Integraatioliittymien kehitys ERP-projekteissa

Aleksi Mutanen

2021 Laurea

Laurea-ammattikorkeakoulu

Integraatioliittymien kehitys ERP-projekteissa

Aleksi Mutanen

Tietojenkäsittely

Opinnäytetyö

Toukokuu, 2021

Laurea-ammattikorkeakoulu **Tiivistelmä**

Tietojenkäsittely

Tradenomi (AMK)

Aleksi Mutanen

Integraatioliittymien kehitys ERP-projekteissa

Vuosi 2021 Sivumäärä 42

Opinnäytetyön tavoitteena oli seurata ja dokumentoida aloittelevan ohjelmistokehittäjän työtä ja kehitystä Microsoftin Dynamics 365 ERP -projekteissa. Opinnäytetyön tarkoituksena oli kehittää tekijän ymmärrystä ja osaamista tietojenkäsittelyn alan töistä. Työn toimeksiantajana oli eCraft Oy Ab ja työn aikana oli tarkoitus saada juuri koulutusohjelmasta valmistuneen työharjoittelijan taitotasoa ja kehittymistä selville tulevia koulutuksia varten.

Opinnäytetyö on tehty päiväkirjamallisena ja siinä kuvataan työssä keskeiset toimintaympäristöt, teknologiat ja työskentelytavat. Tietoperustana on Microsoftin ja Atlassianin tuotteiden dokumentaatio sekä muiden projekteissa tarvittavien teknologioiden dokumentaatiot. eCraftin ohjelmistoprojekteissa on määritelty käytettävät teknologiat ja työskentelytavat ja niitä noudatetaan myös opinnäytetyön aikana. Opinnäytetyön tuloksena on seurantajakson dokumentointi aloittelevan ohjelmistokehittäjän työstä ja kehittymisestä sekä kaikki keskeiset työssä käytettävät teknologiat ja miten niiden käyttämisessä on kehitytty. Toimeksiantajayritys voi lukea opinnäytetyön ja saada ideoita toimintaprosessien parantamiseksi. Lisäksi yrityksen tulevissa koulutusohjelmissa opiskelevat oppilaat voivat lukea opinnäytetyön ja saada parempaa kuvaa tulevasta työstä.

Asiasanat: Azure, integraatiot, Dynamics 365

Laurea University of Applied Sciences

Degree Programme in Business Information Technology

Bachelor of Business Administration

Aleksi Mutanen

Integrations Development in ERP Projects

Year 2021 Pages 42

Abstract

The goal of this Bachelor's thesis was to track and documentate the job and improvement of a beginner employee in Microsoft Dynamics 365 ERP projects. The purpose of the thesis was to improve the knowledge and know-how of the author in information technology business. The employer of this thesis was eCraft Ltd. The aim of this thesis was to find out the level of expertise and improvement of a recently graduated intern from eCraft's training degree to improve their future trainings.

The thesis was made as a diary model and it tells the crucial environments, technologies and working conventions of the job. Knowledge base to do this is from Microsoft's, Atlassian's and other system's required technologies' documentation. In eCraft's software projects there are defined technologies and working conventions and they are strictly followed during the thesis. The result of the thesis is a documentation of the tracking period from beginner Software Developer's improvement and how the developer has improved in using the crucial technologies required to do the job. The employer can read the thesis and get ideas on how to improve the company's workflow. Also in the company's future training programs the students can read the thesis to find out what to expect in software projects.

Keywords: Azure, integrations, Dynamics 365

Sisällys

1 Johdanto

Tämän päiväkirjamuotoisen opinnäytetyön tarkoituksena on seurata aloittelevan integraatiokehittäjän työskentelyä 10 viikon mittaisella aikajaksolla. Seuranta alkaa kolmen kuukauden työharjoitteluajan jälkeen, joista kaksi ensimmäistä oli työnantajan koulutusohjelmassa kouluttautumista. Harjoittelujakso jatkuu vielä koko seurannan ajan. Seurannan aikana viikon tulevat tapahtumat suunnitellaan ja asetetaan tavoitteet omaan kehittymiseen integraatiokehittäjänä sekä kartoitetaan tarvittava tietämys viikon aikana tuleviin tehtäviin. Päivittäin tehtävissä merkinnöissä kerrotaan työpäivän aikana tehdyistä tehtävistä ja niihin liittyvien ongelmien ratkaisuista sekä tarkastellaan oman osaamisen kehittymistä.

Työn toimeksiantajana toimii eCraft Oy Ab, jonka työntekijänä suoritan kehitystyötä useammassa eri asiakasprojektissa. eCraft kuuluu kansainväliseen Fellowmind-konserniin. Yrityksen ydintoimintaan kuuluu mm. Microsoftin Dynamics 365–tuoteperhe. Yritys on jaettu eri liiketoimintalinjoihin ja Modern Work & Cloud Infrastructure -linjan alaisuudessa toimii Integrations & API -liiketoimintayksikkö, johon kuulun itse yhdeksän muun työntekijän lisäksi.

2 Integraatiokehityksen keskeiset ympäristöt ja teknologiat

Työni menestyksekäs hoitaminen edellyttää integraatiokehitykseen liittyvien teknologioiden ja ympäristöjen tuntemista. Toimintaympäristöissä toimiminen vaatii työssä käytettävien käyttäjätilien pääsyn ympäristöjen resursseihin ja käytettäviin komponentteihin. Tietoturvasyistä jokaiselle käyttäjätilille ei anneta pääsyä järjestelmiin, vaan yksittäisille käyttäjätileille annetaan lupa tarpeen vaatiessa. Uuteen asiakasympäristöön saapuessa usein pyydetään projektin vanhemmalta työntekijältä pääsy esimerkiksi Azure-tilaukseen, jotta kyseisen projektin sisällä oleviin komponentteihin pääsee käsiksi.

2.1 Käsitteet

Azure

Azure on Microsoftin luoma pilvipalveluympäristö, jossa on saatavilla monia työssäni käytettäviä komponentteja ja ympäristöjä. Sen tuotevalikoimiin kuuluu esimerkiksi virtuaalipalvelimet, kehitysalustat ja tietokannat (Microsoft 2019a). Azure-portaalin ja sen sisällä olevien toiminnallisuuksien tunteminen ja osaaminen on tärkeä osa työtä.

Azuren sisällä on oma hierarkinen mallinsa käyttäjien ja resurssien hallintaan. Yrityksellä voi olla useampi tilaus, jonka sisällä on lisenssejä, joita annetaan käyttäjille, jotka ovat Active Directoryn sisällä olevassa haltijassa (Microsoft 2020a).

Azure Logic App

Azuren Logic Appit ovat keskeinen integraatiokomponentti integraatioiden toteutuksessa. Logic App on pilvessä oleva palvelusovellus, jolla voidaan luoda erilaisia työnkulkuja integraation aikana (Microsoft 2021d). Esimerkiksi Logic App voi ottaa vastaan HTTP-kutsun ja tehdä sen jälkeen erilaisia tehtäviä ja palauttaa HTTP-vastauksessa tehtävien tuloksen. Yleinen käyttötarkoitus Logic Appille on myös se, että se mahdollistaa integraatioiden suorituksien tarkistuksen.

Azure Function App

Azure Function App on Azuressa pyörivä pilvipalvelu, johon voidaan julkaista ohjelmistoja ajattelematta taustalla pyörivää palvelinpuolta ja näitä ohjelmistoja tai funktioita voidaan kutsua monella eri tavalla (Microsoft 2020c). eCraftilla yleisimmät käyttötapaukset ovat integraatioiden tekeminen järjestelmien välille ja API-rajapintojen luominen data-analytiikkaa varten.

Azure Service Bus

Azure Service Bus on Azureen tehtävä viestinvälitys komponentti, jolla voidaan vastaanottaa viestejä erilaisista järjestelmistä (Microsoft 2021e). Viestit syntyvät Service Busin alle luotavaan jonoon, josta muut applikaatiot voivat ottaa viestejä vastaan. Yleinen käyttökohde Service Busille on luoda lähdejärjestelmään tapahtuma, joka lähettää Service Busiin viestin tapahtumasta ja Logic App myöhemmin ottaa viestin vastaan ja tekee viestin sisällöstä riippuen erilaisia toimintoja.

Visual Studio

Visual Studio on Microsoftin kehitysympäristö ohjelmistokehitykseen ja toimii ohjelmakoodien tekstieditorina ja projektin- sekä ja versionhallinnan ohjelmana.

Visual Studion uusin versio on 2019-malli, jossa on sisäänrakennettuja toiminnallisuuksia auttamaan kehitystyötä. Sen sisällä on toiminnallisuutta esimerkiksi versionhallintaan ja Azuren integraatiokomponenteille julkaisemiseen (Microsoft 2019c). Jälkimmäistä ei käytetä paljoa, koska monimutkaisien ympäristöjen julkaiseminen on huomattavasti helpompaa ja nopeampaa ARM-mallien avulla. Integraatioiden logiikassa käytettyyn OData on myös sisäänrakennettua konfiguraatiotoiminnallisuutta, mutta eCraft on luonut oman NuGet-paketin OData clientille, jonka implementoiminen ja käyttäminen projektissa on tehty erittäin helpoksi.

Visual Studiossa työskentelyyn liittyy aina projektihierarkia. Ylimpänä tasona on Visual Studio-ratkaisu, jonka alla on projektit yksittäisempää laajempaa kokonaisuutta varten (Microsoft 2020b). Esimerkiksi Azure-funktioiden projektiin ei sekoiteta SQL-tietokantaan liittyvää EntityFramework-kokonaisuutta vaan se luodaan omaan projektiin.

CRM

CRM eli customer relationship management on asiakkuudenhallintaan tehty järjestelmä, jossa luodaan esimerkiksi uusia käyttäjiä asiakkaille ja heille liittyviä tietoja (Microsoft 2019d). CRM:n sisällä on myös laajennuksia muuhun tarkoitukseen esimerkiksi Field Service. Järjestelmästä käytetään hyvin montaa eri nimeä työelämässä, mutta tässä työssä järjestelmästä käytetään nimeä Dynamics 365 CRM, jolla tarkoitetaan pilvessä olevaa järjestelmää.

ERP

ERP eli enterprise resource planning on toiminnanohjausjärjestelmä, joka on laaja järjestelmäkokonaisuus, jossa käsitellään yrityksen laskutusta, kirjanpitoa, tuotantoa, jakelua ja varastonhallintaa muuta vastaavaa yrityksen toimintaan liittyvää logiikkaa. Yksi Microsoftin tarjoamista ERP:stä kulkee nimellä Dynamics 365 Finance & Operations (Encore 2021.) Microsoft Dynamics AX on laajemmin tunnettu nimi, joka oli järjestelmän vanha nimi. Tässä työssä järjestelmästä käytetään nimeä Dynamics 365 ERP, jolla tarkoitetaan pilvessä olevaa järjestelmää.

Dynamics 365 ERP:n ja integraatioiden ohjelmointikielet

C# on Microsoftin kehittämä ohjelmointikieli, joka julkaistiin vuonna 2000. C# on olio-ohjelmointiin sopiva kieli ja se on laajamittaisessa käytössä integraatio-ohjelmista videopeleihin. Sen juuret tulevat C, C++ ja Javan syntakseista. (Microsoft 2021a.)

X++ on Dynamics 365 ERP:ssä käytettävä ohjelmointikieli ja sen taustalla toimii MorphX-ohjelmointiympäristö. Kieltä käytetään aktiivisesti Dynamics 365 ERP:n Application Object Tree -hierarkian kanssa, josta löytyy järjestelmän kaikki käytettävät tietotyypit, taulujen kentät, taulut, järjestelmän entiteetit, lomakkeet ja muut vastaavat järjestelmän komponentit. (Microsoft 2015a.) X++-kielellä on paljon samanlaisia ominaisuuksia C#:n kanssa, mutta niiden käyttö kehitysympäristöissä on kuitenkin erilaista.

Ohjelmistokehityksen ympäristöt

Järjestelmäkokonaisuuksien kehitystyössä jatkuva testaaminen kehittäjien ja käyttäjien näkökulmasta on elintärkeää, mutta käyttäjiä ei kuitenkaan pitäisi päästää testaamaan vielä vaiheessa olevia ominaisuuksia (Codebots 2017). Siksi ohjelmistokehityksessä käytetään useita ympäristöjä kehityksen eri vaiheissa. Nämä ympäristöt tunnetaan eCraftilla usein nimillä DEV

eli development, UAT eli user acceptance testing ja PROD eli production. DEV-ympäristöä käytetään uusien ominaisuuksien kehittämiseen ja testaamiseen. UAT-ympäristössä asiakkaalle annetaan pääsy järjestelmään, jolloin loppukäyttäjät voivat testata ominaisuuksia. PROD-ympäristössä on valmis versio tuotteesta, joka on jo asiakkaan liiketoimintakäytössä.

HTTP

HTTP tai Hypertext Transfer Protocol on ohjelmakerros, joka on tarkoitettu dokumenttien ja tiedon välittämiseen verkkoselainten ja verkkopalvelinten välillä (Mozilla 2021). HTTP-kutsuihin liittyy monia eri osia riippuen käyttötapauksesta, mutta tärkeimmät ovat URI-merkkijono ja HTTP-metodi. Esimerkiksi verkkopalvelimien rajapintoihin voidaan käyttää mm. GET- ja POST-metodeja, joilla haetaan tai lisätään tietoa rajapinnan alla olevaan tietokantaan.

OData

OData tai Open Data Protocol on rajapintastandardi, jolla voidaan luoda REST-pohjaisia rajapintapalveluita (OData 2021). Vaikka OData luo HTTP-kutsuja, eCraft on luonut NuGet-paketin, jolla voidaan C#-ohjelmakoodin sisällä luoda rajapintakutsuja LINQ-kirjaston kanssa. Yleinen käyttökohde ODatalle on ottaa yhteys CRM- tai ERP-järjestelmän tietokantaan ja tehdä sillä kyselyitä tiedon hakemiseen, lisäämiseen tai muokkaamiseen.

DevOps

DevOps tai development and operations on toimintamalli ohjelmistokehityksessä, joka mahdollistaa monien kehitysprosessin automatisoinnin. DevOpsin kautta voidaan automatisoida ohjelmiston testaamisprosessi, käännösprosessi ja julkaisuprosessi. (Microsoft 2021b.) Käytännössä tämä tarkoittaa esimerkiksi sitä, että kun DevOpsiin konfiguroituun versionhallinnan repositorion haaraan tulee muutos, DevOps suorittaa ohjelman testauksen, käännöksen ja julkaisun kohdeympäristöön.

