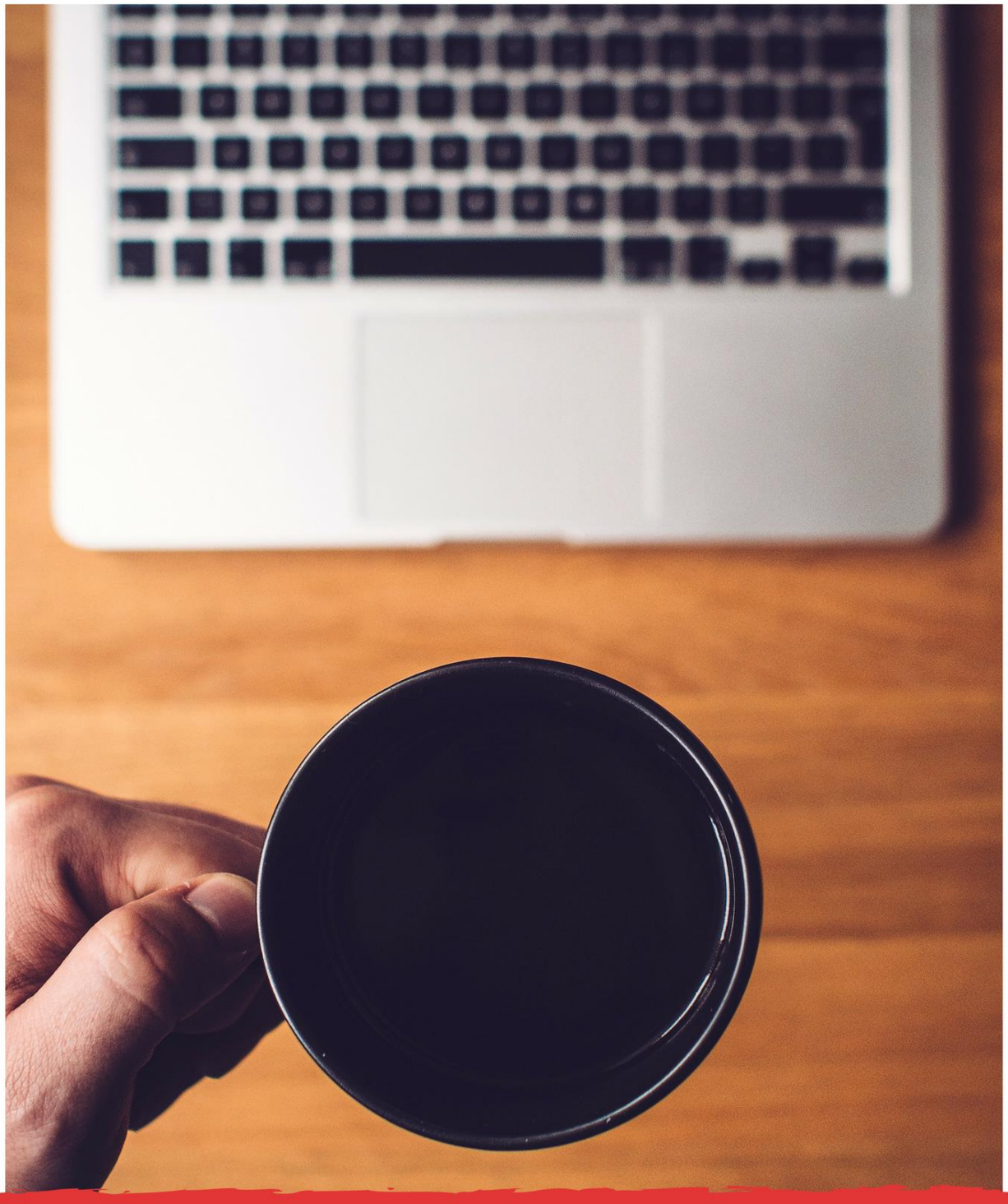
ALFAME



AVAIN KETTERÄN LIIKETOIMINTA-STRATEGIAN TOTEUTTAMISEEN

API/Ohjelmointirajapintaopas IT-päättäjille

Sisällys

ALFAME

Johdatus aiheeseen

Laatutietoisuus työssä käytettävien tietojärjestelmien ominaisuuksista ja hyödykkeistä on kasvanut runsaasti viimeisten vuosien aikana. Enää ei tyydytä hyvään, vaan vaaditaan parasta. Joskus hankintoja tehdään jopa IT-strategian vastaisesti, sillä erilaisia sovelluksia ja järjestelmiä on niin helppo ostaa pilvestä.

