Järgnevalt saab välja toodud tänapäeval ühe kõige populaairsema HCI metoodika kirjeldus ning erinevad meetodid, mis selles rakendatakse. Just nimelt nende meetodite sidumist agiilsete arendusprojektidega püüabki kõnealune seminaritöö arendada.

1.2.1 Kasutajakeskne disain

Kasutajakeskne disain (*UCD - User Centered Design*) on disainipraktika, mille raames arvestatakse kogu disaini- ning arendusprotsessi jooksul põhiliselt kliendi soovida, vajaduste ning piirangutega (User Experience Professionals Association)asutajakesksele disainimeetodile tuginedes loodud tooted on oma olemuselt suunatud lõpp-tarbijale - see tähendab, et kogu keskne probleem arendustsüklis on lõpp-kasutaja eeldatav visioon tootest, mitte tema harjumuste muutmine vastavalt toote kasutamisnõuetele.

UCD järgib suurt hulka erinevaid meetodeid ning praktikaid, et analüüsida, disainida ja ellu viia nii raudvaralisi, tarkvaralisi kui ka kasutajaliidese alaseid lahendusi. Kasutajakeskne disain on loomuselt iteratiivne protsess, kus disain ning selle elluviimine leiavad aset projekti algusfaasis. Jeffrey Rubin on toonud oma teoses "*Handbook of Usability Testing*" välja kolm põhiprintsiipi, mida kasutajakesken disain järgib (Gilbert Cockton, 2008)

1. Varajane keskendumine kasutajatele ning nende eesmärkidele;
    1.1 Struktureeritud ning süstemaatiline infootsing antud teemal;
    1.2 Disainerid koolitatud ekspertide poolt enne andmete kogumise alustamist;
2. Empiirilised mõõtmised ning tootevaldkonna senise kasutamisajaloo uurimine;
    2.1 Fokuseerida lihtsalt õpitavusele ning kasutatavusele;
    2.2 Testida prototüüpe reaalsete kasutajate peal;
3. Iteratiivne disain;
    3.1 Toodet testitakse, muudetakse ning disainitakse korduvalt;
    3.2 Lubatud on tootedisaini täielik ümbertöötamine peale varajaste kontseptsuaalsete disainiideede läbitestimist.

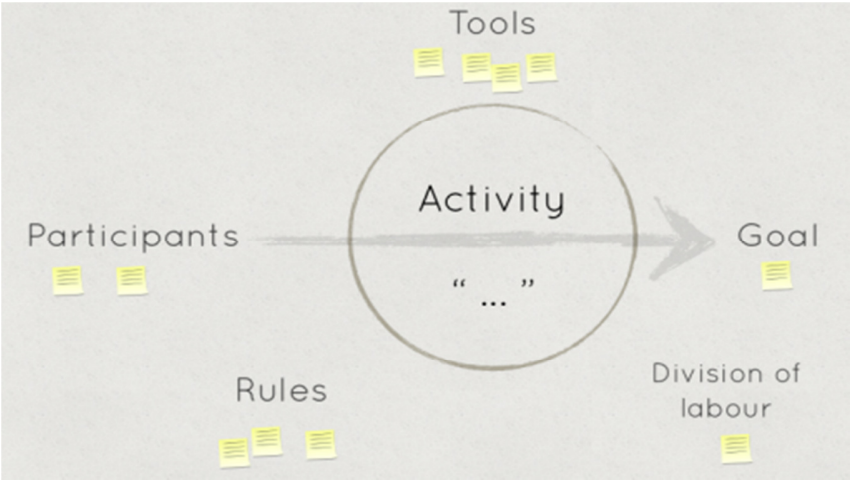
Vastavalt ISO standardile 9241 (p. 11), mis tegeleb inimese-arvuti interaktsiooni ergonoomika defineerimisega, on UCD eesmärgiks luua tooteid, mis on kõrge kasutatavustasemega (Bevan, 2006).

Arendusprotsessides tähendab kasutatavusteooria meetmete rakendamine kogu projekti vältel antud teemaga tegelemist. Erinevaid meetodeid, millega parandada loodava toote kasutatavustaset, tuleb rakendada erinevatel projektietappidel ning vajadusel ka korduvalt. Sellest tulenebki kasutajakeskse disaini aeganõudvus ning põhiline vastuolu agiilsete arendusmetoodikatega. UCD nõuab kasutajate kaasamist kõikides etappides, alates nõuete defineerimisest ning lõpetades turustamisega.

1.2.2 Tegevuskeskne disain

Tegevuskeskne ehk *activity centered* disain on üks kehtivatest alternatiividest kasutajakesksele disainile. Selle kohaselt peavad olema disainitöö keskmes rakenduses läbiviidavad tegevused, mida kasutajad teevad ning mida süsteem peab toetama. Näiteks fotoalbumite loomiskeskkonda arendav meeskond peaks keskenduma erinevatele tegevustele, mida kasutaja võib seal läbi viia - fotode üleslaadimine, albumite loomine ja redieerimine jne. Kõnealuse metoodika toetajad väidavad, et UCD viib mitmetel põhjustel tupikseisudeni arendusprojektides, eestkätt seetõttu, et tehnoloogiat on keerulisem ümber vormida kasutajate soovidele vastavalt kui kasutajate harjumusi tehnoloogiale vastavalt (Norman). Tegevuskeskse metoodika peamiseks eeliseks peetakse tema laia haaret. Kasutajakeskne disain keskendub mingile kitsale audientsile (rahvus, kultuur), mistõttu ühele sihtgrupile loodud toode võib osutuda teisele vägagi ebamugavaks. Protsessidele keskenduv disainiprojekt ei erista aga rahvusi, vaid loob erinevatele protsessidele keskendudes rakenduse, millega suudavad harjuda kõik erinevad sihtgrupid. Lisaks sellele võib kasutajakeskne metoodika end kiirelt ammendada, kuna inimeste harjumused ning käitumismustrid võivad aja jooksul üsna kiiresti muutuda. See tähendab, et tegevuskeskse metoodika pooldajad peavad kasutajakeskset disaini liialt staatiliseks inimeste jaoks, kes on oma olemuselt üsna dünaamilised (Variano, 2012).

Tegevuskeskses metoodikas leiavad sageli kasutust tegevusdiagrammid. Tegu on graafiliste skeemidega, kus on välja toodud erinevate tegevuste jaoks nõutud sammud (algoritmid).



Pilt 2 Aktiivsusdiagramm

1.2.3 Geniaalne disan

Geniaalne disainimetoodika vastandub enamustele teistele metoodikatele, kuna selles ei lähtuta disainiprotsessis kelelgi või millegi eelistustest, vaid usaldatakse kogu disainitöö disaneri “geniaalsusele”. Erinevalt nt. kasutaja- ja tegevuskesksest metoodikast puuduvad geniaalses disainis ka eelnevalt defineeritud meetodid ja kindel dokumentatsioon, mille abil disainiprotsessi läbi viia. See eeldab omakorda, et arendajad tunnevad väga läbivalt oma sihgruppi ning nende ootuseid, garanteerimaks projekti edukuse (Nielsen, 2007)

.

