jamk

Projektinhallintasovellusten vertailu pelinkehityksen tuotantoprosessissa

Tomi Vainionpää

Opinnäytetyö, AMK
Marraskuu, 2021
Luonnontieteiden ala
Tradenomi (AMK), Tietojenkäsittelyn tutkinto-ohjelma

jamk | Jyväskylän ammattikorkeakoulu
University of Applied Sciences

jamk

Kuvailulehti

Vainionpää, Tomi

Projektinhallintasovellusten vertailu pelinkehityksen tuotantoprosessissa

Jyväskylä: Jyväskylän ammattikorkeakoulu. Marraskuu 2021, 40 sivua

Luonnontieteiden ala. Tietojenkäsittelyn tutkinto-ohjelma. Opinnäytetyö AMK

Julkaisun kieli: suomi

Verkkojulkaisulupa myönnetty: kyllä

Tiivistelmä

Tutkimuksen tavoitteena oli selvittää toimeksiantajan tuotantoprosessia sekä vertailla kahta projektinhallinnan sovellusta, Jiraa ja Notionia pelinkehityksen näkökulmasta. Tehtävänä oli löytää vastaukset asetettuihin tutkimuskysymyksiin, jotka olivat: mihin osiin ominaisuus tulee pilkkoa, jotta tuotantoprosessista saadaan toimiva, millaisia mahdollisuuksia projektin pilkkomiseen sovelluksista löytyy ja mitä eroja on pienen ja suuren organisaation projektinhallinnan tarpeissa?

Tutkimusotteena työssä hyödynnettiin kvalitatiivista tutkimusta ja vertailevaa kehittämistutkimusta, jossa aineistonkeruumenetelmänä käytettiin teemahaastattelua ja projektinhallintasovelluksien testaamista. Teemahaastattelussa haastateltiin toimeksiantajayrityksen avainedustajaa, jonka avulla saadaan tarkkaa taustatietoa toimeksiantajan prosesseista.

Tutkimuksen tulokset osoittavat, että Jirasta löytyy monipuolisesti ominaisuuksia kaiken kokoisten organisaatioiden tarpeisiin. Jirasta löytyy muun muassa roadmap -ominaisuus, jonka avulla voidaan seurata useiden eri tiimien toimintaa samanaikaisesti. Notionia voi puolestaan helposti kustomoida ja soveltuu pienempien tiimien tarpeisiin. Teemahaastattelussa selvisi, että toimeksiantaja hyödyntää vain osaa ominaisuuksien pilkkomismahdollisuuksista ja sen käyttämästä Notion sovelluksesta puuttuu tarvittavia projektinhallinnallisia seurantaominaisuuksia.

Johtopäätöksenä todetaan, että suuren yrityksen tarpeisiin Jira on ominaisuuksiltaan kattavampi projektinhallintasovellus pelinkehityksen tarpeisiin. Notion toimii pienemmissä projekteissa ja organisaatioissa hyvin, mutta siitä puuttuu Jirasta löytyviä projektinhallinnallisia ominaisuuksia, jotka ovat suoraan tarkoitettu ketterän kehityksen toimintamalleihin. Toimeksiantajan projektinhallinnan prosessisien tarkkuudessa on kehittämisen varaa. Toimeksiantajayrityksessä luotetaan liikaa suulliseen tiedonkulkuun ja dokumentointi jää toissijaiseksi.

Avainsanat (asiasanat)

Pelinkehitys, Projektinhallinta, Ketteräkehitys, Scrum, Kanban, Jira, Notion, User story

Muut tiedot (salassa pidettävät liitteet)

-

jamk

Description

Vainionpää Tomi

Comparing project management softwares in the game development production process

Jyväskylä: JAMK University of Applied Sciences, November 2021, 40 pages

Natural Sciences. Business Information Technology. Bachelor's thesis.

Permission for web publication: Yes

Language of publication: Finnish

Abstract

The aim of this study was to find out assignee company's production process and to compare two project management tools, Jira and Notion from the game development perspective. The aim was to find answers to predetermined research questions which were the following ones: in what parts should features be broken down to make the production process more effective and functional, what are the possibilities in the project management tools to split the project down and what are the main differences between large and small organization's needs for project management?

This study was conducted using qualitative research and comparative development research. In this study the research material was gathered by thematic interview and testing the two project management tools. In the thematic interview a key representative of the assignee company was interviewed, which provided specific background information about the assignee company's development processes.

The results of the study show that Jira has a wider range of features for the needs of organizations regardless of the size. Amongst other things Jira has a roadmap feature that allows you to track the activities of several different teams simultaneously. On the other hand, Notion can be easily customized to fit smaller team's needs. In the thematic interview it became clear that the assignee company utilizes only a part of the possibilities for splitting workload. The assignee company uses Notion as their project management tool, which lacks some necessary process tracking features.

In conclusion, for the needs of larger organization, Jira is more comprehensive project management tool for game development. Notion works well in smaller projects and organizations, but it lacks a lot of the project management tools that are included in Jira and are directly intended for agile development frameworks. There is room for improvement of the accuracy of the project management processes in assignee company. The assignee company relies too much on face-to-face communication and documentation remains secondary.

Keywords/tags (subjects)

Game development, Project management, Agile, Scrum, Kanban, Jira, Notion, User story

Miscellaneous (Confidential information)

-

Sisältö

Kuviot

Taulukot

1 Pelialan projektinhallinta

Projektinhallinta on tärkeä osa kaikkien projektien onnistumista. Projektinhallintaa hyödyntää niin pienet kuin suuretkin tiimit. Sovelluksia projektinhallinnan avuksi on monia, ja voikin olla haastavaa valita niistä omalle projektilleen sopiva. Peliala on nopeasti kehittyvä ja sen prosessit muuttuvat jatkuvasti. Haasteena onkin pysyä uusimmissa muutoksissa ajan tasalla ja ketterä kehitys auttaa organisaatioita pysymään muutoksissa mukana.

Opinnäytetyö laadittiin aitoon työelämätarpeeseen ja sen toimeksiantajana toimi mobiilipeliyhtiö Skunkworks Oy. Työn tavoitteena oli selvittää toimeksiantajan tuotantoprosessia sekä vertailla kahta eri projektinhallinnan sovellusta, Jiraa ja Notionia. Tämän lisäksi selvitettiin, kuinka projektinhallinnalliset tarpeet eroavat pienen ja suuren organisaation välillä.

Aiheesta löytyy aikaisempia tutkimuksia lähinnä sovelluskehityksen näkökulmasta. Tässä työssä pureudutaankin projektinhallintaan pelikehityksen suunnasta. Tutkimus toteutetaan kvalitatiivisin menetelmin hyödyntäen kehittämistutkimusta. Aineistoa tutkimukseen kerättiin vertailemalla sovelluksia sekä hyödyntämällä teemahaastattelua.

2 Tutkimusasetelma

Toimeksiannon tavoitteena oli selvittää ja määritellä tarkka tuotantoprosessi projektinhallinnallisesta näkökulmasta. Tämän lisäksi selvitettävänä oli, kuinka parantaa ja kehittää työn kulkua. Osaksi työssä sivutaan myös pienen ja suuren organisaation eroja projektinhallinnassa ja sitä, kuinka tämä vaikuttaa tarvittaviin vaatimuksiin. Tutkimuksessa tarkkailun alla oli kaksi sovellusta: Notion ja Jira software. Notion on yrityksellä tällä hetkellä käytössä ja Jira softwarea on käytetty aiemmin. Tutkimus toteutettiin hyödyntäen kvalitatiivisia tiedonkeruumenetelmiä, joilla vertaillaan edellä mainittujen ohjelmien eroja toteuttaa ominaisuuksien pilkkomista sekä projektin kokonaisvaltaista seuraamista. Vertailun kohteena on myös organisaation kokoon liittyvät erot ja kuinka nämä ohjelmat soveltuvat eri kokoisille organisaatioille.

Jotta voidaan tutkia eroavaisuuksia ohjelmien välillä sekä millaisia ominaisuuksia ja käytäntöjä tällä hetkellä työnteosta puuttuu, hyödynnettiin työssä kehittämistutkimusta. Kehittämistutkimuksessa tutkija osallistuu aktiivisesti toimeksiantajan toimintaan ja pyrkii yhdessä työyhteisön kanssa rat-

kaisemaan määritellyt ongelmat (Bister, 2019, 42). Tämän opinnäytetyön tapauksessa kehittämistutkimuksen avulla saadaan tietoa siitä, mitä tulisi kehittää eteenpäin, jotta projektien kasvaessa projektinhallinta toimisi paremmin. Vertailemalla sovelluksia, saadaan myös selvitettyä tämänhetkisen käytössä olevan sovelluksen puutteita vertaamalla sitä kattavampaan ohjelmaan, joka on tarkoitettu suoraan ohjelmistotuotannon projektinhallintaan ja laajempien organisaatioiden käyttöön.

Tietoperustana toimii ohjelmien dokumentoinnit, pelialaan liittyvät ketterän kehityksen kirjat sekä muut verkkolähteet. Ketterään kehitykseen ja scrumiin liittyvät kirjat ja muut lähteet hyödynnettiin myös ohjelmistokehityksen puolelta.

Tutkimuskysymykset

- Mihin osiin ominaisuus tulee pilkkoa, jotta tuotantoprosessista saadaan toimiva?
- Millaisia mahdollisuuksia projektin pilkkomiseen sovelluksista löytyy?
- Mitä eroja on pienen ja suuren organisaation projektin hallinnan tarpeissa?

2.1 Aineistonkeruumenetelmät

Aineistonkeruu suoritettiin lukemalla Jiran ja Notionin dokumentointia ja testaamalla sovelluksia. Tämän lisäksi suoritettiin teemahaastattelu, jossa haastateltiin toimeksiantajayrityksen avainedustajaa. Sen avulla saatiin taustatietoa toimeksiantajayrityksen projektinhallinnallisista tarpeista ja prosesseista.

Tutkimuksen tavoitteena on muodostaa käsitys sovelluksien toiminnasta, ja tämän vuoksi dokumenttien analysointi sekä kokeilun ja testaamisen kautta saatu tieto on työn kannalta olennaista. Testaaminen toteutetaan omatoimisena kokeiluna. Omatoimisessa kokeilussa astutaan järjestelmän käyttäjän rooliin ja näin kerätään tietoa aiheesta (Bister 2019, 34).

Teemahaastattelu sopii työn tiedonkeruumenetelmäksi erittäin hyvin, sillä teemahaastattelun tavoitteena on keskustelunomaisesti saada selville haastateltavan kokemuksia aiheesta. Teemahaastattelussa haastattelun aiheet ovat etukäteen tarkasti määriteltyjä, mutta aiheiden käsittelyjärjestys tai tarkka esitysmuoto voi määräytyä haastatteluvaiheessa (Bister 2019,36).

3 Ketterä kehitys

Agile -terminä kuvaa tietynlaista lähestymistapaa ohjelmistokehitykseen. Agile lähestymistapana korostaa asteittaista toimitusta, tiimin yhteistyötä, jatkuvaa suunnittelua ja oppimista sen sijaan, että kaikki toimitettaisiin heti kerralla alusta loppuun samalla kaavalla. (What is Agile Software Development? N.d.)

Ketterä projektinhallinta on iteraation kautta lähestyvä hallintomalli sovelluskehityksen projekteihin. Tämän keskipisteenä on jatkuvat toimitukset ja palaute asiakkaalta jokaisessa iteraatiossa. Tiimit, jotka käyttävät ketterän projektinhallinnan metodeja parantavat tuotantonopeuttaan, yhteistyökykyä ja onnistuvat vastaamaan paremmin markkinoilla oleviin trendeihin. (Drumond, C. N.d.)

3.1 Periaatteet ja manifesti

Ketterän ajattelutavan lähtökohtana on agile manifesti. Agile manifesti on laadittu useiden sovelluskehittäjien toimesta. Agile manifestin avulla sovelluskehittäjät löytävät toimivampia toimintatapoja kehittää sovellusta työstämällä sitä ja auttamalla muita samanaikaisesti. Tätä työskentelytapaa noudattaen on muodostettu seuraavat viisi manifestin arvoa. (The Agile Manifesto. N.d.)

1. *”Yksilöt ja kanssakäymiset ovat tärkeämpiä kuin prosessit ja työkalut.”*

2. *”Toimiva ohjelmisto on tärkeämpi kuin laaja dokumentointi.”*

3. *”Asiakasyhteistyö on tärkeämpää kuin sopimusneuvottelut.”*

4. *”Muutoksiin reagointi on tärkeämpää kuin suunnitelman noudattaminen.”*

(The Agile Manifesto. N.d.)

Ketterät periaatteet on laadittu agile manifestin pohjalta. Alla olevat kaksitoista kohtaa kuuluvat ketteriin periaatteisiin, jotka ohjaavat ketterän tiimin toimintaa laajemmin, kuin pelkät manifestin pääarvot. (12 principles Behind the Agile Manifesto. N.d.)