Tietoturva ja järjestelmien yhteentoimivuus jäävätkin usein jälkijättöisesti IT-hallinnon päänvaivaksi – esimerkiksi miten asiakastiedot saadaan siirtymään ajantasaisesti CRM-järjestelmästä muihin järjestelmiin tai tuntikirjaukset laskutuksen järjestelmiin.

Palvelulaadun ja ketteryyden takaamiseksi IT-hallinnot joutuvatkin panostamaan integraatiotyökaluihin entistä enemmän. Vakiintunut ratkaisumalli on ESB (*Enterprise Service Bus*), jonka avulla saadaan rakennettua keskitetty palveluväylä ja löyhemmät riippuvuudet (*Loose Coupling*) tietojärjestelmien välille. Tiukat riippuvuudet (*Tight Coupling*) ovat yleensä suurin hidaste kehitykselle, sillä muutokset heijastuvat useisiin eri järjestelmiin.

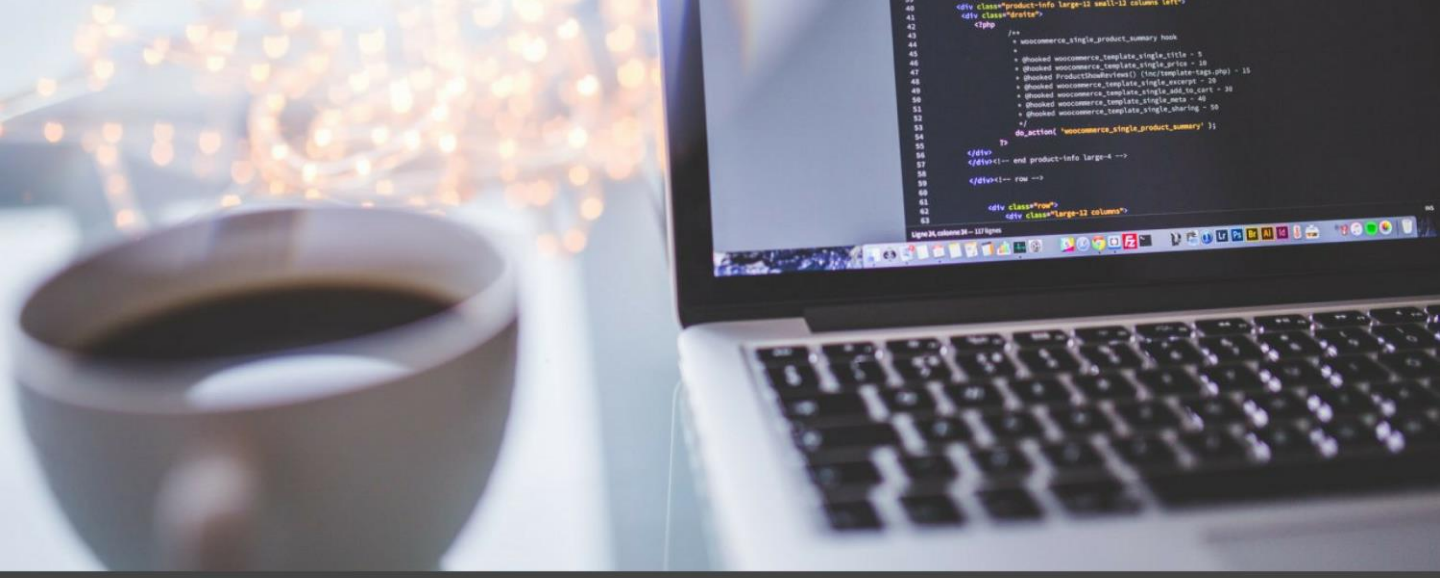
Tietojärjestelmien integrointi toisiinsa ei silti itsessään täytä kaikkia nykyisiä ketteryysvaatimuksia. Tietojärjestelmien sijaan halutaankin tyypillisesti tarjota palveluja, jotka on mahdollista integroida sellaisenaan niitä tarvitseviin sovelluksiin tai toisiin palveluihin.

