Frends-integraatioalustan hoitomalli

HÄMEEN AMMATTIKORKEAKOULU
HÄME UNIVERSITY OF APPLIED SCIENCES

Ammattikorkeakoulututkinnon opinnäytetyö

Tietojenkäsittelyn koulutus

Kevät 2022

Niko Lönnroos

HAMK
HÄMEEN AMMATTIKORKEAKOULU
HÄME UNIVERSITY OF APPLIED SCIENCES

Tietojenkäsittelyn koulutus Tiivistelmä
Tekijä Niko Lönnroos Vuosi 2022
Työn nimi Frends-integraatioalustan hoitomalli
Ohjaaja Lasse Seppänen

Tämän opinnäytetyön tarkoituksena oli tuottaa hoitomalli Yhtiö Y:n Frends-integraatioalustalle. Tarve hoitomallille on tullut, koska Frends-integraatioalustan häiriöiden, ongelmien ja muutosten valvominen on ollut aikaisemmin pelkästään Yhtiö Y:n vastuulla ja tähän haluttiin parannuskeinoja, joiden avulla luodaan turvallisempaa tulevaisuutta liiketoiminnalle. Hoitomallin tarkoituksena on antaa selvät ohjeet ja vastuualueet häiriöiden, ongelmien ja muutosten aikana toimimiseen. Opinnäytetyön toimeksiantajana toimii Yhtiö Y, joka toimii finanssialalla.

Opinnäytetyön tietopohja koostuu netistä etsityistä artikkeleista. Teoriaosuudessa kerrotaan perustiedot integraatiosta, integraationmalleista, API:sta, iPaaS:sta ja Frends-alustasta. Opinnäytetyö on toiminnallinen. Käytännönosuus on toteutettu kehitysprojektina. Kehitysprojekti toteutettiin yhdessä alustan toimittajan kanssa. Opinnäytetyöntekijän vastuulla oli etsiä ja raportoida aineistoa asiantuntijoiden avustuksella kehitysprojektin palavereihin, mutta varsinaista päätäntävaltaa tekijällä ei ole ollut lopullisiin ratkaisuihin.

Opinnäytetyön tuloksena saatiin luotua toimiva hoitomalli Frends-integraatioalustalle. Suurin muutos on, että vastuu häiriöiden seuraamisesta sekä selvitystyön aloituksesta Frends-integraatioalustalla siirtyi toimittajalle. Kehitysprojektin avulla tuotettiin tarpeelliset ohjeet molemmille osapuolille (Yhtiö Y – toimittaja) häiriötilanteissa toimimiseen. Molemmilla osapuolilla on myös tiedossa, miten muutosten ja ongelmien kanssa toimitaan. Tulevaisuudessa yhteistyötä toimittajan kanssa on tarkoitus tiivistää esim. integroimalla tikettijärjestelmät yhteensopiviksi.

Avainsanat Frends, integraatio, pilvipalvelut ja hoitomalli
Sivut 27 sivua ja liitteitä 1 sivu

HAMK
HÄMEEN AMMATTIKORKEAKOULU
HÄME UNIVERSITY OF APPLIED SCIENCES

Degree Programme in Business Information Technology — Abstract
Author Niko Lönnroos — Year 2022
Subject Frends integration platform management model
Supervisors Lasse Seppänen

The purpose of this thesis was to produce a management model for corporation Y Frends-integration platform. The need for the model has arised, because Frends integration platform faced disruptions or troubles due to the fact that Company Y has been in charge of the changes happening on the platform. To this Frends was looking for improvement in order to create a safer business environment towards the future. The purpose of the management model is to give clear instructions and divide responsibilities while working on disruptions or problems. The thesis was commissioned by company Y which operates in Finances.

The thesis knowledge foundation consists of internet articles. In the theory portion the basic information about integration, integration models, API´s, iPaaS and Frends-platform is covered. The practical portion was accomplished as a development project. The development project was done in partnership with the provider of the platform. The person doing the development project was responsible for seeking information and utilizing experts to deliver the found material for project meetings, however he did not have final decision power for the chosen solutions.

A result of the thesis was a working care-model for Frends integration platform. The biggest change was that the responsibility of following disruptions and the start of an inquiry on the Frends integration platform was shifted to the provider of the platform. With the help of the development project sufficient guides were offered to both parties Company Y and Frends on how to operate during disruptions. Both parties also now acknowledge how to act if any problems arise. In the future the goal is to continue and intensify the partnership for example by integrating the ticket systems to match.

Keywords Frends, integration, cloud services and management model
Pages 27 pages and appendices 1 page

Sanasto

API	Application Programming Interface, ohjelmointi rajapinta
APIM	Applivation Programming Interface Management eli API-hallintaa
BPMN	Business Process Model and Notation, graafinen liiketoimintaprosessien esitysmuoto
CSV	Comma-seperated values, tiedostomuoto, jossa tiedot erotellaan pilkun avulla ja tallennetaan tekstitiedostoksi
DIH	Digital Integration Hub, Gartnerin lanseeraama nykyaikainen integraatiokeskus
EAI	Enterprise application integration, integraatiot englanniksi
EDIFACT	Electric Data Interchange for Administration, Commerce and Transport, Euroopan talouskomission määrittelemä tiedon esitystapa
EiPaaS	Enterprise Integration Platform as a Service, Yritystason iPaaS muoto
ESB	Enterprise service bus, integraatiomalli
FTP	File Transfer Protocol, tiedonsiirtomenetelmä
FTPS	File Transfer Protocol Secure, turvallisempi tiedonsiirtoprotokolla
HTML	Hypertext Markup Language, avoimesti standardoitu merkintäkieli
HTTP	Hypertext Tranfer Protocol, protokolla tiedonsiirtoon, etenkin selaimille ja www-sivuille
iPaaS	Integration Platform as a Service, pilvipohjainen integraatioalusta, palveluna
ITIL	Information Technology Infrastructure Library, prosessikehys
JSON	JavaScript Object Notation, tiedostomuoto tiedonvälitykseen
PHP	Hypertext Preprocessor, ohjelmointikieli
REST	Representational State Transfer, ohjelmointirajapintojen toteuttaminen
SFTP	SSH File Transfer Protocol, salauksella suojattu tekstitiedosto
SLA	Sevice Level Agreement, palvelulle tietyt vaatimustasot määrittävä sopimus asiakkaan ja palveluntarjoajan välillä
SNOW	ServiceNow, pilvipohjainen automatisointialusta – toimii myös tikettijärjestelmänä
SOA	Service Oriented Architecture, suunnittelutapa liiketoiminnan prosesseille
XML	Extensible markup language, merkintäkieli

Sisällys

Kuvat ja taulukot

Liitteet

1 Johdanto

Tämän opinnäytetyön tarkoituksena on luoda toimiva hoitomalli Yhtiö Y:n Frends-integraatioalustalle. Digitalisaatio kehittyy hurjalla vauhdilla, mikä luo paljon mahdollisuuksia, mutta asettaa samalla omanlaisensa haasteensa. Finanssialalla teknologian tulee olla korkealla tasolla, jotta palvelut saadaan jouheviksi, luotettaviksi ja nopeiksi. Nykypäivänä odotetaan nopeita ja helppokäyttöisiä sovelluksia ja jos niitä ei pystytä tuottamaan, niin asiakkaat saatetaan menettää kilpailijalle. Tietoturvan rooli on korostunut, GDPR-laki on tuonut omat haasteensa ja eikä sovi unohtaa saavutettavuutta. Ei riitä, että olemassa on hienoja järjestelmiä ja sovelluksia, vaan ne pitää saada toimimaan erillään sekä yhdessä. Järjestelmiä yhdistäessä esiin nousevat integraatiot, jotka Yhtiö Y:ssä toteutetaan Frends-integraatioalustan avulla.

Työssä tullaan kertomaan perusasiat integraatioista, jotta lukija ymmärtää, mistä käytännössä puhutaan. Teorian jälkeen tutustutaan tarkemmin Frendsiin, jonka jälkeen esitellään projektiryhmä lyhyesti. Käytännönosa tulee sisältämään nykytilan kartoituksen, jonka pohjalta nykytila analysoidaan. Tämän jälkeen tullaan hahmotelemaan tavoitetila. Mietitään myös, mitä vaatimuksia on tavoitetilan saavuttamiseksi ja viimeisenä kehitetään keinot, jolla tavoitetilaan päästään. Lopputuloksena tavoitellaan toimivaa kokonaisuutta, jossa Yhtiö Y ja toimittajat pystyvät hoitamaan tilanteen, joka esiintyy Frends-integraatioalustalla.

Opinnäytetyön tutkimuskysymykset ovat:

- Mikä on Frends-integraatioalustan hoitomallin nykytilanne?
- Mikä on tavoitetila, johon halutaan päästä Frends-integraatioalustan hoitomallin kanssa?
    - Mitä vaatimuksia Yhtiö Y:llä on toimittajalle, jotta saadaan toimiva hoitomalli?
- Millä toimenpiteillä päästään tavoitetilaan Frends-hoitomallin kanssa?

2 Integraatio

Integraatiolla tarkoitetaan kykyä yhdistää järjestelmät toimivaksi kokonaisuudeksi, jossa tieto kulkee näppärästi järjestelmästä toiseen (Toivanen, n.d.). Integraatioilla voidaan yhdistää järjestelmien lisäksi myös ohjelmistoja. Integraatiot ovat yleensä yksi tai kaksisuuntaisia. Yksisuuntaisessa integraatiossa tietoja vain lähetetään, kun taas kaksisuuntaisessa integraatiossa tietoja lähetetään sekä vastaanotetaan. Integraatioiden avulla tietoa liikutetaan automaattisesti taustalla, jolloin manuaalisen työn tarve poistuu (Tasanen, 2019).

Integraatioiden tarve syntyy, kun eri järjestelmien toiminnallisuudet eivät täsmää toisiinsa. Integraation suurimpia hyötyjä ovat kustannussäästöt ja toiminnan tehostaminen eli manuaaliseen työhön menevä aika pienentyy merkittävästi, jolloin työpanosta voidaan keskittää muuhun oleelliseen. Manuaalisen työn vähentyessä yleensä myös virheiden määrä pienenee, joka voidaan ajatella säästönä. Integraatioiden hyödyt eivät kosket pelkästään yhtiöiden IT-puolta, vaan se antaa koko liiketoiminnalle hyvinvointia (Haglund, 2018). Toiminnan tuottavuuden ja laadun kehittäminen ovat tärkeimmät syyt yhtiöille ottaa käyttöönsä järjestelmäintegraatiot. Järjestelmäintegraatiot tulevat välttämättömiksi tulevaisuudessa, yritysten sisäisessä sekä ulkoisessa viestinnässä (Lehtonen, 2018).