1. *”Tärkein prioriteetti on asiakkaiden tyytyväisyys sekä ohjelmien aikainen ja jatkuva toimitus”*

2. *“Muuttuvat vaatimukset ovat tervetulleita jopa myöhäisessä kehitysvaiheessa. Ketterissä prosesseissa valjastetaan muutos asiakkaan kilpailueduksi.”*

3. ”Toimitetaan toimiva ohjelmisto säännöllisesti, muutaman viikon tai muutaman kuukauden välein, suosien lyhyttä aikajännettä.”

4. ”Business-puolen työntekijöiden ja kehittäjien tulee työskennellä päivittäin yhdessä koko projektin läpi.”

5. “Rakennetaan projektit motivoituneiden yksilöiden ympärille. Annetaan heille ympäristö ja tuki sekä luotetaan siihen, että he saattavat työnsä loppuun saakka.”

6. “Kaikista tehokkain ja toimivin tapa siirtää informaatiota kehitystiimin välillä on keskustelu kasvotusten.”

7. “Toimiva ohjelmisto on ensisijainen mitta kehitykselle.”

8. ”Ketterät prosessit kannustavat kestävään kehitykseen. Rahoittajien, kehittäjien ja käyttäjien tulisi pystyä ylläpitää tasainen tahti loppuun saakka.”

9. “Jatkuva huomio tekniseen osaamiseen ja suunnitteluun kehittää ketteryyttä.”

10. “Yksinkertaisuus eli turhien työvaiheiden karsiminen on välttämätöntä”

11. “Parhaat arkkitehtuurit, vaatimukset ja suunnitelmat syntyvät itseohjautuvissa tiimeissä.”

12. “Säännöllisin aikavälein tiimi reflektoi sitä, miten he voisivat kehittää tehokkuuttaan ja reflektoinnin myötä tarvittaessa muokkaavat ja hienosäätävät toimintatapojaan.”

(12 principles Behind the Agile Manifesto. N.d.)

3.2 Scrum

Scrum on yksi ketterän kehityksen viitekehyksistä. Sen tarkoituksena on auttaa ihmisiä havaitsemaan monimutkaisesti mukautuvat ongelmat. Samanaikaisesti tulisi toimittaa korkea-arvoisia tuotteita tehokkaasti ja luovasti. Scrum on kevyt toteuttaa ja yksinkertainen ymmärtää mutta se on vaikea hallita täydellisesti. (Schwaber & Sutherland 2020)

Scrumin viitekehys koostuu scrum tiimistä, jossa on eri rooleja, tapahtumia, artefakteja ja sääntöjä. Jokaisella komponentilla on viitekehyksessä oma rooli, joka on tärkeä scrumin onnistumisessa ja käytössä. (Schwaber & Sutherland 2020)

Scrum tiimi

Scrum tiimi koostuu product ownersista, kehittäjistä ja scrum masterista. Scrum tiimit ovat itsenäisiä ja hallinnoivat omaa toimintaansa. Itse organisoituva tiimi pystyy päättämään, kuinka he toteuttavat työnsä parhaiten. Työnjako tapahtuu täysin tiimin kesken, eikä ulkopuolista tahoa, joka antaisi ohjeita työntekoon ole. Tiimeissä on useita eri osaajia ja tämän vuoksi tiimin ulkopuolista osaamista ei tarvita projektin edistämiseen. Scrum-tiimin toimintamalli on suunniteltu joustavaksi, luovaksi ja tuottavaksi. Tämä tekee scrum toimintamallin todella tehokkaaksi. (Schwaber & Sutherland 2020)

Product ownerin tehtävänä on selkeän vision omaaminen ja suunnan näyttäminen projektin aikana. Hän varmistaa tuottavuuden maksimoinnin ja omaa vision scrum tiimin tuottamasta arvosta asiakkaalle. Product owner on vastuussa product backlogista, julkaisuaikataulusta ja sidosryhmien hallinnasta. Product owner määrittelee tuotantotiimille tärkeät sprintin aikaiset tehtävät, jonka myötä luottamus hänen ja tuotantotiimin välillä on onnistumisen kannalta merkittävää. (West n.d.)

Scrum master on tiimin jäsen, joka on vastuussa siitä, että scrumin käytänteet toteutuvat sekä mahdollistaa tiimin tehokkuuden. (Schwaber & Sutherland 2020) Scrum masteria voi kuvailla johtajana, joka palvelee tiimiä. Hän hoitaa käytännön asiat kuten product ownerin auttamisen arvojen määrittelyssä sekä avustaa tuotantotiimiä saavuttamaan nämä määritellyt arvot. Scrum master on product owneria ja muuta tiimiä auttava taho, joka huolehtii tiimin välisestä kommunikaatiosta ja backlogin hallinnasta. (West n.d.)

Sprintti

Sprintti on sovitun pituinen ajanjakso, jonka aikana projektia työstetään eteenpäin. Sprintin pituus voi vaihdella viikoista kuukauteen. Ajanjakson tarkoituksena on luoda säännöllisyyttä työskentelyyn. Seuraava sprintti alkaa välittömästi edellisen päätyttyä. Tarvittava työ toteutetaan sprintin aikana. Sprintti sisältää suunnittelun, daily scrumit, sprintin katsauksen ja retrospektiivin. Sprintin aikana ei tehdä enää suuria muutoksia, jotka vaikuttaisivat koko sprintin lopputulokseen. Product backlogia voidaan tarkentaa tarvittaessa sekä laajuutta voidaan selkeyttää ja uudelleen neuvotella product ownerin kanssa. (Schwaber & Sutherland 2020)

Sprintit mahdollistavat ennustettavuuden työnteossa, kun tietyin väliajoin voidaan tarkastella työn etenemistä ja suunnitella uudestaan seuraava sprintti. Jos sprintti on liian pitkä voi sprintin tavoitteista tulla liian haastavia toteuttaa ja tämä voi lisätä riskitekijöitä. Lyhyemmät sprintit ovat näissä tilanteissa turvallisempia toteuttaa. Jokaista sprinttiä voi ajatella oppimiskokemuksena ja omana projektinaan kokonaisuuden sisällä. (Schwaber & Sutherland 2020)

Sprinti alkaa suunnitteluvaiheesta. Tämän aikana määritetään sprintin tavoite ja sen sisältämä työmäärä. Suunnitteluvaiheeseen osallistuu koko scrum-tiimi. Product owner varmistaa, että osallistujat ovat selvillä backlogissa olevista tärkeistä tehtävistä ja siitä, kuinka ne määrittävät projektin tavoitteita. (Schwaber & Sutherland 2020)

Sprintin katsauksessa tiimi esittelee sprintin aikaisen edistymisen ja määritetään tulevat lisäykset. Tässä katsauksessa mukana voivat olla kaikki avain sidosryhmät, joiden kanssa keskustellaan projektin etenemisestä. Product backlogia voidaan muokata vastaamaan uusia tavoitteita, jotka ovat tulleet esille katsauksen aikana. Katsaus on keskustelunomainen tilaisuus, jota käytetään kehityksen työkaluna, joten tiimin tulee välttää pelkkää esitysmuotoista lähestymistapaa katsaukseen. (Schwaber & Sutherland 2020)

Sprintin viimeinen osa on retrospektiivi, jonka tarkoituksena on suunnitella, kuinka parannetaan laatua ja tuottavuutta. Scrum-tiimi käy läpi sprintin kulkua prosessien, työkalujen, interaktioiden, yksilöiden ja definition of done:n näkökulmasta. Tiimi keskustelee missä onnistuttiin sprintin aikana ja mitä ongelmia tuli vastaan ja miten ne käsiteltiin. Retrospektiivissä ilmi tulleet asiat otetaan käyttöön seuraavassa sprintissä, jotta voidaan parantaa tehokkuutta. (Schwaber & Sutherland 2020)

Product backlog

Product backlog on lista kaikesta siitä, mitä tiedetään projektiin tulevan. Product owner on vastuussa backlogista eli hän toteuttaa kaikki muutokset, lisäykset ja ylläpidon backlogiin. Product backlog muuttaa muotoaan jatkuvasti, kun siihen lisätään uusia tehtäviä, joita tiimin tulee toteuttaa. Backlog listaa kaikki ne ominaisuudet, tehtävät, vaatimukset, parannukset ja korjaukset, jotka tuotteeseen tulee tehdä tulevaisuudessa. Tehtävillä on ominaisuuksia, jotka kuvailevat työtä. Ominaisuuksia voi olla esimerkiksi lyhyt kuvaus, aika-arvio ja arvo mitä asia tuo. Nämä tehtävät ovat myös user storyja, joita käydään opinnäytetyössä myöhemmin lisää läpi. (Schwaber & Sutherland 2020)

Product backlogissa tehtävien prioriteetit määrittävät sen, kuinka nopeasti ne tulisi saada tuleviin sprintteihin. Korkean tason prioriteetilla olevat tehtävät ovat usein selkeämpiä ja sisältävät enemmän tietoa, kuin alemman prioriteetin tehtävät. (Schwaber & Sutherland 2020)

Sprint backlog

Sprint backlogiin kerätään sprintin aikana suoritettavat tehtävät product backlogista. Nämä toimivat sprintin tavoitteina. Sprint backlog on tiimin kesken suunniteltu ja sovittu määrä työtä, joka toimii ennusteena siitä mitä saadaan aikaan sprintissä. (Schwaber & Sutherland 2020)

Sprint backlog visualisoi työn, jonka tiimi näkee tarpeelliseksi saavuttaakseen sprintin tavoitteen. Jotta jatkuva kehitys saavutettaisiin, tulee sprinttiin sisällyttää ainakin yksi korkean prioriteetin omaava tehtävä. Tämä tehtävä määritellään tiimin kesken jo edellisessä retrospektiivissä. (Schwaber & Sutherland 2020)

Sprint backlog on suunnitelma, jossa on tarvittavat tiedot tuotannossa tapahtuvien muutoksien ymmärtämiseen daily scrumien aikana. Daily scrum on hyvin lyhytkestoinen päivittäinen tapaaminen tiimin kesken, jossa käydään läpi tehdyt työt ja mitä tullaan tekemään jatkossa. Tämän perusteella tiimi pystyy tekemään muutoksia helposti ja nopeasti. Tuotantotiimi muokkaa jatkuvasti sprint backlogia sprintin aikana ja tällöin lopullinen sprint backlog syntyy sprintissä, kun siihen tehdään muutoksia. Luomisprosessi tapahtuu tiimin edetessä suunnitelman mukaan. Tällöin tiimi oppii uutta tietoa, jonka pohjalta he voivat tehdä muutoksia sprintin tavoitteisiin. (Schwaber & Sutherland 2020)

Tuotantotiimi tarvittaessa lisää työn sprint backlogiin. Saatettaessa työ loppuun tai sitä toteutettaessa jäljelle jäävän työn arvio päivitetään. Tarpeettomaksi todetut työt poistetaan backlogista, jotta ne eivät vääristä arvioita. Sprint backlog on näkyvillä oleva reaaliaikainen kuvaus tiimin työstä, jonka he aikovat suorittaa sprintin aikana ja se kuuluu ainoastaan tuotantotiimille, joten kukaan muu ei voi siihen tehdä muutoksia kuin tiimi itsessään. (Schwaber & Sutherland 2020)

3.3 Kanban

Kanban on ketterän ohjelmistokehityksen runko niin kuin scrumikin. Kanban eroaa scrumista siten, että työ tehdään jatkuvalla toteutuksella verrattuna scrumin sprintti malliin. Vaatimuksina kanbanissa on jatkuva kommunikointi ja täydellinen läpinäkyvyys työn määrästä. Työtehtävät näkyvät

kanban taulussa, josta tiimi näkee työn visuaalisessa muodossa. Tämä mahdollistaa tiimiläisille jatkuvan mahdollisuuden nähdä jokaisen osan työstä. (Rehkopf n.d.f.)

Kanban tiimin suunnitteluprosessi on joustavaa ja tiimin toiminta keskittyy vain aktiivisena oleviin työvaiheisiin. Kun tiimi saa työn alla olevan tehtävän valmiiksi, lisätään backlogista seuraavana oleva tehtävä tauluun. Product ownerin tulee pitää backlog ajan tasalla, jotta prosessi on toimiva. Tällöin tiimin ei tarvitse huolehtia backlogiin liittyvistä asioista, vaan he huolehtivat vain työn alla olevista tehtävistä. Product owner voi vapaasti tehdä muutoksia backlogissa oleviin tehtäviin. (Rehkopf n.d.a.)

Taulu

Kanban tiimien työ keskittyy kanban taulun ympärille. Taulu on työkalu, jota käytetään työn visualisointiin ja jonka avulla optimoidaan työn sujuvuus tiimin sisällä. Fyysiset taulut ovat melko suosittuja, mutta virtuaalinen taulu antaa tiimille paljon laajemmat mahdollisuudet toteuttaa ketterää kehitystä. Virtuaalinen taulu tarjoaa jäljitettävyyttä ja mahdollisuudet parempaan yhteistyötoimintaan eikä saavutettavuus ole kiinni fyysisestä paikasta. (Rehkopf n.d.a.)