Perinteisillä ESB-työkaluilla tai monoliittisillä applikaatioilla tällaiset räätälöinnit vaativat laajoja käyttöönottoprojekteja useidenkin toimittajien välillä. “Tätä on siis syytä vältellä” tulee helposti ensimmäisenä mieleen – vaikka todellisten vaatimusten täyttämiseksi vaaditaankin räätälöintiä. Palvelutuotantoon tarvitaan siis uudenlainen lähestymistapa: API (*Application Programming Interface*) ja tuottamiseen sopivat API:n hallintatyökalut (*API Management*).

Tämän materiaalin tarkoituksena on jakaa tietoutta rajapinnoista ja niiden mahdollisuuksista ketterään IT-strategiaan eli API-strategiaan. Jos mieleesi herää kysymyksiä tai haluat tietää aiheesta lisää, otathan yhteyttä!

Mukavia lukuhetkiä oppaan parissa toivottaa,

Alfamen väki



API ELI OHJELMOINTIRAJAPINTA

API eli Application Programming Interface on ohjelmointirajapinta, joka kuvaa vuorovaikutuksen tarjottujen palveluiden kanssa. Se rajaa riippuvuudet varsinaisen toteutuksen ja asiakkaan välillä minimiin eli on eräänlainen menettely löyhään sidontaan. API:a käyttämällä saadaan siis liiketoimintaan teknologia- ja toimittajariippumattomuutta, mikä taas lisää jatkokehitysmahdollisuuksia ja taloudellisia etuja.

Ilman rajapintamäärityksiä sovelluksista tulee monoliittisia, jossa kaikki tarjotut toiminnot on sidottu kiinteästi yhteen. Silloin yksikin muutos voi heijastua koko toteutukseen ja aiheuttaa ennalta tuntemattomia ongelmia. Sen vuoksi sovelluksille määritetään komponenttiarkkitehtuureja ja lisätään rajapintoja komponenttien vuorovaikutuskohtiin, jotta osia olisi mahdollista kehittää tai korvata riskittömämmin.

Web-teknologioiden kehittymisen myötä API:t ovat siirtyneet vahvemmin sovellusten julkaisemiksi palveluiksi sisäisen toteutuksen selkiyttämisen lisäksi. Web Service API:n asiakkuus ei enää sido toteutusteknologiaan tai ajoympäristöön, joten toimittajayhteistyö ja komponenttikohtainen uudelleenjakelu on mahdollista.

Kehitystyö painottuu nykyään API:n määrittämiseen, toteuttamiseen ja hyödyntämiseen. Kehitystä tehdään siis niin sanotusti API edellä (*API first*). Kehityksen tehokkuutta ohjaa kehityksessä hyödynnettävien API-työkalujen laatu. API-työkaluista voit lukea lisää myöhemmin tässä oppaassa.

Rajapintatuotannon kehittyessä API:n julkaisukanava nousee tärkeään rooliin. API:t tulee versioida, jotta asiakkailla on mahdollisuus varautua muutoksiin ja migroitua itse valitsemanaan hetkenä uuteen versioon. Tällöin ikäviä ja etenkin monitoimijaympäristössä työllistäviä palvelukatkoja ei myöskään enää synny.

API:n pääsynhallinta tulee voida konfiguroida hyvinkin dynaamisesti, sillä niiden toteutuksissa ei ole voitu, eikä yleensä kannatakaan, ottaa kantaa jatkuvasti muuttuviin pääsynhallintavaatimuksiin: uusia asiakkaita tulee voida palvella ilman käyttöönottohankkeita.

