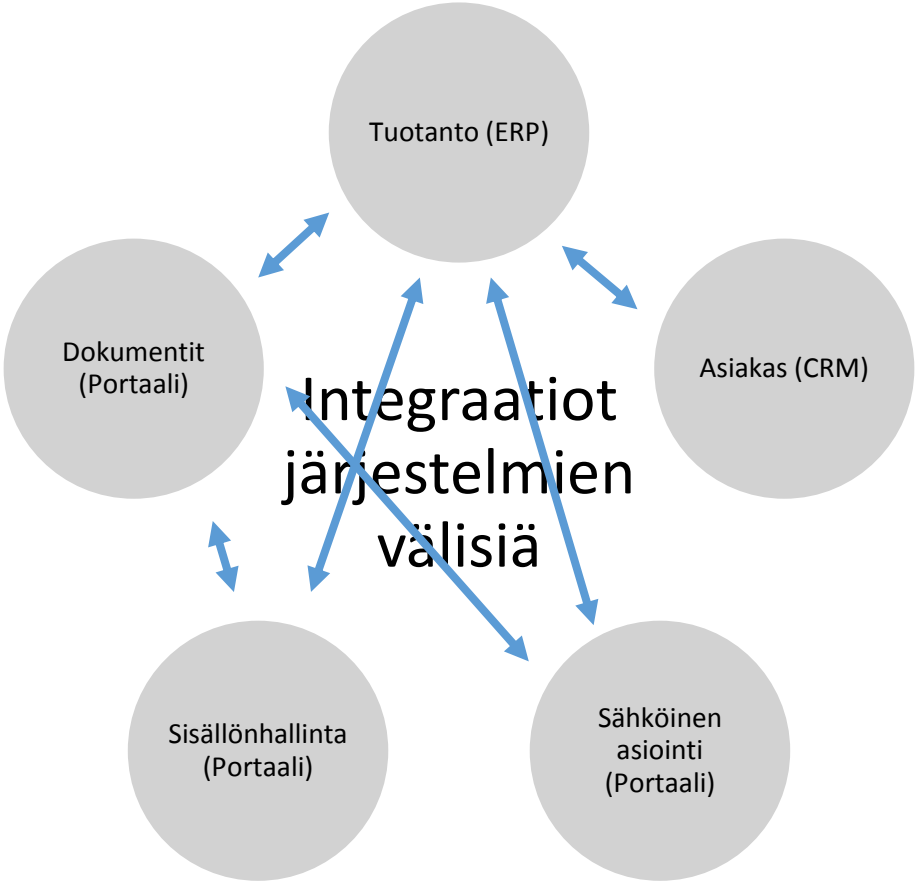
# Toimittaja – Looginen väylärakenne + kirjastot

