

Kohti palvelukeskeistä arkkitehtuuria

**Palvelukerros
Proof Of Concept
selvitys Metropoliassa**

RAPORTTI 1.3.2010

**Koostanut:
Tapio Ekholm**

Johdanto

Testauksen tehtävänä oli selvittää onko SOA-konsepti (Service Oriented Architecture) toimiva ja toteutettavissa oleva malli Metropolian tietojärjestelmien järjestelmäintegraatio-ongelmien ratkaisemiseksi.

Palvelukerros PoC testauksen (Proof Of Concept) konkreettisena kohteena oli selvittää voidaanko nykyiset vuosisuunnitteluun liittyvät palvelut uudistaa siten, että nykyiset "hämähäkki"-integraatiot poistetaan ja palvelut kytketään käyttämään palvelualustaa. Vaikka testaus kohdistui järjestelmien välisten integraatioiden toteutuksen, tulokset antavat viitteitä myös varsinaisen liiketoimintalogiikan toteuttamisesta palvelurajapintaa hyödyntämällä.

Testauskohteeksi valittiin Tuubin toteutuksen suunnitteluun liittyvän palvelun eli Totsu- palvelun uudistaminen, koska siihen liittyy nykyisessä muodossaan useita integraatioita eri järjestelmiin.

Testaukseen osallistuivat Tomi Blomqvist ja Simo Tiainen Metropolista sekä Eero Manninen toimittajan (Eduix) edustajana. Testauksessa tarvittavat skeemat ja niihin liittyvät impletoinnit teki Tuukka Puumala yhteistyössä Tapio Ekholmin kanssa.. Valtaosa raportin sisällöstä on syntynyt PoC prosessin aikana, josta olen sen costanut.

Alustavassa selvitystyössä kartoitettiin mahdollisia palvelualusta tuotteita ja teknologioita, joista seuraava taulukko on yhteenvedona:

Tuote	Lisenssi	Tuetut standardit	Referenssien määrä	Teknologia	Muuta
OpenESB	Common Development and Distribution License (CDDL) Version 1.0	Java Business Integration (JBI, SOAP, WS-*	Useita kymmeniä referenssejä listattuna tuotteen sivuilla	.Java	Netbeans sisältää hyvät graafiset työkalut integrointiin ja palveluiden toteuttamiseen. 43 JBI-komponenttia. Seuraava versio tullaan näillä näkymin rakentamaan OSGI-teknologian päälle (). Project Fuji
JBoss ESB	ESB GNU General Public License, v. 2.0	SOAP	JBossESB-sivujen mukaan : "in use by a large insurance company for 3 years"	Java	Perustuu Jbossin JMS-teknologiaan (JBoss messaging)
BizTalk	Kaupallinen.	SOA		Net	
Apache Synapse	Apache license Version 2.0	SOAP, WS-*		Java	Axis 2-tuotteen päälle rakennettu
MuleESB	Common Public Attribution License Version 1.0 (CPAL)	SOAP	Yksi käytetyimmistä Open source -ESB tuotteista	Java	Toimii yhdessä JBI-standardia tukevan tuotteen kanssa
ServiceMix	Apache license Version 2.0	Java Business Integration (JBI), hallintaan tuki JMX-teknologialle	Yksi käytetyimmistä Open source -ESB tuotteista. Käytössä apachen sovelluspalvelimessa Geronimossa.	Java	24 valmista JBI-omponenttia
WSO2 Enterprise Service Bus	Apache license Version 2.0	SOAP, WS-*		Java	Perustuu Apache Synapse -tuotteeseen

Testauksessa päätettiin nojautua JBI 1.0 standardia tukeviin tuotteisiin, koska niiden yhteensopivuus, odotettavissa oleva elinkaari ja soveltuvuus Metropolian teknologia-arkkitehtuuriin on todennäköisesti paras. Tällä perusteella testattaviksi palvelu/integraatio-tuotteiksi valikoituivat OpenESB ja ServiceMix.