Confluence

Confluence on Atlassianin luoma yhteistyötila, jota voidaan käyttää esimerkiksi työyhteisön toimintamallien luontiin ja projektikohtaiseen- sekä graafiseen järjestelmädokumentaatioon (Atlassian 2020). eCraft käyttää Confluencea projektien ohjelmistojen suunnittelun dokumentointiin, joita asiakkaat ja muut projektin osalliset voivat tarkistella. Dokumentointi sisältää mm. kaavioita ohjelmiston toiminnasta tai ohjelmaluokkien kenttäkartoituksia.

2.2 Integraatiokehityksen erilaiset tehtävät

Integraatiokehityksen aikana on myös tarve tietää kaikki erilaiset tehtävät ja mitä niiden suorittamiseen vaaditaan. Erilaisten ympäristöjen kanssa toimimiseen tarvitaan paljon pientä nuanssitietoa, esimerkiksi App Registeration, jolla voidaan käyttää vaikkapa CRM:n OData- rajapintaa Azure-funktioilla.

2.2.1 Integraatioputken suunnittelu

Joissain projekteissa toimitaan vain Microsoftin tuotteiden parissa, jolloin eri järjestelmien tietorajapintojen kanssa toimiminen on tehty suhteellisen helpoksi. Kuitenkin hyvin usein integraatioissa toimitaan jonkun Microsoftin järjestelmän kanssa, joka integroidaan asiakkaan kolmannen osapuolen järjestelmän kanssa. Näissä tapauksissa on useita eri vaihtoehtoja, millä integraatio kehitetään, mutta myös joissain tapauksissa voi joutua käyttämään jotain tiettyä komponenttiputkea kolmannen osapuolen rajapinnan teknisen toteutuksen vuoksi.

Esimerkiksi jos kolmannen osapuolen API on toteutettu SOAP:lla, niin joskus Logic App voi olla kykenemätön muuttamaan sen metadataa oikeaan muotoon. Logic App ei pysty tehdä XML pyyntöjä, joten SOAP-rajapinnat täytyy kääntää Custom Connectionilla sellaiseen muotoon, että niitä voi kutsua JSON-malleilla Logic Appin kautta.

2.2.2 Autentikoituminen eri järjestelmiin

Tietoturvasyistä jokaiseen integroitavaan järjestelmään täytyy autentikoitua luvitetulla käyttäjällä, jottei projektikuplan ulkopuoliset pääse tietoihin käsiksi. Lisäksi integraatioputken komponentit ovat salattu joko Azuren autentikoinnin taakse tai funktioiden headerit vaativat autentikaation. Koko integraation sisällä olevien järjestelmien ja komponenttien yhteen liittäminen vaatii paljon työtä ja tietämystä niiden toiminnasta.

Esimerkiksi Azure-funktion kautta Dynamics 365 ERP:n OData-rajapintaan autentikoituminen vaatii Azure-tilauksen alla olevasta Active Directory-haltijasta sen uniikin tunnisteen, Active Directory ohjelmarekisteröinnin ja sen salaisuuden, Dynamics 365 ERP:n pohja URL-osoitteen sekä sen API-kutsuja varten olevan aliosoitteen.

3 Nykytilanne

Jokaisessa uudessa työssä on paljon opittavaa ja vähän aikaa oppia vaadittavat taidot. Tavoitteena on kasvaa mahdollisimman nopeasti työpaikan tuottavaksi jäseneksi, mutta täysin uutena ihmisenä IT-alalla tämä ei kuitenkaan ole mahdollista. Vertaistyöntekijöiden joukossa on ihmisiä, jotka ovat jo olleet IT-alalla töissä. Tästä huolimatta jokaisella on mennyt kauan

työhön vaadittavien taitojen omaksumisessa. Pelkkien työtapojen ja työn tekemiseen liittyvien järjestelmien opetteluun menee suuri määrä aikaa. Uutena työntekijänä uudella alalla suorittamisen paineet ovat kovat ja uusien haastavien tehtävien myötä virheitä sattuu joka päivä. Virheiden kautta oma osaaminen ja kokemus kuitenkin kehittyy joka päivä.

3.1 Nykyinen työ ja osaaminen

Työtehtävät ja osaaminen

Pääasiallinen työtehtäväni on toimia ohjelmistokehittäjänä asiakasprojekteissa ja tarkemmin sanottuna integraatioliittymien kehittäjänä. Konkreettisella tasolla työtehtävinäni projekteissa on Azure–pilvipalvelun integraatiokomponenttien luominen, konfiguroiminen ja muokkaaminen tiettyä tarkoitusta varten. Projektin oman osa-alueen vastuu on aina itsellä, mutta projektikokonaisuuden toiminnallisuudesta vastaa aina projektin vanhempi työntekijä.

Työssä tarvitaan olio-ohjelmoinnin lisäksi tietämystä järjestelmien ja rajapintojen toiminnasta. Kuten missä tahansa ohjelmointityössä, jossa käsiteltävä ohjelma tai järjestelmä ei ole tuttu tai suoritettava työ ei ole rutiininomainen, vaaditaan pitkäjänteisyyttä ja kärsivällisyyttä toimintojen opettelussa. Työn opettelu on ollut haastavaa omalta osaltani, mutta onnistumisien kautta myös palkitsevaa.

Oman osaamisen arviointi

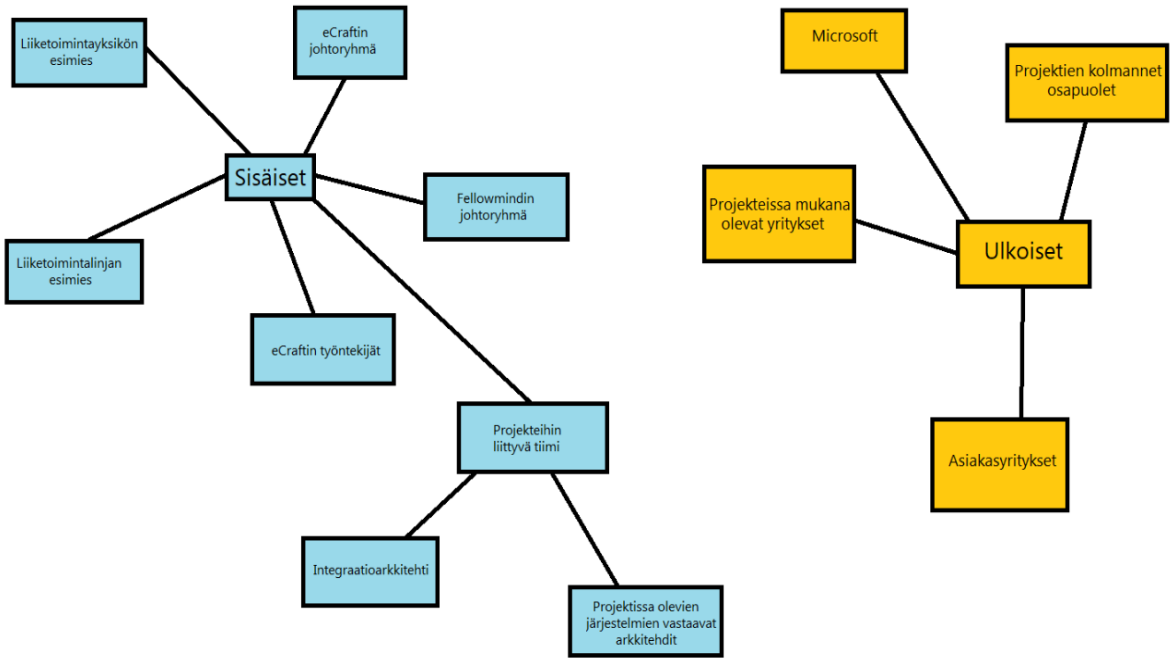
Näissä tehtävissä olen aloitteleva toimija. Työssä käytettävistä teknologioista ainoa tuttu suuri kokonaisuus on C#-kieli ja silläkään en ollut tehnyt mitään soveltavaa harjoittelua ennen työssä aloittamista. Koulutusjaksolla olevien harjoittelutehtävien aikana piti kysyä apua jokaisen komponentin tekemisen ohella. eCraftissa on kuitenkin yhteisöllinen kulttuuri, joten apua saa aina tarvittaessa. Pienempien projektien osa-alueiden korjauksia ja muunnoksia olen saanut suoritettua myös yksin.

Kehittyminen

Olen vielä lähtökuopissa, mutta kehittymisen merkkejä näkyy kuitenkin jo. Integraatiokomponenttien luominen ja niiden toiminnan ymmärtäminen on jo paremmalla tasolla, kuin työssä aloituksen aikana. Niiden luomisessa voi kuitenkin käyttää useampaa eri tapaa ja jokainen pitäisi osata. Jatkossa työn alla on ohjelmointitaidon parantaminen ja projektitöissä käytettävien järjestelmien osa-alueiden toiminnan kertausta.

3.2 Sidosryhmät

eCraftilla työntekijällä on monia sidosryhmiä ja riippuen työntekijän asemasta niiden tärkeys vaihtelee (Kuvio 1).



Kuvio 1 Sidosryhmät

Työharjoittelijana suurimmat vaikuttavat sisäiset sidosryhmät ovat oman liiketoimintayksikön ja -linjan esimiehet, eCraftin muut työntekijät ja projekteissa olevat tiimit. Integraatiokehittäjän pitää olla tarvittaessa yhteistyössä integroitavien järjestelmien arkkitehtien tai vastaavien henkilöiden kanssa. Ilman selkeää kommunikaatiota pienienkin asioiden tekeminen voi muuttua työlääksi. eCraftin ja Fellowmindin johtoryhmät eivät vaikuta tällä hetkellä niin paljoa, koska työharjoittelijana keskityn enemmän oppimiseen ja koulutuksiin, kuin yrityksen johtoryhmän tekemisiin. Tulevaisuudessa tämä varmasti muuttuu, kun alkaa ajattelemaan korkeamman päätäntävallan tekemisiä enemmän.

Ulkoisien sidosryhmien joukosta ainoa, joka ei ole työharjoittelijalle tärkeä on projektien mukana olevat yritykset. Tähän mennessä en ole ollut vielä tekemisissä muiden yrityksien kanssa, jotka ovat mukana samassa projektissa. Tämäkin varmasti muuttuu tulevaisuudessa, kun omien töiden vastuullisuus ja kokemus kasvaa.

3.3 Vuorovaikutustaidot

Miltei poikkeuksetta työpäivän aikana saa olla kokouksessa projektiin liittyvän henkilöstön kanssa. Kokouksia voidaan pitää paikan päällä, mutta koronatilanteen vuoksi lähes kaikki kokoukset tähän mennessä on pidetty etänä. Kokouksissa käsitellään työn etenemiseen, käytettäviin järjestelmiin ja toimintatapoihin liittyviä asioita ja sen aikana kukin kertoo omasta näkökulmasta asioiden edistyminen tai käsitellään vaikeudet miksi asiat eivät edisty. Koska oma roolini työn alla olevissa projekteissa ei vielä vakiintunut ja tehtävät eivät ole rutiininomaisia, huomaan, että joskus on vaikea osallistua keskusteluun. Tilanteita on myös tullut eteen, jossa on täytynyt kyseenalaistaa vanhemman työntekijän teknistä suunnitelmaa. Työharjoittelijana kynnys tähän on kuitenkin korkea.

3.4 Kehittäminen

Toistaiseksi en ole kehittänyt työtehtäviäni eteenpäin, koska jotta jotain voisi kehittää, täytyy ensin osata asia. Joidenkin toimintaprosessien kehittämisestä on ollut esimiehen kanssa puhetta. Esimerkiksi integraatiokomponenttien julkaisuprosessissa käytetty ARM-mallin osa-alueiden luonnissa tarvittavia JSON-pätkiä voisi luoda johonkin käytettävään liittymään.

3.5 Opinnäytetyön tavoitteet

Työn tavoitteena on kehittyä integraatiokehittäjänä ja saada syvempää ymmärrystä usein käytettyjen järjestelmien toiminnasta, esimerkiksi CRM ja ERP. Integraatioita voi tehdä myös tietämättä järjestelmien toimintaa, mutta se hidastaa ja vaikeuttaa kehittämistä huomattavasti, koska oikean tiedon hakeminen tai oikean avun saaminen voi kestää hyvinkin kauan.

Opinnäytetyön seurantajakson aikana on tarkoitus toimia asiakasprojekteissa ja niiden sisältö vaihtelee Dynamics 365 ERP:n tehtävistä laajennoksista kolmannen osapuolen rajapintaintegraatioihin. Miltei jokaisessa projektissa on vanhempi työntekijä mukana ja opastamassa, mutta sellaisen puuttuessa apua on mahdollista saada myös projektitiimin ulkopuolelta.

eCraftin tavoitteena on validoida omaa projektityöskentelyn muotoa ja tiimityöskentelyä. Aloittelevien kehittäjien tehokas ja onnistunut koulutus on elintärkeää uusia ja tulevia koulutusohjelmia ajatellen. Päiväkirjamuotoisella opinnäytetyöllä voidaan tarkastella projektityössä eteen tulevia ongelmia niin teknisistä kuin suunnitelmallisista näkökulmista ja täten saada tietoa ja ymmärrystä koulutusohjelmien kehittämiseen.

