SOA ja mikropalvelut

In house kehitys:
- Korppi 4 kehittäjää
- Roti ó, etc 4 kehittäjää
- Python jusu 4

Käytössä olevia kieliä:
- Java
- Python
- Smalltalk (Pharo)
- Go

Merkittäviä "tuotteita":
- Korppi
- Koppa
- Moniviestin
- Tilavuokraus
- Verkkomaksut
- ...

Sopimalla käytännöistä sähkeneistin konfiguraation tarvetta.

KISS!?

SOA ja mikropalvelut

Panu Wetterstrand

JYVÄSKYLÄN YLIOPISTO
IT-PALVELUT

In house kehitys:

- Korppi 4 kehittäjää
- Roti & etc 4 kehjittäjää
- Python jutut 4

Käytössä olevia kieliä:

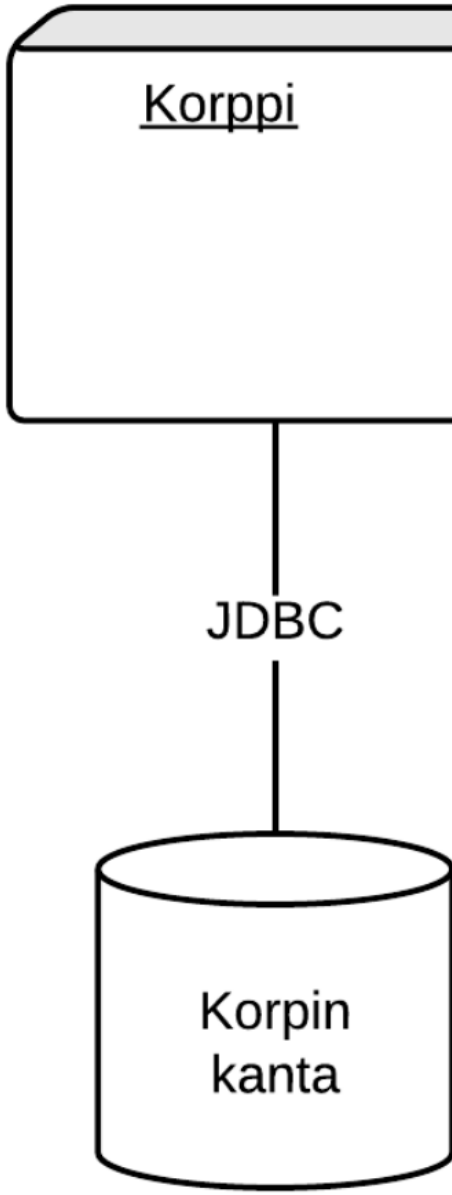
- Java
- Python
- SmallTalk (Pharo)
- Go

Merkittäviä "tuotteita":

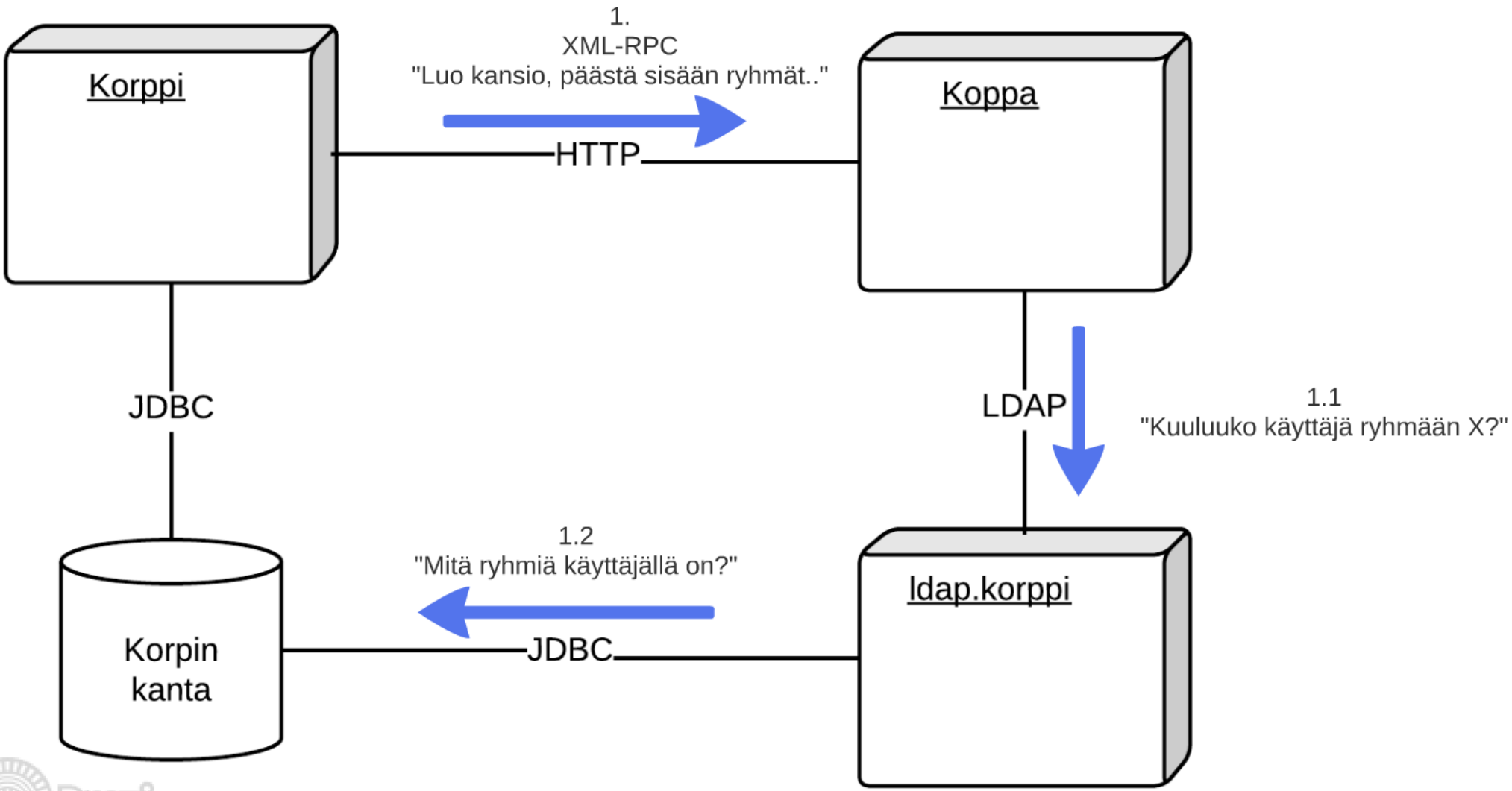
- Korppi
- Koppa
- Moniviestin
- Tilatallennus
- Verkkomaksut
- ...