Palvelualustalle asetettavia yleisiä vaatimuksia erilaisten skenaarioiden kautta

Termit

SOA, Service Oriented Architecture, palvelukeskeinen arkkitehtuuri

Palvelu, joukko yhteenliittyviä toimintoja, esim. opintosuunnitelman hakupalvelu.

, tietojärjestelmä joka liittää palvelut yhteen.

Palvelualusta, palvelun toteutus.

Komponentti, palvelun toteutus.

Järjestelmä, palvelualusta ja siihen kiinteästi liittyvät muut tietojärjestelmät

Skenaariot eri roolien näkökulmista

Palvelun kehittäjä

Palvelun kehittäjä tarkoittaa sitä tahoa, joka vastaa uuden palvelun kehittamisestä, hankkimisesta tai käyttöönotosta tai joka osallistuu varsinaiseen kehitystyöhön.

Tilanne	Vaatus	Ongelma jos vaatimus ei toteudu
Yksittäinen palvelu on ollut käytössä 6 vuotta ilman että palvelua on jouduttu kertaakaan muuttamaan. Palvelu tarjoaa rajapinnan ja tätä vasten on tehty 8 kriittistä integraatiota, joiden tulee olla jatkuvasti toiminnassa. Palveluun on tarkoitus tehdä uusia ominaisuuksia sekä laajentaa ja muuttaa palvelun ulkoista rajapintaa.	Palvelun uusi versio on voitava asentaa tuotantoon vanhan tilalle siten, että vanhaa rajapintaa vasten tehty 8 integraatiota säilyvät ehjinä asennuksen jälkeen ilman muutoksia integraatioihin ja mahdollinen päivityskatko on mahdollisimman lyhyt.	Palveluiden kehittäminen vaikeutuu merkittävästi. Yhden palvelun päivitys voi tuoda mukanaan huomattavan määrän piilokustannuksia.
Liiketoiminta muuttuu ja tätä varten tarvitaan uusi palvelu tukemaan uutta tarvetta. Palvelulle löytyy valmis komponentti, mutta se on toteutettu eri tekniikalla/eri kielellä kuin muut alustaan liittyvät palvelut	Palvelu on voitava liittää palvelualustaan vaikka palvelun toteuttava komponentti on tehty eri tekniikalla kuin millä olemassa olevat komponentit on tehty.	Palvelua varten täytyy kehittää uusi komponentti vaikka tarjolla olisi valmis koeteltu komponentti. Pahimmassa tapauksessa komponentit voidaan ostaa vain tietyltä toimittajalta.
Järjestelmään ollaan tekemässä uutta palvelua, joka toteutetaan koostamalla olemassa olevia palveluita. Palvelun toiminta tulee testata kunnolla muita palveluita vasten ennen sen tuotantoon asennusta.	Järjestelmästä voidaan luoda testiympäristö sekä olemassa olevia palveluita vastaavat testipalvelut. Testien aikana tietovirtaa palveluiden välillä voidaan tarkastella.	Tuotantoon asennetaan virheellisesti toimivia komponentteja
Palvelun kehittäjä on tekemässä uutta hakupalvelua, mutta ei tiedä että sellainen on jo olemassa.	Palvelualusta voidaan listata siihen liitetyt palvelut ja niiden toimintaperiaatteet.	Tehdään päällekkäisiä palveluita, jolloin ylläpidon määrä ja kustannukset kasvavat.
Palvelun kehittäjä tekee uutta palvelua ja huomaa että palvelussa voitaisiin hyödyntää olemassaolevia palveluita. Olemassaolevat palvelut eivät kuitenkaan tarjoa ulkoista rajapintaa.	Palvelualustaan on voitava tehdä palveluita siten, että muut palvelut voivat hyödyntää helposti palvelun tarjoamia rajapintoja. Vanhojen järjestelmien on pystyttävä kommunikoimaan palvelualustan kanssa tietokantayhteyden tai siirtotiedoston avulla.	Tehdään päällekkäisiä palveluita, jolloin ylläpidon määrä ja kustannukset kasvavat.
Palvelualustan kehitys ja tuki loppuu.	Palvelualusta voidaan vaihtaa toiseen	Uusi palvelualusta ja integraatiot