EAI (Enterprise Application Integration) on englaninkielinen nimitys integraatioille. Integraatioille on erilaisia malleja. Niitä voidaan tuottaa on-premises-järjestelmien välille, jolloin integraatio rakentuu lisenssienohjelmien välille tai yrityksille itselleen suoraan rakennetuista integraatioista. Pilvi-integraatiot taas ovat pilvessa aktiivisena olevien ohjelmistojen välisiä integraatioita. Hybridi-integraatio kulkee edellä mainnittujen välimaastossa eli on-premises-järjestelmää integroidaan pilvipohjaiseen sovellukseen (Flashnode, 2021).

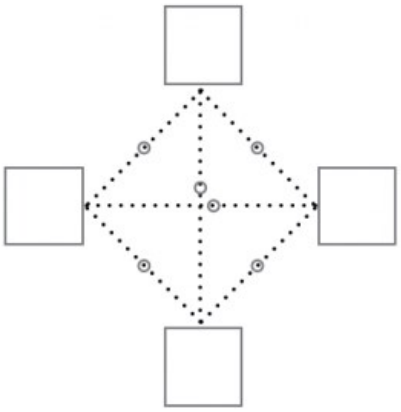
2.1 Integraatioiden toteutusmallit

Integraatioihin on monenlaisia malleja ja niiden toteutustapaa miettiessä tulee yrityksen miettiä liiketoiminnan tarpeita ja yrittää tukea niitä. Yrityksen tulee kartoittaa järjestelmien lukumäärä, jotka integroidaan sekä tulevaisuuden näkymä niiden varalle. Integraatioita toteuttaessa on hyvä miettiä, miten tiedonsiirrot saadaan toimimaan jouhevasti. Ainakin on suunniteltava, miten tietoja luovutetaan/vastaanotetaan ja missä muodossa edellä mainitut tehdään? Myös salaukset, säännöt ja muunnokset tulee pohtia ennen toteutusta. Tulevissa alaluvuissa esitellään integraatiomalleja (Flashnode, 2021).

2.1.1 Point-to-Point-integraatio

Point-to-Point on suora kahden järjestelmän välinen integraatio. Mallin ideana on tehdä kaikkien ohjelmistojen välille erilliset liitännät, kuten Kuva 1 osoittaa. Mallin vahvuutena on yksittäiset integraatioprojektit tai pienet kokonaisuudet, jolloin se on varsin edullinen vaihtoehto. Point-to-Point-malli asettaa seuraavia haasteita ylläpitäjille: huono skaalautuvuus ja uusien integraatioiden lisääminen/kehittäminen voi olla haastavaa sekä mutkikasta. (Flashnode, 2021).

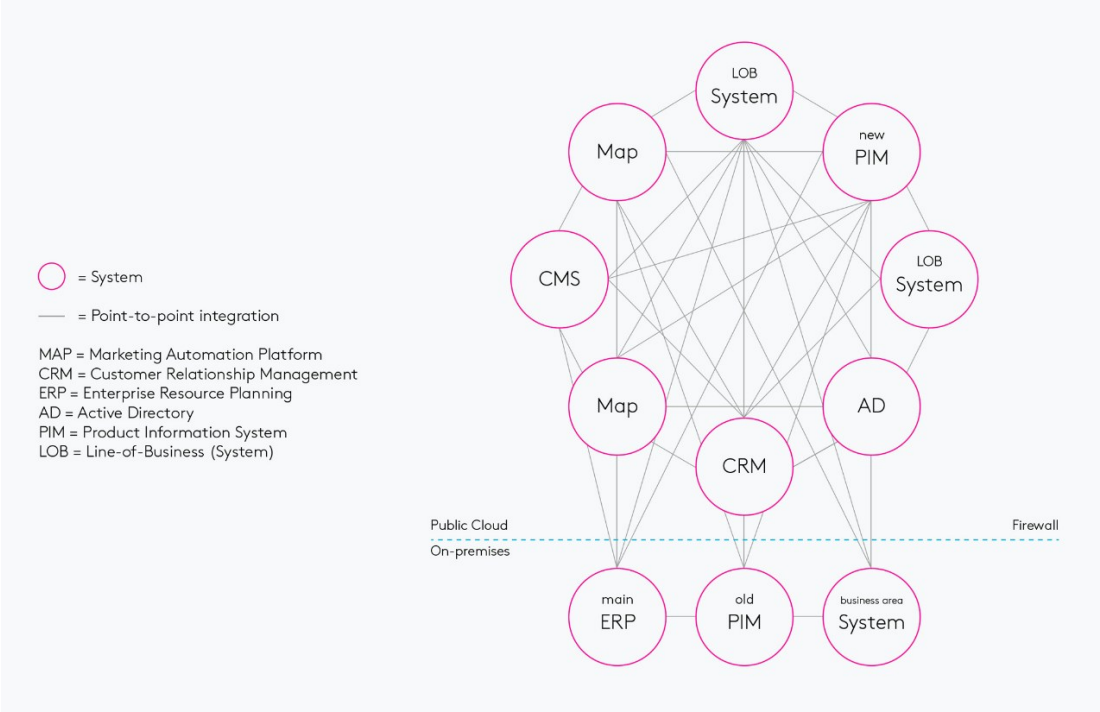
Kuva 1. Point-to-Point-integraatiomalli havainnollistettuna (Flashnode, 2021)



Toivasen mukaan malli sopii erityisesti, jos kyseessä on siis väliaikainen integraatio tai tietovirtojen liikuttelu saman toimittajan järjestelmissä, joissa käytettävät moduulit ovat erilaisia. Toimittaja vastaa tällöin kokonaisuudesta, muissa tapauksissa Point-to-Point ei ole toimiva kokonaisuus

yritykselle. Kuva 2 nähdään hyvin Point-to-Point-mallin heikkoudet, kun integroitavat järjestelmät lisääntyvät (Toivanen, n.d.).

Kuva 2. Laajempi Point-to-Point-malli (Toivanen, n.d.)



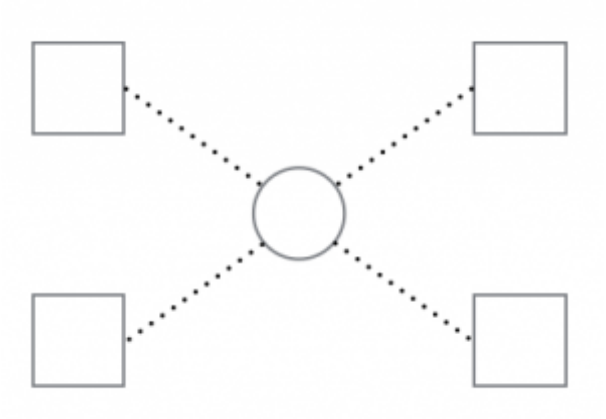
2.1.2 Hub and spoke -integraatio

Hub and spoke on malli, joka toimii hubin eli keskuksen ympärillä. Hub siis toimii keskuksena ja yhtä aikaa viestienvälittäjänä järjestelmille. Sen tarkoituksena on välittää viestit oikeaan paikkaan, kuten Kuva 3 nähdään. Hub vastaanottaa, kohdentaa, muuttaa ja reitittää viestejä. Se osaa myös määrittää oikein vastaanottavat järjestelmät sekä kohdentaa niihin tulevat viestit oikein. Hub and spoke-mallin hyviä puolia ovat tiedon hallinta sekä yksinkertainen toteutus laajentaessa integraatioita ja järjestelmiä.

Hub and spoke-mallissa heikkous kohdistuu juuri hubiin, koska sen toimimattomuus heijastuu suoraan muihin järjestelmiin. Esimerkiksi tiedonsiirrot voivat katketa, joka suuntaan. Toinen haaste liittyy dataan, erityisesti datanmuotoon, koska kaikkiin vastaanottaviin järjestelmiin tulee muotoilla samanlainen datamuoto (Flashnode, 2021). Haasteisiin voidaan lisätä myös, kun

integroitavia järjestelmiä kasvatetaan, jolloin hubin suorituskyky saattaa hidastua ja koko toiminta on alttiina vaaroille. (Abbasi, n.d.-b).

Kuva 3. Hub and spoke -integraatiomalli (Flashnode, 2021)

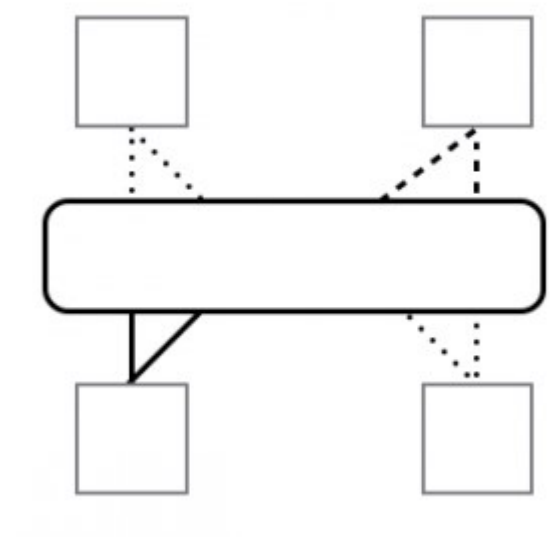


Hub and spoke-malli on tunnettu yleisimmin teollisuudesta. Päivittäistavarakaupat käyttävät tätä menetelmää jakeluun, jolloin yhdessä suuremmassa pisteessä lastataan ja pakataan tuotteet ja ne lähetetään eteenpäin yksittäisiin kauppoihin. Myös lentoyhtiöt käyttävät jakeluun Hub and spoke-mallia (Redwood Logistics, n.d.).

2.1.3 Enterprise service bus

ESB (Enterprise service bus) on kehittyneempi versio Hub and spoke-mallista. Malli toimii hajautusperiaatteella, missä ei ole yhtä pääkeskusta. Bus toimii ikään kuin älykkäänä laiturina, jonne voidaan tuoda ja sieltä voidaan viedä dataa. Olemassa olevat järjestelmät yhdistetään bus-väylään, kuten Kuva 4. Laituri on kykeneväinen muuttamaan datamuotoja järjestelmille oikeiksi, jolloin esim. Järjestelmä A voi lähettää tietoja datamuodossa x ja bus muuttaa datan toimivaksi Järjestelmälle B. Kaikki järjestelmät, jotka ovat integroituna, voivat tuoda ja hakea tietoja, jolloin laiturin rooli voidaan ilmaista myös väliohjelmistona (Abbasi, n.d.-a).