Korppi

- Kehitetty vuodesta 1998 lähtien
- Hyvin perinteinen Java web-sovellus
  - Java, JSP, Tomcat, Postgresql
  - Monoliittinen arkkitehtuuri



Ensimäisiä integraatioita yhteys kurssimateriaali järjestelmään (Koppa)



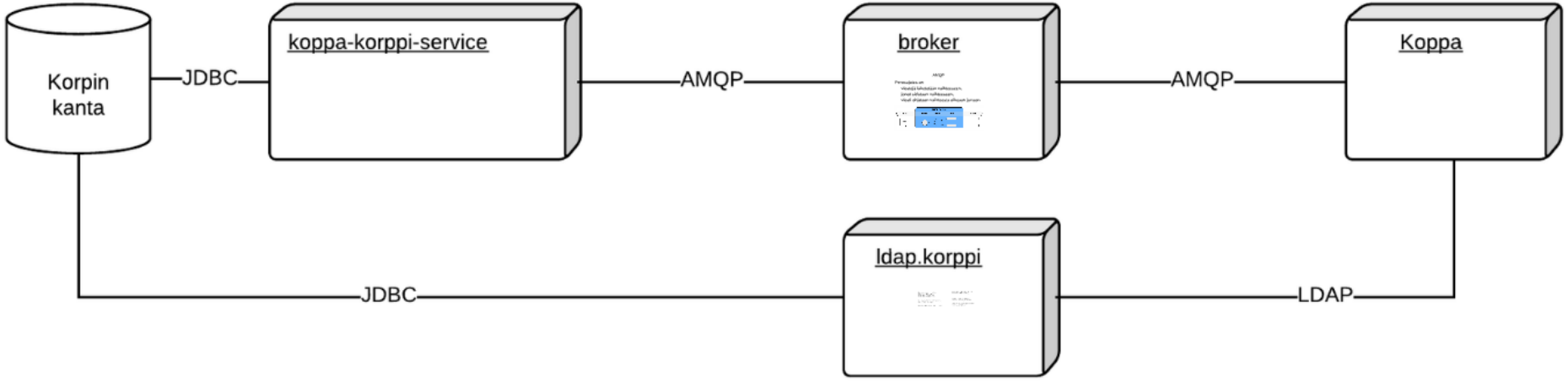
"Voitteko siirtää kentän Z?"

"Kyllä, mutta voi hiukan kestää.."

- Tee muutokset
- Käännä koko korppi
- Valitse / luo sopiva testikanta
- Sovi asiasta Koppa-puolen kanssa
- Käynnistä Korppi testausta varten
- Testaa
- ...

Korpin päivitysväli 2 viikkoa, koska päivitys häiritsee käyttäjiä.