Palvelualustaan on liitetty lukuisa määrä palveluita ja integraatioita.	tuotteeseen kohtuullisella vaivalla ja ilman että palveluita joudutaan tekemään uudelleen.	joudutaan toteuttamaan alusta alkaen, jolloin kustannukset ovat erittäin suuret.
Palvelun kehittäjä on tekemässä integraatiota, joka johon liittyy 5 palvelua. Palveluiden rajapinnat eivät tue transaktioita. Palvelun kehittäjän tulee olla varma että häiriön sattuessa integraatio ei jää virheelliseen tilaan eikä tieto pääse vääristymään.	Palvelualustaan on voitava tehdä transaktionaalisia palveluita ja palvelualustan on tuettava hajautettuja transaktioita.	Monimutkaisia integraatioita ei voida tehdä tai ylläpito vaikeutuu merkittävästi. Integraation testaaminen on vaikeaa ja vauriolähtökohta kasvaa.
Palvelualusta ei toimi odotetulla tavalla.	Palvelualustaan liittyvän dokumentaation taso tulee olla hyvä sekä mahdollinen tuki helposti saatavilla. Palvelualustan toiminta voidaan tarkastaa alustan lähdekoodista ja tarvittaessa voidaan suorittaa pikakorjaus itse palvelualustan lähdekoodiin.	Palvelut joudutaan tekemään toisin kuin on suunniteltu. Palvelua ei voida toteuttaa. Pahimmassa tapauksessa koko palvelualusta joudutaan vaihtamaan.
On tehty uusi palvelu, joka on tarkoitus asentaa tuotantoon.	Uuden palvelun asennus ei saa aiheuttaa käyttökatkoja muissa alustaan liitetyissä palveluissa.	Ylläpito vaikeutuu ja hidastuu. Järjestelmäpäivitykset vaikeutuvat.
Komponenttien kehittämiseen tarkoitettu kehitystyökalu toimii huonosti.	Komponentti voidaan kehittää/toteuttaa usealla eri kehitystyökalulla. Komponentin rakenteen tulee olla niin yksinkertainen, että sitä voidaan tarkastella ilman erityistä kehitystyökalua. Komponentin onfiguraatiot ovat löydettävissä nopeasti.	Uusien palveluiden kehittäminen on hidasta ja kallista.

Palvelualustan ylläpitäjä

Palvelualustan ylläpitäjällä tarkoitetaan tahoa, joka vastaa palvelualustan toiminnasta tai osallistuu sen ylläpitotyöhön.

Tilanne	Vaatusmus	Ongelma jos vaatimus ei toteudu
Palvelualustassa on tapahtunut vakava häiriötilanne.	Ylläpitäjän tulee saada tieto tapahtuneesta häiriöstä.	Palvelun käyttökatkopitenee
Ylläpitäjä on saanut viesti vakavasta häiriötilanteesta.	Ylläpitäjä voi nähdä yhdestä paikasta järjestelmän kokonaistilan sekä järjestelmässä tapahtuneet häiriöt ja niiden aiheuttajat.	Ylläpito vaikeutuu ja aiheuttaa pitkiä käyttökatkoja.
Ylläpitäjä A on saanut ilmoituksen, että palvelu A ei toimi oikein. Ylläpitäjä B on saanut ilmoituksen, että palvelu B ei toimi oikein. Ilmoituksista ei kuitenkaan näe häiriön syytä. Palvelut A ja B on integroitu kolmanteen palveluun, joka aiheuttaa häiriön molemmissa järjestelmissä.	Ylläpitäjät näkevät helposti järjestelmien väliset riippuvuudet ja löytävät häiriönaiheuttajan nopeasti.	Ylläpitäjät tekevät turhaan työtä etsiessään häiriön syytä väärästä paikasta.
Palveluiden käyttö kasvaa rajusti.	Palvelualustan suorituskykyä voidaan lisätä klusteroinnin avulla	. Koko palvelualusta joudutaan vaihtamaan tai osasta palveluita luopumaan.
Palvelualustaan on kytketty uusi palvelu, jossa oleva vika aiheuttaa liikaa kuormaa palvelualustalle säännöllisin väliajoin.	Palvelualustan hallintatyökaluilla voidaan tarkastella palvelun tilaa ja siihen liittyviä tietovirtoja.	Ongelmien selvitys vaikeutuu. Kynnys asentaa uusia palveluita tai päivityksiä kasvaa
Palvelu joudutaan päivittämään. Palvelun rajapintoja käyttää kriittinen integraatio, joka lähettää palvelulle viestejä asynkronisesti. Integraation täytyy pystyä toimimaan huoltokatkon ajan.	Palvelun päivityskatkos ei saa aiheuttaa häiriötilannetta integraatioihin mikäli käytetään asynkronista tiedonsiirtoa.	Lisää ylläpitotyötä ja kynnys asentaa uusia palveluita tai päivityksiä kasvaa.
Järjestelmään tehdään tietoturva-auditointi tai epäillään että järjestelmään on yritetty tehdä tietoturvahyökkäys.	Palvelualustaan liitetyt palvelut ja integraatiot voidaan listata ja tarkastaa niiden näkyvyys sisäverkon ulkopuolelle. Palvelualustassa tapahtuvaa tietovirtaa voidaan tarkastella.	Riski hyökkäyksen onnistumisesta kasvaa, jos tietoturvan varmistaminen keskitetysti ei onnistu.