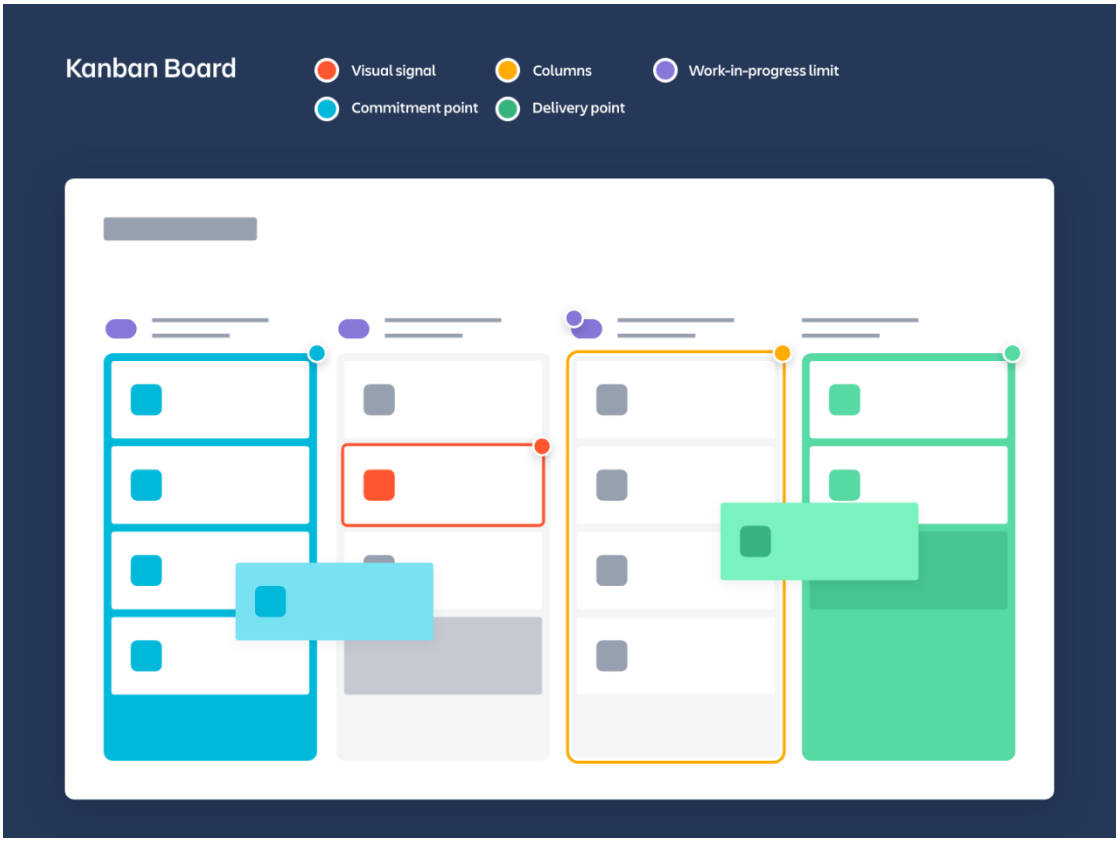
Taulu voidaan jakaa viiteen eri komponenttiin: visuaalisiin signaaleihin, sarakkeeseen, työrajoituksiin, sijoittumispisteeseen ja toimituspisteeseen. Kortit ovat visuaalisia elementtejä, jotka auttavat hahmottamaan tiimin työtä sillä kortti kuvastaa aina yhtä työtehtävää. Visuaalisilla signaaleilla tiimi ja muut asianosaiset näkevät nopealla katsauksella mitä tiimi on työstämässä. (Rehkopf n.d.a.) Kuviossa 1 näkyy esimerkkikuva kanban taulusta ja sen toimivuudesta.

Sarakkeet kuvaavat taulussa jokaista työvaihetta minkä läpi kortit menevät, eli tuotantoprosessia. Sarakkeet voivat koostua yleisesti seuraavista sarakkeista: to-do, in-progress, testing ja done. Tähän voidaan lisätä useita monimutkaisempiakin järjestelyjä tuotantoprosessin läpimenemiseksi. (Rehkopf n.d.a.)

WIP (work in progress) limits eli wip rajoitteet ovat maksimimäärä kortteja, joita voi olla sarakkeessa samanaikaisesti työn alla. Wip rajoitteet ovat kriittisiä pullonkaulojen löytämisen ja työn tehokkuuden maksimoinnin vuoksi. Wip rajoitteet antavat varoitussignaaleja, jos johonkin työhön keskitytään liikaa. (Rehkopf n.d.a.)

Kanban tiimeillä on usein backlog taululleen. Tähän backlogiin asiakkaat ja tiimin jäsenet lisäävät projekti-ideoita, joista tiimi valitsee toteutettavat ideat. Commitment pointilla eli sitoutumispisteellä tarkoitetaan sitä hetkeä, kun tiimi ottaa idean tuotantoon ja alkaa työstää projektia. (Rehkopf n.d.a.)

Delivery point eli toimituspiste on Kanban tiimin tuotantoprosessin päätepiste. Usein tämä tarkoittaa esimerkiksi tuotteen luovuttamista asiakkaalle. Tiimin tavoitteena on siirtää Kanbanin kortit mahdollisimman nopeasti sitoutumispisteestä toimituspisteeseen. Näiden prosessien välissä kulunutta aikaa kutsutaan toimitusajaksi (Lead time). Kanban tiimien tavoitteena on jatkuva toimitusajan tehostaminen. (Rehkopf n.d.a.)



Kuvio 1. Kanban taulun visualisointi (Rehkopf n.d.a.).

Kortit

Kanban kortti on artifakti joka kulkee työprosessin läpi. Jokainen kortti edustaa yksittäistä työtehtävää sen mennessä useiden työvaiheiden läpi, jotka on esitetty kanban boardilla. Kortit visualisoivat tiedon ja helpottavat tärkeän informaation näkemistä jo vilkaisulla. Jokainen kortti sisältää lyhyen kuvauksen työstä, kortin tekijästä, deadlinesta ja työn statuksesta. Näiden tietojen lisäksi kortti voi sisältää lisäinformaatiota kuten linkkejä materiaaleihin ja listoja, jotka kertovat mitä asioita tulisi olla tehtynä ennen tätä korttia. (Rehkopf n.d.b.)

Suoritteiden luovuttaminen tapahtuu sulavasti ja tehokkaasti. Kanban kortit kannustavat tiimejä luomaan selkeitä ja johdonmukaisia oletuksia jokaiselle toiminta-alueelle. Nämä käytänteet helpottavat tunnistamaan sen, kenen omistuksessa työvaihe on ja mikä vaihe on seuraavaksi tulossa. (Rehkopf n.d.b.)

Kanban kortit parantavat työn tehokkuutta tekemällä kortin läpimenoajan seurannasta paljon helpompaa. Kortin läpimenoajalla tarkoitetaan sitä aikaa, joka tehtävän vieminen alusta loppuun vaatii. Kanban kortit yhdistettynä kanban tauluun auttavat tiimiä tunnistamaan mahdollisia pullonkauloja tuotannossa ja näin ollen toiminnasta saadaan sulavampaa. Tiimien tavoitteena on korttien läpimenoajan pienentäminen ja tämä tarkoittaa tuotantoprosessin nopeuttamista. (Rehkopf n.d.b.)

4 Ketteryyttä kehittävät työkalut

Opinnäytetyössä tutkitaan kahta eri projektinhallinnan työkalua Jiraa ja Notionia. Molemmat näistä on laajasti käytettyjä projektinhallinnan työkaluja. Notion on edullinen vaihtoehto ja Jira on kaupalliseen työhön tarkoitettu sovellus. Kummassakin sovelluksessa on samanlaisia ominaisuuksia, joista on hyötyä pelikehityksen projektinhallinnassa. Tässä luvussa käydään läpi ketterän kehityksen käsitteitä, jotka auttavat hahmottamaan projektinhallintasovellusten käyttötarkoituksia.

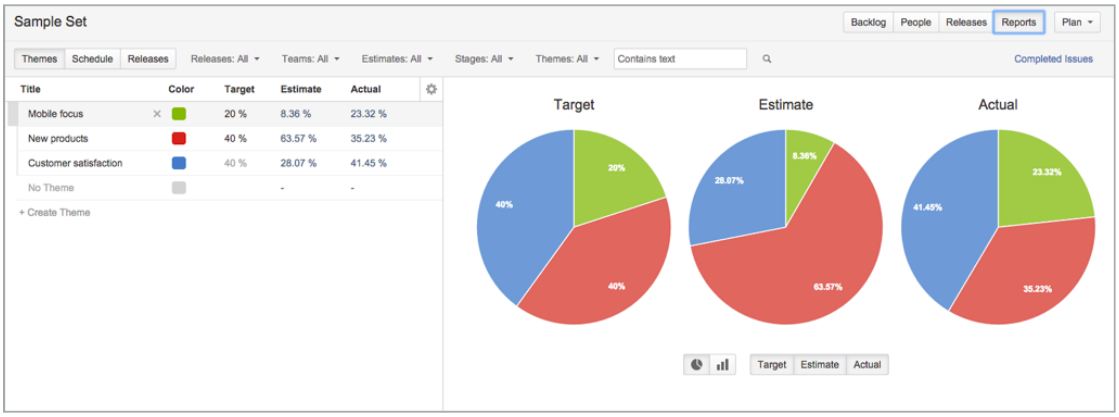
Jira on ketterää kehitystä tukeva projektinhallintasovellus, joka mahdollistaa useiden tiimien välisen yhteistyön. Jiran avulla voidaan seurata projektin etenemistä alusta loppuun ja tehdä työskentelystä mahdollisimman tehokasta. Jiran avulla tiimit voivat suunnitella, seurata ja raportoida työskentelyään sekä hallita projektejaan. (A brief overview of Jira n.d.)

Notion on tuottavuutta edesauttava työkalu, jossa on paljon ominaisuuksia ja työkaluja projektien hallintaan ja muuhun yhteistyötoimintaan. Notion mahdollistaa organisaation työn seurannan ja dokumenttien hallinnan. (D'Alessio 2018)

4.1 Teema

Teemat ovat laajoja alueita, joihin organisaatio keskittää huomiotaan. Teemat ovat tapa merkitä backlogia strategisten painopisteiden, arvovirtojen tai investointikategorioiden mukaan. Teemojen avulla sidosryhmät voivat nähdä mihin organisaatio käyttää aikaa ja resursseja verrattuna siihen, mitä on suunniteltu. (Smith 2015)

Jiran teemoissa on kolme osiota, joiden kautta voidaan seurata teeman etenemistä. Kohde -osioon voidaan itse määritellä arvot, joita voidaan käyttää perustana muiden taulukoiden vertailussa. Arvio -osioon allokoidaan kaikki teeman backlogissa olevat asiat ja tästä lasketaan prosenttiosuus suhteessa kaikkiin backlogissa oleviin tehtäviin. Viimeisenä osiona on todellisuus. Todellisuus -osiossa näkyy allokoituna kaikki työ, joka on tehty. Tämä siis näyttää lopullisen tuloksen siitä kuinka paljon työtä on jo tehty. (Smith 2015) Kuviossa 2 havainnollistetaan näitä Jiran teemojen osioita.



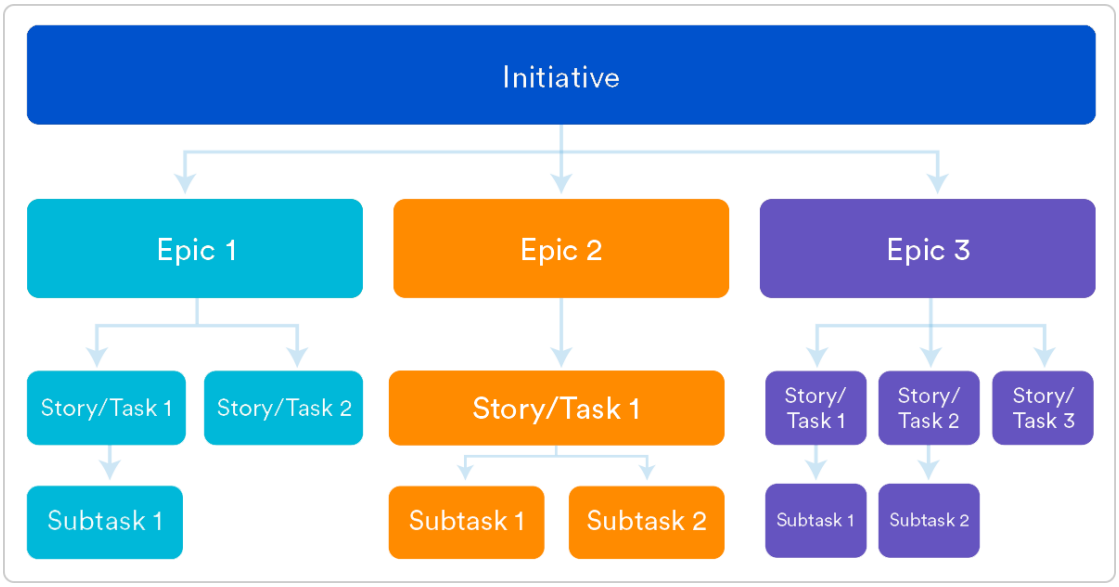
Kuvio 2. Teemojen seurannan näkymä Jirassa (Smith 2015).