4 Päiväkirjavaihe

4.1 Viikko 1

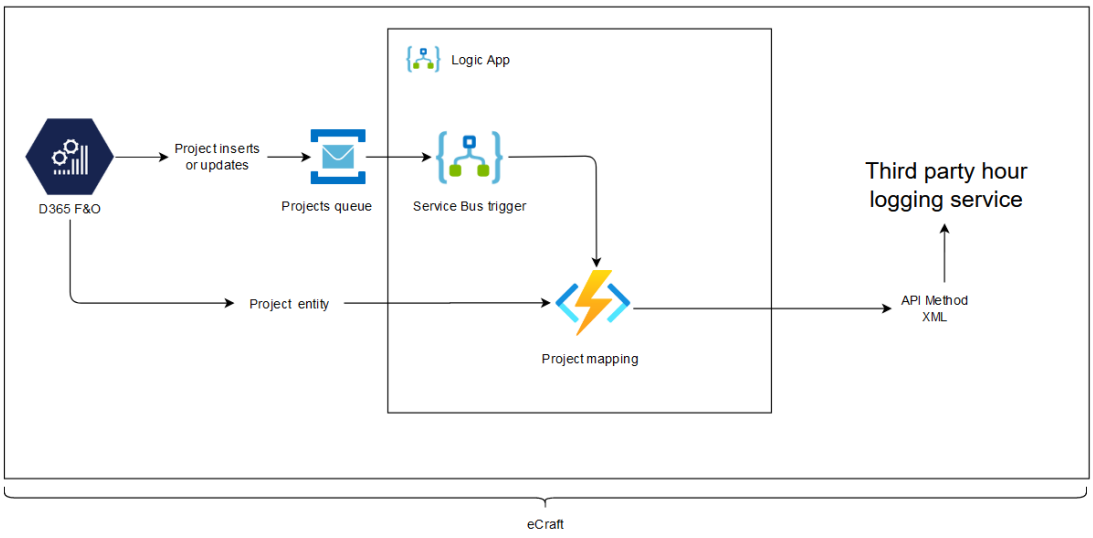
Ensimmäisen viikon tavoitteena on saada Visual Studio -ratkaisu rakennettua tulevaa ulkoista palkkajärjestelmän integraatiota varten, ratkaista DevOps-putken ongelma ja saada käyttäjätili luvitettua asiakkaan Azure-tilaukseen toisen projektin osalta. Muissa projekteissa ongelmien sattuessa suoritetaan testejä ja tarvittaessa korjataan vikoja. Visual Studio -ratkaisun pystyyn saaminen on tärkeä alku projektissa, sillä se pitäisi saada luotua ja siirrettyä versionhallintaan, jotta muut projektin kehittäjät pääsevät siihen käsiksi ja itse kehittäminen voi alkaa.

Maanantai 25.1

Päivän aikana on tavoitteena saada Visual Studio -ratkaisu tehtyä ensi viikolla alkavaa ulkoisen palkkajärjestelmän ja Dynamics 365 ERP:n välistä integraatiota varten ja siirtää se versionhallintaan. eCraftilla on valmis Visual Studio -ratkaisumalli, jossa on valmiina koottuna monia Integrations & API -liiketoimintayksikön luomia työkaluja erilaisia API-kutsuja ja virheenhallintaa varten sekä valmiita ARM-mallien kokonaisuuksia julkaisua varten.

Tiistai 26.1

Päivän keskeisimpänä aiheena on saada kokeiltua joulukuussa aloitettua ulkoiseen tuntikirjausjärjestelmän rajapintaan tehtävää integraatiota. Integraation päätavoitteena on saada Dynamics 365 ERP:n projektinhallinnassa olevista projekteista luotua projektihierarkia tuntikirjausjärjestelmään (Kuvio 2).



Kuvio 2: Dynamics 365 ERP - ja tuntikirjausjärjestelmien välinen integraatioputki

Dynamics 365 ERP:stä lähtee tieto tapahtumalla, joka laukaistaan aina, kun uusi projekti luodaan tai olemassa olevaa projektia muokataan ja projektin status on aloitettu tai valmis. Tapahtuma lähettää viestin viestijonoon ja viestin sisällä on projektille annettu tunniste, sekä yrityskohtainen tunniste. Logic App -suorituksen aikana viestin sisällöstä etsitään projektitunniste ja yritystunniste ja ne lähetetään Azure-funktiolle. Funktion sisällä validoidaan projektin olemassaolo ja sen jälkeen kutsutaan tuntikirjausjärjestelmän rajapintaa luodun projektiluokan sisällöllä.

Keskiviikko 27.1

Päivän ongelmana on ollut etsiä vikaa YAML-ohjelmointikielellä kirjoitetusta DevOps-käännösputkesta. Samaa putkea käytetään projektin muissa osissa, joissa siitä ei tule vikaa, mutta jostain syystä tuntikirjausjärjestelmän integraation Visual Studio -projektin käännösputki ei mene läpi. Virhelokissa olevista viesteistä päätellen ongelma on käyttäjän autentikoinnissa. Tämän ongelman ratkaisuun tarvitaan kokeneempaa työntekijää, mutta kiireisen aikataulun vuoksi tämän ongelman ratkaisu siirtyy myöhempään ajankohtaan.

Torstai 28.1

Päivän aiheena on testata Dynamics 365 ERP - ja tuntikirjausjärjestelmän välisen integraatioputken koko toimintaa. Koska DevOps-käännösputken kanssa on ollut ongelmia, myöskään julkaisuputkea ei ole vielä voitu käyttää. Tämä tarkoittaa sitä, että Azure-funktio täytyy luoda manuaalisesti ja julkaista funktion sisältö manuaalisesti Visual Studio -projektin sisäänrakennetulla työkalulla. Service Bus -viestijono ja siihen liittyvät yhteydet sekä Logic App luodaan manuaalisesti asiakkaan Azure-ympäristössä.

Testaus meni hyvin integraatioputken toiminnan suhteen, mutta uusia kysymyksiä heräsi liittyen integraation suunnitteluun. Tuntikirjausjärjestelmässä projektien hierarkian luonti onnistuu, mutta projektin alla olevat tehtävät eivät siirry projektien lapsiksi. Suunnitelman mukaisesti tekemällä hierarkian luonti ei siis onnistu, joten täytyy kysyä projektin johdolta lupaa ja mahdollisesti myös ohjeita tarvittavien muutoksien tekemiseen, sillä siitä syntyy hyväksymättömiä lisätöitä.

Perjantai 29.1

Päivän aiheena on jatkaa tuntikirjausjärjestelmän integraation testausta. Suurempien projektihierarkioiden luonnissa huomattiin, että Logic App-Service Bus välisessä laukaisimessa on suuri rajoittava pullonkaula. Nykyisellään Logic App tarkistaa annetun aikavälein viestijonosta saapuneita viestejä ja tässä tapauksessa aikaväli on kolmen minuuttia. Ongelma syntyy siinä, kun Dynamics 365 ERP -testiympäristössä luodaan kolme projektia yhden minuutin aikana, integraatioputkella menee mahdollisesti jopa yhdeksän minuuttia suorittaa näiden muutoksien

tekeminen tuntikirjausjärjestelmän rajapintaan. Tämä ratkaistiin muuttamalla Logic App laukaisin yhden viestin tarkistusmuodosta viestierän tarkistukseen.

Viikko 1 yhteenveto

Yksi viikon aikana tehtävistä töistä oli Visual Studio -ratkaisun perustaminen, mutta sen tekeminen siirrettiin lopulta vanhemmalle työntekijälle. Visual Studio -ratkaisujen rakenne on kuitenkin tärkeä tieto, joten sen tutkiminen on hyödyllistä kertausta.

Projektien alkupuolella tehdyt integraatiosuunnitelmat ovat joskus suuntaa antavia ja yleensä vasta testausvaiheessa selviää ongelmia, jotka johtuvat suunnitelman puutteista tai virheistä. Näissä tapauksissa herää oma ongelma siitä, että ehdottaako suoraan toimivaa muutosta suunnitelmaan vai antaako vastuun tarvittavasta muutoksesta suoraan vastuuta kantavalle taholle. Asia täytyy kuitenkin ainakin jollain tavalla nostaa projektijohdon tietoisuuteen, että ongelmaa voidaan alkaa selvittää joko tiimin voimin tai saada ulkopuolista apua.

Hyvin suunniteltu ohjelmisto on jo puoliksi tehty ja tällä tavoin säästetään työtunteja. Tarkistamalla huolellisesti kaikki ohjelmistossa käytettävien rajapintojen palauttamat sanomat tulevilta ongelmanratkaisuilta todennäköisesti säästytään. Puutteellisella suunnittelulla tiimi käyttää resursseja ongelmien ratkaisuun ja tarkistamalla kohdejärjestelmän rajoitteita suunnittelun aikana virhe olisi huomattu jo ennen työn aloittamista (Hyperext 2017.)

Ohjelmistoprojektien suunnitteluun kuuluu monta eri osaa, joita yleensä hoitaa projektipäällikkö ja arkkitehdit. Ylimpänä on projektin ”scope” eli mitä kaikkea projektissa tarvitsee saada tehtyä. Scopen määrittely on erittäin tärkeää ohjelmistoa toimittavan yrityksen kannalta, koska sopimus tehdään sovituista töistä ja työmääristä, jolloin sopimuksen ulkopuoliset työt määritellään muutoksina eikä korjauksina. Tästä lisää myöhemmillä viikoilla. Scopen alla on työmäärä- ja muut kustannusarviot sekä kehitykseen tarvittava aikataulu (javatpoint 2021.)

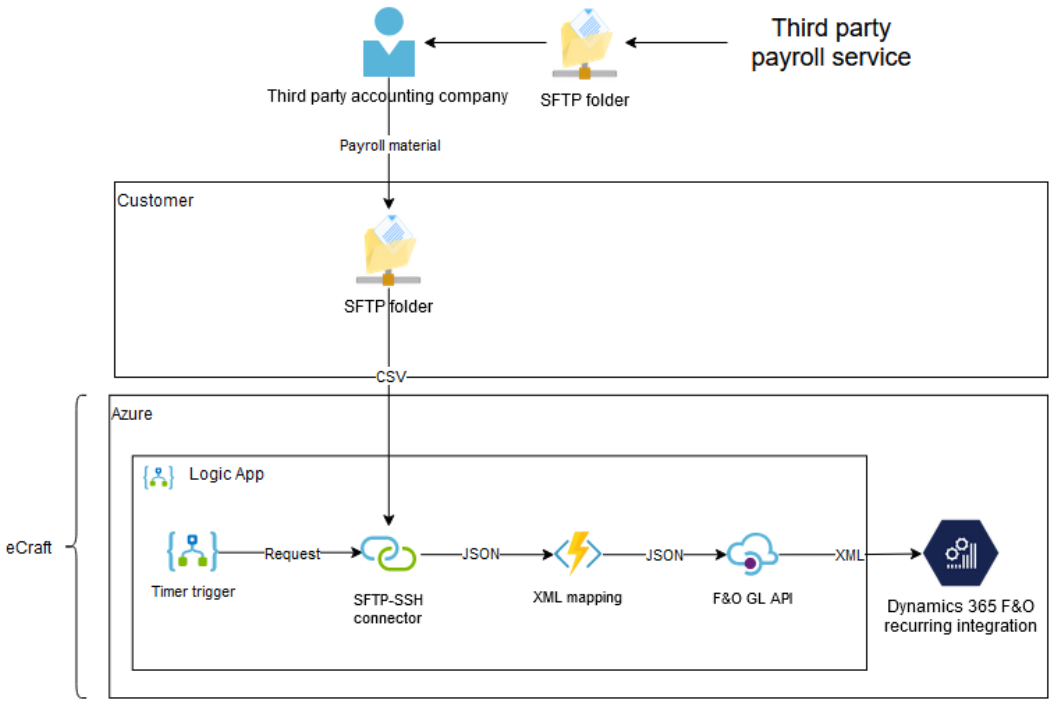
Projektien tai sen osa-alueiden suunnittelu ei kuulu työharjoittelijalle, mutta tulevaisuutta ajatellen on kuitenkin tärkeää kyetä arvioimaan jonkin työn resurssivaatimuksia. Integraatioiden osa-alueilla on oma työmääräarvio tehtäväkohtaisesti. Työmääräarviot ovat vain arvioita, mutta sovitun budjetin ylittyessä lisäbudjettia on paljon vaikeampi saada.

4.2 Viikko 2

Viikon aikana on tarkoitus aloittaa ulkoisen palkkajärjestelmän ja Dynamics 365 ERP:n välisen integraation tekeminen. Tavoitteena on saada liittymää aloitettua sekä saada ymmärrystä kirjanpidon logiikasta yleisesti ja Dynamics 365 ERP:n General Ledgerin toiminnasta.

Maanantai 1.2

Integraation toisena osapuolena on palkkajärjestelmä, jota käytetään laajasti palkanlaskentaan, henkilöstöhallintoon ja raportointiin. Tässä projektissa on tarve integroida palkkajärjestelmästä tulevien kulutiedostojen sisältö Dynamics 365 ERP:n General Ledger -kirjanpitotietokantaan (Kuvio 3).



Kuvio 3: Dynamics 365 ERP - ja palkkajärjestelmien välinen integraatioputki

Palkkajärjestelmästä tiedostot liikkuvat asiakkaan tiedostoserverille SFTP-yhteyden kautta ja integraatioputki ottaa yhteyden paikalliseen tietokäytävään palveluliitännällä. Tästä seuraava osa on Logic App, joka toimii ajastetusti joka yö. Logic App saa palveluliitännällä yhteyden käsiteltävään tiedostoon ja antaa sen sisällön HTTP-kutsun rungossa Azure-funktiolle. Azure-funktion sisällä tiedoston sisältö puretaan ja käännetään oikeaan muotoon, josta se lopulta lähetetään Dynamics 365 ERP:n General Ledger -rajapintaan.

Integraatiossa oleva palkkajärjestelmä on laajasti käytetty järjestelmä, joten osaamisesta siihen integroitumiseen on suurta hyötyä. Tällä hetkellä olen osallisena kolmessa projektissa, jossa tehdään samanlainen integraatio.

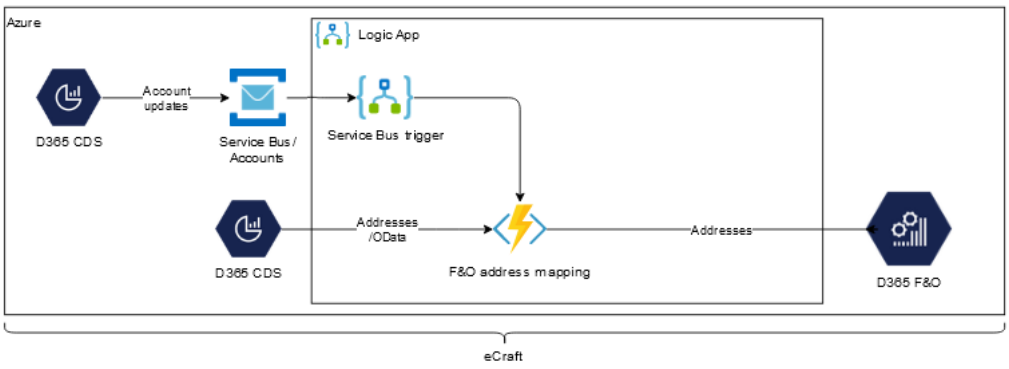
Tiistai 2.2

Päivän tavoitteena on saada viimeiset muutokset palkkajärjestelmän integraatioon ennen UAT-julkaisua. Tähän sisältyy Business Event -tapahtumien luonti Dynamics 365 ERP UAT -ympäristöön ja niihin tarvittavat Service Bus -päätepisteet. Asiakkaan Azure-tilauksen UAT-resurssiryhmään luodaan uusi Service Bus, jotta eri ympäristöjen viestit eivät mene sekaisin.