	Ylläpitä saa kokonaiskäsityksen järjestelmässä käytössä olevista tietoturvaan liittyvistä teknologioista.	
Palvelimelle on asennettu 3 palvelua, jotka on liitetty toisiinsa. Palvelimen osoite vaihtuu.	Palveluiden on pystyttävä olemaan yhteydessä toisiinsa vaikka palvelimen osoite vaihtuisi, ilman että palveluita joudutaan konfiguroimaan uudelleen.	Ylläpidon määrä kasvaa. Riski siihen, että jokin vanha palvelu hajoaa muutoksen yhteydessä on erittäin suuri.

Käytetty testiprosessi ja PoC vaiheet

Vaihe 1:

- skeeman luonti opintojaksototeutusta varten, päätetty toteuttaa MP-käsitemallin pohjalta
- OpenESB asennus
- nykyisen Toisun tietokannan ja HTTP-pyyntöön liittäminen OpenESB:hen
- toisu-portletin konvertointi käyttämään OpenESB:n tarjoamaa rajapintaa
- testaus, aiheutetaan poikkeus kytkemällä Winha pois päältä ja tarkastetaan integraation tilanne

Vaihe 2:

- tehdään tulevaa Toisu-rajapintaa kuvaava transaktionaalinen palvelurajapinta
- tehdään Winha- rajapintaa kuvaava transaktionaalinen palvelurajapinta
- liitetään palvelut OpenESB:hen
- (toive, olisi hyvä toteuttaa myös loki-palvelu)
- testaus, aiheutetaan poikkeus kytkemällä Winha pois päältä ja tarkastetaan integraation tilanne

Vaihe 3:

- muutetaan Toisun rajapintaa
- testaus, tarkastetaan miten palvelua käyttävät toiminnot selviävät muutoksesta
- vaihdetaan OpenESB ServiceMix- tuotteeseen
- testaus, tarkastetaan miten helposti palvelut on siirrettävissä toiseen alustaan
- Raportoidaan PoC- projektin tulos
- Tulosten esittely

Aikataulu:

- 23.11.-27.11. ajalla järjestetään tekninen projektipalaveri
- Joulukuu, Metropolian käsitemallin pohjalta skeeman implementaatio (Tuukka, Tapio)
- Joulukuu, Metropolia asentaa kehityspalvelimen joulukuun loppuun mennessä (Mikko S./JKR)
- Tammikuu, projektipalaveri
- Tammikuu, toisella viikolla OpenESB asennus (Eero M., Tomi B., Simo T.)
- Tammikuu, kolmannella viikolla PoC-toteutuksen aloitus
- Helmikuu, Metropolian pojekti-ihmisten "rajapintakoulutus"
- Viikko 5, PoC testaus valmis
- projektipalaveri, yhteenveto
- Viikot 7 ja 8 raportin valmistelu
- Viikko 9, Tulosten esittely johtoryhmälle

Metropolian käsittemallin pohjalta skeeman implementaatio

PoC-projektia varten määriteltiin yleiset skeemat, jonka avulla viestien rakenne standardoitiin palveluiden välillä. Skeemat sijaitsevat education-planning projektissa.