Koppa viestintä siirettiin omaan palveluunsa



Päivitykset onnistuvat ilman Korpin edustan päivittämistä

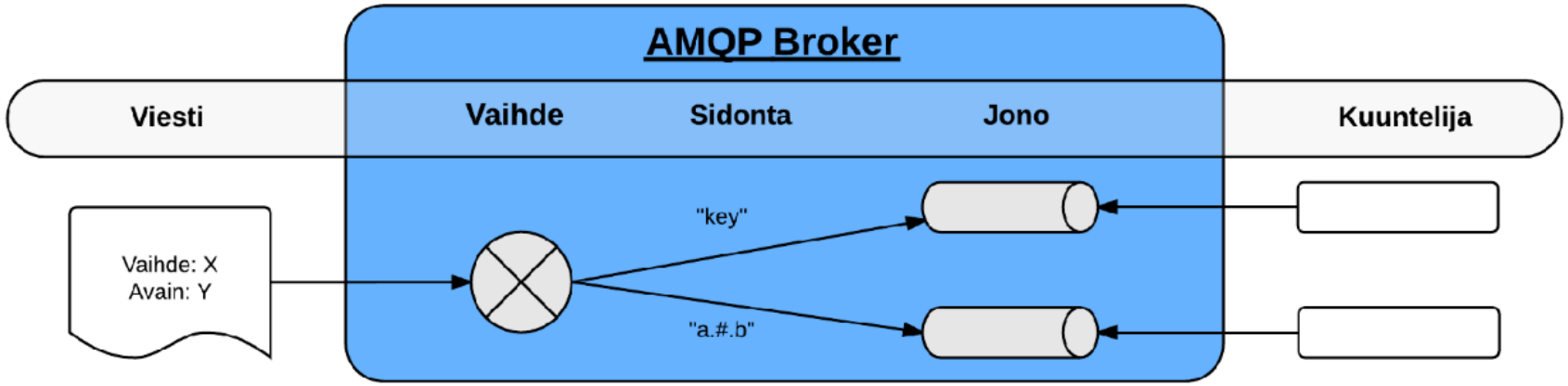
AMQP

Perusajatus on

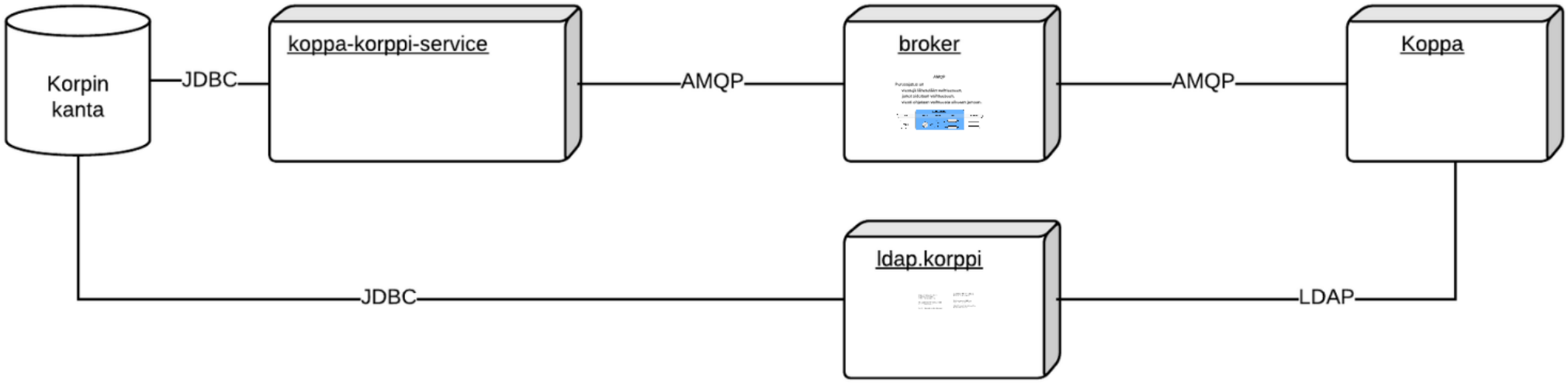
viestejä lähetetään vaihteeseen,

jonot sidotaan vaihteeseen,

viesti ohjataan vaihteesta oikeaan jonoon.



Koppa viestintä siirettiin omaan palveluunsa



Päivitykset onnistuvat ilman Korpin edustan päivittämistä

Korpissa ei oikeasti ole ryhmiä kaikelle tarvitulle tiedolle (kurssilaiset, opettajat yms.)

Tiedot pitää oikeasti muodostaa useilla hauilla. Tämä on hidasta.

Tarvittiin välimuistia, rinnakaistusta, yms.

Lopputulos ei ollut kovin selkeä tai luotettava. (Tosin se toimii)

Ohjelmien tulisi mielellään olla käynnistettäviä ja yksisäikeisiä.

Säikeiden on parempi kommunikoida viesteillä kuin jakaa tilaa.

SOA?

(Service Oriented Architechture)

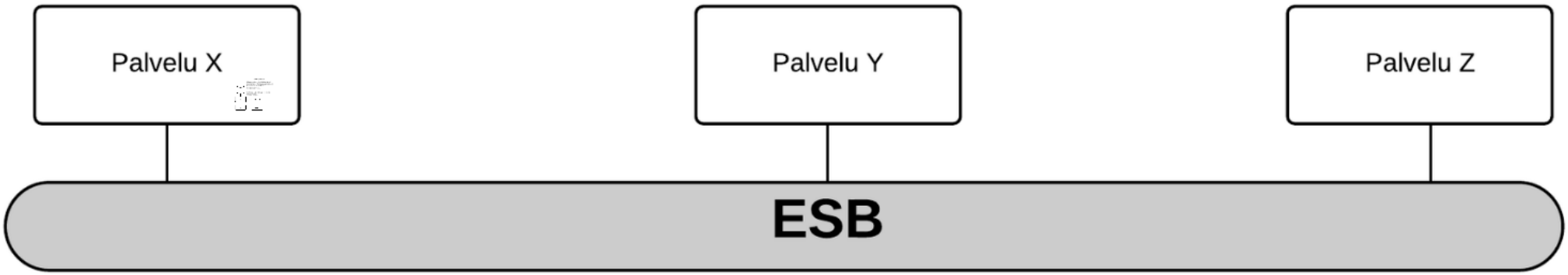
Huomattiin yllättäen, että mehän tehdään SOAa...

Prosessit keskustelevat keskenään yhteisesti sovitulla tavalla.

- Toteutuskieli voidaan valita palvelukohtaisesti
- Kaikki ei kaadu kerralla

Yleensä palvelut rajataan liiketoimintakäsitteiden mukaan.