Kuva 4. ESB-malli, jossa kuvattuna useita järjestelmiä (Flashnode, 2021)



Kuva 5 avulla voidaan hyvin havainnollistaa ESB-mallin prosessin toimintaa. 1 kohta kuvaa järjestelmiä, joissa tiedonvaihto tapahtuu. On lähettävä ja vastaanottava järjestelmä. 2 kohta taas kuvaa kerrosta, jossa tapahtuu tiedonsiirrot. 3 vaihe on rajapinnoille. 4 vaihe prosessoi tietoja oikeaan muotoon. 5 vaihe havainnollistaa prosessihallintaa (Flashnode, 2021).

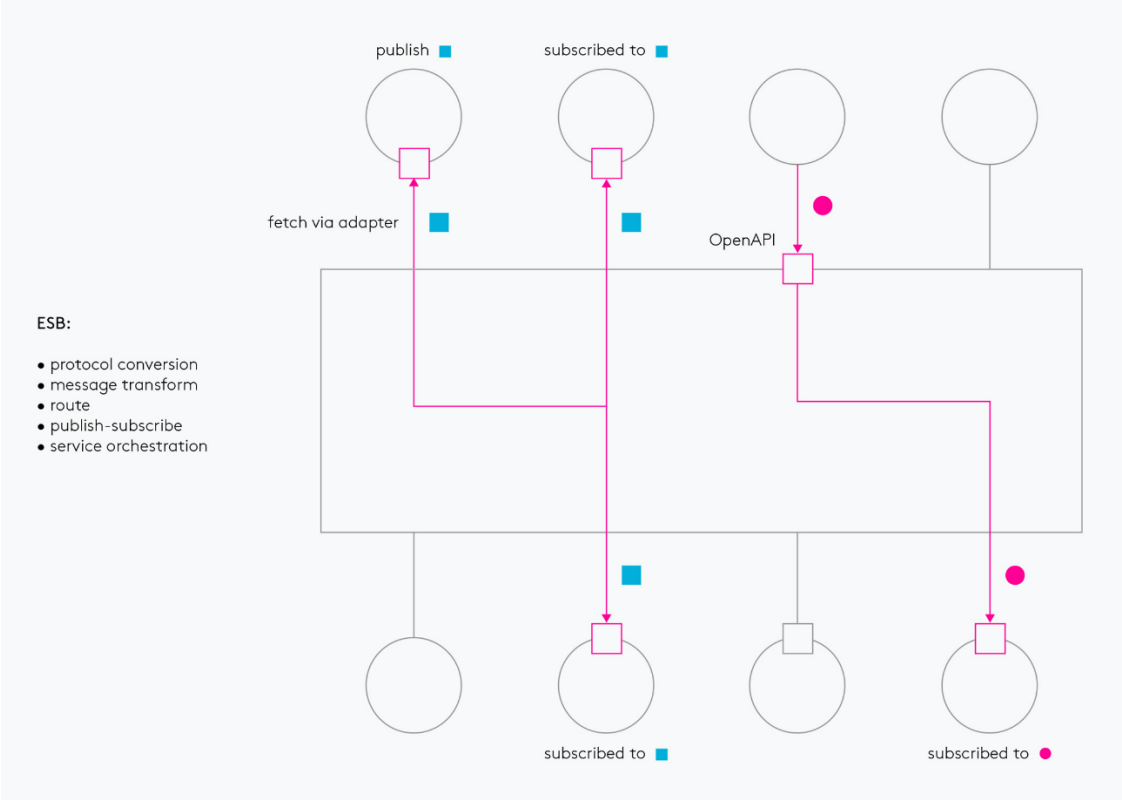
Kuva 5. ESB-mallin prosessi (Flashnode, 2021)

ESB-mallissa pystytään helposti valvomaan integraatioiden vaiheita, jolloin on helppo esim. paikantaa virheiden sijainti. Mallin muita hyviä puolia ovat mallin keveys, skaalautuvuus, uusien järjestelmien lisäys ja mukauttaminen yhtiön haluihin. Vaaraksi voi muodostua, kun järjestelmien

lisääminen on tehty niin helpoksi, jolloin kokonaisuudesta voi tulla raskas ylläpidettävä (Flashnode, 2021).

ESB on suomennettuna palveluväylä. Se on viestienvälittäjä, jolla on kolme yleistä tapaa välitykseen. Jonojärjestelmä, tapahtumapohjainen tiedonvälitys ja publish-subscribe-viestinvälitys. Jonojärjestelmässä data vastaanotetaan ja laitetaan jonoon, josta se lähtee eteenpäin, kun vastaanottava järjestelmä on valmis vastaanottamaan sen. Tapahtumapohjainen tiedonvälitys hyödyntää REST/JSON-rajapintoja, joiden avulla dataa ei laitetakkaan jonoon vaan tietoja välitetään suoraan toiseen järjestelmään. Publish-subscribe-viestivälitys perustuu reititykseen (Kuva 6), jossa Bus eli väylä osaa välittää halutun aiheen/sisällön, mitkä vastaanottaja on valinnut. ESB-malli on yleensä vain suurten yritysten käyttöön (Toivanen, n.d.).

Kuva 6. ESB, Publish-subscribe-viestinvälitysmalli (Toivanen, n.d.)



2.1.4 Service Oriented Architecture

SOA (Service Oriented Architecture) on organisaation arkkitehtuuriin oleva suunnittelu tyyli, jolla yksittäisiä liiketoiminnan prosesseja suunnitellaan toimimaan itsenäisinä palveluina. SOA:n idea on tehdä kerran liittymä järjestelmien välille sekä poistaa kolmansien järjestelmien rooli, jolloin palveluita voidaan käyttää avoimesti (Flashnode, 2021). SOA:n ongelmaksi koitui palvelun raskaus sekä sen avulla tapahtuva kehitys oli liian hidasta. SOA:lle tarvittiin vaihtoehtoinen tapa ja täten kehittyivät mikro- ja minipalvelut (Toivanen, n.d.).

2.2 API

Yksinkertaisimmillaan ohjelmointirajapinta eli Application Programming Interface (API) voidaan selittää toimintona, johon tarvitaan kaksi ohjelmistokomponenttia ja niiden yhdistämisen tekee API. Esimerkkinä, jos haluat tietää tulevan päivän säätiedot puhelimestasi, siihen tarvitaan sään toimittava ohjelmisto ja puhelin. API:en avulla sääohjelmisto tuo puhelimeen automaattisesti uusimmat tiedot tulevan päivän säästä ja päivittää sään tietoja jatkuvasti puhelimeesi (Amazon, n.d.-b).

API:lla siis tarkoitetaan ohjelmointirajapintaa, jonka avulla integraatiot toteutetaan. Käytännössä tämä tarkoittaa, että API:en tehtävä on lähettää ohjelmistolle käskyjä, missä ohjelmisto joko lähettää tai vastaanottaa käskyjä järjestelmien välillä (Tasanen, 2019). API:t ovat saneltuja sääntöjä, joiden avulla sovellukset ja tietokoneet keskustelevat. Ne muodostavat välikerroksen verkkopalvelimen ja sovelluksen välille, missä data voi liikkua. API:t mahdollistavat joustavampia yhteistyö mahdollisuuksia yritysten sisällä sekä yritysten välillä, perustuen API:eissa käytettäviin automaatioihin. Kommunikointi on tällöin helppoa ja vaivatonta (IBM Cloud Education, 2020).

2.2.1 API -rajapinnat

API-rajapintoja ovat OpenAPI, sisäiset API:t sekä ulkoiset API:t. Näiden rajapintojen avulla integraatioalustat voivat olla toteustusalustoja sekä suoritusmoottoreita. OpenAPI on standardi ja

pitää huolen, että API:t kuvataan ja dokumentoidaan samalla tavalla. Swagger on esimerkki OpenAPI:sta (Toivanen, n.d.). OpenAPI mahdollistaa helpomman sovelluskehityksen, kun yhdessä käyttöliittymässä on sisällytetty kaikki tiedot (protokollat, rajapinnat ja ympäristöt). OpenAPI tarjoaa vakiorajapinnan, jota REST on kaivannut API:lle, tällöin kolmasien osapuolienkin on helppo olla vuorovaikutuksessa API:en kanssa (RapidAPI, 2022b).

Sisäisellä API:lla (Internal API) tarkoitetaan yrityksen sisällä kehitettäviä API:ja. Näitä ei silloin ulkopuoliset pääse kehittämään, jolloin ei välttämättä saada niin kehittyneitä ratkaisuja, mutta tämä tapa koetaan varsin tietoturvalliseksi. Sisäisiä API:ja käytetään liiketoiminnan tehostamiseen (Bailey, 2018). Sisäinen API antaa siis kehittäjälle pääsyn pelkästään oman yrityksen tietoihin ja sovelluksiin (RapidAPI, 2022a).

Ulkoisella API:lla (External API) tarkoitetaan API:a, joka on avoinsovellusliittymä. Tällöin kehittäjän ei tarvitse olla yrityksen sisällä töissä, jotta voi kehittää sovellusliittymiä. Tämä mahdollistaa uudet ideat, jolloin sovellus voi kehittyä hurjalla tahdilla. Usein ongelmana voi olla, ettei sovellusliittymää saada tarpeeksi kiinnostavaksi, jolloin kehittäjät eivät jaksa tehdä kehitystyötä (Bailey, 2018). Ideana usein on saada lisäarvoa yritykselle, käyttämättä rahaa tällaiseen kehitystyöhön. Yritysten on usein aloitettava sisäisestä API:sta, sillä ilman suurta ymmärrystä ulkoiset API:t voivat muodostaa tietoturvariskin (RapidAPI, 2022a).