Kalle Launiala, ProtonIT Oy

kalle.launiala@protonit.net

+358445575665

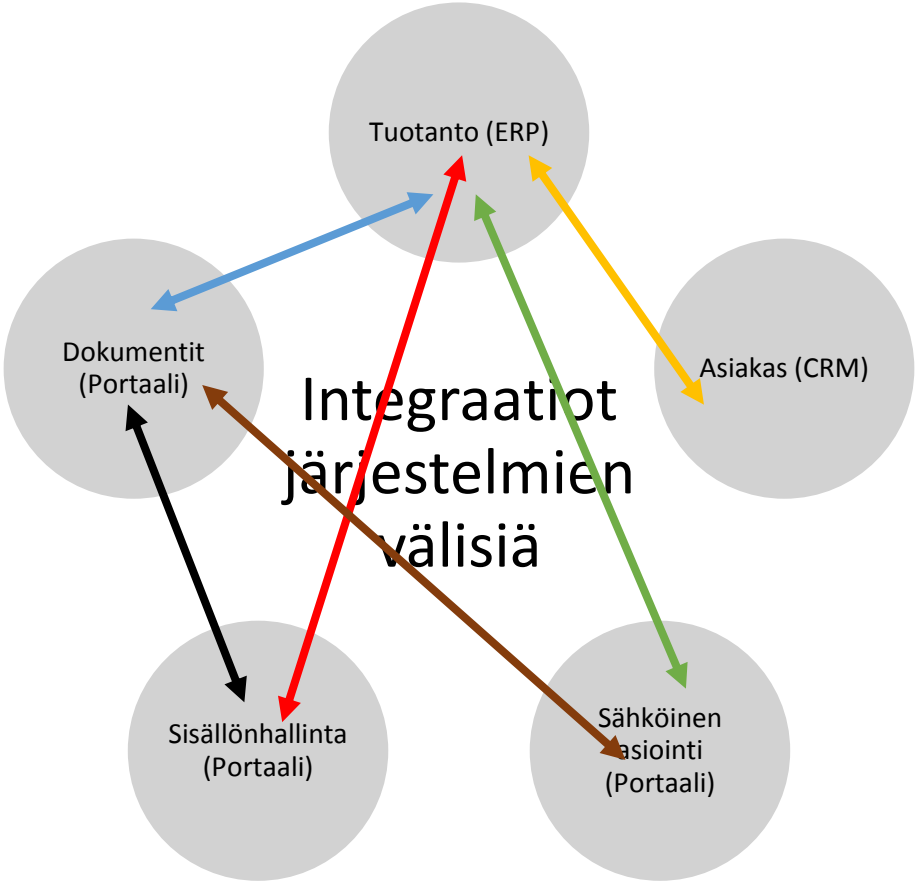
# Nykyiset integraatiot teknisiä, eivät informaatiota kontrolloivia



# Integraation toteutus nykyisin

1. Suunnitellaan ja päätetään, mitä pitää siirtää
    1. Dokumentoidaan asia määrittelyksi
2. Suunnitellaan tekniset kutsut siirrolle
    1. Palvelun suunta, Kuka kutsuu, kuka palvelee
    2. Palveluiden tarkat nimet, parametrit, paluuarvot
    3. Päätetään yhteiset tekniset protokollat
    4. Dokumentoidaan asia määrittelyksi
3. Toteutetaan joko koodaamalla, tai konfiguroimalla
    1. Konfigurointi aktivoi protokollia, toimii paremmin lupauksena kuin käytännössä
4. Ylläpidetään muutokset molempiin päihin yhtaikaa
    1. Jos on säästetty työssä integroimalla samalla useampi taho laajemman tarpeen puitteissa, ylläpidettävä yhtaikaa
5. Koodataan käsin customoinnit, kun käyttäjätunnistus, tiedon omistajuus tai joku muu ero järjestelmien välillä ilmenee
    1. Ei yhtenäistä ylläpidettävyyttä, täysin tapauskohtaista, usein myös toteuttajakohtaista
    2. Erikoistuneet järjestelmät katsovat AINA asioita eri perspektiivistä, jolloin eroja on AINA – siksi ne järjestelmät on alun perin hankittukin
    3. Integrointi menee aina dataan ja sääntöihin asti
    4. (Dokumentoidaan asia päivittäen määrittelyä – ei käytännössä ihan aina tapahdu)