Geniaalse disainimetoodika rakendamist võib võtta kui järk-järgulist protsessi, kus esimese sammuna õpitakse tundma oma kasutajaid. See võib küll tähendada madalamat sissetulekut esimestes projektides, kuid suuremat õnnestumistõenäosust edaspidistes töödes.

Geniaalse disainimetoodika esimene samm on põhjalik uurimistöö. Disaini vaadeldakse kui lahendamist vajavat probleemi, mille olemusse süüvitakse väga põhjalikult. Teatud juhtudel kostatakse ka oma uurimistulemuste kategorisserimiseks katalooge, kuhu märgitakse olulisimad leiud. Selline uurimistöö kestab kogu projekti vältel, analüüsides iga teatud perioodi tagant, kuivõrd edukas on senine lahendus ning kas seda tuleks muuta.

Ka geniaalset disainimetoodikat rakendavad meeskonnad võivad oma töös kasutada personasid ning stsenaariume, kuid ka neid luuakse lähtuvalt oma teadmistest. Põhirõhk asetatakse siiski tulemuslikkuse võimalikult kõrgele tasemele.

Antud lähenemisviis on märksa riskantsem kui teised disainimetoodikad, mis lähtuvad välistest faktoritest oma töös. See tähendab, et kuigi projektidel on geniaalset disainimetoodikat rakendades suur oht läbi kukkuda, on pea sama tõenäoline ka edu saavutamine. Näitena kõnealuse metoodika rakendajatest võib välja tuua tehnoloogiafirma Apple-i, kelle poolt loodud esimese põlvkonna iPod Shuffle oli välimuselt väga innovaatiline ning osaliselt seetõttu ka väga edukas (Walters, 2008)

2 Agiilsed arendusmetoodikad

Agiilseteks nimetatakse hulka erinevaid arendusmetoodikaid, mis vastanduvad oma olemuselt traditsiooniliste ehk fundamentaalsete metoodikatega. Agiilsete lähenemisviiside eesmärgiks on ettevõtete kulude kokkuhoid ning parem valmisolek erinevateks ootamatusteks - näiteks muutused disainis või funktsionaalsuses (Abrahamsson, Salo, Ronkainen, & Warsta, 2002).

Väledad arendusmetoodikad hakkasid laiemalt arenema 1990. aastate keskpaigas, kuigi agiilseid tunnuseid kandvaid projektimetoodikaid võib täheldada ka 1960. aastatest.

Kuigi metoodikaid on erinevaid, olid neil kõigil ühised iseloomujooned - nad sidusid seni eraldiseisnud arendusmeeskonna, äritöötajad ning kliendid ühtseks tihedalt suhtlevaks meeskonnaks. Senise kirjaliku dokumentatsiooni asemel hakkas olulisemaks muutuma näost-näkku kommunikeerumine ja pidevate lühikeste iteratsioonidega uute tarkvarajuppide väljatöötamine. Agiilsete arendusmetoodikate "ametlikuks" sünniajaks võib aga pidada 2001. aasta veebruari, mil 17 tarkvaraarendajat kohtusid Snowbirdis USA-s, et arutada omavahel erinevaid kergemaid-kiiremaid arendusviise. Selle kohtumise tulemuseks oli agiilse tarkvaraarenduse manifest, millest kasvas lõpuks välja *Agile Alliance* - mittetulunduslik assotsatsioon, mille eesmärgiks on agiilsete tarkvaraprojektide propageerimine. Agiilse tarkvaraarenduse manifest kõlas järgmiselt (Beck, et al., 2001):

Tarkvara luues ning teisi tarkvara loomise juures aidates oleme leidnud selleks tööks paremaid viise.

Oleme hakanud hindama:

- *inimesi ja nendevahelist suhtlust rohkem, kui protsesse ja arendusvahendeid*
- *töötavat tarkvara rohkem, kui kõikehõlmavat dokumentatsiooni*
- *koostööd kliendiga rohkem, kui läbirääkimisi lepingute üle*
- *reageerimist muutunud oludele rohkem, kui algse plaani järgimist*

Ka parempoolsetel teguritel on väärtus,

kuid me hindame vasakpoolseid tegureid kõrgemalt.

Lähtuvalt agiilse tarkvaraarenduse manfiestist, võib seega agiilset tarkvaraarendust üldistavalt pidada filosoofiaks, mille rakendamiseks on välja töötatud erinevad metoodikad. Neist populaarseimateks võib pidada selliseid metoodikaid nagu *Extreme Programming* (*XP*), Scrum, Kanban jm.

2.1 Extreme Programming

Extreme Programming-u (XP) algusajad ulatuvad 1996. aastasse. Tänaseks päevaks on sellest välja kasvanud üks enimkasutatavamaid metoodikaid agiilsetes arendusmeeskondades, mis on küll viimastel aastatel pisut populaarsust kaotamas (Smith, 2010).

Kõnealuse metoodika põhitunnuseks võib pidada klassikaliste arendusmetoodikatega võrreldes vähem paberimajandust - dokumentatsioonile keskendumise asemel on arendusemeskonna peaeesmärgiks kvaliteetse toote võimalikult kiire arendamine.

Tööprotsessis kannavad olulist rolli kõik meeskonnaliikmed - klient, arendusmeeskond ning projektijuht. Kogu meeskond lähtub oma töös erinevatest kasutuslugudest (vt. UCD meetodid), saamaks aru, kuidas süsteem peaks toimima (Jeffries, 2001).

Klient mängib kasutuslugude paikapanemisel olulisimat rolli, kuna tema otsustada on loodava toote arengusuunad. Klient või tema esindaja on isik, kes lähtub tööprotsessis tellija ärihuvidest ning nende võimalikult edukale realiseerimisele arendatavas tootes.

Arendaja kanda on XP projektis mitu rolli - nii analüüsimine, disain, arendamine kui ka testimine. Ekstreemarenduses on kasutusel ka mitmeid arendusmeetodeid, mille abil muuta loodava toote koodi kvaliteetsemaks, näiteks paarisprogrammeerimine (lühikeste ajaliste intervallidega kordamööda arendamine).

Projektijuhi põhitööks on arendusmeeskonna- ning kliendivahelise suhtluse võimalikult sujuvaks muutmine. See tähendab, et tema ülesandeks on kogu tööprotsessi juhendada ning organiseerida, kuid mitte sinna isiklikult sekkuda.

XP tööprotsess näeb ette võimalikult tihedate ja lühikeste iteratsioonidega perioodiliste tooteversioonide väljatöötamist. See tekitab kliendile võimaluse anda jooksvalt tehtud tööle tagasisidet, millega arvestada järgmiste iteratsioonide etteplaneerimisel.