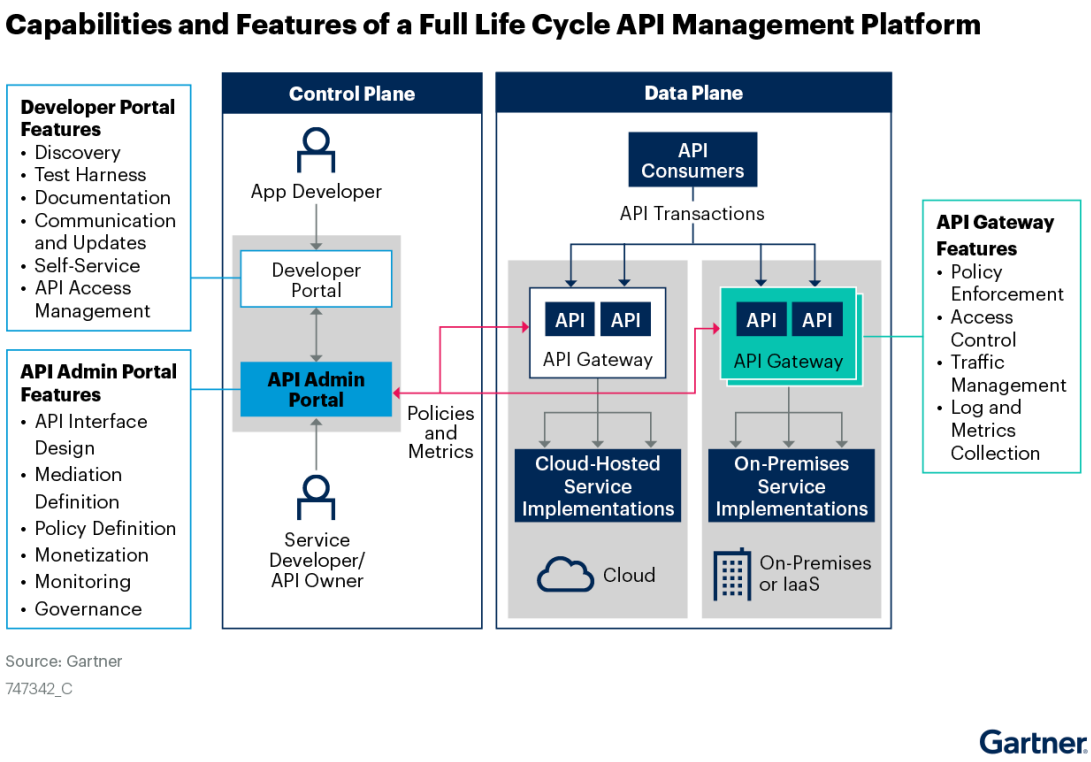
2.2.2 API:n hallinta

APIM (API Management) on API:en hallintaa, joka on tarkoitettu ylläpitäjälle tai kehittäjälle. Sen avulla voi luoda, hallita, analysoida ja käyttää API: a. API:en hallinta ominaisuus on tärkeää, jotta saadaan paras käyttökokemus käyttäjälle, turvataan sovellus, seurataan suorituskykyä ja hallitaan käyttöoikeuksia (Amazon, n.d.-a). API:en hallinnan on siis tarkoitus suojella API:n elämänkaarta ja suojella etteivät ne päädy vääriin käsiin (Ismail, 2021).

API:en hallintaa työkaluun kuuluu yleensä kehittäjäportaali (Developer Portal), hallintaportaali (Admin portal) ja API-käytävä (API Gateway). Nämä ovat komponentteja ja ne tulee integroida yhteen toimivaksi toimintaympäristön kanssa, jossain alustoissa nämä ovat kuitenkin jo valmiina.

Kehittäjäportaali on tarkoitettu API:en kehittäjälle, jossa voidaan testata, dokumentoida, kommunikoida ja kehittää API-toimintoja. Hallinnassa asetetaan suojaukset ja vaatimukset. Yhdyskäytävässä tarkistetaan, voiko API toteuttaa pyyntöä suorituksen aikana. Se siis tarkistaa määritykset tai vaatimukset ja joko päästää toiminnon läpi tai hylkää sen. API-hallinnan käyttö ei ole pakollista, mutta integraatioiden lisääntyessä, se tulee välttämättömäksi. API:en hallinta on esitetty myös Kuva 7. Parhaiten API-hallinnasta saa irti, kun tuntee oman yhtiön API-käyttötapaukset, jolloin osaa hallita kokonaisuutta (Steve et al., 2021). API-hallinnassa on usein myös mukana raportointi sekä analytiikkaominaisuus (IBM Cloud Education, n.d.).

Kuva 7. API:en hallinta kuvattuna (Steve et al., 2021)



2.2.3 API-suojaus

API-turvallisuus ja suojaus on välttämätöntä API:a käytettäessä. Parhaita suojausmuotoja ovat käyttäjätunnukset palveluun, jolloin saadaan identiteetti määritettyä. Apukeinoina tulisi myös käyttää salausta ja allekirjoitusta (TLS). API-Gateway on tärkein suojauskeino API:lle, sillä se toimii valvontapisteenä ja sen avulla hallitaan, miten yhtiön API-liittymiä hyödynnetään. API-hallinta on

myös tärkeässä roolissa, kun suojellaan API:ja. Useat ympäristöt tukevat kolmea suojausjärjestelmää. Ne ovat API-avain, todentaminen ja OpenID Connect. API-avain on merkkijono, perustodennus viittaa käyttäjänimeen ja salasanaan. OpenID Connect taas on yksinkertainen identiteettikerros OAuth-kehyksen yllä (Red Hat, 2019).

2.2.4 REST

REST (representational state transfer) tunnetaan paremmin RESTful API:na, joka on sovellusohjelmointirajapinta. Se noudattaa REST-arkkitehtuuri tyyliä ja sallii vuorovaikutuksen muiden RESTful verkkopalveluiden kanssa. Tällä tarkoitetaan, että tiedot, joita lähetetään REST:n kautta tulee välittää HTTP:n avulla, käyttäen JSON, HTML, XLT, Python tai PHP-muotoa. Myös pelkkä tekstimuoto käy välitykseen. REST-rajapinta vaatii myös otsikon ja parametrin toimiakseen oikein. Niiden avulla saadaan tärkeitä tietoja esim. välimuistista, valtuuksista ja evästeistä. Vaikka REST vaatii monia kriteereitä, pidetään sitä silti helppokäyttöisempänä kuin SOA. REST-vahvuuksia ovat sen keveys ja nopeus sekä sen skaalautuvuus (RedHat, 2020).

REST hyödyntää HTTP-pyyntöjen tietoja. Tietoja hyödynnetään GET, PUT, POST tai DELETE komennoilla, joita API:t käyttävät ohjelmistojen väliseen kommunikoimiseen. RESTful API toimii hajotus periaatteen muodossa, missä isompi kokonaisuus jaetaan pienempiin moduuleihin, jotka luovat omat prosessinsa ja jokainen moduuli käsittelee oman osansa, luoden yhtenäisen kokonaisuuden (Gillis, 2020).

2.3 iPaaS

Integration platform as a Service (iPaaS) eli pilvipohjainen integraatioalusta palveluna. Se mahdollistaa yrityksille huipputehoisia ratkaisuja, jolloin järjestelmiä voidaan integroida eli yhdistää toisiinsa, joka tarjoaa tehokkaampaa liiketoimintaa. iPaaS:n avulla voidaan tehdä API-hallintaa, eri API:n sulauttamista yhteen, automatisaatiota sekä low-code integraatioita. Sen on huomattu myös parantavan yhtiöiden välistä sekä sisäistä kommunikaatiota (Frends, n.d.-e).

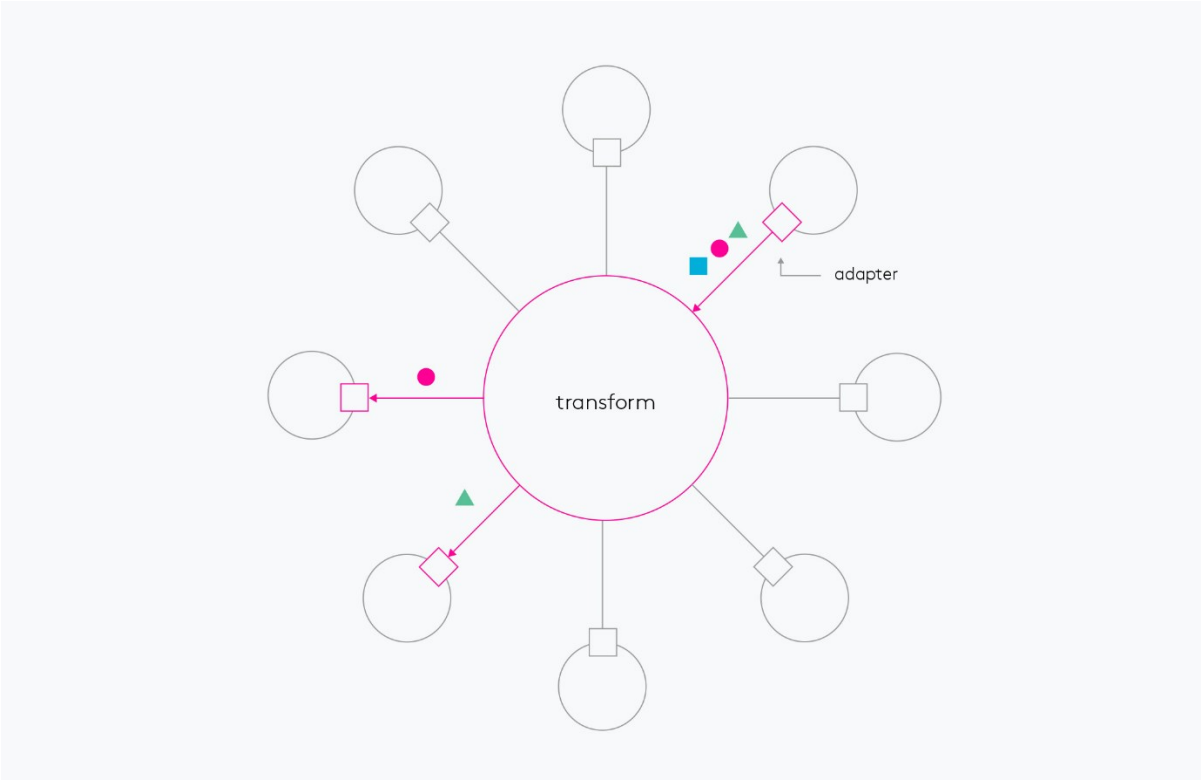
iPaaS-idea on siis tarjota alusta palveluna, jota palveluntarjoaja ylläpitää. Ylläpitäjän vastuulla on turvallisuuden hoito, tietojenhallinta, päivitykset ja ominaisuuksien kehittyessä lisäpalvelun mahdollistaminen. Palvelun hinta muodostetaan joko käytön mukaan tai kuukausihinnoittelulla. iPaaS:n suurimpia hyötyjä ovat integraatio kustannusten pieneneminen, yrityksen sisällä tiedon saaminen reaaliajassa helpottuu, tehokkuus paranee ja integraatioiden toteuttaminen on helpompaa, kun yritys voi itse tarttua toimeen alustan avulla. Myös API:en hallinta kuuluu yleensä iPaaS-alustaan, jolloin API-hallinta on helpompaa ja turvallisempaa (IBM Cloud Education, 2021).