Lisäksi tarvitaan jonkinlainen ESB

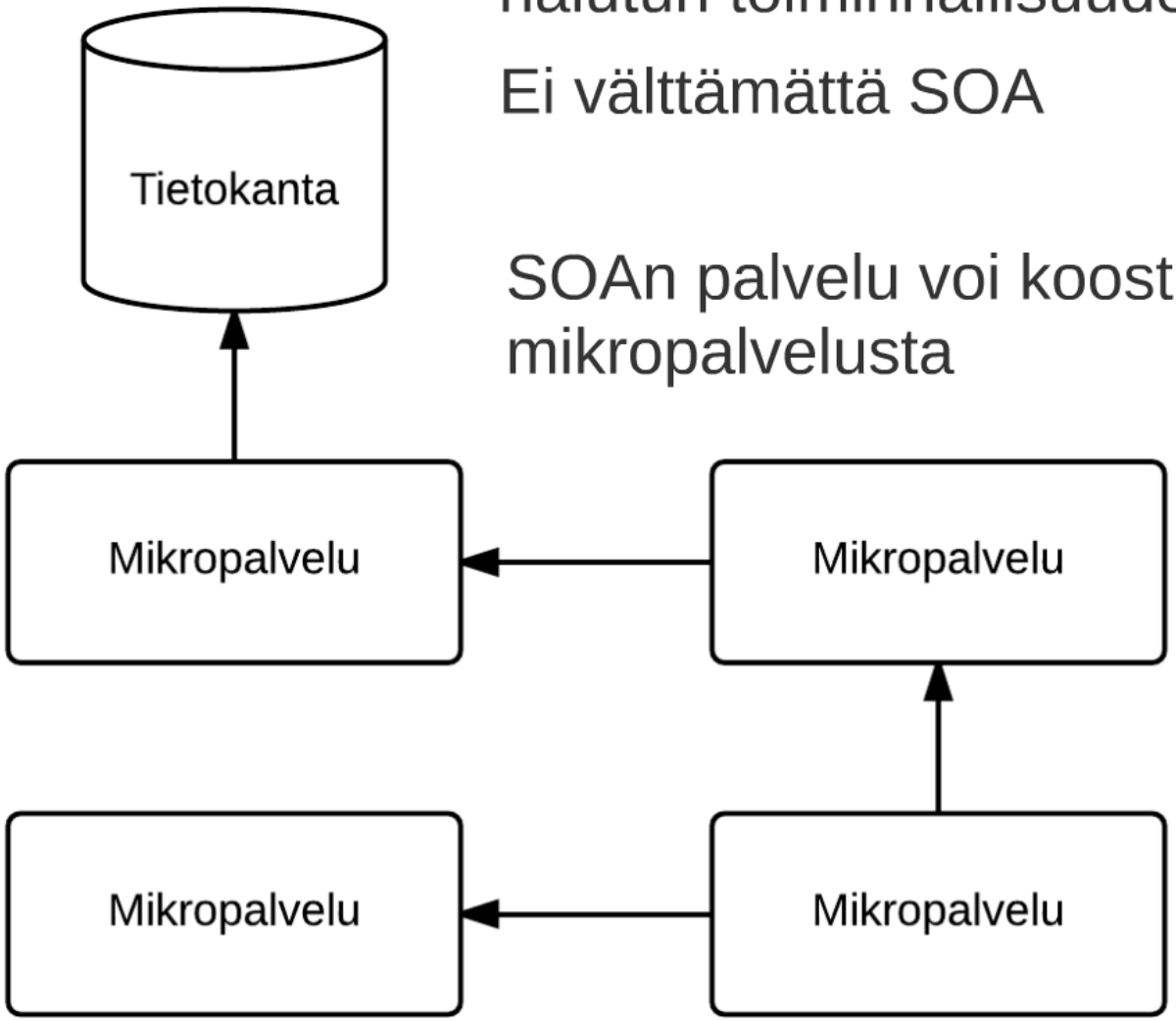


Mikropalvelut

Mikropalvelut ovat arkkitehtuurityyli, jossa pienet erilliset prosessit tuottavat halutun toiminnallisuuden.

Ei välttämättä SOA

SOAn palvelu voi koostua useasta mikropalvelusta



Yleisiä periaatteita

Tehdään pieniä, ymmärettäviä ja yksinkertaisia ohjelmia.

Pieniä asioita tulee paljon, eli toistuva työ pitää minimoida.

Viestinnän halutaan olevan keskitettyä

- Seuraaminen on helpompaa
- Jonotukset riippumatta lähettäjästä/vastaanottajasta
- Ei pisteestä pisteeseen säätämistä

Palvelimen malli

java -jar ohjelma.jar *parametreja*

Silmukassa:

- Odottele pyyntöä
- Käsittele pyyntö
- Palauta vastaus

https://bitbucket.org/jyukopla/it-paivat-2014

Tekniikoita

Yksinkertainen tapa luoda komentoriviohjelmia

- jyu-java-runner: parametrien parsiminen, ohjeet yms.
- Maven Shade plugin (Capsule?)

Kommunikointiin

- ZeroMQ

Muut ohjelmointikielet...

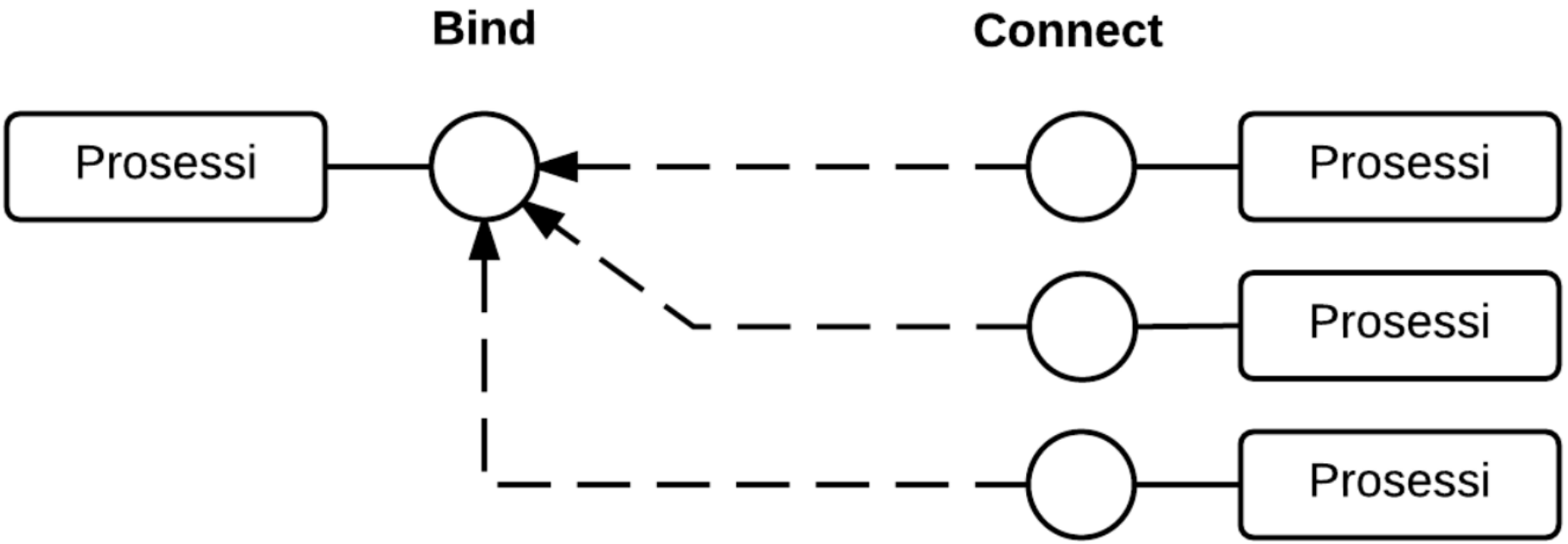
```
public Parameter[] programParameters() {
    return new Parameter[]{
            CreateParameter.named("address")
                    .beRequired()
                    .description("Where service will bind? For example tcp://*:2014").build()
    };
}

public void configure() throws ProgramConfigurationException {
    context = ZMQ.context(1);
    address = getParameter("address");

}

public void run() {
    try(ZMQ.Socket socket = context.socket(ZMQ.REP)) {
        socket.bind(address);
        while (true) {
            byte[] request = socket.recv();
            socket.send("Hello " + new String(request));
        }
    } finally{
        context.term();
    }
}
```