4.2 Initiative

Initiativet ovat isoja projektin osia, jotka koostuvat useista epiceistä. Varsinkin useiden tiimien välisissä projekteissa initiativet ovat hyvä tapa seurata edistymistä. Useiden tiimien työstäessä samaa ominaisuutta voidaan heidän työstämät epicit kasata samaan initiativeen jolloin nähdään koko ominaisuuden edistyminen. (Radigan 2014)

Verrattuna muihin pienempiin elementteihin initiativen saaminen valmiiksi kestää hyvin pitkään. Valmiiksi saamisessa voi kestää jopa vuosi, kun esimerkiksi epicit valmistuvat muutamissa kuukausissa ja storyt sprinttien aikana. Koska initiativet koostuvat epiceistä, valmistuu initiative samalla, kun kaikki epicit on saatu valmiiksi. (Rehkopf n.d.c.)

Pelissä voi olla esimerkiksi moninpeli ja yksinpeli, jotka toimivat iniativeina. Nämä ovat isoja osia, jotka pilkotaan useisiin eri epiceihin jotka myöhemmin vielä jaetaan pienemmiksi storyiksi. Moninpeli on tällöin suuri osa-alue, jossa voi olla useampi tiimi työstämässä ja epicit jaetaan tiimien kesken. Iniative toimii tällöin ylempänä tasona koko moninpelin seuraamiseen. Kuviosta 3 nähdään esimerkkirakenteen ominaisuuden pilkkomiselle.



Kuvio 3. Ominaisuuden pilkkomisen esimerkkirakenne (Rehkopf n.d.c.).

4.3 Epic

Epicit ovat suuria työkokonaisuuksia, jotka voidaan pilkkoa pienempiin osiin. Näitä epiceistä pilkottuja osia kutsutaan user storyiksi. Epicien tarkoituksena on hallinnoida suuria osioita työstä saman nimikkeen alla. Tämän nimikkeen alle kerätään kaikki osioon liittyvät työtehtävät yhtenäiseksi osaksi, jota on helppo hallita ja seurata. Epicit mahdollistavat hierarkkisen järjestelytavan user storyille ja taskeille. Epicit kirjoitetaan tarinamuotoisesti kuten user storytkin. Tarkoituksena on pilkkoa laajempi kokonaisuus pienempiin osiin, jotka voidaan saada valmiiksi palasiksi. Tämä mahdollistaa työlle pienemmät välitavoitteet, joita kohti on helppo edetä ja samalla jatketaan projektin lopputavoitetta kohti työstämistä. (Rehkopf n.d.d.)

Epicit jakautuvat useiden sprinttien ajalle ja niitä voi työstää useampi tiimi. Epicejä seurataan useampien taulujen kautta. Kehityksen edetessä tiimi oppii uusia asioita, jolloin epiceihin lisätään uusia user storyja ja poistetaan vanhoja. Tämän takia epicit elävät jatkuvasti ja auttavat tiimiä ketterämpään kehitykseen. (Rehkopf n.d.d.)

Initiative pilkottiin moninpeliin ja yksinpeliin. Epicejä voidaan tästä lähteä pilkkomaan pienenpiin kokonaisuuksiin tarinallisessa muodossa. Epic voisi olla esimerkiksi seuraavanlainen: Pelaajana haluan pystyä muokata hahmoni ulkonäköä, jotta hahmo olisi persoonallinen. Tämä tarkoittaisi, että tiimin tulisi kehittää peliin ominaisuus, jonka avulla pelaaja voi muokata hahmoa.

4.4 User story

User story on työkalu, jota käytetään ketterässä kehityksessä niin pelikehityksessä kuin ohjelmistotuotannossakin. User storyssa kuvataan pelin ominaisuutta selkeällä käyttäjälähtöisellä kielellä. User storyssä määritellään, kuka käyttäjä on, mitä hän haluaa tehdä ja miksi hän haluaa tämän tehdä. User storyjen tarkoituksena on luoda yksinkertainen kuvaus ominaisuudelle. (What is User Story n.d.)

User storyt kuvaavat pelin vaatimuksia käyttäjän näkökulmasta. Tarkoituksena ei ole ominaisuuden pikkutarkka kuvailu, vaan user storyjen tarkoituksena on antaa selkeä kuva ominaisuuden toimivuudesta. Tällöin jokainen osapuoli ymmärtää mistä on kyse, kun keskustellaan ja suunnitellaan mitä peliin halutaan tehdä ja mitä halutaan saavuttaa. (Keith 2010, 88)

Kasvokkain käytävä keskustelu on hyvin tärkeää, jotta tiimin sisällä vältyttäisiin väärinymmärryksiltä. Suunnitteluvaihe tulee tehdä hyvin ja asioista tulee keskustella useiden eri tahojen kanssa, jotta vältytään turhilta ongelmilta esimerkiksi, kun user story saadaan valmiiksi eikä se toimikkaan niin kuin pitäisi. Kun peli ei toimikkaan halutunlaisesti voidaan törmätä suorituskykyongelmiin. (Keith 2010, 103)

Hyvä user story sisältää definition of done:n, eli sovitun määritelmän siitä milloin user storyt luokitellaan valmiiksi. Definition of done on usein tarkistuslista, jonka avulla määritellään testauksessa tarkistettavat asiat, jotka tulee läpäistä, jotta user story voidaan hyväksyä valmiiksi. (Keith 2010, 99)

Epicissä annetun esimerkin perusteella ei vielä tiedetä tarkempaa toimivuutta hahmon muokkaukseen, joten tässä tulevat user storyt apuun kertomaan mitä toiminnallisuuksia vaaditaan. Epicistä pilkottuja user storyjä voisi olla esimerkiksi: Pelaajana haluan navigoida muokkausvalikossa, jotta löydän haluamani muokkaus mahdollisuudet sekä pelaajana haluan nähdä hahmoni samalla kun teen muokkauksia, jotta näen reaaliajassa tekemäni muutokset. Näillä user storyillä saadaan jo tarkennettua koko epiciä siten että hahmonmuokkaus ominaisuuteen tulee rakentaa toimiva navigointi käyttöliittymä, jossa samanaikaisesti nähdään hahmoon tehtävät muokkaukset.

INVEST-malli

Hyvä user story voidaan määrittää INVEST mallia käyttämällä. Invest mallin attribuutit koostuvat seuraavista osa-alueista: independent, negotiable, valuable, estimable, small ja testable. (Keith 2010, 92)

Independent

Itsenäiset user storyt eivät ole riippuvaisia muiden user storyjen implementointijärjestyksestä, vaan ne voidaan toteuttaa ilman muita user storyja. Kun user story on riippuvainen muista user storyista, tulee ongelma priorisoinnin ja arvioinnin kanssa. (Keith 2010, 92)

Negotiable

User storyt eivät ole sopimuksia, vaan niiden tulee olla apuna keskustelussa ja niitä on mahdollisuus muuttaa. Liian tarkasti määritellyt user storyt estävät muutosten tekoa luomalla harhakuvaa siitä, ettei keskustelua tarvitse käydä. (Keith 2010, 93–94)

Valuable

User storyjen tulee luoda arvoa pelaajalle, tuotantotiimille ja pelin markkinoinnille. Product owner määrittelee user storyjen tärkeyden backlogissa, ja tämän vuoksi hänen tulee nähdä user storyn arvo, jotta hän osaa määrittää sen prioriteetin. (Keith 2010, 95)

Estimatable

Jokaisen user storyn tulee olla arvioitavissa ajallisesti ja resurssitarpeellisesti. Riittämättömän tiedon takia user storyn scope voi olla liian suuri, jolloin arviointia ei voida tehdä kunnolla. Tämä voi aiheuttaa riskejä aikataulutuksessa ja eri roolien tehtävissä. User storyjen valmiiksi saaminen voi hyvin usein vaatia jotakin teknologiaa, esim. työkalua, valmistuakseen ja tämän vuoksi arviointi voi olla haastavaa. (Keith 2010, 95)

Liikaa tietoa tai vastaavasti liian vähän tietoa omaavan user storyn työstäminen ja valmiiksi saaminen voi viedä hyvinkin lyhyen ajan tai kestää jopa useita kuukausia. Tällaisissa tilanteissa tulee käyttää aikaa tutkimiseen, jotta saataisiin kaikki tarvittava tieto työn tekemiseen eikä mennä päät-

tömästi eteenpäin tietämättä mitä on tulossa. Tutkimalla voi esimerkiksi selvittää löytyykö työkaluista jo olemassa olevaa ominaisuutta, joilla tehtävän voi toteuttaa vai joudutaanko tällainen rakentamaan alusta asti. (Keith 2010, 96)

Tuotantoprosessissa otetaan sprinttejä, joissa toteutetaan tämä tutkimis- ja kehitysvaihe. Näitä sprinttejä kutsutaan piikeiksi (spike). Tiimin ollessa pieni ei ole mahdollisuutta sijoittaa jatkuvasti väkeä tutkimustoimiin, joten ovat piikit tällöin hyödyllisiä. Tällöin saadaan kaikki tarvittavat työkalut ja lisätiedon hankkiminen toteutettua kerralla, jonka jälkeen päästään jatkamaan tuotantoprosessia normaalisti ilman esteitä. (Keith 2010, 96)

Sized appropiately

Hyvän user storyn tulee olla tarpeeksi pieni. User storyn kooksi voi arvioida esimerkiksi työntekijän muutaman viikon työtunnit. Pienemmän storyn kokoa on helpompi arvioida ja tällöin siitä tulee paljon tarkempi. (Wake 2003) User storyn koon tulee olla sellainen, että se saadaan mahdutettua yhteen sprinttiin (Keith 2010, 96).

Testable

Sprintille määriteltyjen user storyjen tulee olla sellaisia, että ne saadaan valmiiksi ennen sprintin päättymistä. Jotta user story voidaan saattaa valmiiksi, voidaan kortille määritellä ehdot, jotka sen tulee läpäistä testauksessa. Taululta löytyy tätä varten testaus -osio tai jokin muu varmistussarake, jossa testaaja ja muut hyväksynnän antavat henkilöt voivat käydä tarkistamassa user storyn ja todeta sen valmiiksi. (Keith 2010, 97)

4.5 Task

Task on user storysta pilkottu pala työtä. Taskit ovat useimmiten yhden ihmisen työstämiä kokonaisuuksia. (Task n.d.) Taskeja voidaan tallentaa ja seurata monella eri tavalla. Yksi toimiva tapa seurata taskeja on tehtäväkortit. Korttien avulla kaikki tiimin jäsenet voivat osallistua tehtävien luomiseen ja hallintaan. Kortteja on helppo kustomoida ja niistä saa helposti visuaalisen kuvan meneillään olevista töistä. (Keith 2010, 69) Taskit ovat helppo tapa seurata user storyn kehittymistä. Taskien avulla nähdään kaikki mitä user storyn valmiiksi saamiseksi tulee tehdä. (PEARL IV: INVEST in good User Stories and SMART Tasks 2014)

Usert story esimerkeistä voidaan lähteä pilkkomaan toimivuudet pienemmiksi taskeiksi, jotka ovat tiimin yksinkertaisia työtehtäviä. Esimerkiksi näistä user storyistä voidaan luoda taskeksi: Navigoinnin logiikan ohjelmointi, käyttöliittymän graafiset elementit, 3D mallin näyttäminen hahmosta taustalla.

Smart malli

Smart mallin avulla voidaan määritellä taskeille toimivia tavoitteita. Smart malli koostuu seuraavista osista: specific, measurable, achievable, relevant ja time bound.