Käytettävät skeemat

Projektissa käytettävä tietomalli noudattaa mahdollisimman pitkälle Raketti-hankkeessa tehtyjä tietomallimäärittelyksiä. Määrittelytyö on parhaillaan käynnissä, joten täysin valmiita malleja ei ole vielä saatavilla. Lisäksi malli on määritelty vasta käsitetasolle, joten toteutimme projektissa käytettävän XML-skeeman. Skeeman toteutuksessa noudatetaan suositusta [JHS 170 Julkishallinnon XML-skeemat](#)

Toteutetut skeemat ovat:

- Tietotyytit - yleisiä käsitteitä kuten henkilö, rooli, kieliteksti (lokalisoitava teksti) ja laajuus
- Opinto - Opintojen tietomalli, sisältää tällä hetkellä ainoastaan Opintojakso-elementin
- Opintototeutus - Toteutusten tietomalli, sisältää tällä hetkellä ainoastaan Opintojaksototeutus-elementin

PoC pilotointiympäristö

PoC varten pystytettiin virtuaalikone poc.metropolia.fi, johon asennettiin seuraavat tuotteet:

- OpenEsb v.2.2
- NetBeans
- Tehdyt konfiguraatiot:
- Tallennettu MySQL-ajuripaketti sovelluspalvelimen lib/ext kansioon ja määriteltä tietokanta-asetukset sovelluspalvelimen admin-liittymässä.

Moduulit:

- ToisuEndpointImpl, toimii testirajapintana, joka tallentaa sanoman tiedot tietokantaan
- WinhaEndpointImpl, toimii transaktionaalisena rajapintana, joka aiheuttaa järjestelmäpoikkeuksen tietyn tyyppiselle viestille
- totsService, bpel-palvelu transaktionaalisen bpel-prosessin testaamiseen. Tarjoaa soap-rajapinnan, jonka avulla viedään tiedot kahdelle eri palvelulle.
- RealizationStorageApp, paketoit totsServicen asennuspaketiksi.
- RealizationService, bpel-palvelu, joka palauttaa toteutuksen tiedot tietyn skeeman mukaisena Toisun tietokannasta.
- RealizationSearchServiceCompApp, paketoit RealizationServicen asennuspaketiksi

Versionhallinta:

- trooper.metropolia.fi

PoC kuuluvien palveluiden toteutuksia:

Opiskelijarekisteripalvelu

Opiskelijarekisteripalvelulla tarkoitetaan Metropolian standardoidun toteutus-skeeman mukaisen tiedon siirtämistä konkreettiseen opiskelijahallintojärjestelmään. Oiskelijahallintojärjestelmänä Metropoliasa on Winha joka tarjoaa alunperin Toisua varten tehdyn Web Service -rajapinnan toteutusten siirtämiseksi Winhaan.

Kytkeytyminen Winhan rajapintaan

Winhan rajapinnan tekee erityisen ongelmalliseksi se, että se ei noudata täysin XML-standardia SOAP-viestin nimiavaruuksien määrittämisessä. Winha-rajapinta edellyttää, että nimiavaruus pitää olla määriteltynä SOAP- envelopen lisäksi myös itse lähetettävässä viestissä. Käytännössä kaikki standardia noudattavat työkalut optimoivat SOAP- envelopen generointia siten, että ylimääräiset nimiavaruuden määrittelyt poistetaan. Tästä syystä toimivan SOAP- envelopen generointi on tehtävä hyvin matalalla tasolla, jotta halutunlainen viesti voidaan muodostaa.