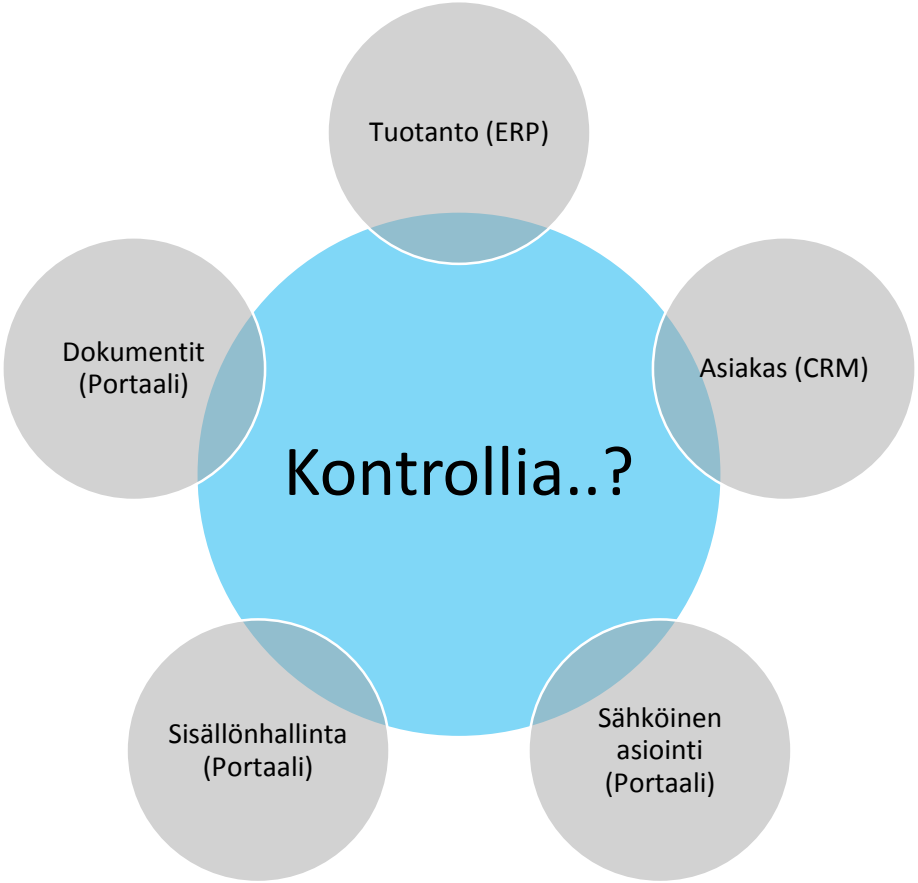
# Nyt integraatiot ovat uniikkeja, dataan ja sen sääntöihin asti meneviä



# SEIS!


Tuotanto (ERP)
Dokumentit (Portaali)
Asiakas (CRM)
Mitä integraatiolla tavoitellaan?
Sisällönhallinta (Portaali)
Sähköinen asiointi (Portaali)

# Kokonaisuuden automatisointia, kokonaisuuden hallintaa..?



# Teknisesti integraatio tapahtuu järjestelmien rajapinnoissa

# Integraation määritys loogisesti

1. Suunnitellaan ja päätetään, mitä pitää siirtää
    1. ~~Dokumentoidaan asia määrittelyksi~~ Kuvataan loogisella tasolla => sisältää myös dokumentaation
2. Suunnitellaan ~~tekniset~~ loogiset kutsut siirrolle
    1. Palvelun suunta, Kuka kutsuu, kuka palvelee
    2. Palveluiden tarkat nimet, parametrit, paluuarvot
    3. ~~Päätetään yhteiset tekniset protokollat~~
    4. ~~Dokumentoidaan asia määrittelyksi~~ Kuvataan loogisella tasolla => sisältää myös dokumentaation
3. ~~Toteutetaan joko koodaamalla, tai konfiguroimalla~~
    1. ~~Konfigurointi aktivoi protokollia, toimii paremmin lupauksena kuin käytännössä~~
4. ~~Ylläpidetään muutokset molempiin päihin yhtaikaa~~
    1. ~~Jos on säästetty työssä integroimalla samalla useampi taho laajemman tarpeen puitteissa, ylläpidettävä yhtaikaa~~
5. ~~Koodataan käsin customoinnit, kun käyttäjätunnistus, tiedon omistajuus tai joku muu ero järjestelmien välillä ilmenee~~
    1. ~~Ei yhtenäistä ylläpidettävyyttä, täysin tapauskohtaista, usein myös toteuttajakohtaista~~
    2. ~~Erikoistuneet järjestelmät katsovat AINA asioita eri perspektiivistä, jolloin eroja on AINA – siksi ne järjestelmät on alun perin hankittukin~~
    3. ~~Integrointi menee aina dataan ja sääntöihin asti~~
    4. ~~(Dokumentoidaan asia päivittäen määrittelyä – ei käytännössä ihan aina tapahdu)~~
6. Määritellään loogiset tarpeet viite-arkkitehtuurin kontrolloimiin paikkoihin
    1. Yhteisten tarpeiden ja yhtenäistämis-kandidaattien tunnistaminen
    2. Tunnistetaan myös jatkossa loogisesti yhtenevät tarpeet