Tietoturvaoikeudet ovat kriittisiä isommissa ohjelmistoprojekteissa ja tuntikirjausjärjestelmän integraatiota tehdessä olen saanut parempaa ymmärrystä Azuren käytöstä ja erilaisten pääsyoikeuksien luonnissa. Muissa projekteissa tarvitaan täysin samanlaista tietämystä, joten Dynamics 365 ERP:n Business Eventien teosta on suuri hyöty.

Keskiviikko 3.2

Päivän aikana tuli aikaisemman projektin jo toteutetun integraatioliittymän suunnitelmaan muutos. Integraation alkupää on Dynamics 365 CRM-järjestelmä, joka on tässä tapauksessa laajennettu XRM:ksi, koska järjestelmässä on käyttäjätileissä asiakkaiden lisäksi myös toimittajien ja muiden yhteyshenkilöiden tietoja. Integraation pää on Dynamics 365 ERP –järjestelmä. Dynamics 365 CRM:n puolella tapahtuvasta muutoksesta käyttäjätietojen taulun tietueeseen laukaistaan tapahtumaketju, jonka tavoitteena on kopioida tapahtuman juuri tehty muutos Dynamics 365 ERP:n puolella oleviin tauluihin tai vaihtoehtoisesti luoda uusia tietueita tauluihin (Kuvio 4).



Kuvio 4: Dynamics 365 CRM - ja Dynamics 365 ERP -järjestelmien välinen integraatioputki

Kesken projektin tehtävät muutokset eivät ole mitenkään harvinaisia. Tässä tapauksessa asiakas huomasi, että kutakin osoitetyyppiä voidaan tarvita useampi per asiakas. Dynamics 365 ERP:n entiteetit eivät tue tätä toimintoa, joten tässä tilanteessa muutos oli tarpeellinen.

Torstai 4.2

Päivän aiheena oli jatkaa käyttäjätili-integraation muutosta. Miltei poikkeuksetta projektin vastaavat järjestelmäarkkitehdit ovat tehneet valmiin Confluence-dokumentaation muutoksen suunnitelmasta, mikä auttaa kehittäjän työtä suuresti. Tästä huolimatta täytyi kysyä pari kertaa Dynamics 365 CRM-arkkitehdilta tunnisteisiin liittyviä kysymyksiä Dynamics 365 ERP:n automaattisen tunnisteluomisen vuoksi. Ilman dokumentaatiota työ olisi edistynyt huomattavasti hitaammin. Palaan tähän aiheeseen myöhemmin seurantajakson aikana.

Perjantai 5.2

Päivän tavoitteena oli saada palkkajärjestelmän integraatioliittymä hyvään vaiheeseen, minkä jälkeen alkoi tilannekatsaus vanhemman projektijäsenen kanssa. Liittymän yksikkötestit menivät läpi sekä General Ledger-rajapintakartoitus tuotti viestit oikeassa muodossa. Visual Studio-projektin kansioista piti vain siirtää ja poistaa tarpeetonta tai sinne kuulumatonta sisältöä. Seuraava askel on tehdä koodikatselmointipyyntö, jotta ohjelmakoodi voidaan tarkistaa ja viedä projektin integraatiokokonaisuuden versionhallintaan. eCraftin versionhallintakäytäntöjen mukaan kaikki tehty ohjelmakoodi täytyy katselmoittaa ennen versionhallintaan siirtämistä.

Joskus työpäivän aikana voi ilmetä pieniä töitä, joita esimies tai projektin vanhempi työntekijä antaa nuoremmalle työntekijälle. Tänään sain iltapäivällä tehtäväksi luoda uuden Logic Appin samaan projektiin, jossa olen tehnyt Dynamics 365 CRM-Dynamics 365 ERP käyttäjäintegraatiota. Tapauksessa on tarve saada haettua General Ledger-rajapinnasta tämän kuun ja viime kuun kirjaukset. Logic App palauttaa HTTP-vastauksen rungossa rajapinnan palauttamat datat. Viimeisestely täytyy jättää ensi viikolle, koska en saanut tehtävän antajalta vielä tietoa, että missä muodossa data täytyy palauttaa.

Viikko 2 yhteenveto

Viikon aikana on ollut työskentelyä kolmessa eri projektissa ja näin toivoisin asioiden olevan myös jatkossa. Erilaisissa projekteissa mukana olemisella kertyy aina uutta osaamista ja kokemusta erilaisista tarpeista. Ongelmia on syntynyt pienistä puutteista yhden integraation suunnitelmassa, mutta yhtä lailla myös omasta toiminnasta. Ottamalla ongelmat heti käsittelyyn joko omassa työssä tai projektiryhmän työhön liittyvissä ongelmissa niistä selvitään jollain ratkaisulla.

Koodikatselmointi joskus tuntuu prosessia hidastavalta toimenpiteeltä, mutta se on kuitenkin erittäin tärkeä oman oppimisen ja projektin toiminnan suhteen. Työtä tehdessä virheitä tulee tehtyä huomaamattaan, mutta katselmoinnissa virheet tai vaihtoehtoiset tavat tehdä asioita tulevat ilmi ja niistä oppii. Gitflow-mallinen versionhallinta on ollut hyvä tapa toimia ainakin

tähän mennessä, mutta sen rajoitteena on ylimääräinen työ ja ajan käyttö (Gadzinowski 2021). Tästä lisää myöhemmin.

Koodikatselmointien tekeminen on tärkeää, koska hyvin usein omassa katselmointipyynnössä on sisältöä, jonka ei pitäisi olla siellä. Joskus Visual Studio -projektihierarkiassa on kymmeniä tiedostoja ja niiden muutoshistoriaa on melkein mahdotonta muistaa itse. Katselmoinnilla annat työsi tuloksen muille tarkistettavaksi, jotta työn laatu on parempi ja että potentiaalisilta virheiltä vältytään (Larsen 2018). Lisäksi muiden ehdotuksilla on helppo oppia uusia tapoja ja tekniikoita tehdä asioita.

Katselmointien tekeminen on hidasta, mutta varmasti muodostumassa jokapäiväiseen työhön. En ole vielä oppinut tarkistamaan muiden tekemiä katselmointeja, mutta todennäköisesti niitäkin pääsee tekemään lähitulevaisuudessa. Olisi hyvä, jos jossain vaiheessa ehtisi käyttää aikaa työharjoittelijatovereiden kanssa muiden tekemien ratkaisujen tarkistukseen. Tällä tavalla olisi hyvä jakaa osaamista ja mahdollisia yleisiä kompastuskiviä tiettyjen liittymien toteutuksessa.

4.3 Viikko 3

Viikon tavoitteena on saada viime perjantaina aloitettu Logic App viimeisteltyä, jatkaa palkkajärjestelmän integraatiota ja mahdollisesti saada myös käyttäjäintegraatio valmiiksi. Palkkajärjestelmä- ja käyttäjäintegraatioissa tarvitsen tietoa ohjelman suunnittelijalta yksityiskohdista. Palkkajärjestelmän integraatiota on testattu yksikkötesteillä, muttei vielä oikealla General Ledger -rajapinnalla. Käyttäjäintegraatiota on kokeiltu omassa ympäristössä ja todettu sen toimivan, mutta yleensä pieniä virheitä ei huomaa tässä vaiheessa työtä.

Maanantai 8.2

Päivä alkoi viime perjantaisen General Ledger -rajapinnan Logic App -haun palautusmuodon luomisella. Aluksi olin luonut palautusmuodon, jossa oli eriteltynä listoissa tämän kuun ja viime kuun merkinnät. Tarkoitus oli kuitenkin saada molempien kuiden merkinnät samaan listaan ja palauttaa lista sellaisenaan.

Päivän aikana sain tehtyä integraatiopuolen koodikatselmoinnin tuntikirjausjärjestelmän integraatioon. Vanhemman kehittäjän pieniä muutosehdotuksia lukuun ottamatta Visual Studio -projektit olivat hyvällä tasolla. Aloitin myös koodikatselmoinnin palkkajärjestelmän integraatiossa.

Suurimpana työnä oli tänään virheen selvitys käyttäjäintegraatiossa. Mielestäni integraatiossa kulkevat viestit olivat oikein, eikä virhettä olisi pitänyt sattua. Sain esimieheltä hyviä ohjeita virheiden selvittämiseen, jotka eivät aluksi toimineet. Sain myöhemmin lisätietoa sellaisista

Visual Studion virheenjäljittäjän toiminnoista, joiden olemassaolosta en tiennyt. Ongelman syy ei ole vielä selvinnyt.

Tiistai 9.2

Päivän tavoitteena on saada palkka- ja tuntikirjausjärjestelmien integraatioiden koodikatselmoinnit läpi sekä jatkaa käyttäjäintegraation testausta. Tuntikirjausjärjestelmän integraatio on menossa tauolle ja on tärkeää saada ohjelmakoodit versionhallintaan, jotta ne ovat muiden käytettävissä tarpeen vaatiessa. Palkkajärjestelmän integraatiossa koodikatselmoinnin jälkeen on tarkoitus siirtyä testaamaan General Ledger -rajapintakutsua, mutta tämä voi vaatia vielä muun projektitiimin työn edistymistä.

Päivä alkoi jatkamalla kahden viikon takaista DevOps-käännösputken ongelmanratkaisua. Kokeneemman työntekijän kanssa saimme muutettua käännösputken logiikkaa, mutta oikea ongelman syy ei selvinnyt. Koska tuntikirjausjärjestelmän integraatio on menossa tauolle, kyseisen ongelman selvitys jää myöhemmälle ajalle. Tuntikirjausjärjestelmän integraation koodikatselmointi meni myös läpi, joten tässä vaiheessa on hyvä siirtää se tauolle.

Suurin osa päivästä meni käyttäjäintegraation ongelmanselvitykseen. Perusteellisen selvityksen jälkeenkin ongelma oli suuri mysteeri myös kokeneille työntekijöille. Ongelma ratkesi sillä, että entiteetti luodaan aina uudestaan vakioarvojen asetuksen yhteydessä ja korvataan vanha uudella.

Keskiviikko 10.2

Sain kolme viikkoa sitten tehtävän olla kahden tuotantoon menneen integraation valvojana, koska integraation kehittäjä lähti kahden viikon lomalle. Viime yönä sain viestin Azure-tilauksesta, että integraatiossa on tapahtunut virhe, joten tehtävänä on tarkistaa virhe ja yrittää selvittää mitä on tapahtunut ja miksi. Koska itselläni ei ole pääsyä asiakkaan Azure-tilaukseen, ehdotin projektitiimin jäsenelle Logic Appin manuaalista uudelleenajamista Azuren puolelta. Uudelleenajamisella voidaan tarkistaa, onko ongelma toistuva.

Palkkajärjestelmän integraation koodikatselmointi meni läpi, mutta käyttäjäintegraation ongelmanratkaisuun meni koko päivä. Eilen saatu ratkaisu ongelmaan ei ollutkaan niin hyvä. Ongelmaa voidaan kiertää luomalla aina uusi entiteetti samoilla tiedoilla, joilla vanha entiteetti on jo olemassa tietokannassa. Tämä ei ole oikeanlainen ratkaisu ongelmaan, mutta muita ratkaisuja ei ole tiedossa.

Torstai 11.2

Eilen tarkistin tuotantoon menneen integraation virhettä ja ehdotin Logic Appin manuaalisesti uudelleenajamista. Koska integraatio on asetettu ajettavaksi automaattisesti joka yö, jos

jokin virhe on toistuva sen pitäisi toistua joka yö, mutta koska viime yönä ei ole sattunut virheitä todennäköisesti oikeaa ongelmaa ei ole ollutkaan. Tehtävän päivystysvuoro päättyy tällä viikolla.

Sain päivällä myös uuden tehtävän. Tehtävänä oli selvittää Azuren UAT-ympäristössä epäonnistuneita suorituksia. Ongelma ei ratkennut iltapäivän aikana ja jatkan sen parissa huomenna.

Perjantai 12.2

Päivän tavoitteena on jatkaa tuoteintegraation ongelman selvitystä ja saada tietoa käyttäjäintegraation osoitemuutoksen jatkokehityssuunnasta. Muutos toimii testien mukaan tällä hetkellä halutulla tavalla, mutta ratkaisu ei ole ideaali duplikaattien takia. Nykyinen versio on kahdella vanhemmalla työntekijällä tarkistettavana ja testattavana.

Tein käyttäjäintegraation osoitemuutoksesta koodikatselmoinnin ja se hyväksyttiin. Siitä tehdään julkaisu UAT-ympäristöön jonkin ajan päästä ja laajemmassa testauksessa selviää, löytyykö jotain rajoituksia nykyisestä toteutuksesta. Tuntikirjausjärjestelmän integraatiossa tehtiin viimeiset julkaisut UAT-ympäristöön ennen projektin keskeyttämistä, jotta integraation toimivuutta voidaan testata tarvittaessa.

Viikko 3 yhteenveto

Viikon aikana tuli vastaan käyttäjäintegraatiossa pieniä ongelmia, jotka eivät lopuksi olleet niin pieniä. Pienienkin ongelmien ratkaisussa voi mennä hyvinkin pitkä aika ja joskus oikeaa ratkaisua ei löydy ikinä. Tällaisissa tapauksissa nuorella työntekijällä tulee mieleen oman työn tehokkuus ja sen suhtauttaminen tehtävän arvioituun työmäärään. Arvioituihin työmääriin olisi tarkoitus aina päästä, mutta tosielämässä uusien ongelmien vastaan tullessa nämä usein eivät kuitenkaan toteudu.