ZeroMQ

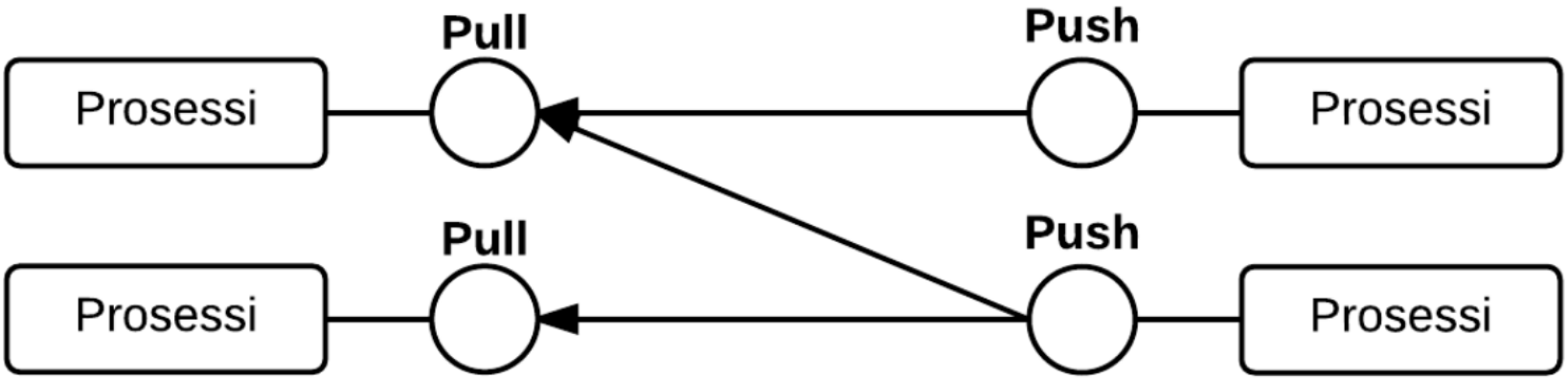
C++ kirjoitettu kirjasto viestien välittämiseen

Prosessin sisäinen ja ulkoinen kommunikaation

Yhteyden ottaja ja kuuntelija on oma abstraktionsa.
Palvelin voi ottaa yhteyden.

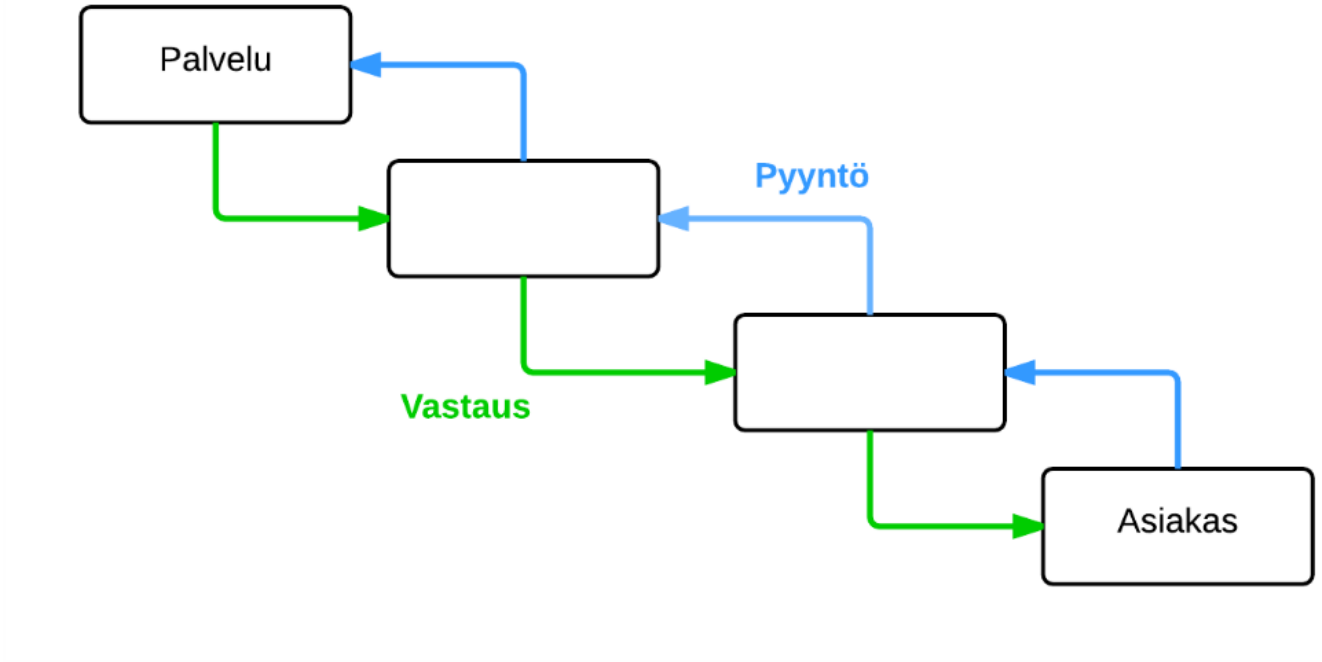


Yhden soketin takana voi olla lukuisia tosiasiallisia yhteyksiä.