Pilt 3 XP erinevatest iteratsioonidest, nende seostest ning ajalistest kestustest.

2.2 Scrum

Scrumi algusaastad ulatuvad ekstreemarenduse omadest kaugemale, kuigi täpse loomisaja osas leidub eriarvamusi. On spetsialiste, kes omistavad Scrumi loomise au Jeff Sutherlandile, John Scumniotalesile nng Jeff McKennale, kuid samuti tuuakse välja Scrumi autoritena ka Hirotaka Takeuchit ning Ikujiro Nonakat, kes tulid kõnealuse arendusmetoodika plaaniga välja 1986. aastal. Hoolimata sellest, kes tuli esmakordselt välja Scrumi alusprintsiipidega, oli Jeff Sutherland see, kes pani need kirja ning esitas OOPSLA (*Object-Oriented Programming Systems, Languages & Applications*) konverentsil 1995. aastal. Kuna Scrum hakkas aja jooksul koguma üha enam populaarsust, otsustati luua ühtne platvorm, mis hakkaks haldama kõike Scrumiga seonduvat ning andma välja Scrumi ekspert-staatust tõestavaid sertifikaate/tiitleid - *Certified ScrumMaster (CSM)* (Krishnamurthy, 2012).

Scrumis eksisteerib kolm erinevat põhirolli, mis moodustavad scrum-tiimi. Nendeks on:

- Tooteomanik- ehk *product owner*, kes on olemuselt toote tellija (e. kliendi) esindaja. Tooteomaniku ülesandeks on kindlustada toote äriväärtuse püsimajäämine arendusprotsessi käigus. See tähendab, et tema tööks on nii erinevate kliendikesksete ülesannetega tegelemine (nt. kasutajalugude kirjutamine) kui ka tööülesannete prioritiseerimine.

- Arendusmeeskond - arendusmeeskonna ülesandeks on valmistada mingis ulatuses lõpuniviidud toode iga iteratsiooni lõpuks. Meeskond koosneb reeglina 3-10 inimesest, kattes võimalikult palju ekspertiisivaldkondasid (disain , arendus, testimine jne).
- *Scrum Master* - kuigi teatud mõistes on kõnealuse rolli näol tegu tiimijuhiga, erinevad tema ülesanded tavapärase projektijuhi omadest. Tema ülesandeks on garanteerida ühtlane töövool ning jälgida kõikide ülesannete täituvust ja vastavust Scrum-kriteeriumitele. *Scrum Master* erineb tavapärasest projektijuhist eelkõige seetõttu, et tema tööülesanded keskenduvad tootele ning tööprotsessile, kuid mitte töötajatele.

Scrum-i tööprotsess koosneb mitmetest iteratsioonidest, mida nimetatakse sprintideks. Viimaste pikkus pole üheselt määratud, vaid sõltub ülesannetest, mis selle sprindi jooksul soovitakse saavutada - tavaliselt jääb ühe iteratsiooni kestus ühe nädala ning ühe kuu vahele. Iga sprindi lõpuks peab olema valminud sajaprotsendiliselt integreeritud ning testitud toode, mida on võimalik juba kliendile esitleda. Sprintide pikkus ning eesmärgid määratakse regulaarsetel koosolekutel, mis toimuvad enne iga iteratsiooni algust. Lisaks nendele toimuvad ka igapäevased kohtumised (*Daily Scrum*), mille raames annavad kõik meeskonnaliikmed ülevaate, kui kaugele nad on oma osaga jõudnud ning mida plaanivad teha.

2.3 Kanban

Kanban on teatud ulatuses lihtsam kui Scrum, koosnedes põhiliselt tööprotsessi visualiseerimisest ning nõudlusmahu piiramisest. Kanbani loojaks võib pidada Jaapani ettevõtet Toyota, kes garanteerib seda kasutades oma varuosade sujuva liikumise erinevate tehaste vahel. Iga varuosa peal asub kaart, mis eemaldatakse varuosa ringlusesse mineku ajal. Kaart saadetakse tagasi varuosa tarnijale, kes kinnitab selle juba uuele tarnitavale varuosale. Kuna kaartide arv on piiratud ning neid taaskasutatakes, garanteeritakse ringluses olevate varuosade kontrolli all hoitud kogus ning uusi juppe tellitakse vaid vajaduse tekkimisel. Tarkvaraarenduses tähendab Kanbani kasutamine visuaalset tahvlit (*Kanban board*), mis on jagatud sektsioonideks (disain, arendus, käikulastud jne). Iga ülesanne asub eraldi kaardil, mis paigutatakse vastavasse sektsiooni. Sealjuures on piiratud kaartide arv, mis tohib ühes sektsioonis ühel ajahetkel asuda (Peterson, 2011).