Koodikatselmointiprosessit tuntuvat joskus aikaa vieviltä, mutta nuorempana työntekijänä niiden tuottama uusi tieto on kuitenkin korvaamatonta. Usein katselmoinneissa kiinnitetään huomiota myös koodin ulkoasuun eikä vain sen sisältöön. Ominaisuuksien kommentointi ja yhteneväinen tyyli ovat yksiä huomioitavia asioita, vaikkei ”oikeaa” tyyliä ole olemassa. Vanhemman työntekijän kokemuksella kuitenkin on arvoa ja juuri aloittanutta työntekijää ohjataan tähän suuntaan. Katselmoinneissa saat palautetta tekemästä työstä ja usein se ei ole pelkkää kehua. Aloittelijana täytyy osata ottaa kritiikki vastaan. Kritiikkiä tulee siksi, että virheistä opitaan (Coutermarsh 2015.)

eCraftin koodikäytännöt pyörivät yleisien käytäntöjen ja Microsoftin käytäntöjen ympärillä. Erityisen tärkeänä asiana pidetään ohjelmiston refaktorointia eli ohjelmiston toiminnallisuuden eriyttämistä eri osiin, jotta sitä on helpompi muokata ja ymmärtää (Microsoft 2018).

Ohjelmakoodin kommentointi on yksi yksinkertaisimmista ja samalla tärkeimmistä tehtävistä asioista. Se on pääasiallisesti tarkoitettu muita ihmisiä varten, jotka lukevat tehtyä ohjelmakoodia. Se on myös tärkeää kehittäjän kannalta tapauksissa, jossa kehittäjä käyttää vuosia sitten tehtyä ratkaisua muussa projektissa, jossa tarvitaan samanlaista ratkaisua. Monimutkaisempia asioita ovat esimerkiksi toimiva ja kattava virheenhallinta ja palautusmuodot http-kutsuihin liittyvissä luokissa. Yksinkertaisempia asioita on muuttujien, metodien ja luokkien oikeanlainen nimeäminen ja ohjelmakooditiedoston rakenne (Microsoft 2015b.)

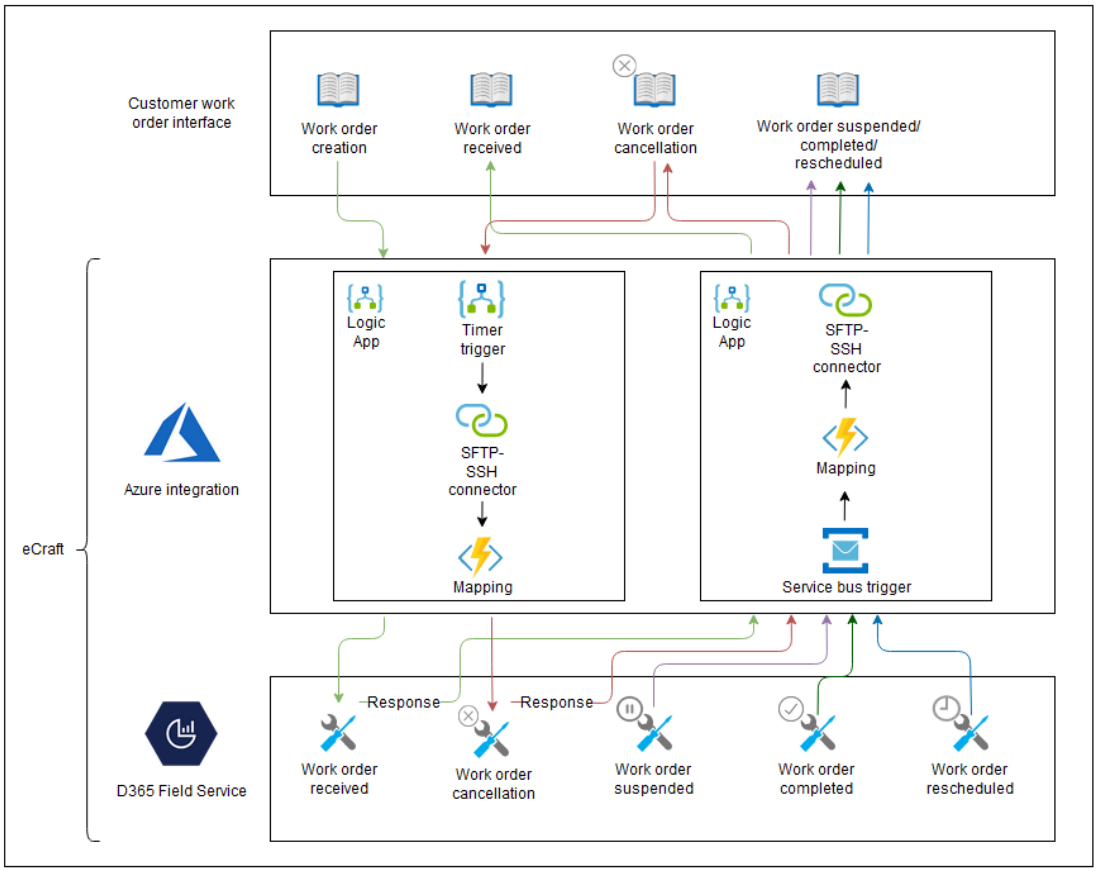
Koodikäytäntöjen perusasiat ovat minulla hyvin hallussa, mutta siitä huolimatta luokkien ja metodien kunnollinen dokumentointi ja kommentointi jäävät usein tekemättä. Syynä on yleensä joko kehittämiseen ja testaamiseen keskittyminen tai budjetin puute. Edellä mainitut asiat ovat erittäin tärkeitä tilanteessa, jossa projekti menee tuotantoon ja esimerkiksi vuoden päästä löytyy jokin ongelma. Sen ratkaisu voidaan antaa jollekin muulle tehtäväksi ja kyseisen henkilön täytyy lukea ohjelmakoodia ymmärtääkseen liittymän toimintaa. Luokkien ja metodien dokumentointi ja kommentointi tekevät tämän henkilön työstä huomattavasti tehokkaampaa.

4.4 Viikko 4

Viikon alussa tulevien töiden määrä ja muoto ei ole vielä tiedossa, koska monet työt ovat testauksessa tai odottavat muiden työntekijöiden työn edistymistä. Pienempiä töitä löytyy usein nopeasti, mutta ne eivät yleensä kestä kovinkaan pitkään.

Maanantai 15.2

Maanantaina on tarkoitus tehdä osa päivästä Azure Developer Associate -sertifikaattia ja loppupäivä uusia integraatioita uuden projektin parissa. Integraatio vaikuttaa hieman monimutkaisemmalta, kuin aikaisemmat tekemäni integraatiot sekä tarkoitus ja kohde ovat täysin uusia (Kuvio 5).



Kuvio 5: Dynamics 365 FS - ja asiakkaan työtilausjärjestelmien välinen integraatioputki

Uudessa projektissa on tarve saada työtilaukset integroitua asiakkaan järjestelmästä Dynamics 365 CRM:n sisällä olevaan Field Serviceen. Tapauksesta tekee monimutkaisen se, että työtilauksien status voi muuttua kummassa tahansa järjestelmässä, joten nämä muutokset täytyy myös integroida, jotta molemmissa järjestelmissä on ajantasainen tieto tilauksesta.

Tiistai 16.2

Päivän aikana jatkoin eilen aloitettua työtilausintegraatiota. Ainoa kompastuskivi oli se, että työtilausjärjestelmästä tulevan tiedoston sisältöä ei voi kääntää suoraan C#-kielen luokaksi, vaan tiedostosta täytyy kartoittaa jokainen kenttä yksitellen.

Keskiviikko 17.2

Päivän aikana pidimme ensimmäisen kokouksen työtilausintegraatiossa olevan tiimin kanssa. Sain hieman paremmin tietoa tehtävän integraation kokonaisuudesta integraatioarkkitehdilta ja projektin CRM-arkkitehdilta kartoitustaulukon Field Servicen työtilausta ja sen relaatioita varten. Confluenceen tehtyjen kartoitustaulukoiden avulla työtilauksien luominen OData-rajapinnalla on huomattavasti helpompaa. Oikeat kentät voi toki päätellä itsekin, mutta

integraation ollessa vasta alussa ei ole järkevää lähteä arvailemaan oikeita kenttiä. Testauksessa yleensä löytyy ongelmat, jos niitä on.

Torstai 18.2

Tärkeimpänä aiheena tänään oli kehityskeskustelu esimiehen kanssa. Kävimme läpi nykyistä osaamistilannetta viime joulukuun jälkeisestä hetkestä. Hyvää kehitystä on tapahtunut, mutta opittavaa on kuitenkin vielä paljon. Tarkistimme myös työn teknisen osaamisen ulkopuolella olevia asioita, esimerkiksi työskentelymotivaatiota, työn merkityksellisyyttä ja tiimityöskentelyä.

Kehityskeskustelun parhaana antina oli tulevaisuutta varten olevat keinot kehitystä varten. Esimiehen mielestä nykyinen osaaminen on jo sellaisella tasolla, ettei tarvitse lähteä kertaamaan mitään työtehtäviä erikseen. Työssä vastaan tulevat tehtävät riittävät tällä hetkellä oppimiseen. Koulusta valmistumisen jälkeen olisi tavoitteena saada vuoden aikana suoritettua Azure Developer Associate -sertifikaatti sekä pitää työmääristä päiväkirjaa jonain ajanjaksona. Päiväkirjamerkinnät ovat tärkeä oppimisen osa-alue työarvioiden ja suunnitelmien tekemiseen.

Perjantai 19.2

Päivän aikana pidettiin kaksi tärkeää kokousta työtilausintegraation osalta. Ensimmäisessä esimies kertoi työprosessin, joka tarvitaan Dynamics 365 CRM:n Business Eventien tekoon. Olin jo tehnyt aikaisemmin vastaavia Dynamics 365 ERP:ssä, mutta niissä prosessi oli hyvin erilainen ja huomattavasti vaikeampi.

Seuraavassa kokouksessa kävimme projektin integraatio- ja CRM-arkkitehtien kanssa tehtävää integraatiota läpi ja tarvittavia toimenpiteitä Field Servicen päässä. Saimme päätöksiä tehtyä esimerkiksi tietyistä vakioarvoista työtilauksen entiteetillä Field Servicessä ja sain lisätietoa tietyistä kartoituksista.

Viikko 4 yhteenveto

Viikolla pääsin aloittamaan jälleen uudessa projektissa työskentelyn ja pääsen samalla työskentelemään uusien ihmisten kanssa. Kaiken lisäksi tehtävä integraatio on uusi tapaus itselle ja hyvinkin kiinnostava.

Kehityskeskustelussa sain hyvää tietoa ja palautetta omasta kehittymisestä niin esimieheltä kuin projektien tiimien jäseniltä. Palautteissa annettiin kehuja kehittymisestä, mutta yhtä paljon kehotettiin luottamaan enemmän omaan osaamiseen. Tämä oli jo tiedossa ja pyrin jatkossa luottamaan enemmän omaan tekemiseen. Projektitöiden aloituksen aikana oma

luottamus tekemiseen oli erittäin heikkoa hyvästä syystä, mutta varsinkin parina viime viikkona on tuntunut, että itseluottamusta tekemiseen on jo kertynyt.

eCraftin integraatiotiimissä on pidetty parin viikon välein välikeskusteluja töiden muodosta, niiden riittävyydestä ja taitojen kehittymisestä. Välikeskusteluilla saadaan pidettyä pidempiaikaista kuvaa kehittymisestä ja kehityskeskusteluissa tarkistetaan uudelleen kehittymisen tasoa, mutta enemmän tulevaisuutta ajatellen. Kehityskeskusteluja pidetään, jotta esimiehellä ja työntekijällä on selkeä kuva työntekijän kuvitellusta taitotasosta ja esimiehen näkemästä oikeasta taitotasosta. Uusien tavoitteiden pariin kannustetaan jatkuvasti, että työntekijä ja esimies eivät tyydy pelkkään perusosaamiseen vaan pyrkivät aina kohti uusia saavutuksia (Murphy 2018.) Kaiken lisäksi esimies saa palautetta työntekijältä omasta toiminnastaan. Laajempaa osaamista muita töistä tarvitaan jatkuvasti IT-alalla.

Aloitteleville työntekijöille on tärkeää pitää kehityskeskusteluja monista eri syistä. Joillain aloittelijoilla voi olla liian suuret odotukset itsestään tai kuvittelevat olevansa jo hyvinkin korkealla tasolla heti aloittamisen jälkeen. Kehityskeskustelussa on hyvä pitää aloittavan työntekijän jalat maassa sekä madaltaa liian epärealistisia tavoitteita. Kukaan ei ole ammattilainen heti aloittaessaan ja pitää ymmärtää, että kehittymiseen menee aikaa. Yleensä aloittelevat työntekijät palkataan potentiaalin takia ja sen kulminoituminen ammattilaiseksi voi viedä vuosia (HR Morning 2021.)

4.5 Viikko 5

Viikon ainoana aikataulutettuna tehtävänä on jatkaa työtilausintegraatiota. Tavoitteena on saada integraatio valmiiksi suunnitellussa työmäärässä ja siirtää toimiva putki testaukseen. Tämän saavuttamiseksi täytyy vielä luoda jokaiselle integraation vaiheelle putki ja testata niiden toimivuutta. Tähän mennessä olen vain keskittynyt Azure-funktion ja Field Servicen väliseen kommunikointiin. Muitakin töitä voi tulla vastaan, kuten aikaisempina viikkoina on huomattu.

Maanantai 22.2

Päivän aikana oli tavoitteena saada työtilausintegraation putket tehtyä. Viime viikolla sain tehtyä Field Servicen päähän toimivat Business Eventit sekä sain asiakkaalta pääsyn SFTP-palvelimelle. Näiden pohjalta on mahdollista luoda kommunikaatio molempiin suuntiin työtilausjärjestelmän ja Field Servicen välillä.

Tiistai 23.2

Sain päivän aikana luotua koko putken työtilausjärjestelmästä Field Servicen suuntaan ja testattua sitä. Tähän mennessä tehtyjen testien perusteella työtilausten luonti ja päivitys sekä niiden onnistumisesta ja epäonnistumisesta luotavat viestit toimivat oikein. Nykyisellä

aikataululla työtilausjärjestelmän ja Field Servicen välinen kommunikointi työtilausten luonnin suhteen pitäisi olla valmiina tällä viikolla.