Joskus iPaaS voi muuttua EiPaaS-malliin, koska EiPaaS tarjoaa paremman hoitomallin yritystason projekteihin. Näistä esimerkkejä on korkeampi saatavuus, turvallisuus ja tietosuoja parempaa, tekninen tuki palvelun tarjoajalta, SLA-sopimukset ja jotkut alustat mahdollistavat DIH (Digital Integration Hub) käytön, joka on tarjolla vain EiPaaS-alustaan (Frends, n.d.-e).

3 Frends

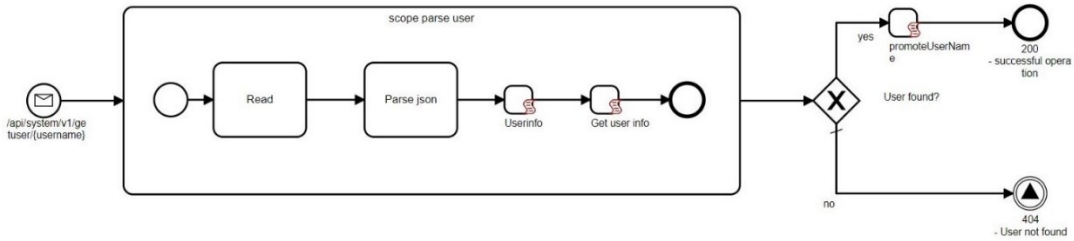
Frends on low-codea hyödyntävä integraatioalusta, jossa integraatiot luodaan BPM-notaatioilla. Frends tasks eli Frends-tehtävien avulla toteutetaan prosessit. Tehtävistä siis syntyy prosesseja (Kuva 9), joita kutsutaan vuokaavioiksi (Kuva 10). Niissä hyödynnetään valmiita komponentteja. Komponentteina toimivat agentit, jotka keskustelevat käyttöliittymien kanssa ja suorittavat toteutettuja integraatioita. Agentteja hallitaan keskuksesta (Control Panel), missä tehdään kehitys, monitorointi ja muu hallinta. Kehitystyö tapahtuu low-coden avulla (HiQ Finland, 2022b). Frendsin avulla voidaan toteuttaa integraatioita tiedonsiirrosta aina hyperautomaatioihin. Frends mahdollistaa API:en toteuttamista sekä niiden hallintaa. Frendsin integraatioalusta toimii keskitetyllä integraatiomallilla, minkä ansiosta alustalla on kyky ajastaa tiedonsiirtoja tai se pystyy tietokanta muutoksiin. Kuva 8 pyrkii havainnollistamaan keskitettyä integraatiota (HiQ Finland, 2022a). Frends myös edustaa EiPaaS-mallia, joka on siis suunnattu yritystason projekteihin ja toteutuksiin. Se myös hyödyntää OpenAPI-rajapintaa omassa API-hallinnassa (Frends, n.d.-e).

Kuva 8. Esimerkki keskitetystä integraatiosta (HiQ, n.d.)



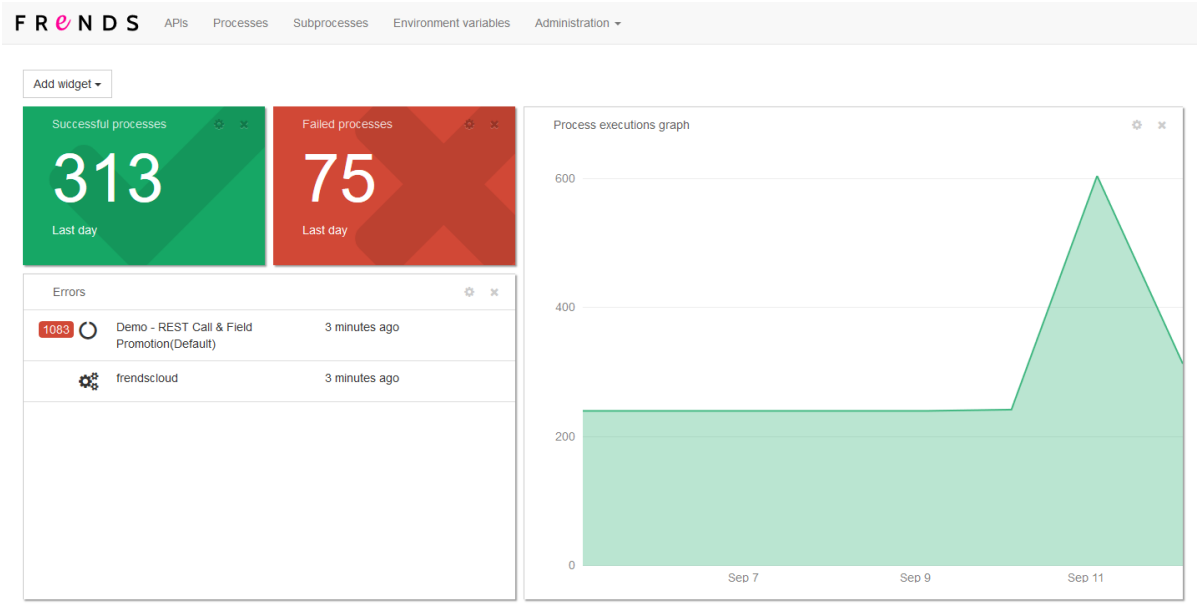
Kuva 9. Yksinkertainen prosessi. Prosessi käynnistyy, missä tapahtuu kolme tapahtumaa, jonka jälkeen se loppuu(Frends, n.d.-b)

Kuva 10. Esimerkki Frends-prosessin vuokaaviosta (HiQ Finland, 2022b)



Frends:ssä on helppo luoda dashboardeja, jossa käyttäjä asettaa tauluja (widgettejä) halutulla tavalla. Taulujen avulla seurataan haluttuja ominaisuuksia tai lukuja. Dashboardia pystyy itse muokkaamaan ja tällä hetkellä tarjolla on neljää erilaista widgettiä, joista tauluja voi luoda. Ne ovat onnistuneet prosessit, epäonnistuneet prosessit, virhetaulu ja viimeisenä on kaavio, jonka avulla näkee onnistuneiden ja epäonnistuneiden prosessien käyrät. Kuva 11 näyttää näistä esimerkin. Widgettien aikarajaa on mahdollista vaihdella, eli niitä voidaan tarkkailla päivä tai kuukausitasolla (Galkin, n.d.).

Kuva 11. Frends-dashboard (Galkin, n.d.)

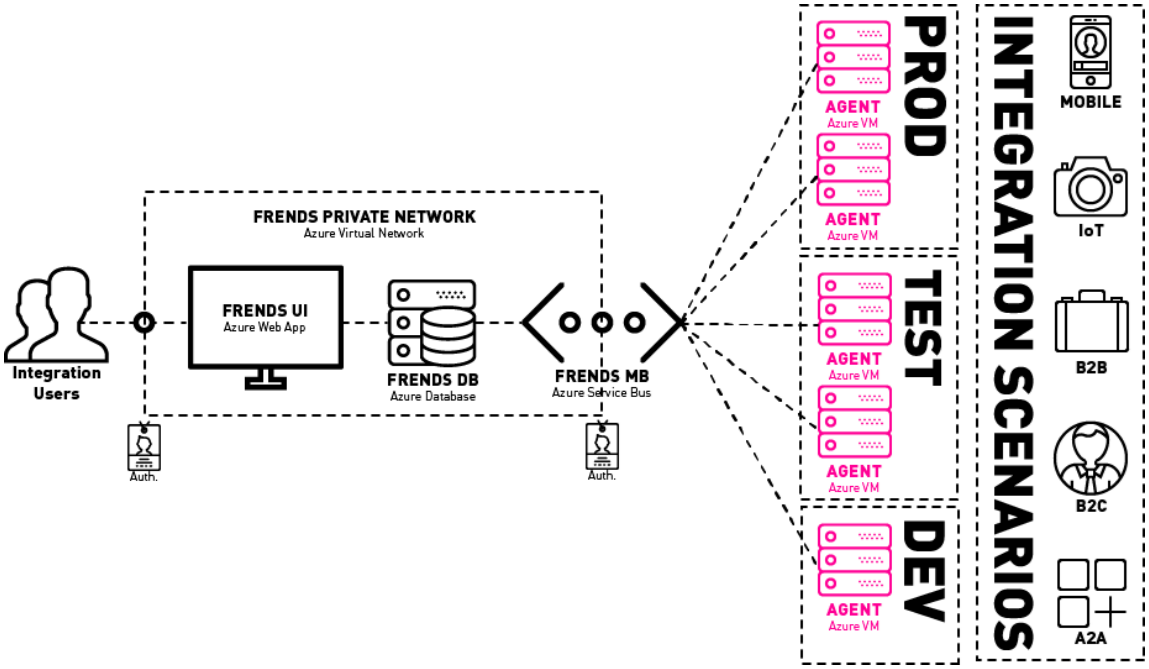


3.1 Frends-agentit

Frends-agenteilla tarkoitetaan Frendsissä toimivia komponentteja. Nämä agentit suorittavat prosesseja ja yksittäisessä prosessissa voidaan hyödyntää useampaa agenttia. Agenttien avulla toteutetaan kehitettyjä integrointikoodeja. Agenteilla on käytössä kolme tasoa: ohjauskerros, liipaisukerros ja toteutuskerros. Nämä kaikki toimivat yhdessä ja muodostavat toimivan kokonaisuuden eli agentin. Ohjauskerroksen (Control Layer) tehtävänä on kommunikointi. Se tapahtuu keskushallintaportaalissa, Azuren Service Bus -jonojen voimin. Jonoja hyödynnetään esim. manuaalisesti käynnistettäviin integraatiovirtoihin tai agentin kokoonpanojen päivittämiseen. Ohjauskerros myös palauttaa valvontatietoja keskusportaaliin. Liipaisukerroksen (Trigger Layer) tehtävänä on laukaista integraatiovirrat automaattisesti toimintaan. Yleensä se laukaisee integraation päätepisteen (API End point) tai ajastetun tapahtuman toimintaan. Toteutuskerroksen (Execution Layer) tehtävänä on suorittaa integraatiovuokoodin prosessi eli se suorittaa asetetut tehtävät koodi muodossa. Toteutuskerros aktivoituu, kun liipaisukerros on käynnistynyt. Toteutuskerros hakee ohjauskerroksesta seurantatietoja integraatiovuokoodin prosessin aikana. Agenttien toiminnasta kertoo Kuva 12 (Frends, n.d.-a).