Rajapinnan epästandardina ominaisuutena voidaan nähdä myös se, että virhetilannetta kuvaava vastaussanoma palautetaan normaalina vastauksena, eikä SOAP- envelopen fault-elementin detail-osassa. Tämä ei ole varsinaisesti ongelma, mutta vastausviestinä käytetään sisältöä, mikä ajatuksellisesti kuuluisi eri paikkaan. Ongelmaksi tämä voisi muodostua siinä vaiheessa, kun ryhdytään toteuttamaan palveluja, mitkä myös palauttaa aidosti jotain sisältöä, koska tällöin joudutaan määrittelemään varsinainen vastaus ja virhesanoma samaan sanomaan.

Alla testatut liittymäkomponentit (binding component) ja niihin liittyvät ongelmat. Lopuksi esitetään toimiva ratkaisu kytkeytyä Winha-rajapintaan.

Liittymisestä Winha-rajapintaan.

SOAP-bindin component

Ongelmana SOAP- komponentissa on se, että lähtevä SOAP- envelope normalisoidaan ja nimiavaruus määrittely puuttuu halutulta tasolta. Koska lähetettävää dokumenttia välitetään käytännössä olio-tietorakenteena, ei algoritmiin, mikä muodostaa xml-rakenteen merkkijonoksi, voi mitenkään päästä käsiksi.

HTTP-binding component

HTTP- liittymäkomponentille (binding component) on mahdollista luoda SOAP- envelopelle halutunlainen merkkijonoesitys. Ongelmaksi muodostuu kuitenkin http-liittymäkomponentille välitettävän muuttujan tyyppi. Jos viestin lähettää liittymäkomponentille merkkijonona, määärtyy http-pyynnön sisältö tekstityyppiseksi, mitä Winha- rajapinta palvelin ei ota vastaan väärän tyyppisenä viestinä. Jos muuttujan määrittää XML:nä, niin viestissä olevat "<" ja ">" -merkit muutetaan entiteettiviittauksiksi "<" ja ">". Koska http-pyynnön tyyppiä ei voi muuttaa riippumatta http-komponentille välitettävän muuttujan tyylistä, on oikeanlaisen http-pyynnön luonti mahdotonta.

TCP/IP- binding component

Komponentti edellytti yllättäen sisällökseen xml-muotoista viestiä, jolloin tällä liittymäkomponentilla http-pyyntöjen mukaisen viestin generoiminen osoittautui mahdottomaksi.

Toimiva tapa kytkeytyä Winhan rajapintaan

Kytkeytyminen Winhan rajapintaan ei onnistunut edellä kerrotun mukaisesti millään testatulla liittymäkomponentilla. Ainoaksi tavaksi keksimme toteuttaa kytkeytymisen EJB- komponentilla, mikä julkaistiin palveluksi openESB:n käyttöön. Eli huonosti käyttäytyvä rajapinta piilotettiin ("wräpättiin") toimivan rajapinnan alle. Itse kytkeytyminen toteutettiin Apache Commons:in http-kirjaston avulla käyttäen suoraan http-protokollaa. Apachen http- rajapinat tarjosi mahdollisuuden määrätä http pyyntöjen sisällön tyyppi täysin sisällöstä riippumatta, mikä mahdollisti http-liittymäkomponentista poiketen toimivan ratkaisun toteuttamisen.

Tämä wräpätty rajapinnan toiminta muutettiin alkuperäisestä rajapinnasta siten, että se palauttaa virhesanoman SOAP- envelopen fault-elementin detail-osassa.

Opiskelijarekisteripalvelun logiikan toteutus

Opiskelijarekisterin logiikkaa (ts. Metropolian standardin skeeman muuntaminen Winhan- skeemaan) lähetettiin toteuttamaan kahdella eri openESB:stä löytyvällä Moottorilla (service engineillä): XSLT service engine ja BPEL-service engine.

XSLT-service engine component

XSLT-service engine:llä voidaan tehdä palvelu siten, että voidaan määrittää XSLT- muunnos lähtevän viestin ja paluuviestin väliin. Näin tehden XSLT- service engine toimii siltana kohdejärjestelmän ja kutsujan välissä, missä viestiliikenne käy mennessä tullen läpi XSLT- muunnoksen. Johtuen Winhan ongelmista tästä ratkaisusta luovuttiin, koska service engineen toimintaa on vaikea lokittaa. Tästä syystä siirryimme käyttämään BPEL-service engineä.