3 HCI ning agiilsete metoodikate ühendamine

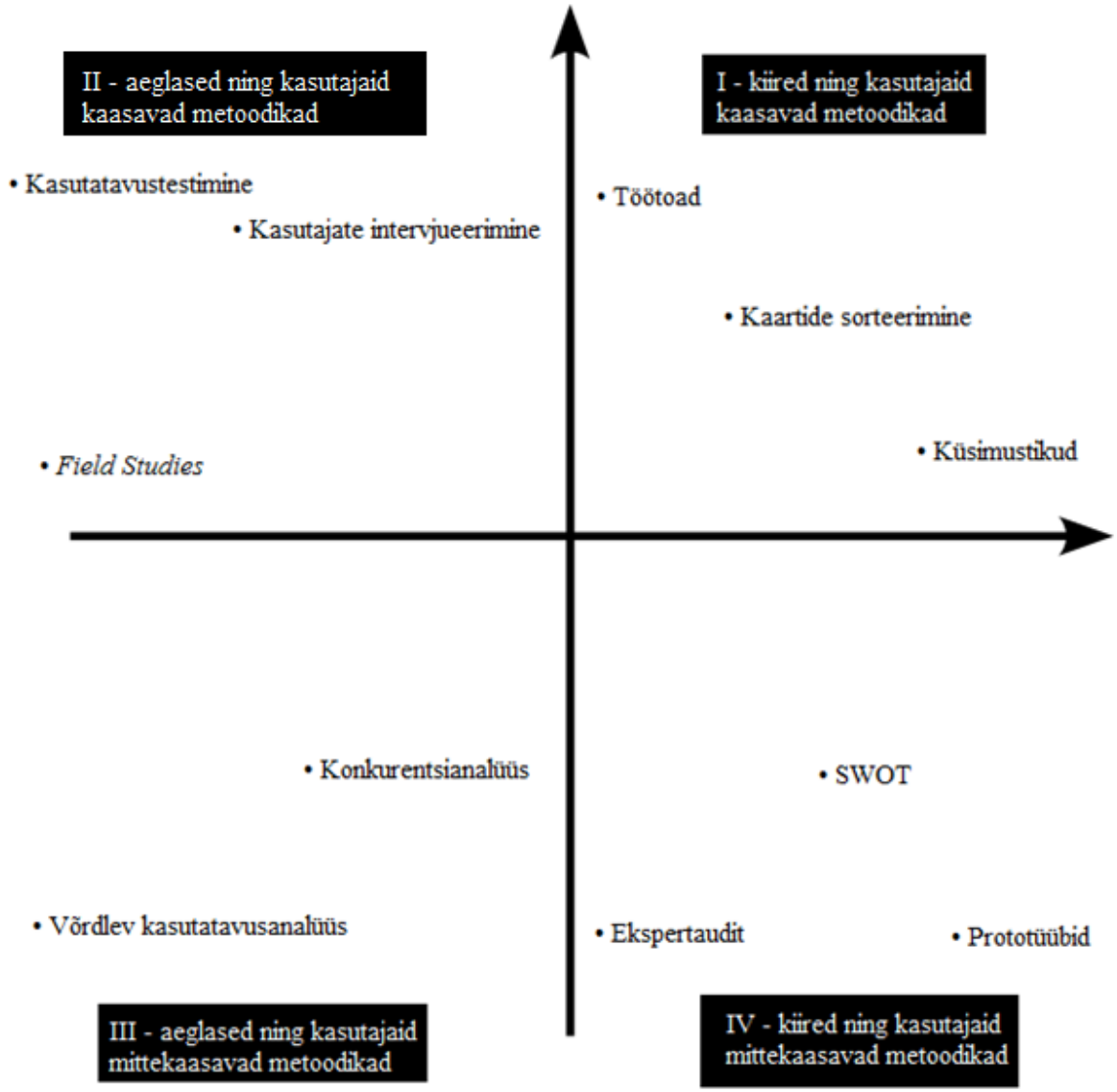
Nagu eelnevatest kirjeldustest võib lugeda, on nii agiilsetel arendusmetoodikatel kui ka HCI praktikatel väga oluline roll tarkvaraarenduses. Agiilsete arendusmetoodikate peamiseks ühiseks omaduseks võib pidada võimet paindlikult ning mõistliku ressursikuluga saada valmis toode, mis vastaks võimalikult täpselt kliendi ärinõudmistele, keskendudes põhiliselt kolmele printsiibile - kiirusele, adaptiivsusele ning inimkesksusele. Agiilsed metoodikad on seetõttu muutunud üha populaarsemaks, kuna nendepoolne tulemus on kliendi jaoks, kes hoitakse kogu arendustööga jooksvalt kursis, märksa rahuldavam. HCI kohaselt tuleb aga ärinõudmisi silmas pidades suunata meeskonna tähelepanu ka kasutatavusfaktorile - ebamugavalt kasutatava toote asemele hakkavad kasutajad esimesel võimalusel otsima alternatiive.

HCI metoodikates kasutatavad meetodid ja praktikad on vastupidiselt agiilsetele iteratsioonidele üsnagi aeganõudvad ning nende lõpp-eesmärk võib valdkonnavõõrale kliendile tunduda esimesel pilgul segane. Toote tellija jaoks asuvad prioriteedid muudes valdkondades - toote korrigeerimises vastavalt turuolukorrale ning *time-to-market* aja võimalikult palju kärpimises.

Lisaks kliendipoolsetele eelistustele mängivad HCI metoodikate madala populaarsuse taga rolli ka nendes esinevate iteratsioonide ajaline pikkus. Kui agiilne arendusmeeskond on võimeline iga tööpäeva lõpuks produtseerima mingis ulatuses valmis tarkvarajupi, siis sama ajaperioodiga jõuab UX-disainer määrata äärmisel juhul fookusgrupi, kelle peal tootedisaini looma ning katsetama hakata. Kõik eelmainitu on tekitamas olukorda, kus standartne agiilne arendusmeeskond koosneb ainiti arendajatest, kes suudavad oma kiirete arendustsüklitega hoida projekti tähtaja fikseeritud (lühikestes) raamides ning tagada sellega tellija rahulolu.

Positiivne kasutuskogemus on aga olulisel kohal toote edukusastme määramisel. See tähendab, et on tarvis leida meetodid, mida on kasutuskeskkonna kujundamisel agiilsetes protsessides mugavaim kasutada.

HCI alla kuulub rohkelt erinevaid metoodikaid, neist igalühel on iseloomulikud praktikad ja meetodid, mille abil määratakse kasutuskeskkond. Neid meetodeid saab aga kirjeldada põhiliselt kahe karakteristika abil - reaalsete kasutajate kaasamise tase ning ajaline kestus. Sellest lähtuvalt võime erinevad meetodid paigutada teljestikule, milles deklareerime neli erinevat piirkonda (Puks & Kirillov, 2013).



Pilt 5 – Teljestik vastavalt paigutatud meetoditega, mille X-tasand näitab meetodite kiirust ning Y-tasand kasutajate kaasamise taset

Teljestiku X-tasandi järgi saab järjestada erinevad meetodid kiiruse järgi, paigutades negatiivsele poolele arglasemad ning positiivsele kiirema lõpptulemuse produtseerivad meetodid. Y-telg näitaks seevastu kasutajate kaasamise taset, kus madalamal asuvad meetodid, kus kasutajat kaasatakse minimaalselt või üldse mitte ning kõrgemal meetodid, kus kaasatakse kasutajat enim. Antud teljestikule kaardistamine aitaks määrata, millised metoodid on olemuselt sobivamad agiilsetes protsessides kasutamiseks ning millised mitte.

3.1 Kiired ning kasutajaid kaasavad metoodikad

I grupp on olemuselt üks keerukamaid, kuna vastavalt eelnevalt püstitatud nõuetele peavad siia langema meetodid ning tavad, milles saavutatakse reaalsete kasutajate abil võimalikult kiireid tulemusi.

Reaalsete kasutajate otsimine ehk fookusgrupi määramine erinevate meetodite kasutamiseks on juba omaette väga aeganõudev protsess. Seetõttu võib ka üldistavalt öelda, et antud grupi tingimustele vastavaid kasutatavuspraktikaid pole võimalik leida. Situatsioon muutub pisut juhul kui fookusgrupp on eelnevalt määratud. Sellisel juhul saab pidada aja suhtes vähenõudlikeks üsna teatud tavasid.

Töötoad - eeldusel, et töötoas osalejad on eelnevalt määratud ning toimumisaja leidmisega ei teki probleeme, on töötubade näol tegu efektiivse viisiga koguda aktiivset tagasisidet erinevate disainilahenduste kohta enne reaalse arendustöö alustamist. Töötubade jooksul oleks võimalik luua koostöös osalejatega erinevaid prototüüplahendusi, milles on oma panuse andnud nii kasutajad, arendajad kui ka disainerid.