Agenteilla on myös vaatimuksia, jotta ne voidaan ottaa käyttöön. Frends-agentti vaatii toimivan internet yhteyden, mutta vain lähtevässä muodossa, koska yhteys muodostetaan pelkästään keskusohjauspaneelin ja Azure Service Bus jonoihin. Agentti tarvitsee joko Linux/Windows palvelimen ja Docker/Kubernetes kontin (Frends, n.d.-a). Kaikki Frends agentit ovat itsenäisiä komponentteja eli ne eivät ole riippuvaisia, muiden toiminnasta (Galkin, 2022).

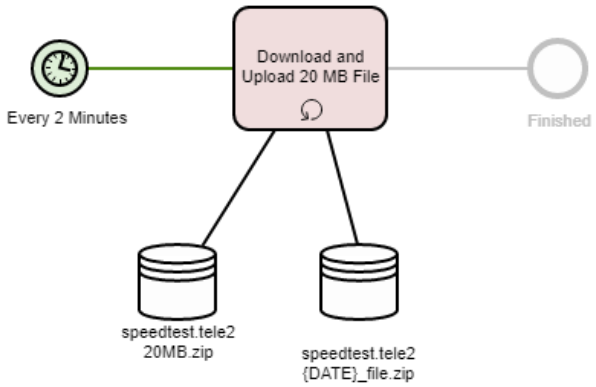
Kuva 12. Agenttien toimintaa (Galkin, 2022)



3.2 Frends-tiedostonsiirrot

Frendsissä tiedostonsiirrot tehdään hallitusti ja tyyliä kutsutaan Cobalt transferiksi. Kuva 13 havainnollistaa hallitun tiedostonsiirron. Tuettuja tiedostonsiirto protokollia ovat FTP, FTPS, SFTP, Windows tai Linux pohjaiset tiedostojärjestelmät. Frendsin lokitietoihin jää myös merkintä, kun tiedostonsiirtoja on tapahtunut.

Kuva 13. Frends-tiedostonsiirto (*Frends IPaaS | Integration Scenarios - Managed File Transfers*, n.d.)



Frendsin vahvuuksiin kuuluu sen kyky käsitellä tietoja lähes missä muodossa vain. Näitä muotoja ovat esim. JSON, CSV, XML, EDIFACT tai mukautettu tekstimuoto. Cobalt transfer tarjoaa myös seuraavia ominaisuuksia: automaattiset varmuuskopioinnit tiedostoille, tiedostojen helppo nimen vaihtaminen ja helppo tiedostonsiirtoprosessi (Frends, n.d.-e).

3.3 Frends-tietoturva

Frends hyödyntää Azure Security Centeriä sekä tukee GDPR (General Data Protection Regulation) lakia, jolloin alusta pystyy tarjoamaan kaikki lakisääteiset velvollisuudet. Laki asettaa velvollisuuden, jossa täytyy olla kaksi toimijaa, tiedonhallitsija sekä tiedon käsittelijä. Hallitsija on alustan käyttäjä, kun tiedonkäsittelijänä toimii Frends. Azure Security Center on kykenevä havaitsemaan haavoja ja hyökkäyksiä 36-tunnin sisällä niiden alkamisajasta. Frends mahdollistaa myös tietojen seuraamisen laajasti, edellyttäen tiedon kulkeneen Frends alustan kautta. Eli jos ei käytetä Frends-alustaa kaikkeen tiedonsiirtoon, niin silloin sieltä ei löydetäkään kaikkea (Frends, n.d.-c).

Frendsillä on tietoturva myös API:en suojaamisen, joissa on käytössä kahta eri mallia. Ensimmäinen on OpenID ja OAuth 2.0 avulla toimiva identiteettihallintajärjestelmä. Tämä menetelmä suojaa parhaiten sovellusliittymiä, kun loppukäyttäjille tai ryhmille rajataan pääsy tiettyihin API:ihin. Toinen menetelmä on käyttää API-avaimia. Menetelmä sopii parhaiten

sovelluksiin, kun loppukäyttäjän ei tarvitse tunnistautua sinne. Säännöt ohjaavat, ketkä saavat käyttää API-avaimia ja avaimen myötä on pääsy itse API:iin. Frends:ssä on myös API-hallinta, jonka avulla voidaan suojata järjestelmää ja alustaa (Frends, n.d.-d).

4 Hoitomalliprojekti

Tämän kehitysprojektin tarkoituksena on luoda toimiva hoitomalli Yhtiö Y:n Frends-integraatioalustalle. Siinä siirretään vastuu toimittajalle, mikäli alustaan kohdistuu muutoksia, ongelmia, häiriöitä tai päivityksiä. Kehitysprojekti toteutetaan yhdessä Yhtiö Y:n (Toimeksiantaja) sekä palvelun toimittajan kanssa. Tarkoituksena on tarkastella olemassa olevaa alustaa sekä siihen liittyvää palvelua ja mietitään, tarvitseeko nykyistä sopimusta päivittää.

4.1 Toimeksiantaja

Toimeksiantajana toimii Yhtiö Y, joka on pitkänlinjan finanssialan toimija. Yhtiön tarkoituksena on luoda taloudellista hyvinvointia asiakkailleen. Yhtiön on pysyttävä teknologian kehityksessä mukana ja se haluaakin tarjota asiakkailleen toimivia, luotettavia, laadukkaita ja tietoturvallisia palveluita, jotka edustavat Yhtiön Y:n hyviä tapoja.

4.2 Toimittaja

Yhtiö Y:n integraatioalustan toimittajana toimii Yhtiö X, joka on pohjoismainen Frends- toimittaja. Yhtiö X on kansainvälinen konsulttiyritys, joka tarjoaa erityisesti ohjelmistoalalle palveluita.

4.3 Projektiryhmä

Varsinaista projektiryhmää ei ole nimetty Yhtiö Y:ssä tähän kehitysprojektiin. Hoitomallin kehittämiseen osallistuu Yhtiö Y:ltä 4 henkilöä ja toimittajalta 2 henkilöä. Kehitysprojektin ajankohta on 12/2021–3/2022.

5 Hoitomalli

Tässä luvussa mennään käytännön osaan, jossa selvitetään nykytila Yhtiö Y:n integraatioalustasta. Hoitomalli käsitteenä avataan ensiksi, jonka jälkeen siirrytään nykytilaan. Nykytilan kartoituksen jälkeen analysoidaan Yhtiö Y integraatioalustan tilanne, nykytila myös raportoidaan toimittajalle. Tämän jälkeen tehdään päätöksiä, tarvitseeko jotakin integraation palvelua päivittää hoitomallin suhteen. Lopuksi esitetään vaatimukset toimittajalle, jotka halutaan uudelle hoitomallille ja tehdään jatkosuunnitelma, miten tavoitteeseen päästään eli valmiiseen hoitomalliin ja mitä jätetään tulevaisuuden kehitykseen.

Hoitomallilla tarkoitetaan tässä yhteydessä sitä, kun Yhtiö Y:n Frends-integraatioalustaan kohdistuu häiriöitä, muutoksia, ongelmia tai päivityksiä, niin on olemassa selkeät toimintaohjeet, jonka mukaan toimitaan. Hoitomalliin kuuluu myös yhteystietojen määrittäminen eli keneen ollaan yhteydessä, jos häiriöitä, muutoksia, ongelmia tai päivityksiä esiintyy. Hoitomalliin tulee määrittää myös SLA-ajat sekä palvelunomistajat. Hoitomallissa Yhtiö Y tekee ohjeet, jotta toimittajan ja Yhtiö Y:n Service Deskit, osaavat toimia oikein eri tilanteissa ja tietävät minne ottaa yhteyttä tilanteen tullen.

5.1 Nykytila

Nykytila osiossa selvitetään, että montako integraatiota Yhtiö Y:llä on tällä hetkellä ja kuinka niitä valvotaan ja ylläpidetään. Nykytilan kartoittaa opinnäytetyöntekijä. Nykytila on kartoitettu Yhtiö Y:n Frends-integraatioalustalta eli manuaalisesti etsitty ja laskettu integraatioiden määrät monitoreilta/API-välilehdeltä. API:en määrä etsittiin, jotta on tiedetty esittää toimittajalle oikeat luvut ja laajuus integraatioiden osalta. Taulukko 1 on kirjattu, minkälaisia API:ja Yhtiö Y:llä on ja montako integraatiota kyseisellä API:lla on.

Yhtiö Y:llä on tällä hetkellä 81 erilaista integraatiota tuotannossa toiminnassa. Tapahtumia oli 30 päivän mittausajankohtana (15.12.2021 – 15.1.2022) aikana, onnistuneita API-kutsuja on noin 39 400 904 ja API-kutsu virheitä noin 213 657. Eli yhteensä API-kutsuja on viimeiseen 30 päivään noin

39 600 000. Arkisin kutsuja on noin 1,5–2 miljoonaa päivässä, viikonloppuisin noin 0,7–1 miljoonaa päivässä. Normaalisti API-virheitä ilmenee 6500–8000 päivässä.

Taulukko 1, API:en määrä

Numero	API	Luku kertoo, montako integraatiota API:lla on
1	Appointments v1	11
2	Cards Life Cycle Management v1	8
3	Cards Management v1	8
4	Chat v1	8
5	Configuration Management Database v1	3
6	Foreign Exchange Rates v1	2
7	Healthcheck v1	1
8	IT Service Management v1	4
9	Leads v1	6
10	Mobile app API root	1
11	Payment Servi ces - Consent Management v1	10
12	Payment Services - Account Information Services v1	3
13	ServiceNow v1	5
14	Settings v1	11
Yhteensä		81

Nykytilasta voidaan päätellä, että integraatioiden määrä on jo laajaa Yhtiö Y:llä, jolloin nykyinen hoitomalli ei ole riittävä, koska se on perustunut pelkästään Yhtiö Y:n huolenpitoon häiriöiden osalta. Hoitomallia tulee siis päivittää turvallisempaan muotoon. Tällä tarkoitetaan, että on ehdottomasti siirrettävä Frends-alustan hoitovastuu toimittajalle.