BPEL-service engine component

Opiskelijarekisteripalveluun liittyvä logiikka on toteutettu BPEL- moottorilla siten, että varsinainen skeemamuunnos tehdään XSLT- muunnoksella. Muunnos tehdään XSLT:llä sen takia, että BPEL:n tarjoamaa mäppäystyökaluja ei voitu käyttää wsdl:n puutteellisten skeema-määritysten takia. Ongelmana on se, että palvelun wsdl- kuvaus ei määrittele vastausviestin sisältämiä elementtejä mitenkään, joten niihin ei voi mitenkään BPEL- moottorissa viitata. XSLT- muunnoksessa voidaan viitata elementtiin, vaikka sitä ei skeemassa olisikaan määritelty.

Virheen kaappausta BPEL- enginessä ei saatu käytettävissä olevan ajan puitteissa toteutettua. Tavoitteena oli, että virhesanoman sisällä palautettu virhesanoma olisi saatu ohjattua esim. tiedostoon. Ongelmaksi jäi, että virheen palauttava muuttuja ei syystä tai toisesta alustunut catch- lohkoissa, vaikka catch.- lohko aktivoitui asianmukaisesti.

XSLT- muunnos ei huomio lähdeaineistosta kuin opinnon koodin ja toimipisteen. Nämä kaksi kenttää riittävät kuitenkin osoittamaan rajapinnan periaatteellisen toimivuuden. Muokkaamalla XSLT:tä pidemmälle, voidaan kyseisellä toteutuksella saada aikaiseksi täysin toimiva siirto, mikäli keskitetyn skeeman ja Winha- rajapinnan skeeman välillä ei ilmene ristiriitaisuuksia (Tätä en ole järjestelmällisesti varmistanut).

Asennusohje

Opiskelijarekisteripalvelu on riippuvainen kolmesta alla luetellusta PoC-repositoryn projektista.

Asennus tapahtuu ajamalla deploy komposiittiprojektille:

studentAdministrationAdministrationServerComposition.

winhaWrapper3: EJB- pavulla toteutettu Winha- rajapinnan "wräppäävä" palvelu.

studentAdministrationService: BPEL moottorilla toteutettu varsinainen opiskelijarekisteripalvelu

studentAdministrationAdministrationServerComposition: Kahden edellisen projektin sisältävä komposiitti-projekti, mikä asennetaan palvelimelle.

Demo

Syötettävä aineisto voidaan kirjoittaa tiedostoon:

studentAdministrationService\inputFilesDataa\inputRealization.xml ja ajaa bat- tiedostolla

studentAdministrationService\inputFiles\siirraInputRealization.bat Winhaan. (Bat-tiedosto tekee olennaisesti edellä mainitun tiedoston kopioinnin inputFiles- kansioon, mitä

studentAdministrationService pollaa.