Töötubade läbiviimine eeldab, et korraldaja (e. arendusmeeskonna) koosseisu kuulub väga kompetentne moderaator, kes suudaks kogu protsessi juhendada.

Küsimustikud - küsimustike laialisaatmise, kokkukogumise ning analüüsimise peale kuluv aeg võib sageli varieeruda, sõltuvalt nii küsimustiku põhjalikkusest kui ka täitjate pealehakkamisest. Sellegipoolest on tegu hea meetodiga koguda statistilist tagasisidet mingil teemal, rakendades küsimustike näol nt. A/B testimist. See tähendab kasutajatele kahe erineva disainilahenduse näitamist ning palvet valida endale meeldivam, eelistatavalt selgituste ja põhjendustega.

Kaartide sorteerimine - töötubades on võimalik läbi viia ka kaartide sorteerimist. Tegu on meetodiga, mille abil on võimalik saada rohkelt tagasisidet disainilahendustele. Osalejatele antakse kätte hulk kaarte, millele võib olla kirjutatud:

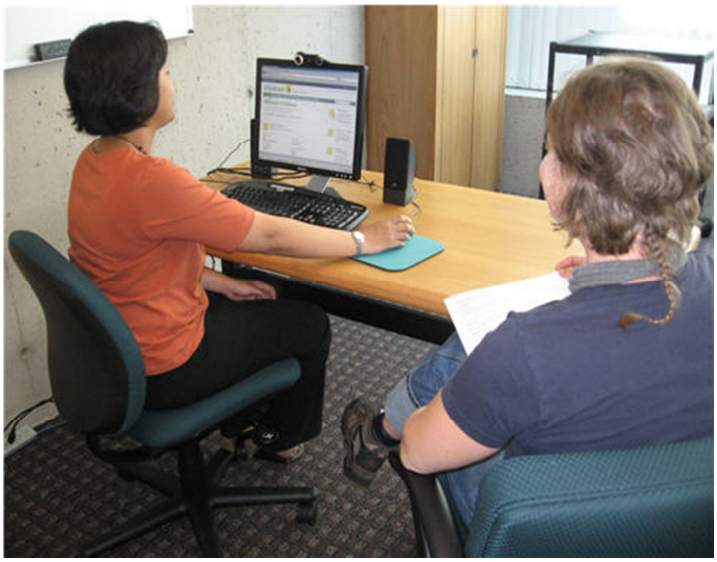
- loodaval veebilehel asuvad kategooriad;
- rakendust kirjeldavad väited;
- muu loodava tootega seonduv

Osalejate ülesandeks on antud kaardid panna järjekorda, kus kõrgematel positsioonidel asuvad vastavalt olulisemad kategooriad või tõesemad väited. Kaartide sorteerimisest kogutud tagasisidele tuginedes on lihtsam erinevaid disainiülesandeid prioritiseerida.

3.2 Aeglased ning kasutajaid kaasavad metoodikad

Teise gruppi, kuhu on koondatud aeganõudvad ning reaalseid kliente rakendavad meetodid, on võimalik paigutada väga mitmeid HCI praktikaid. Nagu varasemalt selgitatud, on mitmed HCI tavad üsna ressursinõudlikud, põhjustades sellega vastuolu agiilsete arendusmetoodikate kiire olemusega. Antud gruppi kuuluvad tavad võib pigem liigitada agiilsete arendusprotsesside jaoks ebasobivaks.

Kasutatavustestimine - olemuselt üks tulemuslikumaid (kuid ka ressursinõudlikumaid) kasutatavustavasid, kuna kaasab protsessi loodava toote potentsiaalsed lõpp-kasutajad (Nielsen Norman Group, 2013). Kasutatavustestimine eeldab mingil tasemel tooteprototüübi olemasolu - olgu tegu kas paberprototüübi, klikitava rakenduse (millel puudub funktsionaalsus) või esialgse beeta-variandiga loodavast tootest. Oluline on see, et testkasutaja saaks ettekujutuse loodavast tootest ning oskaks sellest lähtuvalt anda ka tagasisidet. Testimine ise koosneb eraldiseisvatest etappidest. Esmalt tuleb paika panna testkasutaja profiil, lähtuvalt toote sihtrgrupist (mees-/naissoost, vanusegrupp, etnograafiline kuuluvus jne), seejärel tuleb kirja panna ka testimisstsenaariumid - ülesanded, mida kasutajad loodud prototüübi peal lahendama hakkavad. Testimisessiooni ajal annab testi läbiviija (e. moderaator) kasutajale ülesande kätte ning kirjutab üles kõik tähelepanekud. Antud kujul testimine annab disaineritele reaalse ettekujutuse sellest, kuidas nende senine lahendus sobib kasutajatele ning milliseid muutuseid tuleks sisse viia.



Pilt 5 Tavaline kasutatavustestimise sessioon

Kasutajate intervjueerimine - kasutajate reaaleluline küsitlemine on pisut aeganõudvam protsess kui ühtse töögrupi korraldamine, kuna nõuab iga kasutaja puhul individuaalset lähenemist. Erinevalt kasutatavustestimisest ei ole siinkohal aga nõutud reaalse toote või prototüübi olemasolu, kuna intervjueerida võib ka eeltööna arendustööle. Intervjuudes on võimalik koguda kasutajate unikaalseid arvamusi ning neid lahata põhjalikult, vältides korraldaja ning osaleja vahelisi arusaamatusi. Oluline on intervjueerija pädevus, kuna kogutavad andmed on mittestatistilised.

***Field studies* -** ehk kasutajate väliuuringud kujutavad endast väga tulemuslikku, kuid ka väga kulukat uurimismeetodit. Tegu on tehnikaga, milles disainimeeskond jälgib kasutajat tema harjumuspärases keskkonnas, saamaks tagasisidet tema käitumismaneerida ning eelistuste kohta. Näiteks klientide jälgimine ostukeskuses annab e-kommerts lahenduste loojatele üsna väärtuslikku tagasisidet selle kohta, kuidas kasutajad oma ostunimekirju koostavad ning millest sõltuvalt nad erinevaid impulssoste sooritavad. Väliuuring annab projektimeeskonnale infot nii terminoloogia osas (mis väljendeid inimesed kasutavad, kuidas suhtlevad), asutustevahelist erinevuste kohta jms.Kuigi väliuuringud võivad olla äärmiselt produktiivsed, nõuavad ka väga palju ajalist panustamist ning pühendumist. Küll aga saab ühe uuringu tulemusi kasutada tõenäoliselt mitmetes järgnevates projektides.