Lisäksi palveluiden performointia tulee pystyä seuraamaan ja mukauttamaan, jotta ne vastaavat kulloistakin palvelukutsujen määrää. Julkaisukanava voidaan rakentaa API:n hallintatyökaluilla (*API management*).

Myös API-palveluiden tuotteistaminen ja myyminen (*SaaS*) on mahdollista. API:n julkaisukanaviin tehtävä panostus avaa samalla mahdollisuuden tarjota palveluita ulkoisille asiakkaille ja rahastaa niistä. Mikäli olet kiinnostunut API-ratkaisujen tuotteistamisesta, otathan yhteyttä!

API-TYÖKALUT

API:n määritykseen käytettävät teknologiat ovat kaikkien API-työkalujen lähtökohta. Esimerkkeinä teknologioista:

- WSDL
- RAML
- Swagger

Nämä teknologiat mahdollistavat API:n vuorovaikutuksen kuvaamisen sekä dokumentoinnin sitomatta asiakas- tai palvelutoteutusta mihinkään tiettyyn toteutusteknologiaan. Niitä tulkitsemalla on siis mahdollista oppia, miten API:a tulisi käyttää ja kuinka toteutus on rakennettava.

Koska kuvaukseen käytettävät teknologiat ovat vakiintuneita, niiden pohjalta on mahdollista myös automaattisesti rakentaa toteutuksia: toteutusteknologiakohtainen ratkaisun runko syntyy generoimalla kuvaimesta koodi, olipa kyseessä asiakas- tai palvelutoteutus.

Esimerkkeinä:

- Wsdl2Java
- Mulesoft ApiKit
- Postman

API:n tarjoaman löyhän sidonnan johdosta se tarjoaa myös luonnollisen paikan testiautomaatiolle ja väliaikaistoteutuksille. Testaamisen yleisin ongelma on, miten testattava kohde irrotetaan ajoympäristöön niin, että säilytetään riittävä tarkkuus testaamiseen. Koska API-kuvaimista pystytään generoimaan koodirunkoja, siitä voidaan myös helposti generoida väliaikainen palvelutoteutus, jota käytetään testin aikana (Mock tai Stub). Itse testaaminen voidaan tehdä asiakastoteutuksen roolissa eli juuri siten kuin tuotannossakin.

Esimerkkejä testaustyökaluista:

- QUnit
- Robot Framework
- SOAP UI

API MANAGEMENT

API Management on laaja kokonaisuus, jonka pääpainona on tehdä rajapintojen suunnittelusta, käyttöönotosta ja hallinnasta helpompaa. API Management pitää sisällään tietoja, kuten millaisia rajapintoja on käytössä, ketkä niitä käyttävät sekä mitä ja missä niitä tarjotaan. API:n toteuttavien taustajärjestelmien ei tarvitse ottaa kantaa lopullisesta asiakasympäristöstä, vaan niissä voidaan keskittyä tehokkaaseen rajapinta- ja palvelutuotantoon.

API Managementilla helpotetaan API:n käyttöönottoa tarjoamalla julkaisualusta API-kuvaimille. API:t ja niiden tarjolla olevat versiot löytää selaamalla katalogia, josta ne on mahdollista ottaa käyttöön kuvainten tai koodiesimerkkien avulla. Julkaisualusta myös dokumentoi vallitsevaa nykytilaa ja helpottaa siten hallintaa.

Esimerkkejä toteutuksista:

- Azure API Management
- Mulesoft API Manager
- Kong
- WSO2

API Managementille ei ole olemassa standardeja, mikä toisaalta muodostaa haasteen sopivan työkalun valintaan. Ideaalitilanteessa API Management sisältäisi seuraavia ominaisuuksia:

Dokumentaatio ja API-kuvaimet
API:t julkaistaan kuvaimen ja palvelun identifioivan polun eli URL:n avulla. Tyypillisesti API:t versioidaan, jolloin API:sta on mahdollista tarjota useampaa yhdenaikaista versiota, kunnes vanhat versiot pystytään lopulta poistamaan käytöstä. API-kuvaimesta generoidaan yleensä Web-sivu, jossa esitetään dokumentaatio sekä käytön kokeilua sallivat toiminnot. Käyttöönoton helpottamiseksi tarjolla voi olla myös valmiita asiakastoteutuksia tai työkalut niiden rakentamiseen.