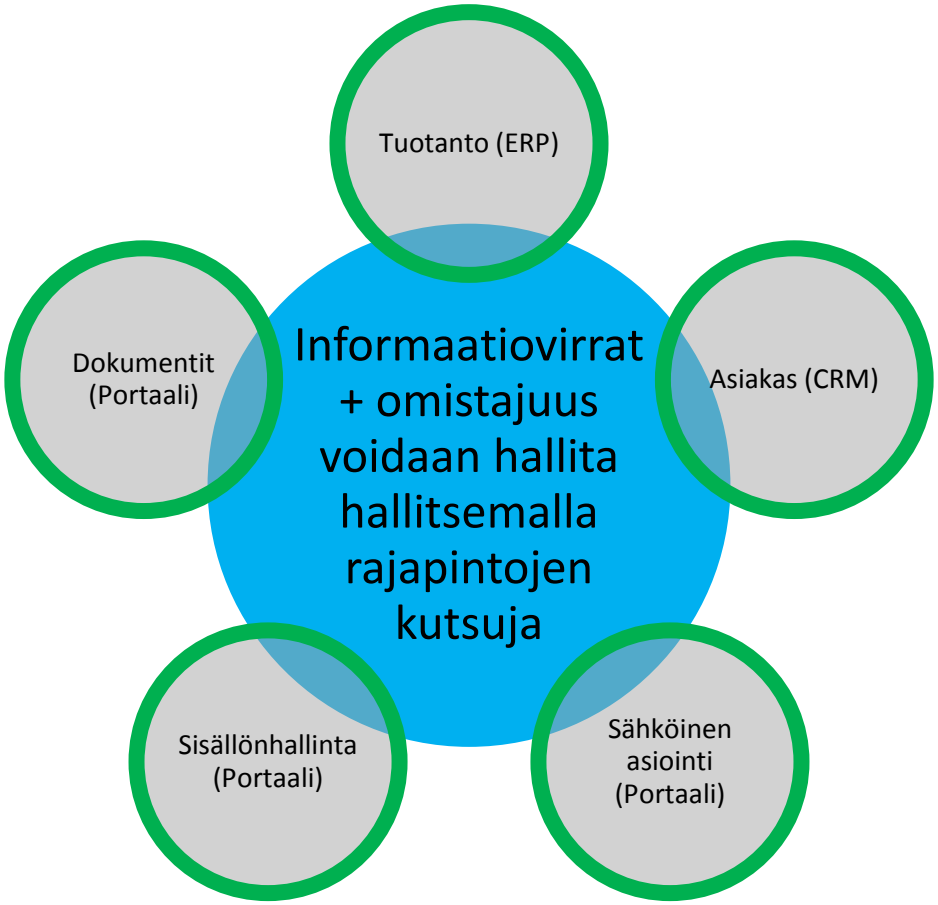
# Loogisen määrittelyn realisointi ADM-automaatiolla

1. Suunnitellaan ja päätetään, mitä pitää siirtää
   1. Kuvataan loogisella tasolla => tuottaa myös dokumentaation
2. Suunnitellaan loogiset kutsut siirrolle
   1. Palvelun suunta, Kuka kutsuu, kuka palvelee
   2. Palveluiden tarkat nimet, parametrit, paluuarvot
   3. Kuvataan loogisella tasolla => tuottaa myös dokumentaation
3. **Mikäli toteutusta tarvittuun teknologiapinoon ei vielä ole**
   1. Tehdään referenssitoteutus tavanomaiseen tyyliin käsin
   2. Tehdään ADM-automaatti siirtämällä referenssi generaattoreiksi
   3. Julkaistaan oma lisäys/muutos/päivitys verkoston repositorioon
4. Toteutetaan customoinnit viite-arkkitehtuurin kontrolloimiin paikkoihin
   1. Yhtenäinen ylläpidettävyys koko verkoston yli, ei henkilöitymistä
   2. Tunnistetaan jatkossa edelleen automatisoitavissa olevat toistuvat patternit => toteutetaan verkostoon

# Toistuva toteutus vähenee

- Toteutus tarvitaan ”ensimmäisen kerran”
  - Viitearkkitehtuurin mukainen ”best-practice” toteutus
- Teknologia-kohtaiset toteutukset tarvitaan kertaalleen
  - Korvaa käsityötoteutukseen verrattuna 80-90% ajettavasta koodista
- Pitää myös ylläpitää, mutta ylläpito ja kehitys kohdistuvat aina **kaikkiin samaa moduulia käyttäviin**
  - Yhteisö hyötyy yhteiskäyttöisyydestä aidosti