3.3 Aeglased ning kasutajaid mittekaasavad metoodiakd

Kolmandasse gruppi langevad aeganõudvad HCI protsessid, mis ei nõua reaalsete kasutajate kaasamist. Sarnaselt teise grupiga on ka siinkohal tegu pigem agiilsetesse projektidesse mittesobivate meetoditega, kuna lisaks nende ajanõudlusele ei ole võimalik kogutud andmeid toetada reaalsete kasutajate abil.

Võrlev kasutatavusanalüüs - ehk CUE (*Comparative Usability Evaluation*) tähendab hulka erinevaid professionaalseid kasutatavusmeeskondasid, kes analüüsivad loodavat toodet. Loodud analüüsitulemused kogutakse hiljem kokku ning püütakse leida vastus hulgale küsimustele:

- Kas kasutatavusanalüüsid suutsid tuvastada üheseid puudujääke? Kas nende poolt kasutatud meetodid kattusid?
- Milline on lähtuvalt analüüside kattuvusest kriitiliseim probleem kehtivas lahenduses/lahendustes?
- Mitu kasutatavusviga suudeti keskmiselt tuvastada.

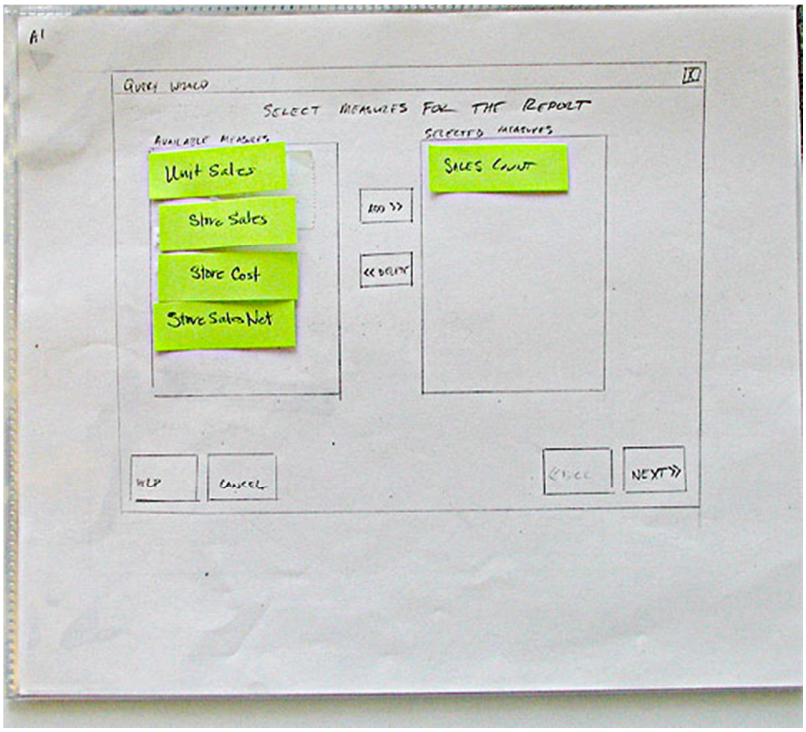
Kõnealust meetodit on kasutatud üsna mitmete tuntud portaalide ja toodete peal, tuntumateks näideteks on e-maili teenus Hotmail (testis 9 erinevat meeskonda) (http://www.dialogdesign.dk/Summary1.htm) ning mööblikaupluste jaeketi Ikea poolt loodud interaktiivne tööriist *IKEA Pax Wardrobe Planning Tool* (testis kolmteist erinevat meeskonda) (Molich, Pero, Schroeder, & Modgil, 2006).

Konkurentsianalüüs - enne prototüüpide kallal töötama hakkamist võib olla kasulik tutvuda turul asuvate konkurentide toodete tugevuste ning nõrkustega. Uurimistöö peab selgitama, mis on iga konkurendi eelised ning puudujäägid turul, sellele põhjenedes leida alternatiivlahendusi. Kuna tegu on üsna põhjalikku uurimist nõudva meetodiga, võib selle läbiviimine olla üsna ajamahukas, alustades planeerimisest ning lõpetades tulemuste kokkukogumise ning analüüsimise ja järelduste tegemisega.

3.4 Kiired ning kasutajaid mittekaasavad metoodikadd

Neljandasse gruppi kuuluvad meetodid, mis on kiirelt teostatavad ning ei nõua reaalsete kasutajate kaasamist. Kuigi puudub võimalus katsetada nende meetodite tulemusi reaalsete kasutajate peal, on nende abil võimalik anda ka projektimeeskonnale selgem ülevaade loodavast tootest, selle sihtgrupist ning nende teel seisvatest probleemidest.

Prototüübid - tooteprototüübid aitavad hinnata erinevaid disainialternatiive tootearenduse varajastes astmetes. Sealjuures on võimalik loodud algnäidiseid kasutada ka reaalsete kasutajate peal ning seeläbi hinnata erinevate lahenduste kasutamise sobivust. Kui tootarenduse algfaasidfes on otstarbekam luua lihtsamaid paber- või HTML-prototüüpe, siis peale esialgsete testide läbimist võib juba luua nn *"High-fidelity"* prototüübi, mis kujutaks endast eelvaadet lõplikule rakendusele. Prototüüpe on võimalik luua üpris erinevalt - alates lihtsast paberprototüübist ning lõpetades pisut komplitseerituma klikitava HTML-prototüübiga. Paberprototüüp kujutab endast suuremat paberpinda, millele on joonistatud markeriga rakenduse üldraamistik. Märkmepaberitega asetatakse peale menüüsid, dialoogiaknaid ning muid elemente, läbipaistva paberiga sisendvälju jne. Paberprototüüpidel on mitmeid eeliseid, millest märkimisväärseim on vähene ressursinõudlikkus ning kiirus. Samuti on neid võimalik luua suurematel gruppidel ühistööna, kaasates vajaduse ja soovi korral ka kasutajaid. Teiselt küljelt ei ole võimalik paberprototüübi abil anda kasutajale täit kasutuskogemust, mistõttu võib tagasiside mingil määral kannatada. Samuti on prototüübi testimise ajal nõutud moderaatori olemasolu, kes vahetab "lehti" vastavalt kasutaja käitumisele.