Keskiviikko 24.2

Päivä alkoi lyhyellä kokouksella työtilausintegraation integraatioarkkitehdin kanssa, jossa opetin integraation toimintaa Azuresta testejä varten. eCraftin projektihallintaa tehdään Jira-järjestelmässä ja arkkitehti tarvitsee tietoa integraation toiminnasta, jotta hän voi luoda Xray-testejä järjestelmään. Jirassa projektit pilkotaan tehtäviksi ja Xray-testeillä voidaan tehdä end-to-end testausta koko tehtävän toiminnasta.

Torstai 25.2

Päivän aikana oli uusi tilannekatsaus CRM-arkkitehdin kanssa. Tarkoitus oli saada lisätietoa työtilausten päivittämisen tapahtumista Field Servicen päässä. Tapahtumia pitäisi laukaista työtilauksen päivittämisestä, mutta koska sanoman sisältö takaisin työtilausjärjestelmän suuntaan vaihtelee riippuen päivityksen muodosta, päivitystä ei voi laukaista mistä tahansa muutoksesta. Työtilausten uudelleenajoituksen tapahtumaa varten luodaan uusi nappula Field Servicen käyttöliittymään, josta työntekijä voi sitten laukaista uudelleenajoituksen uusien päivämäärien asettamisen jälkeen. Keskeytyksessä, perumisessa ja päivittämisessä täytyy käyttää monimutkaisempaa logiikkaa oikean tapahtuman tunnistamiseksi.

Perjantai 26.2

Päivän aikana oli tavoitteena saada työtilausintegraation Visual Studio -projektin ohjelmakoodiluokat muutettua uuteen ja tehokkaampaan muotoon. Integraatiota on tähän mennessä tehty mahdollisimman nopeasti, jotta saadaan ensimmäinen versio testattavaksi ensi maanantaita varten. Tästä syystä Azure-funktiota varten olevat luokat ovat hieman itseään toistavia, koska monissa vaiheissa tarvitaan koodikielen käännös tiedostosta tai kartoitus sanomaa varten.

Viikko 5 yhteenveto

Viikon aikana tein lähes pelkästään työtilausintegraatiota. Viikon tavoitteena oli saada ensimmäisen vaiheen prototyyppi ensi viikon alussa olevaa asiakastestausta varten. Tämäntyyppisissä integraatioissa on tyypillistä, että aluksi luodaan niin sanottu MVP eli Minimum Viable Product. Tarkkaan työn jälkeen ei vielä kiinnitetä niin paljoa huomiota, vaan luodaan toimiva ensimmäinen versio niin nopeasti kuin mahdollista.

Minimum Viable Product ohjelmistokehityksessä tarkoittaa kokonaisuutta, jossa on vain kaikkein tärkeimmät ominaisuudet. Ohjelmakoodin tai Visual Studio -projektin muotoiluun ei kiinnitetä vielä tässä vaiheessa niin tarkkaa huomiota. MVP:n tarkoitus on demonstroida tuotteen

ominaisuuksia ja esittely- ja testausvaiheen jälkeen tuote voi muuttaa muotoaan jopa radikaalisti. MVP:ssä tärkeintä on loppukäyttäjän ja kehittäjätiimin välinen kommunikaatio tarvittavista muutoksista, jotta kehittäjätiimillä on koko ajan selkeä suunta kehityksessä (Techopedia 2020.)

Aloittelevana ohjelmistokehittäjänä on tärkeää osata luoda mahdollisimman nopeasti ja pienellä vaivalla ensimmäinen versio toteutettavasta tuotteesta, jotta se saadaan heti testaukseen. Integraatioliittymissä se on hieman vaikeampaa, koska putkeen liittyy monta eri osaa, jotta viesti kulkee alusta loppuun. Pelikehityksessä MVP-konsepti on hieman toimivampi, koska ominaisuuksia ja grafiikoita voi karsia pois niin helposti. Jos MVP:tä lähdetään kehittämään olisi hyvä tietää alkuun, että mitä kaikkea siihen kuuluu. Tarvitseeko integraatiossa olla kokonainen toimiva putki vai riittääkö kehittäjän oman tietokoneen ympäristössä toimiva versio, jota voi demonstroida asiakkaalle? Jos MVP:lle ei ole määritelty kokonaiskuvaa se voi paisua liian valmiiksi tuotteeksi, jota ei ole lähtökohtaisesti myyty asiakkaalle ja budjetti voi ylittyä turhaan. Laajuus olisi hyvä määrittää asiakkaan kanssa ennen kehittämisen aloitusta.

4.6 Viikko 6

Viikon aikana on tavoitteena saada työtilausintegraatiota edistettyä siihen pisteeseen, että myös peruminen, uudelleenajoittaminen ja päivittäminen toimii halutulla tavalla. Maanantaina on ensimmäinen testaus asiakkaan kanssa.

Maanantai 1.3

Päivän aikana aloitin Field Servicen päästä lähtevän päivityssanoman kartoitusta. Työtilauksen entiteetillä on paljon relaatiotauluja Dynamics 365 CRM:ssä ja monia näistä tarvitaan sanoman kentissä. Yhdeksi ongelmaksi on muodostunut päivämäärä- ja enum-tyyppiset kentät. XML-käännöksessä ei näy kenttiä, vaikka ne on kartoitettu.

Tiistai 2.3

Sain päivän aikana selvitettyä ongelmat XML-käännökseen liittyen. Huomiseksi on tarkoitus saada nykyinen työtilausintegraation versio käyttöön UAT -ympäristöön. Tässä tapauksessa luodaan uusi App Registeration Azuren Active Directoryssä, uusi käyttäjä Dynamics 365 CRM:ssä integraatioita varten ja Logic Appien sekä Azure-funktioiden julkaiseminen. Uuden integraatiokäyttäjän luominen vaatii henkilöä, jolla on hallintaoikeudet Microsoft 365 Admin Centeriin.

Keskiviikko 3.3

Aamulla sain työtilausintegraatioon integraatiokäyttäjän UAT-ympäristöön, joten nyt voin julkaista loput komponentit Azureen, jotta koko integraatioputki toimii Field Servicen suuntaan.

Päivän aikana sain myös päivitys- ja uudelleenajoitussanomia hyvään vaiheeseen. Ne on julkaistu DEV-ympäristöön ja integraatioarkkitehti saa tehtyä testejä nykyisellä versiolla.

Torstai 4.3

Päivän aikana oli tarkoitus saada työtilausintegraation pieniä puutteita korjattua nykyisessä kartoituslogiikassa. Koska Field Serviceen on nyt saatu käyttöliittymään nappula uudelleenajoitusta varten, on tarve tehdä Dynamics 365 CRM -järjestelmään uudet Business Eventit työtilauksen uudelleenajoitusta varten DEV- ja UAT-ympäristöihin. Tapahtumat luotiin Plugin Registration Tool -työkalulla.

Perjantai 5.3

Päivän aikana oli ensimmäiset asiakastestaukset työtilausintegraatiossa. Kaikki toimi muuten hyvin paitsi, että olin jättänyt työtilauksen lisäysosan versioon oman päivityslogiikan. Päivityslogiikka oli siellä siksi, että voin säästää aikaa omassa kehitystyössä. Jos päivityslogiikkaa ei ole jokaisella testikerralla, täytyy luoda uusi työtilausnumero ja siihen kuluu paljon aikaa.

Viikko 6 yhteenveto

Viikon aikana tein vain yhtä integraatiota, jossa on monia erilaisia osa-alueita. Lisäksi kaikkiin osa-alueisiin liittyy tietynlainen toiminnallisuus ja logiikka. Jos olisin aloittanut työn ilman mitään dokumentaatiota, olisin joutunut kysymään apua tai lisäohjeita arkkitehdilta useasti päivässä. Integraatio oli kuitenkin dokumentoitu hyvin Confluenceen, josta on kehittäjänä huomattavasti helpompi katsoa yleisellä tasolla tehtävän osan logiikkaa.

Integraatioiden hyvissä dokumentaatioissa on eriteltynä mm. tehtävä asia, käytettävät ympäristöt ja niihin autentikointiin liittyvät asiat, mahdolliset tarvittavat kartoitukset ja käytettävät entiteetit. Dokumentaation pohjalta kehittäjän työ on huomattavasti helpompaa, koska se poistaa arvailuun pohjautuvan kehityksen (AltexSoft Inc 2018). Dokumentaatio voi olla myös sopimuksessa eritelty asia, joten sen toimittaminen on yhtä tärkeää, kuin itse ohjelmiston.

Dokumentointia tehdään ohjelmistoprojekteissa monista eri syistä. Tärkein syy on se, että asiakkaalla ja toimittavalla yrityksellä on yhteinen kuva sovitusta kokonaisuudesta. Tämä on erityisen tärkeää muutoksen hallinnassa, koska sovittuun kokonaisuuteen ei tehdä muutoksia ilman erillistä prosessia (Fox 2021.) eCraftin ohjelmistoprojekteissa dokumentaatiolla on myös muita syitä, esimerkiksi olemassa olevien liittymien dokumentointi oman liiketoimintalinjan käyttöön. Tämä tulee ilmi tilanteessa, jossa vanhemmalla asiakkaalla on jokin toimitettu liittymä ja hiljattain on kehitetty sama liittymä uudestaan toisessa projektissa. Jos uudemmassa projektissa on tehty parannuksia liittymään, niin vanhemmalle asiakkaalle voi tarjota päivityspalvelua.

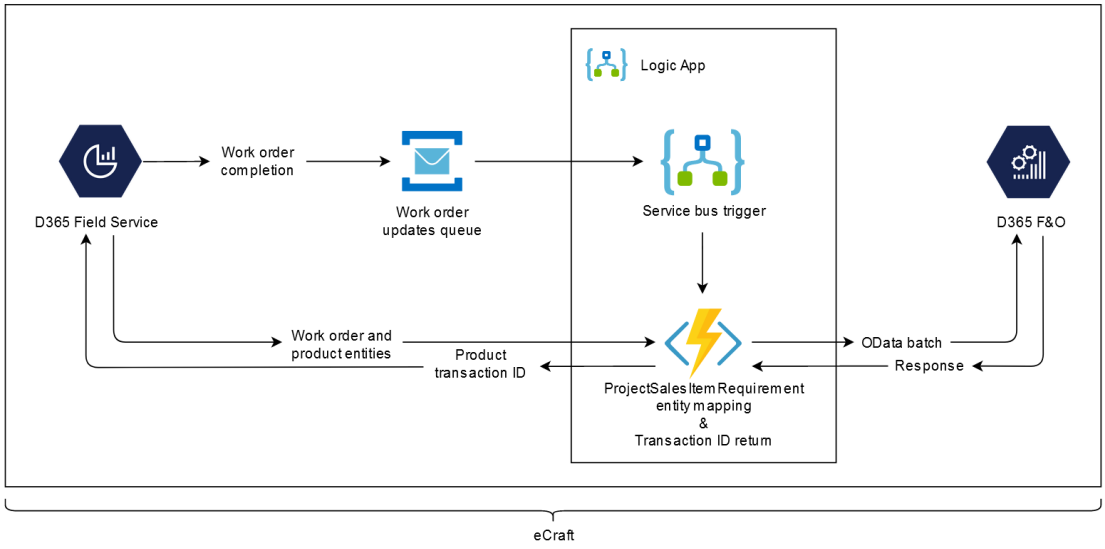
Omassa työssäni ei ole ollut vielä tarvetta tehdä dokumentaatioita itse, koska olen vielä työharjoittelija. Dokumentaatiot tehdään seniori- tai arkkitehtitasoisen henkilön toimesta ja toteutan liittymän sen mukaisesti. Tulevaisuutta varten on toki tärkeä osata myös tehdä itse dokumentaatioita, jotta ne on helpompi selittää asiakkaalle ja mahdollisille tuleville työharjoittelijoille.

4.7 Viikko 7

Viikon aikana jatketaan työtilausintegraatiota. Nykyisellä aikataululla päivitys- ja uudelleenajoittamislogiikat pitäisi saada hyvään vaiheeseen, jotta ne voidaan siirtää UAT-ympäristöön asiakkaan testattavaksi.

Maanantai 8.3

Aloitin tänään uuden integraation samassa projektissa, jossa teen työtilausintegraatiota. Tavoitteena on saada valmistuneen työtilauksen toimitettujen tuotteiden tiedot Dynamics 365 ERP:n sisällä olevien projektien myydyiksi tuotteiksi (Kuvio 6).



Kuvio 6: Dynamics 365 FS - ja Dynamics 365 ERP -järjestelmien välinen integraatioputki

Yksi tärkeimmistä osista integraatiossa on transaktion onnistuminen Dynamics 365 ERP:ssä. Joko kaikkien rivien täytyy onnistua tai ei yhdenkään. Transaktion onnistuessa Dynamics 365 ERP:n rajapinta palauttaa transaktion tunnisteen, joka tallennetaan takaisin Field Servicen toimitetuille tuotteille.

Tiistai 9.3

Jatkoin päivän aikana eilen aloitettua toimitettujen tuotteiden integraatiota. Päivän aikana oli tarkoitus jatkokehittää ja testata eilen saatua ensimmäistä versiota. Ongelmiksi muodostui se, että Dynamics 365 CRM:ään ei ole vielä tehty puuttuvaa relaatiota työtilauksen ja projektin välille, sekä se, että Dynamics 365 ERP:n päähän ei ole vielä tehty tarvittavaa entiteettilaajennosta. Työ jatkuu muun projektitiimin työn edistyttyä.

Keskiviikko 10.3

Päivän aikana jatkoin työtilausintegraation kehittämistä. Viimeisimmän refaktoroinnin jäljiltä integraatioon oli jäänyt ongelma liittyen Azure-funktion palautusarvoon. Kartoitusfunktion sisällä kutsutaan toista luokkaa, joka kartoittaa oikeanlaisia sanomia riippuen käyttäjän antamasta arvosta. Ongelmaksi oli muodostunut se, että funktion sisällä olevat luokat eivät palauttaneet virhettä sellaisen sattuessa vaan omanlaisen palautusobjektin. Poistamalla palautusobjektin käytön virheen sattuessa korjasi ongelman.