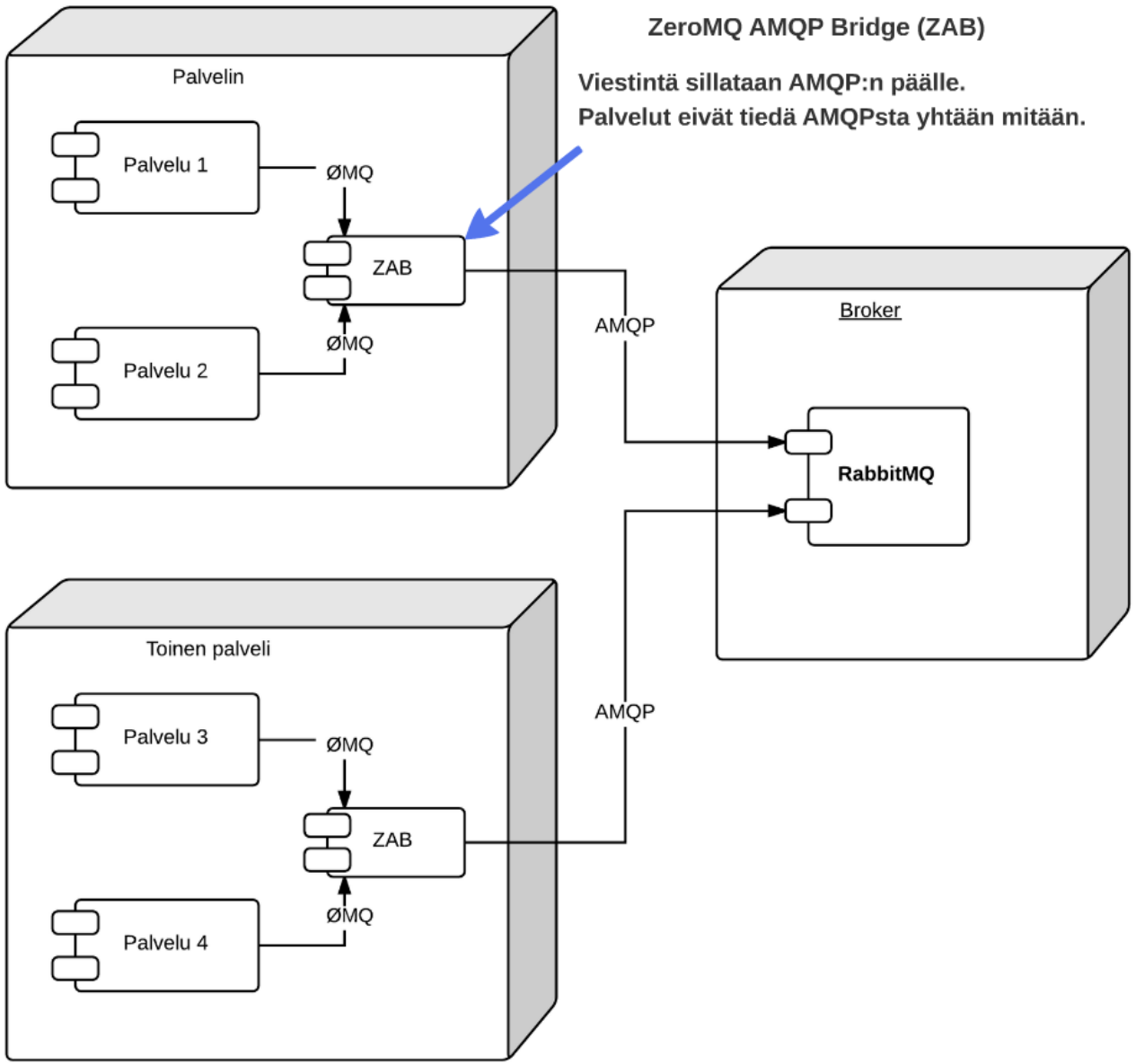
Käytetystä soketti tyypistä riippuu kuinka viestit käyttäyty.

REQ-REP viestinnässä viesti pitää sisällään paluusoitteen.



Mahdollistaa hajautetun viestinnän helpon toteuttamisen.

ZeroMQ on kirjasto viestinnän rakentelusarja....



```
java -jar service.jar bind:tcp://localhost:1234 jdbc:postgresql://kanta.kone.fi/kanta user salasana
```

Siltaavaan ohjelmaan määritellään:

```
connect:tcp://localhost:1234
amqp:user:pass@broker.jyu.fi:5672/vhost vaihde routingKey
```

Ohjelmat päälle Supervisord:llä tai Systemd:llä

- kaatuneet palvelut nostetaan uudelleen pystyyn
- AMQP asetuksia voidaan muuttaa koskematta palveluihin

Yksinkertaista, mutta räjähti käsiin...