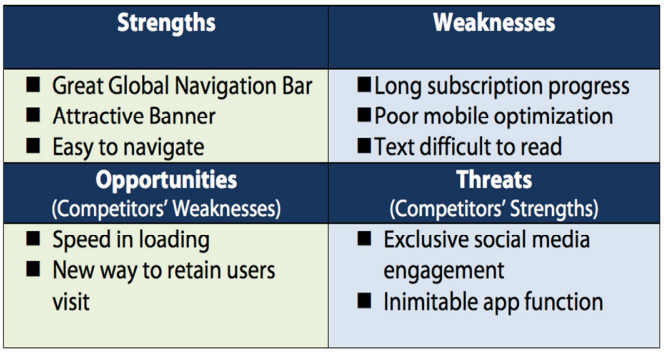
Pilt 6 Näide lihtsast paberprototüübist

HTML-prototüübid jagunevad *low-fidelity* ning *high-fidelity* prototüüpideks. Neist esimesi luuakse reeglina minimaalse vaevaga, tutvustamaks kasutajale elementaarseid interaktsioone rakenduses. Kasutaja saab rääkida rohkem kaasa toote lõpp-disaini valmimises, näidates, kus ta eelistaks teatud elemente lehel näha või mida muuta. *High fidelity* prototüübid on aga pigem toote lõplike astmete disainilahenduste testimiseks otstarbekad. Kasutajad saavad imiteerida toote kasutamist, anda sellele tagasisidet, täita hiljem rahuloluküsitlust jne. HTML-prototüüpide eelised ning puudujäägid on peamiselt vastupidised paberprototüüpide omadele. Kuigi neid on pisut kulukam ning aeganõudvam luua, on nendega saadud kasutuskogemus pisut ehedam ning tulemuslikum.

Ekspertaudit - professionaalne kasutatavusanalüüs on hea võimalus koguda üsna kiiret tagasisidet loodud toote või tooteprototüübi kasutatavuse osas, kasutades ära kasutatavuseksperdi heuristilisi teadmisi ning kogemusi. Ekspert suudab tuvastada toote peamised kasutatavusprobleemid ka ilma reaalseid kasutajaid kaasamata, anda soovitusi vajalike muudatuste osas ning vajadusel luua ka kõrgetasemelise tegevusplaani toote ümbertöötlemiseks. Erinevalt võrdlevast kasutatavusanalüüsist on ekspertauditi näol tegu kiireloomulise protsessiga, kuna tegutseb enamasti üks individuaalne ekspert (äärmisel juhul ekspertmeeskond), kelle töötulemuste kallal alustatakse koheselt tööd.

SWOT - neljatähelise akronüümiga (*Strengths, Weaknesses, Opportunites, Threats*) tähistatakse struktureeritud planeerimismeetodit, milleabil analüüsida erinevaid kasutatavuslahenduse tugevusi ja nõrkusi, võimalusi ning ohtusid. Kuigi eelmainitud meetodit kasutatakse valdavalt ärivaldkonnas, võib sellele ka kasutatavuse loomisel leida rakendust.

SWOT analüüsi tegemisel luuakse neljast lahtrist koosnev maatriks, kus on välja toodud kõik eelmainitud neli karakteristikat. Kirjutatakse üles kõik seniloodud toote tugevad ja nõrgad küljed, kuid samuti ka võimalused ning ohud tuleviku tarbeks. Antud meetodi kasutamine annab meeskonnale hea ülevaate toote hetkeseisust ning erinevatest punktidest, millele tulevikus kõrgendatud tähelepanu osutada.



Pilt 6 Näide SWOT tabeli kasutamisest UX-valdkonnas

Eelmainitud teljestiku sektoritesse langenud valikutega tutvudes on võimalik püstitada hüpotees, et agiilsetesse arendusmetoodikatesse sobivad kõige etemini I ning IV grupi meetodid, mis on kiirelt mõõdetava tulemusega, olenemata reaalsete kasutajate kaasamisest. II ning III grupi meetodid on ajakulu osas üsna nõudlikud ning võivad mõjuda agiilsete arendusmetoodikate kiiretele iteratsioonidele pigem aeglustavalt. Antud hüpoteesi kohaselt on võimalik ka agiilsetes projektides rakendada erinevaid HCI meetodeid ilma kehtiva metoodika reegleid rikkumata.
Eelnevalt toodud metoodikate ajaline kestus ei ole aga üheselt määratud ning lohaka korraldustöö juhul võib ka elementaarseimast töötoast kujuneda pikk mõttevahetus, mille tulemus on kaugeltki liiga inforohke, et seda praktiliselt rakendada. Küll võib agiilsetele arendusmeeskondadele anda erinevaied nõuandeid ning soovitusi, rakendamaks eelnevalt välja toodud meetodeid:

- Garanteerida meeskonnas vähemalt ühe professionaalse HCI-asjatundja olemasolu, paremal juhul koolitada kogu meeskond antud valdkonnas välja. Mitmed HCI meetodid nõuavad moderaatori olemasolu, kes organiseerib uurimusi, aitab neid läbi viia ning hiljem tulemusi analüüsida. Sellise inimese kompetentsus garanteerib, et saadavad uuringutulemused on võimalikult kvaliteetsed ning uuringute pikkus ei kujune liiga pikaks.
- Planeerida kasutajateste ning muid kasutajaid kaasavaid uuringuid võimalikult pikalt enne iteratsioone. See tähendab eelkõige püsivat kasutajate otsimist ning nende peal läbiviidavate testide etteplaneerimist. Testijate leidmisel tuleb eelkõige kasuks nii sotsiaalmeedia kui ka olemasolev klientide andmebaas, kuid äärmisel vajadusel sobivad

selleks ka lähedalasuvad kontoritöötajad, kes pole konkreetse projektiga seotud ning ei ole erialalt arendajad vm.

- Plaanides kasutatavusteste tuleks vähendada testkasutajate arvu võimaliku miinimumini. Kui tavapäraselt eeldatakse vähemalt 5-10 testkasutaja olemasolu, et oleks võimalik luua statistilisi andmeid ning teha kindlaid järeldusi. Agiilsete projektide puhul tasub aga piirduda ligi kolme kasutajaga. Võimalusel saab ka peale iga paari testkasutajaga töötamist viia sisse tema poolt soovitatud muudatused ning neid omakorda katsetada järgmise kasutaja peal. Sellisel viisil pole võimalik avastada kõiki toote kitsaskohti, kuid saab jälile kriitilistele kasutatavusvigadele.
- Lasta kasutatavuseksperdil sooritada perioodilisi heuristilisi analüüse tootest. Need analüüsid ei pea olema väga põhjalikud, piisab ka kriitiliste vigade väljatoomisest. Sedasi takistatakse varases faasis tehtud vigade ülekandumine lõpp-tootesse. Siinkohal oleks väga sobilik leida ekspert, kes elab ning töötab teises ajavööndis. Sellisel juhul on arendusmeeskonnal võimalik talle saata toode ülevaatamiseks oma tööpäeva lõpuks, saades juba järgmiseks tööpäevaks raporti, kus on kirjas edaspidised soovitused.

Kokkuvõte

Tänapäeval on arvutitest saanud igapäevane tarbeese, võrguühendust hakatakse sealjuures pidama inimõiguseks. See on viinud olukorrani, kus iga tarkvaratootja püüab luua toodet, mida oleks läbi hea kasutatavusteguri viia võimalikult suure inimgrupini. Selle saavutamiseks kasutatakse aga erinevaid metoodikaid ning meetodeid, mis kipuvad aga olema üsna ressursinõudlikud.