# Looginen informaation hallinta mahdollistuu => Pallo-malli



# Palvelukirjastot

Hajautetun palveluketjun tai ”väylän” realisointi

# Teknisestä minimistä semanttiseen minimiin

- Tekninen minimi palvelun kutsumiselle
  - Palvelun nimi
  - Palvelun tekniset parametrit
  - Palvelun tekninen paluuarvo
- Semanttinen minimi palvelun määrittelyyn
  - Palvelun nimi
    - Semanttisesti MinunApp.HaeHenkilö vs. VRK.HaeHenkilö
  - Palvelun semanttisesti nimetyt parametrit ja paluuarvo
    - MinunApp.HenkilöTunnus vs. VRK.HenkilöTunnus
- Tekninen + Semanttinen Yhdessä muodostavat palvelun ”sormenjäljen” tai ”signaturen”
  - VRK.HaeHenkilö – niminen palvelu
  - Parametri: VRK.HenkilöTunnus
  - Paluuarvo: VRK.Henkilö

# ADM:n kiihdytys Pallo-palveluihin

- Semanttinen määrittely ja sormenjälki syntyvät ADM:llä automatisoidusta loogisen tason palvelukuvauksesta
  - Automaation määrämuotoisuus ohjaa luontevasti katalogiin/kirjastoon – dynaamiseen malliin
  - Myös kirjastoon rekisteröinti voidaan automatisoida
- Migraatio nykyisestä
  - Motivaationa tuottaa palvelu ”väylään” eli kirjastoon
  - Loogisen tason kuvaus nykyisestä teknisestä tasosta tehdään ADM:llä toteutettuun suunnittelutason abstraktioon
  - Yhtenäistetään palvelurajapinta ottamalla ADM-automaatio käyttöön (kuten edellä kuvattu)
- Uuden tekeminen
  - Määrittelyt kannattaa tehdä loogisesti joka tapauksessa => työmäärä pienempi
  - Lopputulos suoraan väylävalmista tuotantoa

# Kirjasto konkreettisesti on vain palvelu

- Yhtälailla hajautettu palvelu kuin muutkin
  - Kuka tahansa voi julkaista kirjastopalvelun
  - Kehen tahansa ei tarvitse luottaa – eikä automatiikkakaan luota
  - Oleellista on voida päättää kenen kirjastoon luottaa

- Hakukriteerinä lisäksi mahdollisuus liittää metatietoa tarpeen mukaan
  - Kriittinen elementti on ”semanttinen sormenjälki”
  - Yksityisen sektorin vertikaalit määrittävät itse omat metatietonsa

- Kirjasto palauttaa haun perusteella vähintään täydellisen semanttisen kuvauksen + URL:n, josta palvelu löytyy
  - Testi + demokäyttö voidaan mahdollistaa rinnakkaisilla URL:eilla palvelustubeilla, demodatalla jne...
  - Konkreettinen infra voidaan luoda tukemaan ratkaisua, esim JulkICT lab:n toimesta
  - Mitään raskasta demorakennetta ei ole pakko tehdä

# Tekninen kutsu: semanttinen signature + URL, josta palvellaan

# Kirjastosta poimittavien palveluiden hyödyntäminen ”ketjuna”

- MinunApp sisäinen palvelu voi käyttää VRK:n datamalleja ja funktionaalisia palvelulohkoja
  - Datamallit ovat VRK.Henkilö – avaruuden mallit
  - Palvelun funktionaalisuus tunnistaa saman avaruuden mallit
  - VRK:n Pallolla vastuuna on tarjoilla VRK:n hallinnoima data
  - Funktionaalisuus ko. dataa vastaan ajetaan tarvittaessa myös muualla