Spesific

Taskin tulee olla tarkka, jotta jokainen ymmärtää mitä siihen kuuluu. Taskien tarkoituksena on olla niin tarkkoja, etteivät ne sekaannu toistensa kanssa eivätkä sisällä samoja asioita. Tämä auttaa jokaista ymmärtämään miten taski sisältyy lopulliseen user storyyn. (PEARL IV: INVEST in good User Stories and SMART Tasks 2014)

Measurable

Taski tulee olla mitattavissa. Tärkeää on määritellä ne mitta-arvot, jotka määrittelevät sen, milloin tavoite on saavutettu. Mitta-arvot ja erilaiset välietapit tekevät tavoitteista konkreettisempia, sillä niiden avulla voidaan mitata edistymistä. (Lynch 2019)

Achievable

Työ tulee olla saavutettavissa. Tekijän tulee omata taidot, joiden avulla taskin voi suorittaa. Jos osaaminen on puutteellista, tulee sitä hankkia, jotta työ saadaan tehtyä. Tavoitteena on lisätä motivaatiota eikä herättää pelkoa. (Lynch 2019)

Relevant

Jokaisen taskin tulee olla merkityksellinen ja sen tulee tuoda lisäarvoa user storyyn. Kehittäjän työtä helpottamaan user storyt pilkotaan taskeiksi, mutta taskien tulee kuitenkin olla selitettävissä product ownerille ja sidosryhmille. (Lynch 2019)

Time bound

Jos taskilta puuttuu realistinen ajoitus, on onnistuminen hyvin haastavaa. Tavoitteelliset deadlinet ovat onnistumisen kannalta erittäin tärkeitä ja taskeja suunniteltaessa tuleekin pohtia, mitä on mahdollista saada valmiiksi missäkin ajassa. Kun taskeilla on deadlinet määriteltynä, tulee työ varmemmin tehtyä valmiiksi sovittuun päivämäärään mennessä. (Lynch 2019)

Määrittämällä selkeä deadline taskin valmistumiselle, taskin tekijä osaa helpommin arvioida milloin työssä tulee liian kiire ja milloin tulee pyytää apua muilta tiiminjäseniltä. Apua pyydettäessä voidaan esimerkiksi jakaa taski useammalle tekijälle tai muokata sitä, jos siitä on muodostunut odotettua haastavampi. (PEARL IV: INVEST in good User Stories and SMART Tasks 2014)

5 Tulokset

Tässä kappaleessa käydään läpi teemahaastattelussa käydyn keskustelun tuloksia sekä sovellusten vertailun tulokset. Haastattelu käytiin keskustelunomaisesti hyödyntäen haastattelukysymyksiä, jotka löytyvät liitteestä 1. Haastateltavana toimi toimeksiantajayrityksen avainedustaja.

Sovelluksia tutkittiin testaamalla kumpaakin sovellusta käytännössä. Tutkimuksen kohteena olevat sovellukset olivat Jira ja Notion. Sovelluksien tutkittavat ominaisuudet on valittu teemahaastattelusta nousseiden asioiden perusteella.

5.1 Teemahaastattelu

Haastateltava kertoo, että heidän prosesseissansa ei käytetä erikseen user storyja vaan tehtävien kuvaamisessa hyödynnetään taskeja. Taulu koostetaan roadmap ominaisuuksista ja jokaiselle osa-alueelle määritellään siihen kuuluvat epicit. Sprintin alussa käytävässä suunnitteluprosessissa epicit pilkotaan taskeiksi joilla tarkoitetaan työntekijöiden työtehtäviä. Näiden taskien edistymistä voidaan seurata. Haastateltavan mukaan heidän toiminnassaan user storylla ja taskilla tarkoitetaan käytännössä samaa asiaa. Tämä siis tarkoittaa sitä, että pääsääntöisesti käytetään työtehtävää (task) eikä tarinamuotoista selitettä (user story). Haastateltava kokee, että pienen tiimin etuna on helppo kommunikointi, jonka vuoksi heidän prosesseissaan user storyt eivät ole käytössä.

Haastateltava kertoo, että heidän projektinhallintaprosesseissansa käytössä on projektinhallintasovellus Notion. He kokevat Kanban-taulut toiminnalleen elintärkeiksi, sillä ilman niitä edistymisen

seuranta olisi hyvin haastavaa. Notionin erilaiset näkymät joilla taskit voidaan lajitella, helpottavat kokonaisuuksien hahmottamista ja tuovat joustavuutta asioiden hahmottamiseen.

Kustomoinnin koetaan olevan tärkeä osa projektihallintasovellusten toimivuutta. Etenkin joustavuus ticketien teossa on ollut Notionin hyvä ominaisuus. Epicit eivät olleet alussa toimeksiantajayrityksen prosesseissa mukana, mutta nykyään ne koetaan hyödyllisiksi. Epicit pilkotaan taskeihin ja bugeihin ja jokaiseen epiciin on yhdistetty siihen liittyvät taskit. Julkaisuversiot on implementoitu mukaan epiceihin jotta hahmotetaan mihin julkaisuversioon epic liittyy. Kaikille ticketeille on priorisointimahdollisuus. Haastateltava kertoo, että he eivät priorisoi taskeja, vaan bugit laitetaan tärkeysjärjestykseen siten, että mitä tehdään sprintin aikana. Jokaista taskia käsitellään sivuna, jonne voidaan monipuolisesti kustomoida taskin vaatimat tiedot. Notionissa taski voidaan dokumentoida hyvin laajasti auki, jotta tiimi ymmärtää mitä se sisältää.

Haastateltava kertoo, että toimeksiantajayrityksen prosesseissa epiciä voi kuvailla laajakuvaisena user storyna. Epic voi olla esimerkiksi moninpelin tekeminen, jotta peliä voidaan pelata yhdessä. Jos epicinä on esimerkiksi pelkkä ääni, kyseinen epic kuulostaa todella laajalta ja näin ollen olisi järkevämpää, että kyseinen epic olisi tarkemmin määritelty eikä kuulostaisi vain avainsanalta. Tällä hetkellä haastateltavan mukaan ääni-epic vaikuttaa ennemminkin työnkuvalta kuin epiciltä. Tämänlaiset epicit tulisi siis määritellä tarkemmin, jotta havaittaisiin, että se on laajempi kuin user story, mutta pienempi kuin yksi suuri kokonaisuus.

Haastateltavan mukaan heidän sprinttinsä koostuvat tuotteen päivityksistä ja tapahtuvat 1–2 viikon välein. Sprinttien pituus riippuu sprinttiin määriteltyjen ominaisuuksien mukaan. Tuotantoon varataan puolitoista viikkoa, jonka jälkeen sprintin lopussa luodaan release candidate eli RC. RC on versio, jonka ominaisuudet ovat valmiit ja se sisältää kaikki vaadittavat ominaisuudet, jotta sitä voidaan harkita julkaisuvalmiiksi. Tämän jälkeen tyypillisesti käytetään muutama päivä bugien korjaamiseen, jolloin versio hiotaan siihen kuntoon, että se on julkaisukelpoinen.

Sprintin alussa toteutetaan sprint planning jossa suunnitellaan tulevan sprintin sisältö, tavoitteet ja aikataulutus. Haastateltava kertoo, että heillä on käytössä scrum-tyyppinen menettely toiminnassaan. Sprint planningin jälkeen työtehtävät jaetaan osaamisen mukaan eri työntekijöille. Sprint planning koostetaan ottamalla epic ja purkamalla se pienempiin taskeihin. Tämän avulla saadaan realistinen käsitys vaaditusta työmäärästä ja voidaan seurata prosessin etenemistä tarkemmin.

Designer ja product lead pilkkovat taskit mahdollisimman tarkasti, mutta taskin tekijä pilkkoo työtään vielä pienempiin palasiin tarvittaessa. Sprintin aikana työstetään taskeja ja täydennetään niiden kuvauksia tarvittaessa. RC-vaiheeseen mentäessä kaikkien taskien tulee olla tarkistusvaiheessa. Kun taskit ovat valmiina, pidetään tunnin mittainen pelitestaus -sessio, jossa testataan ominaisuuksien toimivuutta ja tarkastellaan taskien onnistuneisuutta. Pelitestauksen aikana käytetään apuna laadunvarmistamislistaa, jonne on määritelty testattavat kohdat, joissa peli voi mennä rikki ja testataan näiden kohtien toimivuus. Testin aikana havaitut bugit listataan ylös Notioniin jonka jälkeen bugit korjataan ja uusi versio julkaistaan. Sprintin lopuksi pidetään vielä retrospektiivi, jossa käydään läpi, miten sprintti sujui. Tämän jälkeen alkaa uuden sprintin suunnitteluvaihe. Välillä sprinttien aluissa ja lopuissa on päällekkäisyyksiä, kun osa työntekijöistä on vailla tehtäviä buginkorjausvaiheessa ja tällöin sprint planning voidaan pitää aikaisemmin.

Notionin sisäinen kalenteri ei ole haastateltavan mukaan toimeksiantajayrityksellä käytössä, sillä se koetaan aikaa vieväksi. Tämän lisäksi se ei tuo riittävästi lisäarvoa ja ominaisuuksia tekemiselle, joten he käyttävät Google Sheet -taulukkoa sprinttien kalenterina. Kalenterin käyttö Notionissa on hidasta, sillä jokaiseen taskiin tulee erikseen määritellä määräajat, joka vie ylimääräistä työaikaa.

Haastateltava kertoo, että product lead käy päivittäin läpi taulunäkymää jota kehitystiimi käyttää aktiivisesti. Taskeja, jotka ovat tarkasteluvaiheessa, ei käydä läpi ennen RC:tä. Pelitestaussession aikana nämä kuitenkin tarkastetaan. Testailua tapahtuu päivittäin, mutta lopullinen pelitestaus suoritetaan RC-vaiheessa.

Haastateltavan mukaan avoin kommunikaatio Notionin kautta dokumentoidaan hyvin. Notionista löytyy hyvä dokumentaatio kaikille ominaisuuksille, joiden avulla tiimin jäsenet pysyvät ajan tasalla ja pääsee helposti tarkastelemaan taskien ja ominaisuuksien vaatimuksia. Designerit ylläpitävät dokumentaatioita, joiden avulla muut pysyvät tietoisina kaikista päivityksistä. Notionin kommunikaation lisäksi haastateltava kertoo toimeksiantajayrityksen kommunikaatiossa olevan käytössä Slack -viestintäsovellus, jonka avulla tiimin jäsenet pysyvät ajan tasalla meneillään olevista asioista.

Haastateltava kertoo, että eri rooleille (esimerkiksi art ja designer) on toimeksiantajayrityksen prosesseissa luotu omia tauluja sekä muita näkymiä joiden avulla pysytään selvillä roolikohtaisista taskeista. Sprintin suunnitteluvaiheessa nämä taulut käydään läpi, jotta jokainen löytää niistä uudet asiat. Kaikkia pieniä taskeja ei välttämättä joka kerta raportoida, vaikka haastateltava myöntää, että näin tulisi tehdä.

Haastateltava kertoo, että heidän prosesseissaan markkinointi pyritään tekemään sprintin kanssa samassa aikataulussa, mutta sitä ei käydä sprintin suunnittelussa läpi. Markkinoinnin taskit pyritään suorittamaan julkaisuun mennessä, jotta ne olisivat samanaikaisesti valmiina seuraavaan julkistukseen. Markkinoinnin taskeihin liittyy esimerkiksi markkinointikampanjat ja mainokset. Art joka ei liity peliin vaan esimerkiksi mainoksiin, pyritään pitämään taskeina Notionissa.

Haastateltava kokee, että organisaation kasvaessa tulisi tarkemmin seurata sitä, kuinka määritellään definion of done. Ajatuksena haastateltavalla oli, että esimerkiksi Jirasta löytyvät valintaruudut voisi olla myös Notionissa toiminnassa, jotta voitaisiin merkitä määritteet valmiiksi, jotka ovat definition of doneen määritetty.

Haastateltava kertoo, että arviointi puuttuu toimeksiantajayrityksen tuotantoprosesseista eikä Notionista löydy suoraan valmiiksi käytettävissä olevaa arviointipohjaa. Notionista puuttuu kokonaan työn arviointipisteet sekä burn down chart jonka avulla voidaan arvioida edellisten sprinttien edistymistä. Näiden avulla pystyttäisiin laskemaan työn tuotantonopeus.

Haastateltava toivoisi bugien tarkempaa seurantaa, sillä tällä hetkellä ne elävät samalla tavalla kuin muutkin taskit ennen kuin ne menevät taas arvioitavaksi ja ne merkitään tehdyiksi. Haastateltava kokisi, että tähän toimiva toimintatapa olisi retrospektin tyyppinen menettely, jossa voitaisiin tarkastella vain bugien edistymistä, kun tällä hetkellä ne näkyvät taskeina.