Markkinapaikat
Julkaistujen API:en etsintä, hankinta ja rahastus tehdään yleensä verkkoportaalin kautta. Maksullisiin palveluihin myönnetään tyypillisesti API-avain (API-key), joka sallii käyttöoikeuden hankittuihin palveluihin.

Analytiikka ja tilastot
Palveluiden käytöstä on tärkeä kerätä dataa, jotta palvelua voidaan tarvittaessa skaalata tarpeen mukaan. Kerätty data toimii myös palautteena palvelulaadusta, joka on pohjana uusille tarvekartoituksille ja kehityshankkeille.

Suorituskyky
Mikäli palvelun resurssit eivät ole kerätyn datan perusteella riittäviä, on niitä mahdollista skaalata eli toteuttaa ”rate limitting” tai ”load balansointi” uusille rinnakkaisjärjestelmille. Myös tiedon puskurointi on mahdollista, mikäli samaa tietoa noudatetaan toistuvasti lyhyen ajan sisään.

Turvallisuus ja hallinta
Kaikki käyttäjät on syytä tunnistaa käyttöoikeuksien vahvistamiseksi, mutta myös mahdollisten väärinkäyttöjen estämiseksi. Lisäksi rahastus on mahdollista kohdentaa tarkasti yksittäisten käyttöoikeuksien perusteella.

API:N HYÖDYT IT:LLE JA LIIKETOIMINNALLE

API tarjoaa mitattavia hyötyjä niin IT:lle kuin koko organisaatiollekin. IT:n näkökulmasta API:n hyödyt kanavoituvat sen tuottavuuteen, kun taas liiketoiminnalliset hyödyt syntyvät lisääntyvän kilpailukyvyn ja myynnin kautta. Seuraavaksi kerromme tarkemmin hyödyistä, jotka API mielestämme tuottaa:

Helppo ylläpito ja jatkokehitys

Julkaistut API:t mahdollistavat jatkokehityksen myös oman organisaation ulkopuolella. Omien palveluiden arvoa on mahdollista kehittää rakentamalla ekosysteemejä kumppaneiden ja asiakkaiden avulla. Esimerkiksi KONE on viime vuosina rakentanut itselleen täysin uuden kehittäjäverkoston, joka pohjautuu API-ratkaisuihin.

Palveluiden tuotteistaminen tukee myös sisäistä kehitystä: API management -alustan julkaisu- ja hallintaominaisuudet palvelevat myös sisäisiä kohderyhmiä. Mitä useamman API:n julkaisuun saadaan jalkautettua sama alusta, sitä tehokkaampaa niiden kehitys, julkaisu ja käyttöönotto on.

Toimittajariippumattomuus

Toimittajariippumattomuus takaa, että rajapinta ja sitä hyödyntävät palvelut eivät kärsi, vaikka palveluja uudistettaisiinkin. Esimerkiksi CRM-järjestelmän vaihtaminen ei vaikuta API:n ansioista muihin yrityksen järjestelmiin tai palveluihin, vaan ne säilyvät ennallaan.

Mikäli käytössä olevaan rajapintaan ei ole enää saatavilla toteutusta, se voidaan esim. tarjota ESB-alustalla toteuttamalla vanha rajapinta uudelleen ja mappaamalla kutsut nykyiseen palveluun. Tällöin uuden palvelun käyttöönotto on ketterää, koska uusiin rajapintoihin voidaan migroitua vaiheittain.