Torstai 11.3

Päivällä pidettiin kahdesti viikossa pidettävä integraatiotiimin kokous. Kokouksesta teki mielenkiintoisen se, että katsoimme yhdessä Microsoft Ignite -tapahtuman yhtä esitystä, jossa kerrottiin uudesta teknologiasta liittyen ARM-mallien luontiin. Koska työtilausintegraatiossa yksi seuraavaksi tehtävistä töistä on ARM-mallien luonti DevOpsia varten, saan hyvän tilaisuuden kokeilla uutta teknologiaa ja raportoida kokemuksia muille tiimin jäsenille.

Perjantai 12.3

Päivän aikana työtilausintegraation CRM-arkkitehti piti testaustilaisuuden asiakkaan kanssa nykyisen työtilausintegraation toiminnasta. Kaikki tuntui toimivan lähellä haluttua tapaa, mutta muutamia korjausehdotuksia tuli. Vastaustiedostot tulivat väärällä enkoodauksella ja muutamia elementtejä piti asettaa vaihtoehtoisiksi työtilausten viennissä Field Serviceen.

Viikko 7 yhteenveto

Viikon aikana tein useampia eri asioita eri projekteissa, jotka vaativat erilaisia toimenpiteitä versionhallinnassa. Kehitystuloksien vieminen versionhallintaan ja niiden julkaiseminen tuotantoympäristöön vaativat omanlaisensa prosessin.

eCraftin projekteissa käytetään usein GitFlow-mallista versionhallintaa. Versionhallinnassa käytetään useita haaroja eri tarkoituksia varten. Integraatioiden kehityksessä käytetään ominaisuus tai feature-haaroja, joissa itse kehitystä tehdään. Kun ominaisuutta saadaan kehitettyä, muutoksia siirretään koodikatselmoinneilla kehitys tai develop-haaraan. Develop-

haarassa säilytetään sen hetkistä kehitysversiota, joka julkaistaan DEV-ympäristöön, jossa kehittäjät voivat testata versiota. Develop-haarasta julkaistaan myös UAT-ympäristöön siinä vaiheessa, kun versio on saatu tarpeeksi hyvään vaiheeseen testausta varten. Feature-haaroista voidaan viedä myös katselmoinnilla tuotanto tai master-haaraan (Atlassian 2021.)

Tuotannossa havaituista puutteista tai huoltotoimenpiteistä luodaan korjaus tai hotfix-haara, joka pohjautuu tuotannossa olevaan master-haaraan. Hotfix-haarassa tehdään tarvittava korjaus, jonka jälkeen haara yhdistetään koodikatselmoinnilla master- ja develop-haaroihin (Atlassian 2021.) Tämän jälkeen haluttavat muutokset viedään tuotantoon DevOpsin julkaisuputkella.

Työssäni versionhallintaa käytetään koko ajan, mutta silti osaaminen on vielä hieman vajanaista. Kokemuksella pienemmät tietämättömyyden osat selviävät, mutta syvempää osaamista olisi hyvä saada. Sain koulutuksen alussa harjoittelumateriaalia Gitin käyttöön, mutta en ole vielä ehtinyt opetella sitä. Harjoittelumateriaalissa on osia Gitin sisällä tapahtuvasta toiminnasta, joten siitä olisi hyvä lähteä liikkeelle.

4.8 Viikko 8

Viikon aikana on tavoitteena saada työtilausintegraation vaiheet valmiiksi, mutta tämän onnistuminen on myös riippuvainen asiakkaan aktiivisesta testauksesta. Kehittäjätiimin näkökulmasta on joskus vaikea nähdä kaikkia asiakkaan tarpeita ja nämä tarpeet huomataan parhaiten asiakkaan testauksella.

Maanantai 15.3

Aloitin päivän aikana yksikkötestien tekemisen työtilausintegraatioon. Yksikkötesteissä tarvitsee kattaa työtilausjärjestelmästä tulevien tiedostojen käsittelyyn, työtilausjärjestelmään menevien vastaustiedostojen kartoitukseen sekä Field Service -yhteyden luomiseen tarvittavat toiminnallisuudet. Yksikkötesteillä varmistetaan DevOpsin käännösputkessa ohjelmiston oikeanlainen toiminta.

Tiistai 16.3

Päivän aikana sain lisää kommentteja työtilausintegraatiossa tehdyistä testeistä asiakkaan toimesta. Asiakas testasi Field Servicen päästä lähettää päivitysviestejä työtilausjärjestelmään muuttamalla UAT-ympäristössä työtilauksen status valmiiksi. Miltei kaikki tuntui olevan asiakkaan näkökulmasta olevan oikein muutamia pieniä yksityiskohtia lukuun ottamatta. Työtilausjärjestelmään menevissä tiedostoissa lähdejärjestelmäksi pitää asettaa asiakkaan vanha järjestelmä, vaikka lähde on oikeasti Field Service. Lisäksi päivämäärien kentissä pitää käyttää muotoa, jolla kohdejärjestelmä tunnistaa ne UTC-ajaksi.

Keskiviikko 17.3

Päivän aiheena oli tehdä Dynamics 365 ERP -kehittämistä samassa projektissa, jossa olen tehnyt käyttäjäintegraatiota. Kehittämiseen sisältyi uusien taulu- ja entiteettilaajennoksien tekeminen sekä olemassa olevien käyttöliittymän lomakkeiden ja valikon valintojen muokkaaminen. Integraatiotiimin vanhempi kehittäjä oli kouluttamassa minua ja toista tiimin nuorempaa työntekijää aiheesta. Viimeisimmästä ERP-kehittämisestä on jo useita kuukausia, joten päivä oli erittäin opettavainen uusien asioiden oppimisesta ja vanhojen asioiden kertaamisesta.

Torstai 18.3

Päivän aiheena oli jatkaa työtilausintegraation yksikkötestien tekoa ja samalla käydä keskustelua CRM- ja integraatioarkkitehtien kanssa asiakkaan toiveista liittyen sanomien muodostukseen työtilausjärjestelmästä ja -järjestelmään.

Perjantai 19.3

Päivän aikana jatkoin toimitettujen tuotteiden integraatiota. Yhdeksi päivän ongelmaksi muodostui Dynamics 365 ERP -järjestelmän tuotteiden konfiguraation puute. Järjestelmä luo sen sisällä olevien projektien tuotteista automaattisesti laskuja ja laskuilla olevilla tuotteilla täytyy olla tyyppikategoria, esim. palvelu tai tuote. Järjestelmän tuotteiden kategorian asetuksen jälkeen integraatio toimi halutulla tavalla.

Viikko 8 yhteenveto

Tein viikon aikana monia erilaisia yksikkötestejä työtilausintegraation Visual Studio -ratkaisussa. Tähän mennessä olen käyttänyt xUnit- ja Moq-kirjastoja testien tekemiseen. Monia muitakin vaihtoehtoja on, mutta en ole vielä kokeillut näitä.

Yksikkötestejä tehdään siksi, että niillä voidaan tarkistaa tuotantoputken eri vaiheissa Visual Studio -projektihierarkiassa olevien liittymien toimintaa. Lisäksi yksikkötestit auttavat sellaisessa tilanteessa, jossa kesken ohjelmiston kehitystä projektiin otetaan uusia kehittäjiä. Testit auttavat uutta kehittäjää ymmärtämään nykyisen version toimintaa. Testeille määritellään oletusarvoinen toiminta, jolla Visual Studio -projektissa olevien liittymien tulisi toimia. Yksikkötestit ovat projektikokonaisuuden alimman tason testejä. Aikaisemmin olen maininnut Xray-testeistä, joita tehdään Jiraan. Xray testit ovat integraatiotason testejä eli niillä testataan koko putkea yhdestä järjestelmästä toiseen (Guru99 2021a).

eCraftilla yksikkötestejä käytetään toiminnallisuuden tarkistamiseen kehittäjän omassa ympäristössä ja myöhemmin DevOpsin käännösputkessa. Jos yksikkötestit eivät mene läpi, käännös epäonnistuu ja julkaisu ympäristöön perutaan. DevOpsista ja sinne julkaisusta lisää myöhemmin. Lisäksi yksikkötestit toimivat oleellisesti muutoksenhallinnassa. Yksikkötestit tarkistavat

liittymän toiminnan muutoksen jälkeen ja varmistavat, ettei liittymä hajoa ennen asennusta (Software Testing Fundamentals 2020.)

Yksikkötestien tekemisestä pidettiin minulle ja muille aloitteleville työharjoittelijoille koulutusta viime syksynä, mutta omakohtainen harjoittelu on jäänyt aika vähälle. Yksikkötestien kehittämien on mielestäni vaikeampaa kuin itse liittymän kehittäminen. Tämä johtuu siitä, että osaamiseni xUnit- ja Moq-kirjastojen kanssa on vielä suhteellisen heikkoa. Monimutkaisen sovelluslogiikan testaaminen on vielä itselle vaikeaa ja tarvitsen vielä lisää opettelua aiheen parissa. Erityisesti callback-logiikka olisi tärkeä osata, jotta voi luoda testitapauksia funktioille, joissa käytetään silmukoita.

4.9 Viikko 9

Viikon aikana on tavoitteena saada työtilausintegraation ARM-mallit tehtyä jatkuvaa julkaisuputkea varten sekä saada toimitettujen tuotteiden integraatio valmiiksi ja asiakkaan testattavaksi.

Maanantai 22.3

Päivän aikana sain tehtyä yksikkötestit toimitettujen tuotteiden integraatioon sekä korjattua muutamia ongelmakohtia rivien kirjoittamisessa Dynamics 365 ERP:hen. Yksi integraation tärkeimmistä teknisistä ominaisuuksista on se, että joko kaikkien rivien kirjoittaminen onnistuu tai ei yhdenkään. Huomasin testauksessa ongelman, jossa yksi rivien kirjoittamisesta epäonnistui, kun yritti kirjoittaa riviä, jolla on väärä määräyksikkö. Tapahtuma epäonnistuu, mutta kaksi kolmesta rivistä kirjoittui kuitenkin järjestelmään, mikä ei ole haluttu toiminnallisuus. Muuttamalla tallennuslogiikkaa hieman ongelma ratkesi.

Tiistai 23.3

Sain päivän aikana tehtyä ARM-mallit työtilausintegraatioon käyttäen Microsoftin uutta Bicep-ohjelmointikieltä. Suurempia ongelmia ei vielä löytynyt, mutta ongelmat yleensä selviävät siinä vaiheessa, kun malleja yritetään käyttää DevOpsin julkaisuputkessa.

Keskiviikko 24.3

Päivän aikana sain uuden tehtävän, jossa jo kehitetyn integraation toimintalogiikkaa tarvitsee muuttaa. Integraatiossa kirjoitetaan Dynamics 365 ERP:n General Ledger -rajapintaan kirjanpitotapahtumia perustuen kortistotiedostosta tuleviin riveihin. Muutoksessa nykyinen liittymä muutetaan käyttämää XML-tiedostoja. Muutoksesta ei ole vielä dokumentaatiota, joten tässä vaiheessa voin vain kehittää alustavan toteutuksen.

Torstai 25.3

Päivän aikana oli tarkoitus saada toimitettujen tuotteiden integraatio hyvään vaiheeseen huomista asiakastestiä varten. Yksi tärkeimmistä asioista oli testata nykyistä ratkaisua ja saada yleisimmät ongelmakohdat ratkaistua. Yksi ongelmista oli saada tietoon kaikki Dynamics 365 ERP:n projektientiteetin ominaisuudet, jotka estävät rivien kirjoituksen. Kaikki yleisimmät ongelmat löytyivät.

Perjantai 26.3

Aamun aikana oli vielä joitain ongelmia toimitettujen tuotteiden integraatiossa rajapintakutsujen suhteen, mutta ne saatiin ratkottua ennen testauksen alkua. Testaus sujui integraation osilta hyvin, koska oikeita virheitä suorituksien aikana ei sattunut.

Viikko 9 yhteenveto

Viikon aikana sain monia muutostöitä ja valvomiseen liittyviä töitä ja siksi en saanut ARM-malleja valmiiksi. Niiden tekeminen siirtyy ensi viikolle.

eCraftilla ja monissa muissa ohjelmistoyrityksissä isoissa projekteissa tehdään laaja suunnitelma tarvittavista kokonaisuuksista ennen työn aloitusta. Suunnitelmassa dokumentoidaan esimerkiksi integraatioliittymän kenttien kartoitus ja toimintalogiikka. Kuitenkin kehityksen aikana tai tuotannossa asiakas voi huomata omasta näkökulmastaan puutteen liittymän toiminnassa ja täten haluaisi muutoksen olemassa olevaan ratkaisuun.

Tällaisissa tapauksissa tehdään muutospyyntö tai Change Request. Muutospyynnöissä asiakas kuvailee nykyisessä ratkaisussa olevan puutteen ja halutun uuden toiminnallisuuden, jonka jälkeen se päätyy käsiteltäväksi ohjelmiston toimittajayritykselle. Muutospyyntöjen kuuluminen projektityöhön riippuu usein asiakkaan kanssa tehdystä sopimuksesta, joten joskus muutospyyntöjä ei käsitellä muutoksina vaan korjaustöinä. Jos muutos on projektin kokonaisuuden ulkopuolella, tällöin kyse on muutoksesta. Useimmiten muutostöissä kyse on vain ylimääräisistä kenttäkartoituksista tai joidenkin kartoituksien muuttamisesta, jolloin pyyntö hyväksytään sovitulla työmäärällä ja toteutetaan sovitulla aikataululla. Jos muutos liittyy CRM:n sisäiseen toiminnallisuuteen, niin silloin projektin vastaava CRM-arkkitehti huolehtii muutoksen toteuttamisesta. Itse muutoksen lisäksi täytyy päivittää olemassa olevat dokumentaatiot vastaamaan muutosta (Tallyfy 2021.)

Jos muutospyyntöön liittyvä toteutus on jo tuotannossa, silloin käytetään aiemmin mainittua hotfix-periaatetta. Muuten kehitys tehdään normaalisti feature-haaran kautta dev-haaraan. Tehtyjen muutoksien jälkeen kyseessä oleva liittymä menee uudestaan testaukseen ja se hyväksytään asiakkaan puolesta (Guru 99 2021b.)

Oma osaamiseni muutoksien teossa on hyvällä mallilla, mutta itse muutospyynnön prosessia en ole vielä oppinut. Olen kyllä ollut asiakkaan kanssa yhteydessä tarvittavista muutoksista, mutta vain käytännön ratkaisujen puolella. Projektinhallinnan osalta tämä on vielä hieman mysteeri. Tulevaisuudessa tulen olemaan enemmän vastuussa omista työmääräarvioista, jolloin muutospyyntöjen prosessi tulee varmasti silloin enemmän tutuksi.