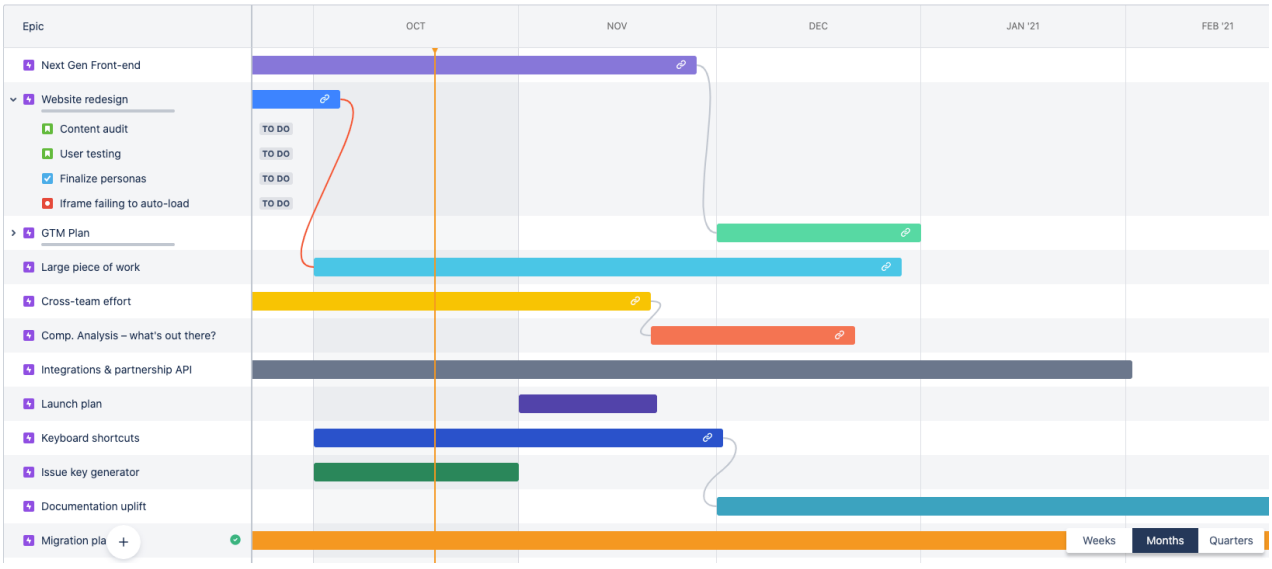
Haastateltava kertoo, että Jiran käytöstä on jäänyt kaipaamaan ticketien etsimistä jquery kielellä. Hakutapojen lisääminen olisi hyödyllinen ominaisuus, joka tällä hetkellä puuttuu Notionista. Haastateltava kaipaisi Notioniin omia kustomoitavia hakusanoja, joilla voisi suodattaa tickettejä.

Haastateltava kertoi, että versionhallinnan systeemien integrointi puuttuu Notionista. Tällä hetkellä heidän versionhallintansa tehdään GitHubin kautta. Jirassa oli ominaisuus, jolla pystyi laittamaan ticketin id:n jolloin sen pystyi merkkaamaan suoraan valmiiksi tai johonkin muuhun tilaan ja tätä ominaisuutta haastateltava kaipaisi Notioniin.

5.2 Sovellusten vertailu

Ajallinen seuranta

Jira mahdollistaa Notionia tarkemman ajallisen seurannan. Jirasta löytyy roadmap -ominaisuus, josta nähdään epicit aikajanassa. Tämän avulla voidaan verrata niitä toisiin epiceihin ja niiden sijoittumisajankohtiin. Jirassa on mahdollisuus laajentaa epiciä, jolloin kaikki siihen liittyvät työtehtävät tulee näkyville. Tässä tarkemmassa roadmapissa on mahdollisuus tarkkailla koko projektin aikajanaa ja epiceille pystytään luomaan linkityksiä toistensa välille. Tämän avulla pystytään seuraamaan sitä, mitkä epicit tulee tehdä ennen seuraavia. Kuviosta 4 nähdään minkälainen näkymä Jirassa on roadmap -ominaisuudelle. Jiran roadmap-ominaisuuden avulla pystytään seuraamaan useiden tiimien työskentelyä samanaikaisesti. Roadmap on siis suuren skaalan kuva koko projektin näkymästä, jolla pystytään seurata helposti ja nopeasti koko projektin etenemistä. Roadmapista löytyy lisä pilkkomismahdollisuuksia (teemat ja initiativet) jotka auttavat organisaatiota tarkkailemaan etenemistä laajemmalla tasolla. Tarkoituksena ei ole saada pientä nippelitietoa, jonka saa kanban- tai scrum-taulu näkymässä, vaan havainnollistaa laajempaa kuvaa.



Kuvio 4. Visualisointi Jiran roadmap-ominaisuudesta (Roadmaps in Jira Software n.d., muokattu)

Notionista löytyy kalenteri välilehti, jonka avulla voidaan seurata eri asioita aikajanallisesti. Kalenteri ei ole yhtä kattava Jiran roadmapiin verrattuna, mutta sillä saadaan perusnäkymä siihen, mitä on tekeillä. Oleellista on seurata mitä tehtäviä on samanaikaisesti työn alla, jossa kalenteria voi hyödyntää. Kalenterin avulla hahmotetaan laaja-alaisesti tekeillä olevat asiat, jolloin nähdään

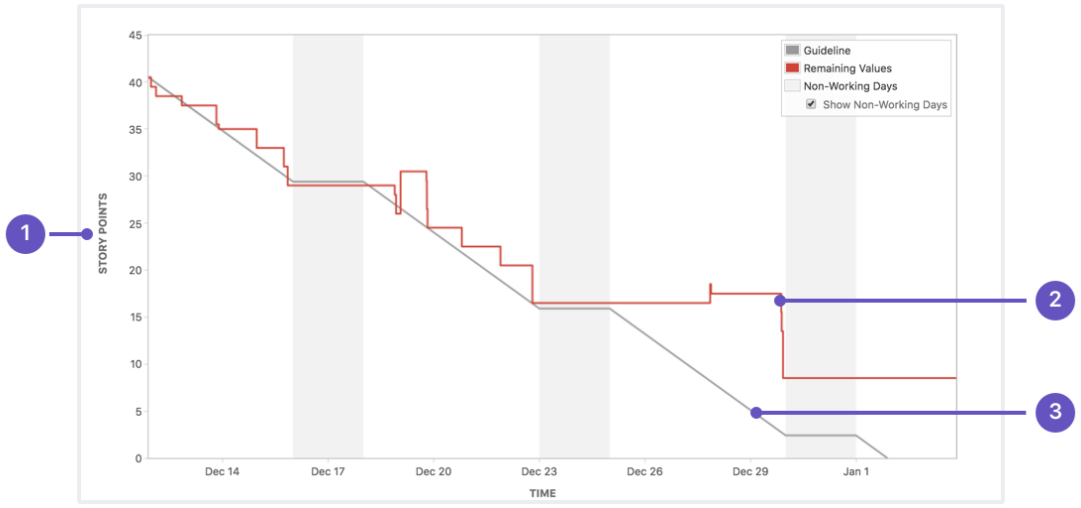
kuinka paljon resursseja, on vielä käytettävissä. Notionin kalenteri on hyvin yksinkertainen näkymä, joka on tarkoitettu enemmän normaalin kalenterin käyttöön ennemmin, kuin projektinhallinnallisien työtehtävien seurantaan. Jos kalenterinäkymään lisää liikaa asioita, tulee näkymästä hankalaselkoinen ja sekava. Tämän myötä useiden storyjen samanaikainen seuranta Notionin kalenteritoiminnolla voi olla hyvin haastavaa. Kalenteri soveltuu esimerkiksi sprinttien ja muiden tärkeiden päivämäärien seuraamiseen. Notionin kalenterin hyötynä on se, että sen käyttötapaa voi muokata omiin käyttötarpeisiin, mutta tämä vaatii ylimääräistä ajankäyttöä saadakseen niistä omalle käytölle sopivia.

Notionin kalenteria voidaan hyödyntää myös muihin tarpeisiin kuin tehtävien ja sprinttien seurantaan. Kalenteria voidaan hyödyntää myös koko tiimin kalenterina, jolloin nähdään missä kukin menee ja mitä tapaamisia on tulossa. Julkaisun aikataulun määrittelyssä kalenterin käytöstä on hyötyä, kun suunnitellaan tiimin prosesseja.

Notionin kalenterista löytyy samanlaisia suodattimia, kuin taulukosta. Näiden avulla voidaan määrittää, minkälaista dataa kalenterimerkinnöissä nähdään ja kenelle ne kuuluvat. Kalentereita voidaan luoda valmiiksi ja niitä voidaan määrittää useampi. Näin voidaan tehdä oma kalenterinsa, vaikka epicien tai sprinttien seurannalle helposti ilman erillisiä suodattimien muokkailuja. Kun ollaan määritelty useampi kalenteri eri käyttötarkoituksille, nähdään vaivattomammin tietyt asiat yhdellä kalenterilla verrattuna siihen, että kaikki olisivat samassa sekavassa näkymässä.

Burndown chart

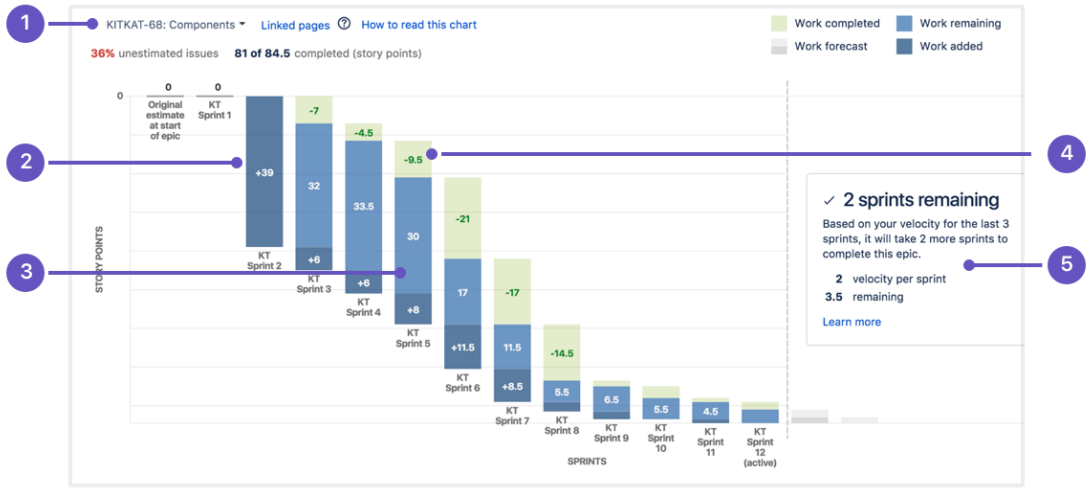
Burndown chart on graafinen kaavio, jolla kuvataan työn etenemistä. Tämä ominaisuus löytyy Jirasta, mutta Notionissa tätä ei ole ollenkaan. Burndown chartissa kuvaaja laskee sprintin ajalle kerättyjen story pointtien määrän ja tekee siitä ajallisen viivan kuvaamaan sprintin etenemistä. Story pointeilla tarkoitetaan työn kuormittavuutta, joka usein mitataan työtunteina (Rehkopf n.d.e). Ajan kuluessa ja storyjen valmistuessa burndown chartilla saadaan viivamuodossa havainnollistettua kaavioon myös töiden todellinen toteuma sprintin ajalle. Kuviosta 5 löytyy esimerkkikuva burndown chartista, jossa nähdään vasemmalla story pointtien määrä, harmaalla arvioitu aikajana ja punaisella reaaliaikainen työn toteuma.



Kuvio 5. Visualisointi Jiran burndown chartista (Rehkopf n.d.e.).

Jirasta löytyy myöskin epic burndown chart jonka avulla voi seurata epicin valmistumista. Tämä kaavio toimii samalla periaatteella kuin normaali burndown chart, mutta näkymä on pylväsdiagrammin muodossa. Normaalisti burndown charteilla mitataan sprintin etenemistä ja sitä, kuinka työ saadaan tehtyä sprintin aikana. Epic burndown chartissa seurataan ja arvioidaan kuinka monta sprinttiä epicin suorittamiseen menee ja mitä tapahtuu jokaisen sprintin aikana. (Rehkopf n.d.e.)

Epic burndown chartissa jokainen sprintti näkyy taulukossa omana pylväänä, josta näkee kuinka paljon töitä, on saatu tehtyä ja onko uusia tehtäviä lisätty sprintin aikana. Kuviosta 6 havaitaan, että tummansiniset palkit ovat sprintin aikana uusia lisättyjä story pointteja. Tämä tarkoittaa sitä, että sprintin aikana lisätyt story pointit tulevat näkyviin epic burndown charttiin tällä värillä. Edellä mainitun kuvion vaaleansininen palkki kertoo epicissä jäljellä olevien story pointtien määrän. Vihreä palkki puolestaan näyttää sen, miten paljon sprintin aikana on saatu tehtyä töitä. Epic burndown chart arvioi tiimin edellisten sprinttien tuotantonopeuden perusteella, kuinka paljon saadaan tulevien sprinttien aikana valmiiksi. Tämä arvio näkyy edellä mainitussa kuviossa harmaana värinä. Epic burndown chart arvioi myös sitä, kuinka monta sprinttiä epicin valmiiksi saamisessa tulee vielä menemään.



Kuvio 6. Esimerkkikuva Jiran epic burndown chartin näkymästä(Rehkopf n.d.e.).

Normaalin sprintin burndown chart auttaa tiimiä seuraamaan omaa työtään ja sen avulla tarvittavat henkilöt saavat kuvan siitä, kuinka paljon töitä saadaan sprintin aikaan tehtyä. Projektinhallinnasta vastaavien henkilöiden kannalta epic burndown chart on tärkeämpi työkalu kuin normaali burndown chart sillä sen avulla pääsee näkemään useampien sprinttien kokonaisuuden. Tämän avulla saadaan parempi kuva siitä, mitä pidemmällä kuin yhden sprintin kestoisella aikavälillä saadaan tehtyä. Notion:n heikkoutena on näiden burndown charttien puuttuminen mikä vaikeuttaa työn tehokkuuden ja ajankäytön arvioimista ja näiden osa-alueiden kehittämistä, kun ei saada visuaalista dataa aiheesta. Notionin ongelmana on myös se, ettei siinä ole valmiina olemassa story pointtien kaltaista työn kuormittavuuden arviointijärjestelmää, mutta tähän voi itse Notionin merkinnöillä tehdä tuntimääräiset arviot jokaiselle tehtävälle. Tämä tapa ei kuitenkaan anna työlle seurantamahdollisuuksia, kun burndown chart ominaisuus puuttuu.