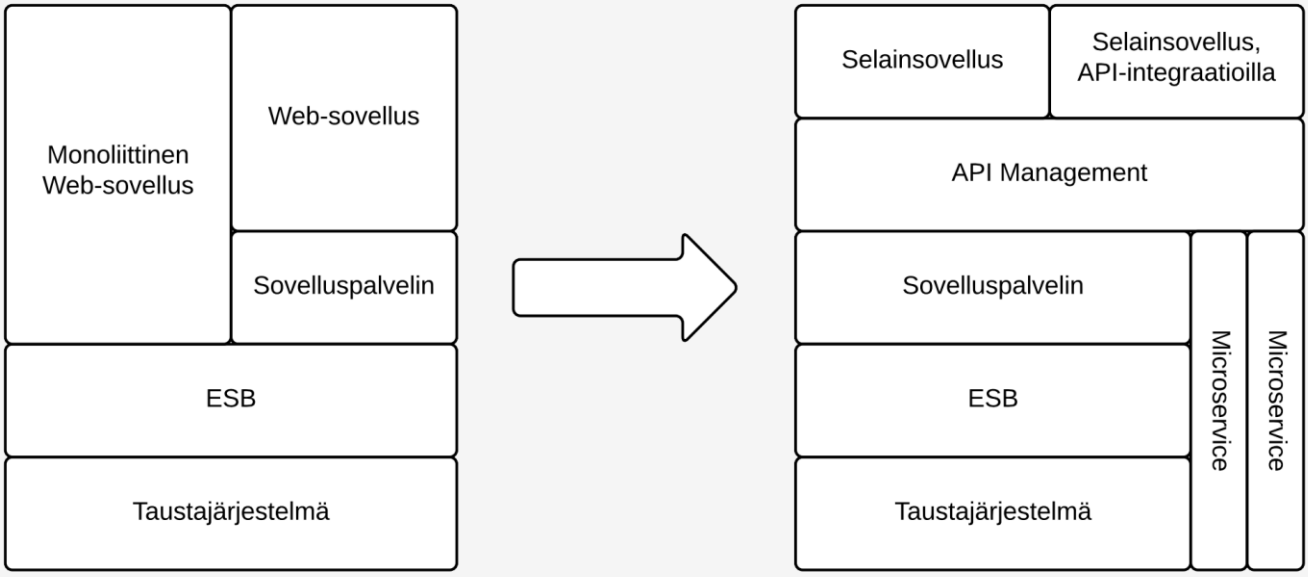
API-strategia osana IT-arkkitehtuuria

IT-ratkaisut ovat tyypillisesti olleet laajoja teknologia- ja toimittajasidonnaisia palvelukokonaisuuksia, mutta API-strategiaan keskittymällä tiukoista sidoksista pääsee irrottautumaan. API management –työkalujen ja ketterien virtualisointi-/konttiteknologioiden ansiosta palveluista on mahdollista rakentaa itsenäisiä (Microservices).

Perinteinen ja harmaita hiuksia aiheuttava tilanne organisaatioissa onkin, että IT on jumissa vanhojen sovellusten ja taustajärjestelmien kanssa, joihin ei uskalleta tehdä muutoksia. Toimittajat eivät osaa ylläpitää tai kehittää vanhoja järjestelmiä. Koko liiketoiminnan pelätään jopa kaatuvan käsiin, jos niihin tehdään muutoksia. API Managementin ansiosta ei tarvitse pelätä järjestelmien hajoamista, vaan rajapinnat mahdollistavat esimerkiksi vaiheittaisen käyttöönoton, ketterän testaamisen ja selkeän palveluiden dokumentoinnin.

API-strategiassa monoliittiset sovellukset pyritään korvaamaan täysin selainsovelluksilla. Tällöin tuotettuja API-palveluja on mahdollista integroida suoraan loppukäyttäjien selaimissa. Silloin voidaan pala kerrallaan tuottaa uusia palveluja julkaisemalla uusia API-versioita hyödyntämällä rakennettua API-julkaisukanavaa.