4.10 Viikko 10

Viikon aikana on tavoitteena saada ARM-mallit tehtyä työtilausintegraatioon ja DevOps-putket tehtyä jatkuvaa julkaisua varten.

Maanantai 29.3

Päivän aikana etsin ongelmaa Bicep-malleistani ja omaksi harmikseni se paljastui olevan vain yhden kirjaimen puuttuminen yhdestä rajapintayhteyden elementistä. Toinen päivän kohokohta oli vanhemman kehittäjän pitämä oppitunti DevOpsin käännös- ja julkaisuputkista. Viikon yhtenä aiheena on saada DevOps kuntoon työtilaus- ja muiden integraatioiden osalta.

Tiistai 30.3

Päivän aiheena oli tehdä tarvittavat muutokset integraatioon, johon kaksi viikkoa sitten tehtiin Dynamics 365 ERP -laajennoksia. Integraatiossa kirjoitetaan tuotteita järjestelmään api-rajapinnan kautta, joten muutos ei ole monimutkainen.

Keskiviikko 31.3

Päivän aiheena oli tehdä DevOps-käännösputkea työtilausintegraatiota varten. Yhdeksi ongelmaksi muodostui se, että käännösputken pitäisi saada Bicep-tiedostot käännettyä JSON-muotoon, että ne voidaan julkaista Azureen.

Torstai 1.4

Päivän aiheena oli saada työtilausintegraation DevOps-julkaisuputki valmiiksi ja testattua. Pieniä ongelmia muodostui parametrisoinnin ja Logic Appien julkaisun kanssa, mutta niistä selvittiin nopeasti. Seuraavana vuorossa on juuri julkaistujen integraatioputkien testaus ennen UAT-julkaisua.

Viikko 10 yhteenveto

Viikon aikana on ollut tehtävänä tehdä DevOpsin kautta logiikkaa jatkuvaa käyttöönottoa varten työtilausintegraation projektissa. Joissain projekteissa DevOpsin rakennus annetaan asian osaaville henkilöille. Tämän projektin lähes kaikki integraatiot on tehty nuorempien kehittäjien toimesta, joten on sopivaa, että samat henkilöt hoitavat koko toimituksen alusta

loppuun. Tässä projektissa integraatioita varten käytetään DevOpsia CI/CD -tarkoituksella eli DevOpsin sisällä integraatioiden lähdekoodit testataan Visual Studio -projektien sisällä olevien yksikkötestien perusteella, projektit käännetään ja lopulta julkaistaan Azureen. DevOps-putki laukaistaan joko manuaalisesti tai automatisoidusti esimerkiksi versionhallinnan haaran commitista (Microsoft 2019e.)

Tästä syystä yksikkötestien tekeminen integraatioihin on tärkeää julkaisun kannalta. Testauksen ja lähdekoodin käännön jälkeen ARM-mallit ja Visual Studio -projektien funktiot kerätään kokoon ja siirrytään julkaisuvaiheeseen. Julkaisussa ARM-mallien perusteella julkaistaan Azure-komponentit ja Visual Studio -projektien sisältö Azure funktioihin. Julkaisun jälkeen komponentit ovat valmiita käytettäväksi, jos yhdessäkään komponentin julkaisuprosessissa ei tapahtunut virheitä (Microsoft 2021c.)

DevOps-osaamiseni käännös- ja julkaisupuolella on jo suhteellisen hyvällä tasolla, mutta hiottavaa toki vielä olisi. eCraftilla ei ole parhaita käytäntöjä DevOpsin putkien suhteen, mutta erilaisia toimintamalleja on ja niiden toimintaa olisi hyvä opiskella osaamisen parantamiseksi. DevOpsin taustalla pyörii käyttäjäprofiileja ja autentikointeja, joista minulla ei ole toistaiseksi vielä mitään tietoa, joten tästä olisi hyvä löytää lisää tietoa tai osallistua koulutuksiin. Integrations & API -liiketoimintayksikössä on myös hyvää osaamista DevOpsiin liittyen, joten hyviä opettajia on saatavilla.

5 Yhteenveto

Opinnäytetyön alussa olin hyvin epävarma omasta osaamisestani ja työprosessit tuntuivat hankalilta. Esimerkiksi muutoksen tekemisen prosessi oli vieras ja vaikea ymmärtää sekä en tiennyt kaikkia toimintatapoja mitä siinä tarvitaan. Työpäivän aikana tuntui, että oli suuret suorittamispaineet osaamisen kehittymiseen.

Kehitys on kuitenkin ollut hyvää. En tiennyt ennen aloitusta mitään yksikkötesteistä tai ohjelmiston osa-alueiden dokumentoinnista ja nyt osaamiseni ja tietämykseni on jo huomattavasti paremmalla tasolla näiden osalta. Ehkä suurimpana heikkoutenani pidin sitä, että omaan tekemiseen ei ollut vahvaa luottoa ja itsevarmuus oli heikkoa. Seurantajakson aikana olen päässyt tekemään monia erilaisia töitä erilaisissa projekteissa erilaisien ihmisien kanssa ja uskon, että tämä on antanut tarvittavaa itsevarmuutta tulevaisuutta varten. Projektitiimien jäsenet myös kannustivat tähän seurantajakson aikana olleessa kehityskeskustelussa. Paremman itsevarmuuden kautta työpäivä tuntuu myös huomattavasti rennommalta, koska paineet eivät ole niin korkealla.

5.1 Oman työn analysointi

Työni analysoinnissa olen huomannut, että pienienkin osa-alueiden hyvään osaamiseen tarvitaan paljon kokemusta ja kertausta. Viime syksynä olin koulutuksessa miltei joka aiheesta, mutta oikeaan osaamiseen kuitenkin tarvitaan toistoja ja käytäntöä. Koulutusohjelman aikana käydyt koulutukset olivat enemmänkin kosketuspinta aiheeseen ja vasta myöhemmin käytännön töissä tai harjoituksissa saa tehokasta oppia ja kokemusta käsiteltävästä asiasta.

5.2 Tulokset ja kehittämisehdotukset yritykselle

eCraft tulee hyödyntämään opinnäytetyötä tulevia koulutuksia varten. Uudet oppilaat voivat lukea opinnäytetyön ja saada parempaa kuvaa tulevasta työstä. Oppilailla voi olla samanlainen näkemys tulevasta työstä kuin minulla oli eli tietää peruskonsepteja ohjelmistokehityksestä, mutta konkreettiset työtavat ja prosessit ovat täysin hämärän peitossa. Opinnäytetyöllä voidaan myös validoida nykyisiä toimintatapoja ja yrittää löytää kehittämisen kohteita koulutusohjelmissa ja prosesseissa. Minun saapumiseräni oli vasta ensimmäinen kokeilu devcrafters-koulutusohjelmassa ja uusia on todennäköisesti tulossa. Yrityksen nykyisellä kasvutahdilla uusia kehittäjiä varmasti tarvitaan tulevaisuudessa ja voi olla, että itsekin pääsen pitämään jotain koulutusta jostakin aiheesta.

Tulevissa kehittäjille tarkoitetuissa koulutusohjelmissa voisi olla enemmän läksyjen tapaisia tehtäviä koulutuksien välissä, jossa kehitetään jotain integraatiota eteenpäin. Esimerkiksi jos viikon alussa opetellaan Logic Appien ja Function Appien toimintaa, niin koulutuksien jälkeen voisi olla yksi välipäivä koulutuksista, jolloin ideoiden perusteella integraatioita voisi kehittää eteenpäin. Lisäksi ensimmäisissä projektitehtävissä olisi tärkeää, että toteutettava liittymä olisi hyvin ja selkeästi dokumentoitu ja suunniteltu. Juuri aloittanut ja kokematon ohjelmistokehittäjä on hyvin epävarma toteutettavasta työstä ja omista taidoistaan.

Lähteet

Sähköiset

AltexSoft Inc 2018. Software Documentation Types and Best Practices. Blogikirjoitus. Prototypr.io. Viitattu 10.3.2021.
https://blog.prototypr.io/software-documentation-types-and-best-practices-1726ca595c7f

Atlassian 2020. What is Confluence Cloud?. Viitattu 8.2.2021.
https://support.atlassian.com/confluence-cloud/docs/what-is-confluence-cloud/

Atlassian 2021. Gitflow Workflow. Viitattu 15.3.2021.
https://www.atlassian.com/git/tutorials/comparing-workflows/gitflow-workflow

Codebots 2017. What are environments in software development? A guide to the development, beta, and production environments. Viitattu 12.2.2021.
https://codebots.com/app-development/what-are-environments-in-software-development-a-guide-to-the-development-beta-and-production-environments

Coutermarsh, M. 2015. Jr. Developers #1: Pull Requests & You. Viitattu 12.4.2021.
https://medium.com/@mscccc/jr-developers-1-pull-requests-you-39a11c3bdd94

Encore 2021. D365 Finance & Operations. Viitattu 8.2.2021.
https://www.encorebusiness.com/solutions/microsoft-dynamics-365-finance-operations/

Fox, C. 2021. Why Writing Software Design Documents Matters. Viitattu 15.4.2021.
https://www.toptal.com/freelance/why-design-documents-matter

Gadzinowski, K. 2021. Trunk-based Development vs. Git Flow. Viitattu 9.2.2021.
https://www.toptal.com/software/trunk-based-development-git-flow

Guru99. 2021a. Unit Testing Tutorial: What is, Types, Tools & Test EXAMPLE. Viitattu 19.3.2021.
https://www.guru99.com/unit-testing-guide.html

Guru99. 2021b. Change Control Process in Software Engineering with Steps. Viitattu 29.3.2021. https://www.guru99.com/change-control-business-analyst.html

HR Morning. 2021. Performance Review Resources. Viitattu 14.4.2021.
https://www.hrmorning.com/performance-review/

Hyperext 2017. The importance of planning when it comes to systems development. Viitattu 29.1.2021.
https://www.hyperext.com/software-development/the-importance-of-planning-when-it-comes-to-systems-development/

Larsen, Z. 2018. The Art of a Well Composed Pull Request. Viitattu 6.2.2021.
https://medium.com/faun/the-art-of-a-well-composed-pull-request-3815fc7e9610

Microsoft 2015a. X++ Language Programming Guide. Viitattu 5.2.2021.
https://docs.microsoft.com/en-us/dynamicsax-2012/developer/x-language-programming-guide

Microsoft 2015b. C# Coding Conventions (C# Programming Guide). Viitattu 12.2.2021.
https://docs.microsoft.com/en-us/dotnet/csharp/programming-guide/inside-a-program/coding-conventions

Microsoft 2018. Refactor code. Viitattu 23.4.2021.
https://docs.microsoft.com/en-us/visualstudio/ide/refactoring-in-visual-studio?view=vs-2019

Microsoft 2019a. Get started guide for Azure developers. Viitattu 4.2.2021.
https://docs.microsoft.com/fi-fi/azure/guides/developer/azure-developer-guide

Microsoft 2019b. Business events and Azure Service Bus. Viitattu 2.2.2021.
https://docs.microsoft.com/en-us/dynamics365/fin-ops-core/dev-itpro/business-events/how-to/how-to-servicebus

Microsoft 2019c. Welcome to the Visual Studio IDE. Viitattu 5.2.2021.
https://docs.microsoft.com/en-us/visualstudio/get-started/visual-studio-ide?view=vs-2019

Microsoft 2019d. Overview of Dynamics 365 Dynamics 365 CRM Professional. Viitattu 5.2.2021.
https://docs.microsoft.com/en-us/dynamics365/sales-professional/sales-professional-overview

Microsoft 2019e. What is Azure Pipelines?. Viitattu 1.4.2021.
https://docs.microsoft.com/en-us/azure/devops/pipelines/get-started/what-is-azure-pipelines?view=azure-devops

Microsoft 2020a. Get Associate or add an Azure subscription to your Azure Active Directory tenant. Viitattu 4.2.2021.
https://docs.microsoft.com/en-us/azure/active-directory/fundamentals/active-directory-how-subscriptions-associated-directory

Microsoft 2020b. Introduction to projects and solutions. Viitattu 5.2.2021.
https://docs.microsoft.com/en-us/visualstudio/get-started/tutorial-projects-solutions?view=vs-2019

Microsoft 2020c. Introduction to Azure Functions. Viitattu 26.4.2021. https://docs.microsoft.com/en-us/azure/azure-functions/functions-overview

Microsoft 2021a. A tour of the C# language. Viitattu 5.2.2021. https://docs.microsoft.com/en-us/dotnet/csharp/tour-of-csharp/

Microsoft 2021b. What is Azure DevOps?. Viitattu 8.2.2021. https://docs.microsoft.com/en-us/azure/devops/user-guide/what-is-azure-devops?view=azure-devops

Microsoft 2021c. What are ARM templates?. Viitattu 15.4.2021. https://docs.microsoft.com/en-us/azure/azure-resource-manager/templates/overview

Microsoft 2021d. What is Azure Logic Apps?. Viitattu 23.4.2021. https://docs.microsoft.com/en-us/azure/logic-apps/logic-apps-overview

Microsoft 2021e. What is Azure Service Bus?. Viitattu 6.5.2021. https://docs.microsoft.com/en-us/azure/service-bus-messaging/service-bus-messaging-overview

Mozilla 2021. HTTP. Viitattu 6.5.2021. https://developer.mozilla.org/en-US/docs/Web/HTTP

Murphy, M. 2018. Here's Why You Still Need To Conduct Performance Reviews With Your Employees. Forbes. Viitattu 19.2.2021. https://www.forbes.com/sites/markmurphy/2018/11/04/heres-why-you-still-need-to-conduct-performance-reviews-with-your-employees/?sh=2850a4c819f8

OData 2021. Etusivu. Viitattu 10.2.2021. https://www.odata.org/

Software Testing Fundamentals. Unit Testing. Viitattu 15.4.2021. https://softwaretestingfundamentals.com/unit-testing/

Tallyfy. 2021. What is a Change Request and How to Manage It. Viitattu 15.4.2021. https://tallyfy.com/change-request/

Techopedia 2020. Minimum Viable Product (MVP). Viitattu 29.2.2021. https://www.techopedia.com/definition/27809/minimum-viable-product-mvp

Kuviot