Kuvitellaanpa palvelua, joka tarvitsee esimerkiksi 5 muuta palvelua toimiakseen.

```
java -jar service.jar bind:tcp://localhost:1234 config.conf

[services]
palvelu1=http://localhost:1230
palvelu2=http://localhost:1231
palvelu3=http://localhost:1232
palvelu4=http://localhost:1233
palvelu5=http://localhost:1234
```

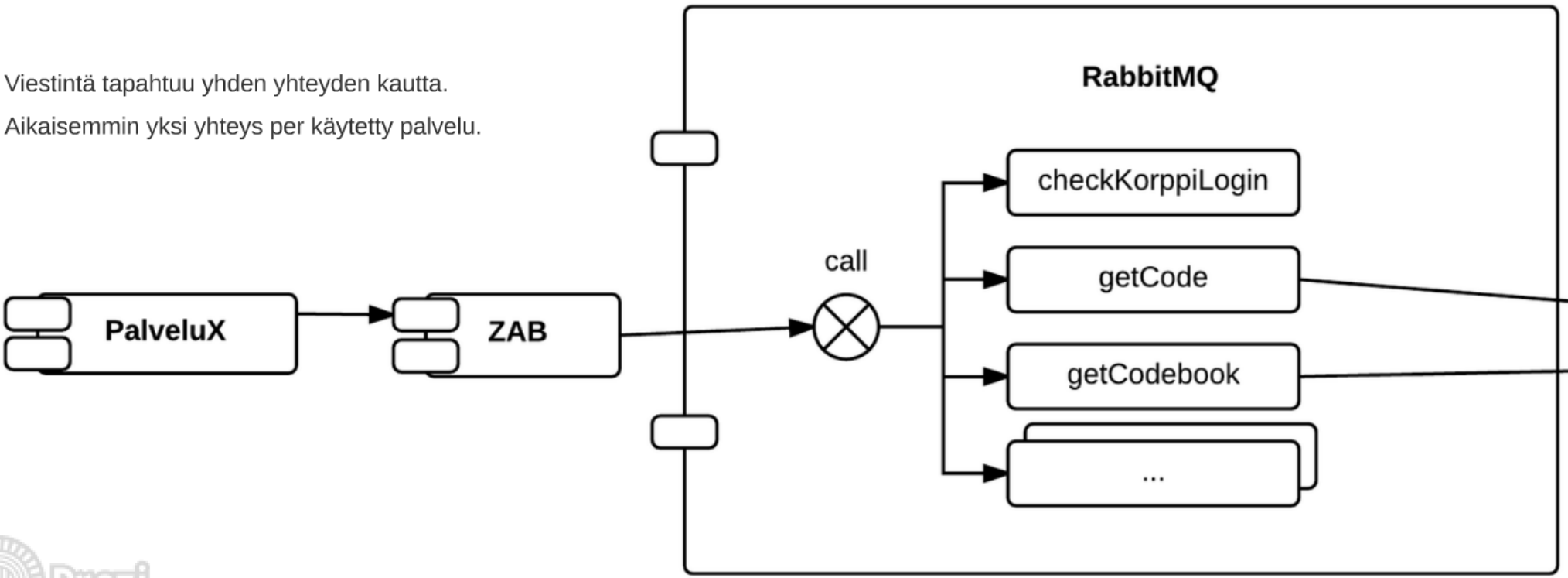
```
connect:tcp://localhost:1230
amqp:user:pass@broker.jyu.fi:5672/vhost vaihde palvelu1
connect:tcp://localhost:1231
amqp:user:pass@broker.jyu.fi:5672/vhost vaihde palvelu2
connect:tcp://localhost:1232
amqp:user:pass@broker.jyu.fi:5672/vhost vaihde palvelu3
connect:tcp://localhost:1233
amqp:user:pass@broker.jyu.fi:5672/vhost vaihde palvelu4
connect:tcp://localhost:1234
amqp:user:pass@broker.jyu.fi:5672/vhost vaihde palvelu5
```

Sopimalla käytänteistä vähennettiin konfiguraation tarvetta.

RabbitMQ + käytänteet muodostavat käytännössä ESB:n.

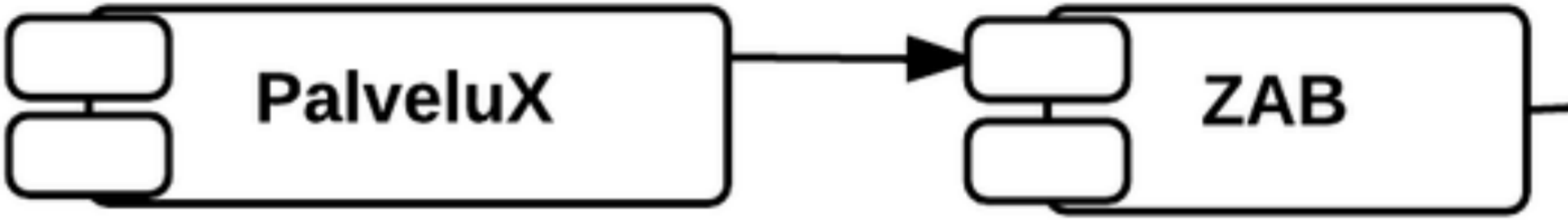
Viestintä tapahtuu yhden yhteyden kautta.