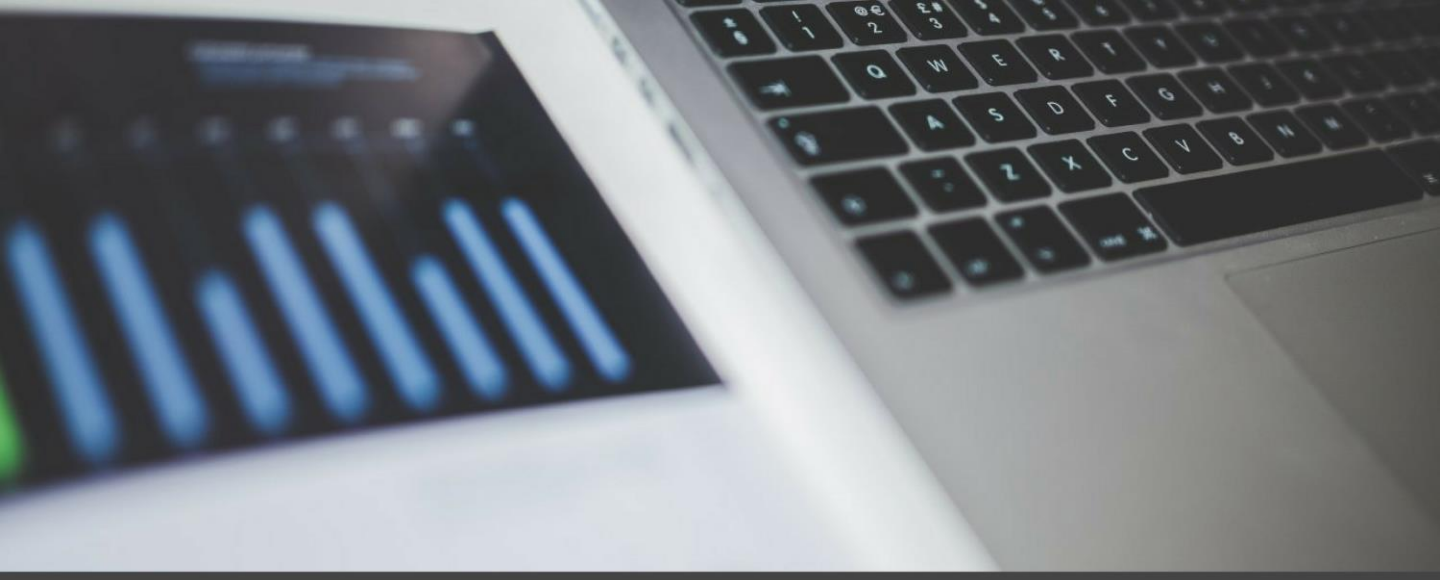
Kuvassa vasemmalla vanhan maailman ratkaisut ja oikealla API-kerroksella varustettu toivetila:



Tuotteena tarjoaminen

API:n tuotteistaminen voi jopa olla lähtökohtana API:n toteuttamiselle. Näin ei kuitenkaan aina ole, vaan yritys saattaa ensin tuottaa palvelua itselleen, kunnes saadaan idea, miksi ei tarjottaisi palvelua myös muille.

API management -alustalla tuotteen julkaisu on helppoa: samaa menettelyä on harjoitettu jo sisäisesti, nyt vain lisätään tuotteesta rahastus ja markkinointi. Yritykselle aukeaakin uusi kanava liiketoiminnan laajentamiseen. Tuotettuja palveluja voi myös markkinoida ulkoisissa API-portaaleissa, kuten API:Suomessa.



INTEGRAATIOT JA API

Integraatiot ovat pitkään olleet kiinteä osa IT-arkkitehtuuria. Ne ovat yleensä erittäin liiketoimintakriittisiä ja vaativat jatkuvaa seurantaa. Siksi integraatioratkaisut pyritään keskittämään esim. ESB-alustalle, joka on tarkasti seurattavissa ja auditoitavissa.