5.2 Hoitomallin vaatimukset ja tavoitetila

Nykytila on selvitetty ja sen perusteella voidaan todeta, että nykyinen hoitomalli ei ole riittävällä tasolla. Tämän perusteella tehtiin hahmotelmaa tavoitetilasta. Tavoite on siirtää vastuu toimittajalle alustan seuraamisesta. Toimittajan vastuulla on päivystää Frends-integraatioalustaa häiriöiden varalta, jolloin häiriöiden huomaaminen ei ole pelkästään Yhtiö Y:n varassa. Hoitomallin

tavoitetilaan kuuluu myös ongelmien, muutoksien ja päivityksien raportointi. Näillä toimenpiteillä halutaan luoda turvallisempaa jatkoa Frends-alustalle.

Yhtiö Y haluaa siis siirtää päivystysvastuuta toimittajalle Frends-integraatioalustan suhteen. Yhtiö Y:llä on myös muita vaatimuksia, joita halutaan hoitomalliin mukaan ja mistä pitää sopia toimittajan kanssa. Osa vaatimuksista tulee suoraan ITIListä. Vaatimuksia ovat palveluaika, SLA, agenttien valvonta, häiriönhallinta, ongelmienhallinta, muutoshallinta ja kommunikointi.

5.2.1 Palveluaika ja SLA

Kaikille API:lle on määritelty oma häiriön päivystysaika, joka on 8–16/5 eli arkipäivisin häiriöitä päivystetään toimittajan päässä kello 08.00–16.00. Jos häiriöitä ilmenee, on toimittajan kahdessa tunnissa reagoitava siihen.

5.2.2 Agenttivalvonta

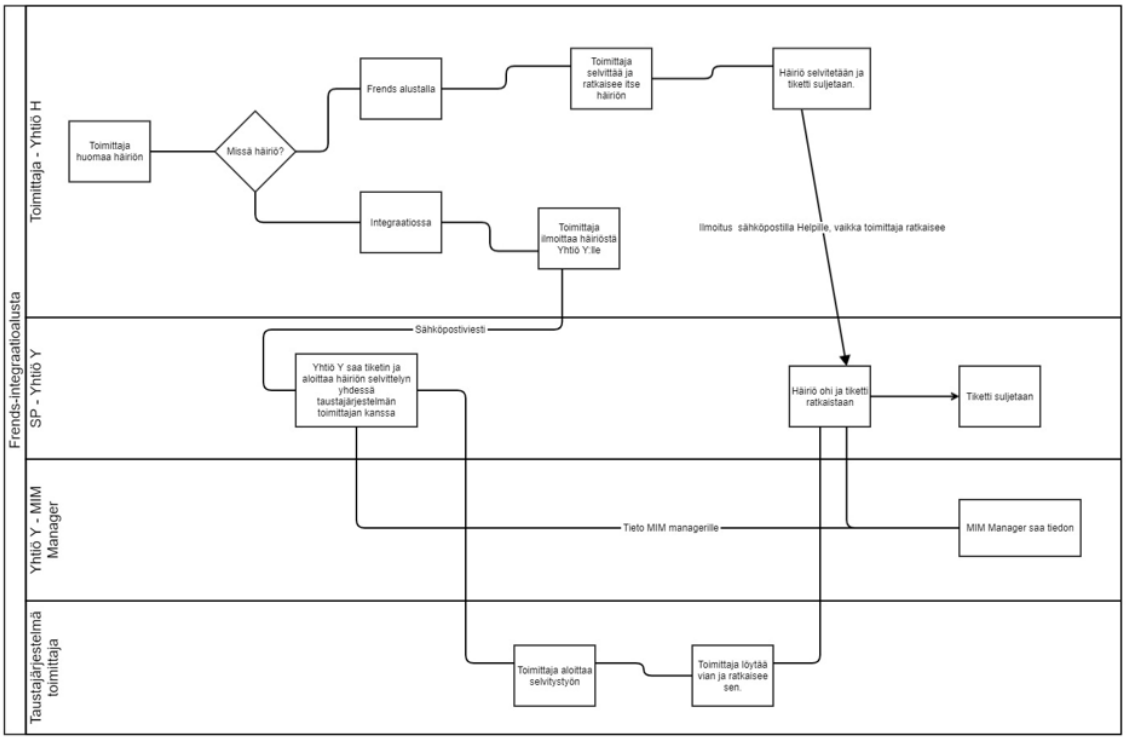
Toimittajalla vastuu valvoa, että Yhtiö Y:n käytössä olevan Frends-integraatioalustan agentit toimivat integraatiorajapinnassa. Mikäli agenttien toiminnassa esiintyy toimimattomuutta, niin toimittajan vastuulla on korjata agentteihin liittyvät ongelmat.

5.2.3 Häiriönhallinta

Häiriönhallinnan osalta on sovittu että, kun toimittaja huomaa häiriön Frends-alustalta, sen tulee aloittaa selvitystyö. Mikäli häiriö koskee Frends-alustaa, tulee toimittajan se itse ratkaista. Jos taas häiriö on integraatioissa niin, siirtyy selvitystyö Yhtiö Y:n Service Deskille. Tällöin tulee toimittajan Service Desk:n laittaa sähköpostiviesti Yhtiö Y:n Service Deskiin. Sähköpostista siis muodostuu tiketti toisen toimijan tikettijärjestelmään. Sähköpostissa tulee olla integraatio nimettynä, jossa häiriö sekä mahdollisimman tarkka selostus, mihin häiriö kohdistuu ja mikä virhekoodi siellä esiintyy. Yhtiö Y:n Service Desk jatkaa tästä häiriön selvittelyä toisen toimittajan kanssa. Yhtiö Y toimittaa toimittajalle toimintaohjeet. Ohjeisiin on myös listattu, keneltä Yhtiö Y:ssä voi tiedustella kyseisen ongelman kohdalla lisää. Yhtiö Y:n häiriöhallinta vastaavalle on myös tultava tieto, mikäli

häiriöitä ilmenee. Yhtiö Y:n Service Deskin vastuulla on tiedottaa häiriönhallinta vastaavaa häiriöstä. Kuva 14 on häiriöhallinnan malliprosessi.

Kuva 14. Häiriönhallinnan prosessikuvaus



5.2.4 Ongelmanhallinta

Ongelmanhallinnalla tarkoitetaan tässä yhteydessä, jos alustassa huomataan jatkuvia ongelmia niin niistä sovitaan palaveri, jossa ongelmiin yritetään löytää pikaista korjausta. Ongelmien havaitseminen voi tapahtua Yhtiö Y:n/Toimittajan toimesta. Yhtiö Y kirjaa ServiceNow järjestelmään ongelmatiketin tästä. Yhtiö Y:n muutosvastaava hoitaa myös ongelmatilanteiden koordinoinnin.

5.2.5 Muutoshallinta

Muutoshallinnalla tarkoitetaan järjestelmässä tapahtuvia muutoksia eli toisin sanoen päivityksiä järjestelmissä. Tämä tulee hoitaa koordinoidusti, jotta vältytään epätietoisuudelta. Muutokset

hoitaa pääosin Yhtiö Y:n muutosvastaava. Jokaisesta muutoksesta tehdään tiketti, Yhtiö Y:n ServiceNow:hun. Yhtiö Y:llä on käytössä myös muutoskalenteri, johon nämä muutokset tuodaan tikettien avulla. Kehitysprojekti on sopinut, kun Yhtiö Y tekee muutoksia Frends:llä, niin siitä tulee ilmoittaa toimittajalle ja käydään palaverin muodossa läpi, mitä on muutettu. Kehitysryhmä myös infoaa Yhtiö Y:n muutosvastaavaa muutoksesta. Muutosvastaava tekee tiketin yleisten ohjeiden mukaisesti.

5.2.6 Kommunikointi

Kommunikaatio tapahtuu yleisesti sähköpostin välityksellä, kiireellisissä asioissa voidaan sopia palavereita esim. Teamsiin. Kommunikaatio häiriötilanteissa aloitetaan sähköpostien välityksellä. Tulevaisuuden tavoitteena on rakentaa integraatio tikettijärjestelmien välille. Kommunikaatiota halutaan parantaa nykyisestä ja on päätetty, että toimittajan Service Deskille sekä kahdelle kehittäjälle annetaan pääsy Yhtiö Y:n Confluenceen, josta näkee operointiohjeita liittyen API:hin ja niissä esiintyviin virheisiin. Epäselvissä tilanteissa Service Desk:t voivat kysyä neuvoa palvelunomistajilta. Palvelunomistajille nimetään myös varahenkilöt, jos palvelunomistaja on sairaana tai lomalla.

Yhtiö Y järjestää myös perehdytyshetken toimittajan Service Deskille, jossa opastetaan, mitä monitoreilta tulee seurata ja kuinka niihin reagoidaan. Toimittajan Service Desk ja kehittäjät saavat oikeudet Yhtiö Y:n Frends-tuotantoympäristöön, missä monitorien seuraamista voidaan tehdä.

5.3 Jatkosuunnitelma

Tavoitetila on saatu määriteltyä onnistuneesti toimittajan kanssa. Tulevaisuudessa tavoitteena on entistä sujuvampi yhteistyö toimittajan kanssa. Tulevaisuuden suunnitelmiin kuuluu integraatio tikettijärjestelmien välille, jolloin pystytään entistä parempaan kommunikaatioon. Integraation avulla pystytään myös raportoimaan tarkemmin, kuinka paljon tikettejä liikkuu ja mistä aiheista häiriöitä syntyy. Näiden perusteella voidaan tulevaisuudessa, jokin palveluaika päivittää esim. 24/7-muotoon. Jos palvelua halutaan päivittää siitä, tehdään erillinen sopimus toimittajan kanssa.

5.3.1 ServiceNow - Jira -integraatio

Tulevaisuudessa on suunnitelmana integroida Yhtiö Y:n ja Toimittajan tikettijärjestelmät toimivaksi kokonaisuudeksi. Yhtiö Y:llä on käytössä ServiceNow-alusta, toimittajalla käytössä tikettijärjestelmänä Jira. Tikettien liikkuessa molempiin suuntiin halutaan helpottaa molempien toimijoiden Service Deskiä.

5.3.2 Palvelunpäivitys 24/7-muotoon

Palveluhinnat ovat tiedossa 24/7-palvelun osalta eli jos jokin palvelu vaatii lisätukea, niin sitä on saatavilla melko nopeasti. Tällöin Yhtiö Y:n on otettava yhteyttä toimittajan vastuuhenkilöön, jonka kautta neuvotellaan sopimuksen laajennuksesta.