Versiohallinta

Jirassa on Notionista puuttuva pelikehitykselle hyödyllinen lisäelementti, jonka avulla voi tarkastella ominaisuuksien seurantaa. Jirassa voi luoda eri versioiden julkaisuille omat versiot, joihin voi liittää kaikki työtehtävät. Tämän avulla päästään tarkastelemaan esimerkiksi mitä kukin versio pitää sisällään ja mitä ominaisuuksia ollaan jo tekemässä tulevia versioita varten. Tämän ominaisuuden avulla tehtävät osataan priorisoida paremmin, kun tiedetään jo valmiiksi mihin versioon mitäkin ominaisuutta tullaan tarvitsemaan. Jos tarkastelua ei ole käytössä, saatetaan työ aloittaa liian aikaisin, kun ei ole tietoa siitä, mihin versioon se on suunniteltu. Yksittäinen ominaisuus saattaa

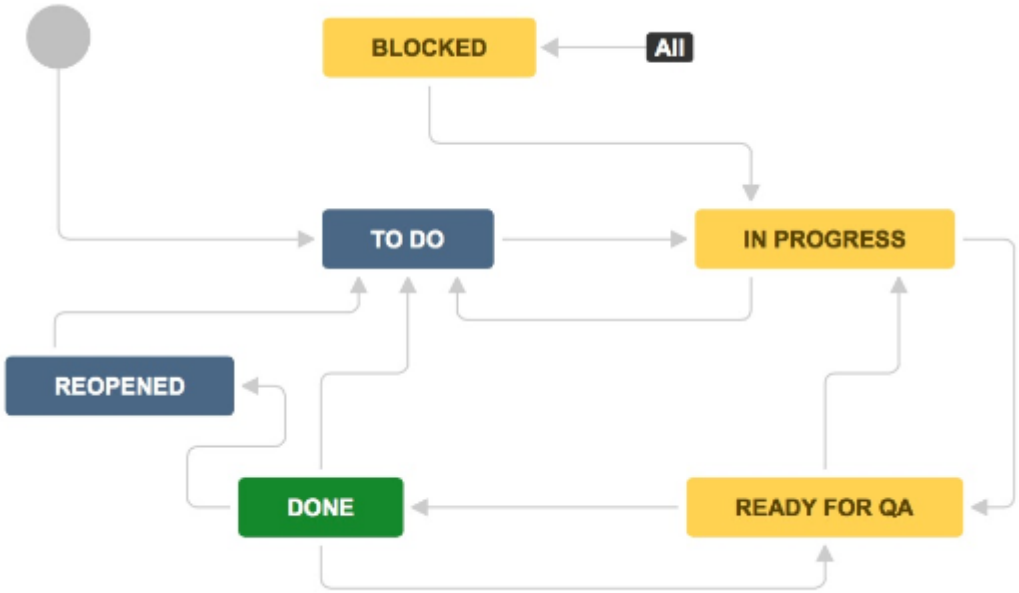
vaatia sen, että jokin toinen ominaisuus on ennen sitä valmis ja ilman tätä tarkastelua saatetaan tehdä turhaa työtä.

Notionissa ei ole valmiiksi integroitua versionhallintasysteemiä, mutta siihen voidaan muokata omat versionhallinta tagit, jotka merkataan aina kyseisiin epiceihin ja storyihin. Tämän myötä niitä voidaan katsastella sivulla, jonne on määritelty näkyväksi versionhallinta tagien mukaisesti kyseiset tehtävät. Tämä ei ole yhtä kattava kuten jo Jirassa valmiiksi oleva ominaisuus, mutta Notionissa pystyy pienellä lisätyöllä tekemään omaa versionhallintaa.

Storyboard

Jira mahdollistaa erillisien taulukoiden luomisen scrumille ja kanbanille. Tällöin ne ovat konfiguroitu juuri sille sopivaan toimintamalliin. Scrum taulukko on suunniteltu sisältämään sprintit ja erillisen product- ja sprint backlogin. Notionin taululla nämä ominaisuudet tulee itse tehdä scrumille ja kanbanille sopiviksi.

Jirassa on mahdollista määritellä storyille tarkka kulkurakenne siihen, kuinka niitä voidaan taululla liikuttaa. Tämä lisää järjestelmällisyyttä eri vaiheiden välillä. Esimerkiksi kuviossa 7 havaitaan kuinka progress-kohdassa oleva story voidaan siirtää takaisin to do -sarakkeeseen tai testaukseen. Jirassa voidaan määritellä, ettei esimerkiksi in progress-kohdasta voi hypätä suoraan done -kohtaan, vaan storyjen tulee läpikäydä määritellyn rakenteen välivaiheet, jotta se voidaan todeta valmiiksi. Rakenteen avulla storyja ei voi vahingossa siirtää vääriin sarakkeisiin, vaan ainoastaan ennalta määriteltyihin paikkoihin. Liiallisten taulujen luominen hankaloittaa työntekijöiden tuotantoprosessia, joten taulut tulisi pitää mahdollisimman yksinkertaisina, mutta niiden pitäisi kuitenkin sisältää kaikki tarvittavat osiot.



Kuvio 7. Story workflow:n rakenne Jirassa (An easy way for Scrum teams to adopt a best practice Jira workflow n.d.)

Notionista löytyy storyboard näkymä, jolla voidaan toteuttaa ketterän kehitystavan työnseurantaa kuten scrumia ja kanbania. Kortteja voidaan suodattaa eri tietojen mukaan ja seurata niitä näiden pohjalta. Group by -valikon kautta voidaan pinota ja tarkkailla esimerkiksi työntekijöihin liitettyjä kortteja, jolloin nähdään selkeästi mikä kortti kuuluu kenellekin. Tällä tavalla voidaan myös seurata miten työt jakautuvat eri roolien kesken kuten artististien ja ohjelmoijien välillä, kun nähdään suoraan, miten paljon millekin roolille on töitä. Properties -valikosta pystytään muokkaamaan korteissa näkyvän tiedon määrää, jolloin visuaalisesti nähdään nopeasti ja helposti juuri ne tiedot mitkä ovat tiimille taikka jollekin muulle tarkastelijalle oleellisia.

Notionissa on mahdollista luoda useita eri taulukkoja valmiiksi valikkoon, jossa on heti tarvittava tieto saatavilla. Tällöin ei tarvitse aina alkaa erikseen suodattamaan tietynlaista valikkoa vaan on voitu valmiiksi tehdä jokaiselle näkymälle oma valmis taulu, josta päästään tarkastelemaan tietoja. Tällaisia voivat olla esimerkiksi edellä mainitut roolien ja työn tekijöiden valikot. Haastattelussa ilmenneeseen bugien seurantaan voitaisiin esimerkiksi luoda oma taulukko, jossa on bugien korjaamiselle sopivat tarkasteluelementit käytössä.

Molemmissa sovelluksissa on mahdollista luoda tageja, joiden avulla pystytään luokittelemaan kortteja tarkemmin omien tarpeidensa mukaan. Erilaiset tehtävät voidaan merkata eri tavalla niin,

että esimerkiksi art ja ohjelmointi -kortit pystytään erottamaan pikaisella vilkaisulla. Näiden tagien avulla pystytään esimerkiksi luomaan Notioniin omat näkymät tietyille työtehtäville. Jokaisen osa-alueen lead pystyy tarkastelemaan oman alueensa työtehtäviä helposti näkymän avulla. Aiemmin esille noussut versionhallinta Notionissa voidaan toteuttaa tällä merkintätavalla.

Yhteenveto

Jira on projektinhallinnallisilta ominaisuuksilta selkeästi laajempi sovellus pelinkehitykseen verrattuna Notioniin. Taulukosta 1 havaitaan sovellusten sisältämien ominaisuuksien kattavuus. Jirasta löytyy paljon laajemmin ominaisuuksia useisiin eri käyttötarkoituksiin. Notion puolestaan jää vajaaksi valmiiksi integroitujen ominaisuuksien puolesta. Monia näitä ominaisuuksia pystytäänkin Notionin kustomoitavuuden avulla tekemään itse, mutta ne eivät yllä samalle tasolle Jiran vastaavien ominaisuuksien kanssa. Taulukosta voidaan päätellä, että Jira on huomattavasti kattavampi projektinhallintasovellus.

Taulukko 1. Yhteenveto sovellusten välisistä eroista

Ominaisuus	Jira	Notion
Scrum -taulu	x	
Kanban -taulu	x	x
Korttien tarkempi muokkaus	x	x
Task	x	x
User story	x	x
Epic	x	x
Initiative	x	
Teema	x	
Raportointi	x	
Tagit	x	x
Kalenteri	x	x
Versionhallinta	x	
Storyworkflow	x	
Tiimienvälinen työskentely	x	
Kustomoitavuus		x
Bugienhallinta	x	x

6 Johtopäätökset

Haastattelusta saatujen tietojen avulla voidaan verrata Jiran ja Notionin välisiä eroja hyvin laajasti. Notionin käyttöön ja tuotantoon liittyvät haasteet johtuvat siitä, ettei Notionia ole suoranaisesti tarkoitettu sovelluskehityksen projektinhallintaan. Suurin osa esille tulleista ongelmista olisi korjattavissa vaihtamalla Notion Jiraan, sillä sen ominaisuudet tukevat suoraan pelinkehitystä.

Pienessä tiimissä Notion toimii hyvin. Ominaisuudet, jotka siitä puuttuvat pystytään helposti käsittelemään ja kommunikoimaan tiimin kesken, tai hyödyntämään jotain toista sovellusta apuna. Suuremmassa organisaatiossa projektinhallintasovelluksena kannattaisi olla suoraan sovelluskehitykseen tarkoitettu ohjelma, joka tukee ketterän kehityksen menetelmiä. Valmiiksi integroidut ketterät toiminnot ehkä puuttuvatkin Notionista. Vaikka Notionissa on puutteita, voidaan ne lisätyöllä itse rakentaa Notionin kustomoitavuuden ansiosta. Pienen tiimin etuna on joustavuus työnteossa, jonka myötä se ei välttämättä edes kaipaakaan kaikkia Jirasta löytyviä ominaisuuksia, jotka ovat tarkoitettu enemmän isojen tiimien käyttöön.

Tiimin koon kasvaessa tarkka dokumentointi on tärkeämpää kuin aikaisemmin. Jotta jokainen tiimin jäsen pysyy kärryillä menevistä asioista, on laajempi dokumentaatio onnistumisen kannalta välttämätöntä. Kaikki olemassa oleva tieto tulee laittaa ylös, sillä suun kautta liikkuva tieto ei välttämättä kulkeudu kaikille ja voi näin aiheuttaa väärinymmärryksiä. Jos on esimerkiksi taski johon ei ole tehty tarvittavia tarkennuksia, voi työntekijä tehdä tietämättään vääriä asioita. Jirassa on Notioniin verrattuna laajempia tarkkailutyökaluja projektinhallintaa varten ja sen avulla pystytään tarkkailemaan, vaikka koko organisaation eri projekteja samanaikaisesti. Projektinhallintasovelluksen tulee pystyä skaalautumaan tiimin koon mukaan ja Jirassa on paremmat mahdollisuudet tälle.

User storyt olisi hyvä lisä toimeksiantajan projektinhallinnan prosesseihin tulevaisuudessa. Organisaation ja tiimin kasvaessa user storyjen avulla saataisiin paremmin selvää siitä, miten ominaisuuden tulisi toimia käyttäjätasolla, jotta jokainen tietäisi mitä ajetaan takaa. Yksinkertaiset taskit voidaan ymmärtää eri tavalla, jos toimivuutta ei ole kuvattu ja tämä voi helposti aiheuttaa väärinymmärryksiä. User storyt voidaan pilkkoa pienempiin taskeihin, mutta taustalla olisi kuitenkin user story josta saadaan tarkempi kuva kokonaisuudesta. Epicien tulisi myöskin olla tarinamuotoisia, jotta vältyttäisiin siltä, että epic kuvaisi yhdellä sanalla isoa kokonaisuutta. Opinnäytetyössä ilmi tullut initiative voisi olla hyvä lisäkäsite, joka kuvaisi isoa kokonaisuutta yksinkertaisessa muodossa.

Tulevaisuudessa tiimin kasvaessa Notionin sisäisen kalenterin käyttö voi olla järkevämpää. Notionin oman kalenterin avulla kaikki tiimin jäsenet pääsevät helpommin kalenteriin käsiksi ja se on helposti saatavilla suoraan sovelluksessa. Tällöin myöskin taskit ja muut merkinnät päivittyisivät suoraan kalenteriin. Pienen tiimin etuna on helppo ja nopea kommunikaatio, mutta isommalla tiimillä Notionin kalenterista olisi varmasti hyötyä. Kalenteri ei ole suoraan kattava verrattuna Jiran roadmap -ominaisuuteen, mutta kalenterin avulla pystytään muokkaamaan tärkeimmät asiat näkyville.