Toiminnallisesta näkökulmasta integraatioita on karkeasti kolmea eri tyyppiä:

- API-palveluiden yhteen liittäminen ja tarjoaminen
- Olemassa olevien applikaatioiden ja datan yhteen liittäminen
- Monimutkainen prosessiautomaatio ja ohjelmistorobotiikka

Julkaistu palvelu on ideaalitilanteessa sellainen, joka on mahdollista ottaa käyttöön ja integroida saumattomasti. Web Service/SOAP-teknologioilla toteutetut palvelut vaativat kohtuullisen monimutkaisen asiakasohjelmiston ja siksi ne ovat tyypillisesti piilotettuja loppukäyttäjiltä osaksi applikaatioita tai integraatioita. Siten jokainen uusi loppukäyttäjälle kehitetty palvelu heijastuu helposti myös IT:lle. Siitä seuraa uusien integraatioiden kehitys- ja käyttöönottohankkeita, jotta tuotetut palvelut saadaan vastaamaan nykyisiä tarpeita.

Palveluiden integraatioista tulisikin saada mahdollisimman lähellä loppukäyttäjää toteutettavia. REST-teknologioilla toteutetut API:t on helppo käyttöönottaa kevyemmissäkin laitteissa, sillä asiakasohjelmistoksi riittää HTTP-asiakasohjelmisto. Sen vuoksi REST API on mahdollista ottaa käyttöön suoraan käyttäjän selaimessa, eivätkä kaikki uudistukset enää heijastu integraatiomuutoksina IT:lle.

Mitä useampi palvelu saadaan julkaistua REST API:na, sitä enemmän niitä on mahdollista integroida API-kerroksen ulkopuolella. Yhä useampi palvelu kehitetäänkin suoraan REST API –palveluksi, joita voidaan jakaa versioituina ketterin DevOps-menetelmin API management –alustan avulla. ESB:n tehtävät API-palveluiden yhteen liittämisessä ovat sen vuoksi vähenemässä ja API-palveluiden seuranta siirtymässä API management –alustan tehtäväksi.

ESB-alustan rooli on jatkossa yhä vahvemmin monimutkaisen prosessiautomaation ja ohjelmistorobotisaation työkaluna. Toimenpiteet, joita tehdään ilman ihmistä, vaativat edelleen jatkuvaa seurantaa ja auditointia. Kehitys painottuu liiketoimintaprosessien (BPMN, BPEL jne.) ja –sääntöjen määrittämiseen (DMN, Drools jne.) sekä API-palveluiden tuottamiseen, jotta ihmiset voivat osallistua osana tekoälyä.

Haluatko lisätietoja?

API:t ovat taloudellinen investointi, joiden käyttöönottoa on hyvä miettiä huolella. Ne eivät ole pelkästään teknisiä päätöksiä, vaan APIen avulla ohjataan koko liiketoimintaa. Mikäli haluat tietää lisää tai sinulla heräsi kysymyksiä aiheesta, otathan rohkeasti yhteyttä.

Tutustu myös blogiimme ja maksuttomaan tietopankkiimme!

Janne Tirkkonen
Toimitusjohtaja
044 906 2216
janne.tirkkonen@alfame.com

Jaana Anglé
Ratkaisumyyjä
050 465 8668
jaana.angle@alfame.com

Jari Saari
Ratkaisumyyjä
050 470 0958
jari.saari@alfame.com

ALFAME

Alfame lyhyesti

Olemme vuonna 2004 perustettu riippumaton tietojärjestelmätoimittaja ja ohjelmistokehittäjä. Autamme asiakkaitamme digitalisoimaan ydinprosesseja, tehostamaan kommunikaatiota ja kasvattamaan kilpailukykyä.

Palvelemme kasvavaa yritys- ja julkishallinnon asiakaskuntaamme valtakunnallisesti – tunnustetulla hyvällä asenteellamme ja korkealla osaamisellamme. Olemme teknologiariippumaton yritys. Toteutamme ratkaisuja sekä Microsoftin että avoimen lähdekoodin teknologioilla.

Haluamme erottua markkinoilla toimintamallillamme, joka perustuu luotettavuuteen, läpinäkyvyyteen ja 100 prosenttiseen tyytyväisyystakuuseen.

http://www.alfame.com