Teisest küljest on olulisel kohal arendusprojektides ka muud tegurid - kliendi rahulolu, projekti kvaliteetne lõpptulemus ning kiire tulem. Selle saavutamiseks on üha enam populaarsust kogumas agiilsed e. väledad arendusmetoodiakad. Hoolimata mitmete erinevate metoodikate olemasolust, jagavad nad kõik ühtset eesmärki - tagada võimalikult kiire realiseerimisajaga turuolukorrale vastav lõpp-toode.

HCI- ning agiilsed arendusmetoodikad on oma olemuselt aga vastuolulised. Kuigi mõlema alla kuuluvad erinevad iteratiivsed meetodid, erineb nende iteratsioonide pikkus märgatavalt. Agiilsete arendusmetoodikate põhirõhk on kiirusel, HCI metoodikates on sellest olulisem aga kvaliteetse kasutuskeskkonna loomine.

Kuna mõlemad on eduka lõpp-toote valmimiseks vajalikud, on tarvis leida nende ühiseks koostööks sobivaim viis. Selleks on mitmeid lihtsamaid meetodeid, alustades oma arendusmeeskonna erinvate HCI-heuristikatega kurssi viimisest ning lõpetades erinevast ajatsoonist pärit spetsialisti palkamisega kiirete auditite saamiseks.

Agiilsetest projektides ei tohiks erinevaid HCI meetodeid sajaprotsendiliselt välistada. Pigem tuleks valida meetodid, mis tagavad tulemuse võimalikult kiiresti. Paigutades hulga HCI meetodeid nelja alamgruppi oma kiiruse ning kasutajate kaasamise taseme järgi, võib püstitada hüpoteesi, et just kiireid lõpp-tulemusi produtseerivaid meetodeid eelistatakse rohkem kasutada agiilsetes meeskondades.

Antud semimnaritöö on plaanis edasi arendada bakalaureusetööks, mille raames analüüsitakse Eesti iduettevõtteid ning nende kogemusi HCI-valdkonnas, kontrollides ka eelnevalt püstitatud hüpoteesi paikapidavust.

Kasutatud allikad

1. Nielsen Norman Group. (2013). *Usability Testing*. Kasutamise kuupäev: November 2013. a., Allikas Evidence-Based User Experience Research, Training, and Consulting: http://www.nngroup.com/courses/usability-testing/
2. *JISC infoNet*. (21. September 2012. a.). Allikas: Planning a participatory workshop: http://www.jiscinfonet.ac.uk/infokits/participatory/
3. Abrahamsson, P., Salo, O., Ronkainen, J., & Warsta, J. (2002). *Agile software development methods.* Espoo: VTT Publications .
4. Beale, R. (2007). *HCI II*. Allikas: Introduction to HCI: http://www.cs.bham.ac.uk/~rxb/Teaching/HCI%20II/intro.html
5. Beck, K., Beedle, M., Bennekum, A. v., Cockburn, A., Cunningham, W., Fowler, M., et al. (2001). *Agiilse tarkvaraarenduse manifest*. Allikas: Agiilse tarkvaraarenduse manifest: http://agilemanifesto.org/iso/et/
6. Bevan, N. (May 2006. a.). International Standards for HCI. London, United Kingdom.
7. Carroll, J. M. (kuupäev puudub). *Interaction Design Foundation*. Allikas: Human Computer Interaction (HCI): http://www.interaction-design.org/encyclopedia/human_computer_interaction_hci.html
8. Gilbert Cockton. (April 2008. a.). Revisiting Usability's Three Key Principles. Sunderland, United Kingdom.
9. Gube, J. (5. October 2010. a.). *SMASHING MAGAZINE*. Allikas: What Is User Experience Design? Overview, Tools And Resources: http://uxdesign.smashingmagazine.com/2010/10/05/what-is-user-experience-design-overview-tools-and-resources/
10. Jeffries, R. (8. November 2001. a.). *Xprogramming.com*. Kasutamise kuupäev: December 2013. a., allikas What is Extreme Programming?: http://xprogramming.com/book/whatisxp/
11. Krishnamurthy, V. (15. October 2012. a.). *A Brief History of Scrum*. Kasutamise kuupäev: November 2013. a., allikas TechWell.

12. Molich, R., Pero, K., Schroeder, W., & Modgil, N. (2006). Tips and Tricks for Better Usability. *DialogDesign*.
13. Nielsen, J. (29. May 2007. a.). *Nielsen Norman Group*. Allikas: The Myth of the Genius Designer: http://www.nngroup.com/articles/the-myth-of-the-genius-designer/
14. Norman, D. A. (kuupäev puudub). *Don Norman: Designing For People*. Allikas: Human-Centered Design Considered Harmful: http://www.jnd.org/dn.mss/human-centered_desig.html
15. Peterson, D. (2011). *What is Kanban?* Kasutamise kuupäev: November 2013. a., allikas Kanban Blog: http://www.kanbanblog.com/explained/
16. Puks, R., & Kirillov, Z. (August 2013. a.). Towards the Better Adoption of HCI Methodologies by Technology Startups. Roger Puks, Zahhar Kirillov, Harjumaa, Eesti.
17. Santafe, I. (1. Juny 2013. a.). *Webcredible customer experience design*. Allikas: Diary study guide: how to get the best results from diary study research: http://www.webcredible.co.uk/user-friendly-resources/web-usability/diary-study-guide.shtml
18. Smith, S. (12. Januar 2010. a.). *DotNetSlackers*. Kasutamise kuupäev: December 2013. a., allikas Is Extreme Programming Dying? Is Agile Growing in Popularity?: http://dotnetslackers.com/Agile/re-290516_Is_Extreme_Programming_Dying_Is_Agile_Growing_in_Popularity.aspx
19. SO, H. C. (2008). Human Computer Interaction: An Overview.
20. User Experience Professionals Association. (kuupäev puudub). *User Experience Professionals Association*. Allikas: UXPA: Usability resource: What is User Centered Design: http://www.usabilityprofessionals.org/usability_resources/about_usability/what_is_ucd.html
21. Variano, M. (5. October 2012. a.). *ICT4D @ Tulane Student perspectives on ICT4D from the Payson Center's IDEV4100 Classes at Tulane University in New Orleans.* Allikas: Human Centered Design vs. Activity Centered Design: http://tulaneict4d.wordpress.com/2012/10/05/human-centered-design-vs-activity-centered-design/

22. *VERSIONONE Agile Made Easier*. (kuupäev puudub). Allikas: Agile Methodologies for Software Development: http://www.versionone.com/Agile101/Agile-Development-Methodologies-Scrum-Kanban-Lean-XP/
23. Walters, H. (8. March 2008. a.). *Businessweek*. Allikas: Apple's design process: http://www.businessweek.com/the_thread/techbeat/archives/2008/03/apples_design_p.html