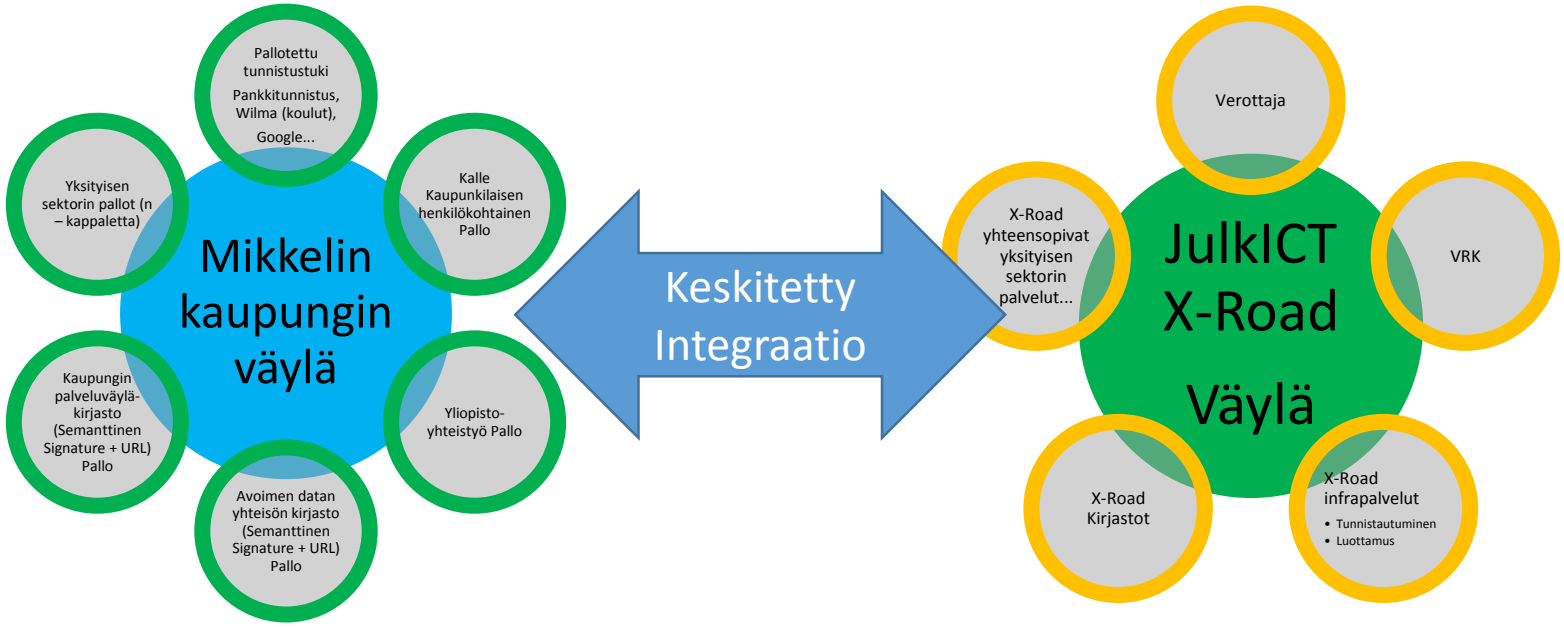
- Sovellus ajaa valmiiksi tehtyjä sovelluksia omaa datavarantoaan vasten
  - Semanttinen malli ei ota kantaa siihen, onko data VRK:n dataa. Semanttinen malli kuvaa VRK:n mallin.
  - Eli VRK:sta luotettavasti tuotu data vaikka VERO:n järjestelmiin jalostetaan eteenpäin, kuten nykyisinkin VERO:n omiin malleihin

- **Uutta – henkilökohtainen alusta:** Yksityisen sektorin tarjottavissa
  - Ohjelmistotuottajat voivat tehdä funktionaalisuutta, joka käyttää VRK.Henkilö ja VERO.Verotiedot2011 tietorakenteita
  - Funktionaalisuus ajetaan kansalaisen omaa tietoa vasten, kansalaisen omassa tietosuojatussa, auditoidussa hiekkalaatikossa
  - Funktionaalisuus = Palvelu, ei tarvitse välttämättä tapahtua verkon/väylän yli

# Väylätason integraatio

Kaupunkitason väyläliitos kansalliseen väylään

# Kokonaisuuksien integraatio voidaan nähdä tavanomaisesti



# Tai käyttäjän näkökulmasta – väylä yhdistää palvelut