Suuren ja pienen organisaation projektinhallinnallisissa tarpeissa on eroja. Pienen tiimin sisällä tieto liikkuu nopeasti ja asiat pystytään toteuttamaan todella ketterästi. Isompi tiimi ja organisaatio tarvitsee enemmän tasoja ja hierarkkista järjestystä, jotta ylin johto saa helposti selville missä mikäkin projekti on menossa juuri kyseisellä hetkellä. Hyvin tarkka dokumentointi on myöskin tärkeää, jotta jokainen tiimin jäsen pysyy selvillä meneillään olevista työtehtävistä.

Epicien käyttö on hyödyllistä aina huolimatta organisaation koosta. Epicit antavat tiimeille ja organisaatioille pienempiä maaleja, joita kohti projekteja työstetään. Tällöin epicit pilkkovat projektin pienempiin osiin, joita voidaan arvioida ajallisesti ja jotka saadaan valmiiksi suunnitellussa ajassa. Tämä edesauttaa projektin hallitumpaa etenemistä. Isommassa organisaatiossa voidaan ottaa käyttöön myös initiativet ja teemat, jotka tuovat lisää hallintatasoja koko organisaation laajuisiin toimintoihin.

7 Pohdinta

Opinnäytetyön tavoitteena oli selvittää toimeksiantajan tuotantoprosessin vaiheita sekä verrata Jiran ja Notionin eroja. Tämän lisäksi selvitettiin minkälaisia eroja pienen ja suuren organisaation projektinhallinnallisissa tarpeissa on. Opinnäytetyö onnistui selvittämään tutkimusasetelmassa määriteltävät asiat. Opinnäytetyön haasteena oli turhan laaja aihealue, jota jouduttiin runsaasti tarkentamaan. Liian laaja opinnäytetyön laajuus aiheutti epäselvyyttä siitä, mikä oli tutkimuksen pääasiallinen tarkoitus. Toimeksiantajayrityksessä tapahtuneiden muutoksien myötä opinnäytetyön ohjaaja vaihtui kesken projektin ja tämä toi lisähaastetta tutkimuksen toteuttamiseen. Tapahtuneiden muutoksien myötä opinnäytetyön tarkoitus ja aihe vaihtoivat muotoaan alkuperäisestä suunnitelmasta, joka toi lisää epäselvyyttä tutkittavasta asiasta.

Opinnäytetyön tuloksia voidaan hyödyntää kasvavien peliyrityksien projektinhallintasovelluksien valinnassa. Työ antaa myöskin ohjeita ketterän kehityksen toteuttamiseen. Samoja periaatteita voidaan soveltaa myöskin sovelluskehityksen parissa tehtävään projektinhallintaan. Toimeksiantajalle tämä opinnäytetyö antaa lisätietoa heidän tuotantoprosessistaan ja tietoja siitä, mitä prosesseissa tulisi kehittää.

7.1 Luotettavuustarkastelu

Opinnäytetyön dokumentoinnin tulee olla riittävää, jotta työn luotettavuutta voidaan arvioida. Työn uskottavuutta lisää harkiten tehdyt valinnat ja niiden perustelut. (Kananen 2015, 112) Työn vahvistettavuuden varmistaminen on yksi tärkeä osa opinnäytetyön luotettavuutta. Vahvistettavuutta voidaan varmistaa esimerkiksi luetuttamalla tutkimusta tietolähteenä olleella henkilöllä (Kananen 2015, 113). Työn vahvistettavuuden lisäämiseksi työ luetutetaan toimeksiantajayrityksen edustajalla, joka vahvistaa tulkinnan oikeaksi.

Lähteistö on hyvin laajaa ja tarkoin valikoitua. Tiedon oikeellisuutta on tarkkailtu useiden eri lähteiden perusteella. Lähteinä on käytetty tutkittavien sovellusten omia dokumentaatioita sekä muiden tahojen dokumentteja ketterästä kehityksestä. Suuri osa lähteistä ei ole suoranaisesti pelialalta, vaan ne ovat sovelluskehityksen näkökannalta, mutta näitä samoja periaatteita pystytään soveltamaan peliteollisuudessa. Muutama lähde on melko vanha, mutta tietoa on luettu useasta eri paikasta ja osan vanhemmista lähteistä on koettu kuvaavan tarkemmin tutkittavaa asiaa, jonka perusteella lähdevalinta on tehty.

Sovellusten toimintoja ja ominaisuuksia on verrattu ja kartoitettu sovelluksia testaamalla, havainnoimalla ja niiden dokumentaatioita tutkimalla. Sovellusten ominaisuuksien tarkkailuun käytetään myös teemahaastattelussa kerättyä tietoa toimeksiantajayrityksen nykyisistä toimintamalleista.

Haastatteluun on valittu henkilö(t) sillä perusteella, että he ovat keskeisessä roolissa tuotannon suhteen, jolloin he osaavat kertoa koko organisaation ja tiimien tilasta ja tarpeista. Teemahaastattelu valittiin aineistonkeruumenetelmäksi, sillä sen avulla saadaan luotettavaa tietoa suoraan toimeksiantajalta heidän tämänhetkisistä prosesseistansa.

7.2 Kehittämisehdotukset

Tässä opinnäytetyössä keskityttiin enemmän sovellusten vertailuun ja ominaisuuksien pilkkomiseen. Jatkon kannalta olisi oleellista tutkia yksittäisiä ominaisuuksia tarkemmin, jotta saataisiin syvällisempää tietoa aihealueesta. Pienemmällä aiherajauksella voitaisiin toteuttaa siis syvällisempää tutkimusta projektinhallinnan piirteistä.

Jos tutkittavana on yksittäinen ominaisuus, voitaisiin sitä kehittää ja demota oikeassa tuotannossakin. Tämän opinnäytetyön aikana saatiin vain kehitysideoita, joita voitaisiin testata, mutta tulevaisuuden kannalta olisi hyvä päästä myöskin tutkimaan kehitysideoiden toimintaa aidossa ympäristössä.

Tutkittavana oli vain kaksi sovellusta, jos tutkimuksesta jätettäisiin tuotantoprosessien kehittämisnäkökulma pois, voitaisiin ottaa useita eri sovelluksia vertailuun ja saada kattavampaa kuvaa siitä, minkälaisia projektinhallinnallisia mahdollisuuksia eri sovelluksissa on. Tällaisella tutkimuksella voitaisiin löytää mahdollisuuksia useisiin erilaisiin käyttötarkoituksiin.

Lähteet

12 Principles Behind the Agile Manifesto. N.d. Artikkeli Agile Alliancen verkkosivuilla. Viitattu 1.11.2021. https://www.agilealliance.org/agile101/12-principles-behind-the-agile-manifesto/

A brief overview of Jira. N.d. Opas Atlassian:n verkkosivuilla. Viitattu 12.11.2021. https://www.atlassian.com/software/jira/guides/getting-started/overview#jira-software-hosting-options

An easy way for Scrum teams to adopt a best practice Jira workflow. N.d. Artikkeli Atlassian Marketplace:n verkkosivuilla. Viitattu 13.3.2021. https://marketplace.atlassian.com/apps/1213278/easy-agile-scrum-workflow-for-jira?tab=overview&hosting=server

Bister, T. Tietojenkäsittelyn opinnäytetyö – Viittoja ja karttoja tutkimisen ja kehittämisen teille. 2019. Jyväskylän ammattikorkeakoulun julkaisuja -sarja.

D'Alessio, F. 2018. The Beginner's Guide to Notion. Artikkeli Keep Productive:n verkkosivuilla. Viitattu 12.11.2021. https://www.keepproductive.com/blog/notion-for-beginners

Drumond, C. N.d. Agile project management. Artikkeli Atlassiansin sivuilla. Viitattu: 23.11.2020. https://www.atlassian.com/agile/project-management

Kananen, J. Kehittämistutkimuksen kirjoittamisen opas – miten kirjoitan kehittämistutkimuksen vaihe vaiheelta. 2015. Jyväskylän ammattikorkeakoulun julkaisuja 212.

Keith, C. 2010. Agile game development with scrum. Boston: Pearson Education inc. E-kirja. http://index-of.co.uk/Agile/Agile%20Game%20Development%20With%20Scrum.pdf

Lynch, W. 2019. Scrum: INVEST in Good Stories by Achieving SMART Tasks. Kirjoitus Warren Lynch:n blogissa. Viitattu 8.11.2021. https://warren2lynch.medium.com/scrum-invest-in-good-stories-and-smart-tasks-63f8432f7840

PEARL IV: INVEST in good User Stories and SMART Tasks. 2014. Kirjoitus Pearls of Wisdom-blogissa. Viitattu 11.8.2020. https://agilepearls.wordpress.com/2014/05/09/pearl-xv-invest-in-good-user-stories-and-smart-tasks/

Radigan, D. 2014. Announcing Jira Portfolio: View, plan, & manage initiatives. Uutinen Atlassianin verkkosivuilla. Viitattu 22.5.2020. https://www.atlassian.com/blog/2014/09/announcing-jira-portfolio-view-plan-manage-initiatives/amp

Rehkopf, M. Agile epics: definition, examples, and templates. N.d.d. Artikkeli Atlassian:n verkkosivuilla. 22.5.2020. https://www.atlassian.com/agile/project-management/epics

Rehkopf, M. Kanban vs. scrum: which agile are you? Artikkeli Atlassian:n verkkosivuilla. Viitattu 23.8.2020. https://www.atlassian.com/agile/kanban/kanban-vs-scrum

Rehkopf, M. Learn how to use burndown charts in Jira Software. N.d.e. Artikkeli Atlassian:n verkkosivuilla. Viitattu 23.11.2020. https://www.atlassian.com/agile/tutorials/burndown-charts

Rehkopf, M. Stories, epics, and initiatives. N.d.c. Artikkeli Atlassian:n verkkosivuilla. Viitattu 12.6.2020. https://www.atlassian.com/agile/project-management/epics-stories-themes

Rehkopf, M. What is kanban board? N.d.a. Artikkeli Atlassian:n verkkosivuilla. Viitattu 23.8.2020. https://www.atlassian.com/agile/kanban/boards

Rehkopf, M. What is Kanban card? N.d.b. Artikkeli Atlassian:n verkkosivuilla. Viitattu 23.8.2020. https://www.atlassian.com/agile/kanban/cards

Roadmaps in Jira Software. N.d. Opas Atlassian:n verkkosivuilla. Viitattu 5.11.2021. https://www.atlassian.com/software/jira/guides/roadmaps/basic-roadmaps#what-is-a-roadmap

Schwaber, K., Sutherland, J. 2020. The 2020 Scrum Guide. Opas Scrum Guides:n verkkosivuilla. Viitattu 5.8.2020. https://scrumguides.org/scrum-guide.html

Smith, A. 2015. Jira Portfolio: the fundamentals. Artikkeli Atlassianin verkkosivuilla. Viitattu 12.1.2020. https://www.atlassian.com/blog/jira-software/jira-portfolio-fundamentals?_ga=2.142559730.1500559380.1582518553-792419064.1577401910

Task. N.d. Agile sanakirja Agile academyn verkkosivuilla. Viitattu 5.11.2021. https://www.agile-academy.com/en/agile-dictionary/task/

The Agile Manifesto. N.d. Artikkeli Agile Alliancen verkkosivuilla. Viitattu 1.11.2021. https://www.agilealliance.org/agile101/the-agile-manifesto/

Wake, B. 2003. INVEST in Good Stories, and SMART Tasks. Artikkelib XP123:n verkkosivuilla. Viitattu 11.8.2020. https://xp123.com/articles/invest-in-good-stories-and-smart-tasks/

West, D. Scrum roles and the truth about job titles in scrum. N.d. Artikkeli Atlassian:n verkkosivuilla. Viitattu 16.10.2021. https://www.atlassian.com/agile/scrum/roles

What is User Story. N.d. Artikkeli Visual Paradigm:n verkkosivuilla. Viitattu 21.5.2020. https://www.visual-paradigm.com/guide/agile-software-development/what-is-user-story/

Liitteet

Liite 1

Haastattelukysymykset

- Minkälaisista osista user storyn muodostamisprosessinne koostuu?
- Minkälaisiin osiin pilkotte ominaisuudet ja muut työtehtävät?
- Minkälaiset ohjelman ominaisuudet ovat tuotantoprosessin kannalta tärkeitä?
- Minkälaisia ongelmakohtia pyritään ratkaisemaan?
- Mitä vaiheita tuotantoprosessinne sisältää?
- Miten tarkastelette prosessin etenemisiä?
- Mitkä ominaisuudet ovat tärkeitä eri tiimihenkilöiden näkökulmasta?
- Miten eri tiimin jäsenet otetaan huomioon prosessissa?
- Minkälaiset ominaisuudet ja prosessit olisivat tarpeellisia organisaation kasvaessa?