6 Tulokset

Opinnäytetyön tuloksena syntyi Frends-integraatioalustan hoitomalli Yhtiö Y:lle. Tämän avulla Yhtiö Y varmistaa itselleen entistä turvallisempaa ja toimintavarmempaa liiketoimintaa. Hoitomalli pienentää häiriötilanteiden ajallista kestoa ja se näkyy parempana käyttökokemuksena asiakkaille. Hoitomallin avulla päästiin tavoitteeseen eli nykytila saatiin kuvattua kirjallisesti. Nykytila osoitti, ettei hoitomalli ollut tarvittavalla tasolla. Tämän perusteella käytiin yhdessä toimittajan kanssa läpi, miten häiriötilanteet saataisiin ratkaistua mahdollisimman nopeasti.

Suurin muutos kohdistuu vastuun siirtymisestä alustan hoidosta toimittajalle. Yhtiö Y:n vastuulle jää kuitenkin alustan kehitystyö sekä integraatioihin kohdistuvat häiriöiden selvitykset, yhdessä taustajärjestelmien toimittajien kanssa. Hoitomalli tuotti selkeät vastuualueet ongelmatilanteiden ratkaisemiseen yhtiöiden välille.

Kehitysprojekti hoidettiin melko suoraviivaisesti, mutta laadukkaasti alusta loppuun. Kehitysprojektin avulla määritettiin häiriö, muutos, ongelma ja päivitystilanteisiin ohjeet. Sen avulla on saatu sovittua, keneen otetaan yhteyttä ja kuinka toimia häiriön, muutoksen, ongelman tai päivityksen tullen. Yhtiö Y:n Service Deskille on tehty PowerPoint, johon yhteystiedot on kerätty eri taustajärjestelmien toimittajista. Molempien osapuolien Service Deskillä on valmius ja tieto toimia, kun Frends-alustaan kohdistuu toimenpide, joka vaatii reagointia.

Hoitomallin myötä myös toimittajan oikeuksia laajennettiin Yhtiö Y:n Confluenceen sekä Frends-integraatioalustalle. Ennen toimittajalla ei ole ollut pääsyä edellä mainittuihin paikkoihin. Kehitysprojektin avulla saatiin myös hahmotelmaa tulevaisuudelle, jossa suurin tavoite on tehdä integraatio tikettijärjestelmien välille.

7 Yhteenveto

Opinnäytetyön tarkoituksena oli siis luoda Yhtiö Y:n Frends-integraatioalustalle toimiva hoitomalli. Teorianosan avulla lukijan tulisi ymmärtää, mitä integraatioilla tarkoitetaan ja minkälaisia malleja niiden luontiin on. Lukijalle yritetään kertoa selvästi myös perusasiat Frends-alustasta. Tutkimuskysymyksiin vastattiin mielestäni selkeästi nykytilan, tavoitetilan ja vaatimuksien osalta. Tulokset opinnäytetyöstä olivat positiiviset toimeksiantajan näkökulmasta. Hoitomallilla toivotaan olevan positiivisia vaikutuksia Yhtiö Y:n liiketoimintaan. Hoitomalli selventää vastuita yhtiöiden välillä.

Opinnäytetyötä aloitettaessa omat tiedot integraatioista ja Frends-integraatioalustasta olivat melko mitättömät ja haasteena on ollut luoda helppolukuista tekstiä integraatioista sekä Frends-alustasta. Opinnäytetyötä tehdessä sain valtavasti tietoa integraatioiden perusasioista ja niiden toteuttamisesta. Opittavaa jäi vielä tulevaisuuteenkin, etenkin integraatioiden teknisestä toteuttamisesta. Kehitysprojekti opetti, kuinka monesta eri näkökulmasta asioita pitää pystyä tarkastelemaan.

Tulevaisuuden tavoitteena itsellä on työskennellä lisää integraatioiden parissa ja oppia rakentamaan niitä ja soveltamaan niiden hyötyjä liiketoimintaan. Tätä pääsen konkreettisesti toteuttamaan, kun hoitomallissa määritelty ServiceNow - Jira -integraatio toteutetaan. Myös low-coden mahdollisuuksista työelämässä olisi mukava oppia enemmän.

Lähteet (suositellaan Mendeleyn käyttöä)

Abbasi, A. (n.d.-a). *ESB Introduction: Enterprise Service Bus Concepts*. Retrieved February 11, 2022, from https://tutorialspedia.com/esb-introduction-an-overview-of-esb-concepts-capabilities-benefits-challenges/

Abbasi, A. (n.d.-b). *Point to Point vs Hub Spoke vs ESB Integration Architecture*. Retrieved February 10, 2022, from https://tutorialspedia.com/point-to-point-vs-hub-spoke-vs-esb-integration-architectures/

Amazon. (n.d.-a). *API Management - API Tools, Services, and Best Practices*. Retrieved February 5, 2022, from https://aws.amazon.com/api-gateway/api-management/

Amazon. (n.d.-b). *What is an API? - API Beginner's Guide - AWS*. Retrieved February 5, 2022, from https://aws.amazon.com/what-is/api/

Bailey, K. (2018). *Internal vs. External APIs. What Is An API? | by Khallil Bailey | Medium*. https://medium.com/@khallilbailey/internal-vs-external-apis-2e54ef28659a

Flashnode. (2021). *Integraatiot | Ite wikin digitalisoinnin opas*. https://www.itewiki.fi/opas/integraatiot/

Frends. (n.d.-a). *Frends iPaaS | Architecture - Frends Agents*. Retrieved February 3, 2022, from https://frends.com/platform/frends-agents

Frends. (n.d.-b). *Frends iPaaS | Capabilities - Orchestration*. Retrieved February 1, 2022, from https://frends.com/platform/orchestration

Frends. (n.d.-c). *Frends iPaaS | GDPR*. Retrieved February 15, 2022, from https://frends.com/gdpr

Frends. (n.d.-d). *Frends iPaaS | Integration Scenarios - APIs*. Retrieved February 15, 2022, from https://frends.com/platform/apis

Frends. (n.d.-e). *Frends iPaaS | iPaaS*. Retrieved January 25, 2022, from https://frends.com/landing/ipaas

Frends iPaaS | Integration Scenarios - Managed File Transfers. (n.d.). Retrieved May 15, 2022, from https://frends.com/platform/managed-file-transfers

Galkin, O. (n.d.). *How to use Frends UI | frends Docs - Integration Platform Documentation*. Retrieved February 14, 2022, from https://docs.frends.com/en/articles/2189208-how-to-use-frends-ui

Galkin, O. (2022). *Agents | frends Docs - Integration Platform Documentation*. https://docs.frends.com/en/articles/2236679-agents

Gillis, A. S. (2020). *What is REST API (RESTful API)?* https://www.techtarget.com/searchapparchitecture/definition/RESTful-API

Haglund, J. (2018). *Järjestelmäintegraatio, mitä se on selkokielellä?* https://www.alfame.com/blog/jarjestelmaintegraatio-mita-se-on-selkokielella

HiQ. (n.d.). *HiQ Finland - Integraatiot ja integraatioalustat - lyhyt oppimäärä*. Retrieved February 14, 2022, from https://hiq.fi/ajankohtaista/integraatio/

HiQ Finland. (2022a, February 1). *HiQ Finland - Frends integraatioalusta*. https://hiq.fi/palvelut/integroi/frends/

HiQ Finland. (2022b, February 1). *HiQ Finland - Mikä on Frends ja millaista sillä on toteuttaa integraatioita?* https://hiq.fi/blogi/mika-on-frends-ja-millaista-silla-on-toteuttaa-integraatioita/

IBM Cloud Education. (n.d.). *What is API Management? | IBM*. 2020. Retrieved February 5, 2022, from https://www.ibm.com/cloud/learn/api-management

IBM Cloud Education. (2020). *What is an Application Programming Interface (API) | IBM*. https://www.ibm.com/cloud/learn/api#toc-how-an-api-BOm6NChy

IBM Cloud Education. (2021). *What is iPaaS (Integration-Platform-as-a-Service)? | IBM*. https://www.ibm.com/cloud/learn/ipaas

Ismail, K. (2021). *What Is API Management (and Why Do I Need It)?* https://www.cmswire.com/information-management/what-is-api-management-and-why-do-i-need-it/

Lehtonen, K. (2018). *What is system integration?* https://www.youredi.com/blog/what-is-system-integration

RapidAPI. (2022a). *Internal vs External APIs (What's the Difference?) | RapidAPI*. https://rapidapi.com/blog/internal-vs-external-apis/

RapidAPI. (2022b). *What is OpenAPI? | OpenAPI Definition | API Glossary*. https://rapidapi.com/blog/api-glossary/openapi/

Red Hat. (2019). *What is API security?* https://www.redhat.com/en/topics/security/api-security#why-is-api-security-important

RedHat. (2020). *What is a REST API?* https://www.redhat.com/en/topics/api/what-is-a-rest-api

Redwood Logistics. (n.d.). *What is a Hub and Spoke Distribution Model? – Redwood Logistics : Redwood Logistics*. Retrieved February 9, 2022, from https://www.redwoodlogistics.com/the-hub-and-spoke-distribution-model-and-why-it-works-for-ltl/

Steve, D., Gary, O., Matt, B., & Kevin, M. (2021). *How to Evaluate API Management Solutions*. https://www.gartner.com/document/4009107?ref=solrAll&refval=314833980

Tasanen, P. (2019). *Mitä integraatio, rajapinta ja api tarkoittavat?* https://valjas.fi/mita-integraatio-rajapinta-ja-api-tarkoittavat/

Toivanen, A. (n.d.). *HiQ Finland - Integraatiot ja integraatioalustat - lyhyt oppimäärä*. Retrieved January 20, 2022, from https://hiq.fi/ajankohtaista/integraatio/

Liite 1: Aineistonhallintasuunnitelma

Kehitysprojekti:

Kehitysprojektin aikana pidetään päiväkirjaa (aineisto), johon kerätään teknistä tietoa projektista. Tämä tieto analysoidaan opinnäytetyötä varten. Päiväkirjaa säilytetään tekijän tietokoneen C-asemalla, ja siitä tehdään säännöllisesti varmuuskopioita tekijän tietokoneen OneDriveen. Päiväkirjaa säilytetään C-asemalla ainakin vuoden verran opinnäytetyön valmistumisesta. Kehitysprojektin aikana pidetyistä kokouksista pidetään pöytäkirjoja, jotka säilytetään työnantajan koneella sekä tekijän omalla koneella, tarkemmin C-asemalla molemmissa. Valmiin projektin onnistumisesta kerätään tietoa työpaikan antamalle koneelle. Toimeksiantaja omistaa opinnäytetyön aineiston ja tulokset.