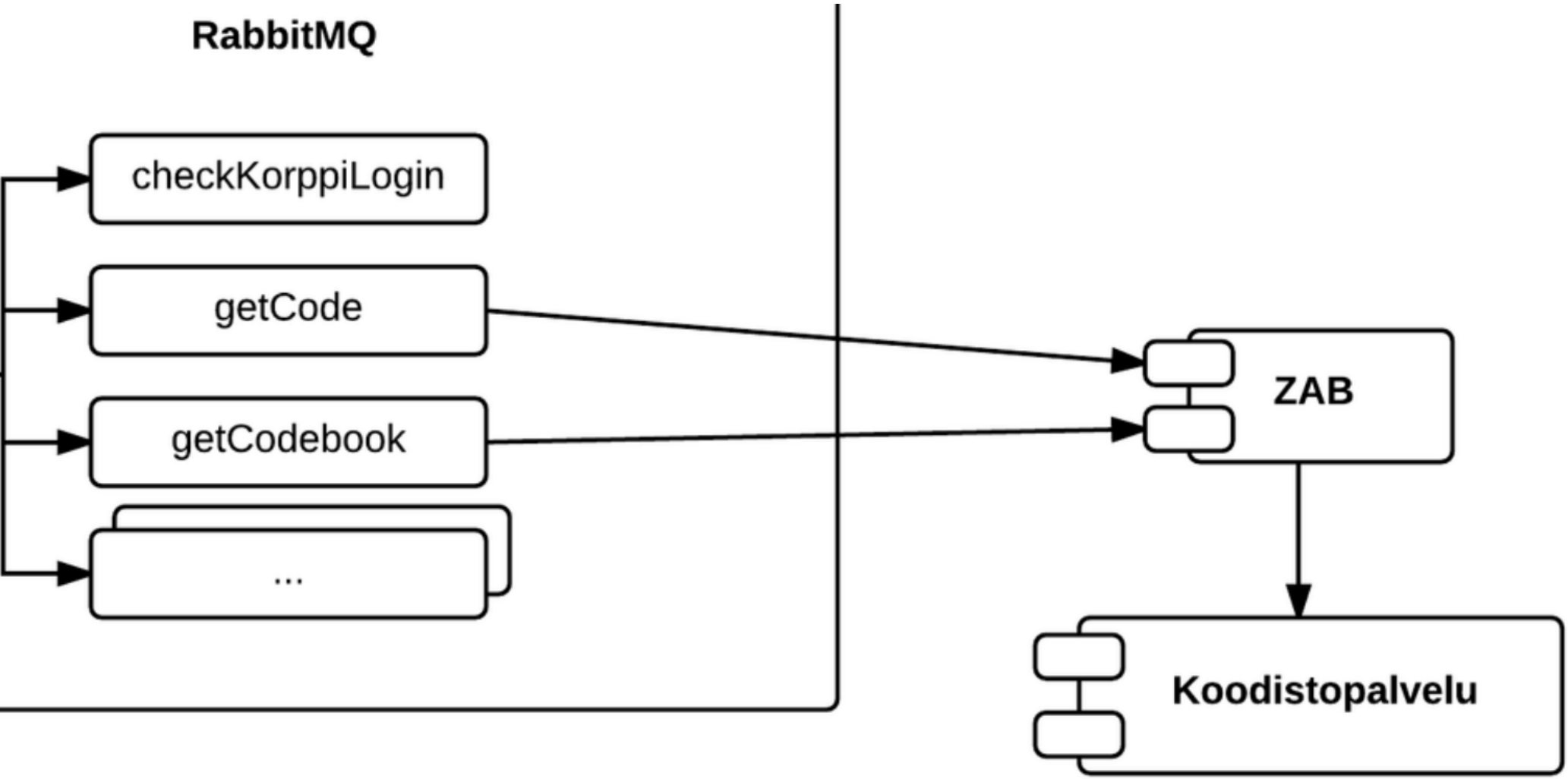
Aikaisemmin yksi yhteys per käytetty palvelu.



Viestintä tapahtuu yhden yhteyden kautta.

Aikaisemmin yksi yhteys per käytetty palvelu.





Palvelu ilmoittaa käynnistyessään viestiväylälle, mitä metodeja se tarjoaa.
Silta huolehtii oikeiden jonojen kuuntelusta.

Palvelu käsite osoittautui yllättäen turhaksi
Pelkkä nimi riittää erottamaan metodit toisistaan.

Case: Koodistopalvelu

Usein prosessien välillä halutaan siirtää pelkkiä viittauksia käsitteisiin, ei käsitteisiin liittyvää tietoa.

Suoritus
- arvosana
- tyyppi
- laitos
- ...

Case: Koodistopalvelu

Arvosana

Arvosana on "2". 1-3? 1-5? 1-100?

Arvosana on "H". Hyväksytty, Hyvä, millä asteikkolla?

Case: Koodistopalvelu

Ilmoitetaan asteikko samalla:

grade:1-3:2

grade:hyv-h-er:h

Kootaan tällaiset listat yhteen paikkaan, koodistopalveluun:

- Tietoja pitää voida hakea.
- Palvelussa pitää voida muokata olemassa olevia koodeja.
- Koodeja ja koodistoja pitää voida versioida.
- ....

Case: Koodistopalvelu

Palvelu "räjähti" tietokantaa käyttäväksi mammutiksi.

... ja ihan turhaan?

KISS!?

Case: Koodistopalvelu

Oikeasti riittää, että tiedot saa haettua yhdellä tavalla

Määritellään *koodi* niin, että sitä ei versioida. Jos tulee jotakin mitä tarvitsee versioida se ei ole enää koodi ja pistetään se jonnekin muualle.

Case: Koodistopalvelu

Tarvittavan toiminnallisuuden toteutti uudestaan yhdessä päivässä ja pisti tuotantoon.

https://source.kopla.jyu.fi/code/go/csv-code-service

Rumahan se on, mutta sen uudelleen kirjoittaminen ei vie kauaa...

Kokemuksia

Palveluiden päivittäminen on nopeaa.
Prosessikohtaiset vastuut helpottavat päivittämistä.
Muutoksiin reagoiminen on helpompaa.
Paikallinen yksinkertaisuus.
Kieli ja ympäristö riippumattomuus.

Valvonta on hankalaa
Tuotanto menee rikki yllättävillä tavoilla
Työkalujen ja tapojen kehittäminen vie paljon aikaa

Tulevaisuudessa

Paketoinnin kehittämistä ja automatisointia.

Tuotosten julkaiseminen.

Autorisaation ja autentikaation toteuttaminen.

Käyttöliittymien palasteleminen.