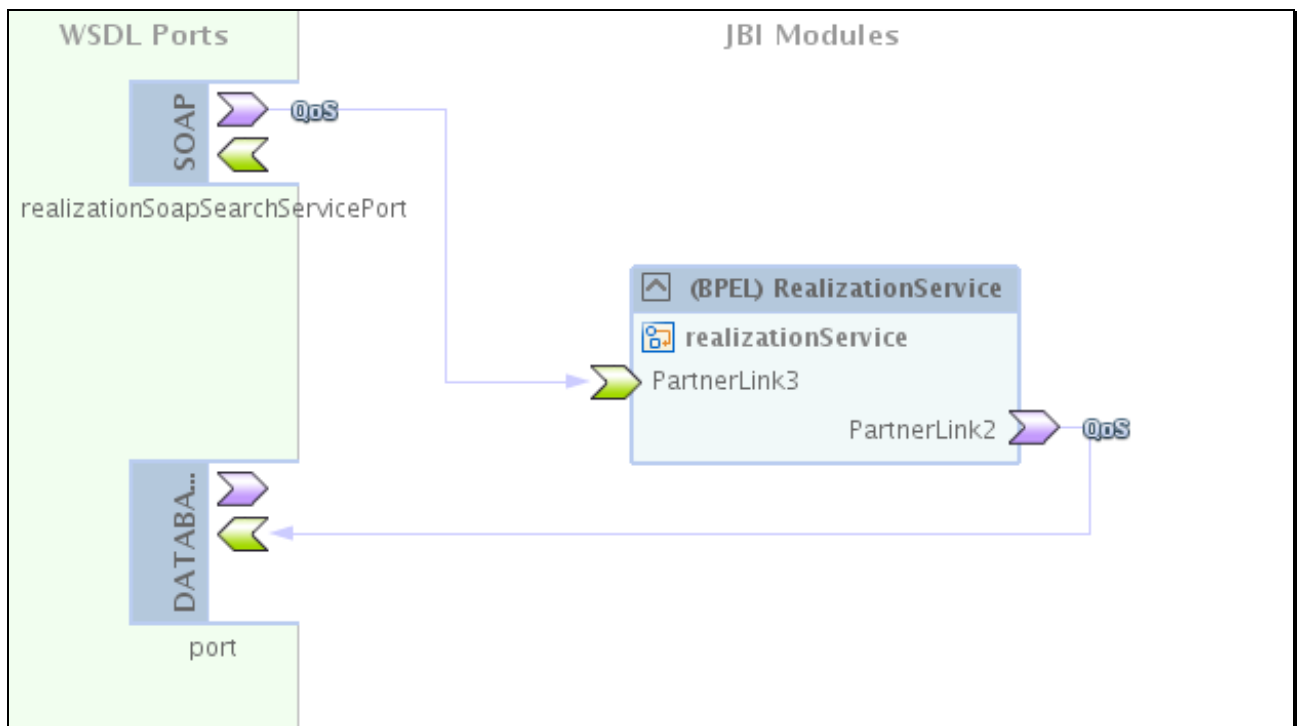
Palvelu Toisua varten

Moduulin tarkoituksena on testata miten Metropolian nykyisten järjestelmien tietokannasta voidaan hakea tietoa ja julkaista ne uudelleen käyttäen yleisiä skeemoja.

Moduulin komponentit

- BPEL-prosessi, joka on yhteydessä Toisun kantaan tietokantakomponentin(database binding component)avulla, muuntaa tietokannan tiedot yleisen skeeman mukaiseksi ja julkaisee SOAP-rajapinnan
- tostu-portlet, joka hakee toteutuksen tiedot SOAP- rajapinnalta ja näyttää ne käyttöliittymässä.

Tietokannan kytkeminen OpenESB:hen onnistui ilman ongelmia. Netbeans sisältää hyvät ja toimivat työkalut palvelun käyttöönottoon.



Transaktionaalisen palvelun testaus

Moduulin tarkoituksen testata OpenESB:n BPEL- moottorin kyky käsitellä transaktoita.

Moduulin komponentit:

- Transaktionaalinen EJB-tekniikalla toteutettu rajapinta, joka tallentaa viestin tietokantaan(kuvaa toisu-rajapintaa)
- Transaktionaalinen EJB-tekniikalla toteutettu rajapinta(kuvaa Winha-rajapintaa), jos viestissä tietty merkkijono, aiheuttaa Runtime poikkeuksen, jolloin transaktion tulee peruuntua.
- BPEL-prosessi, joka ottaa vastaan SOAP-viestin, aloittaa transaktion ja kutsuu järjestyksessä ensin toisu-rajapintaa ja tämän jälkeen winha-rajapintaa.
- Muutettu toisu-portlet lähettämään viesti OpenEsb:n julkaisemaan SOAP-palveluun

Testin kulku

Lähetetään toisu-portletista viesti SOAP-palveluun

Saadaan vastaus ja tarkastetaan tietokannan tilanne

Todetaan että tieto on tallentunut Toisu-palvelun tietokantaan

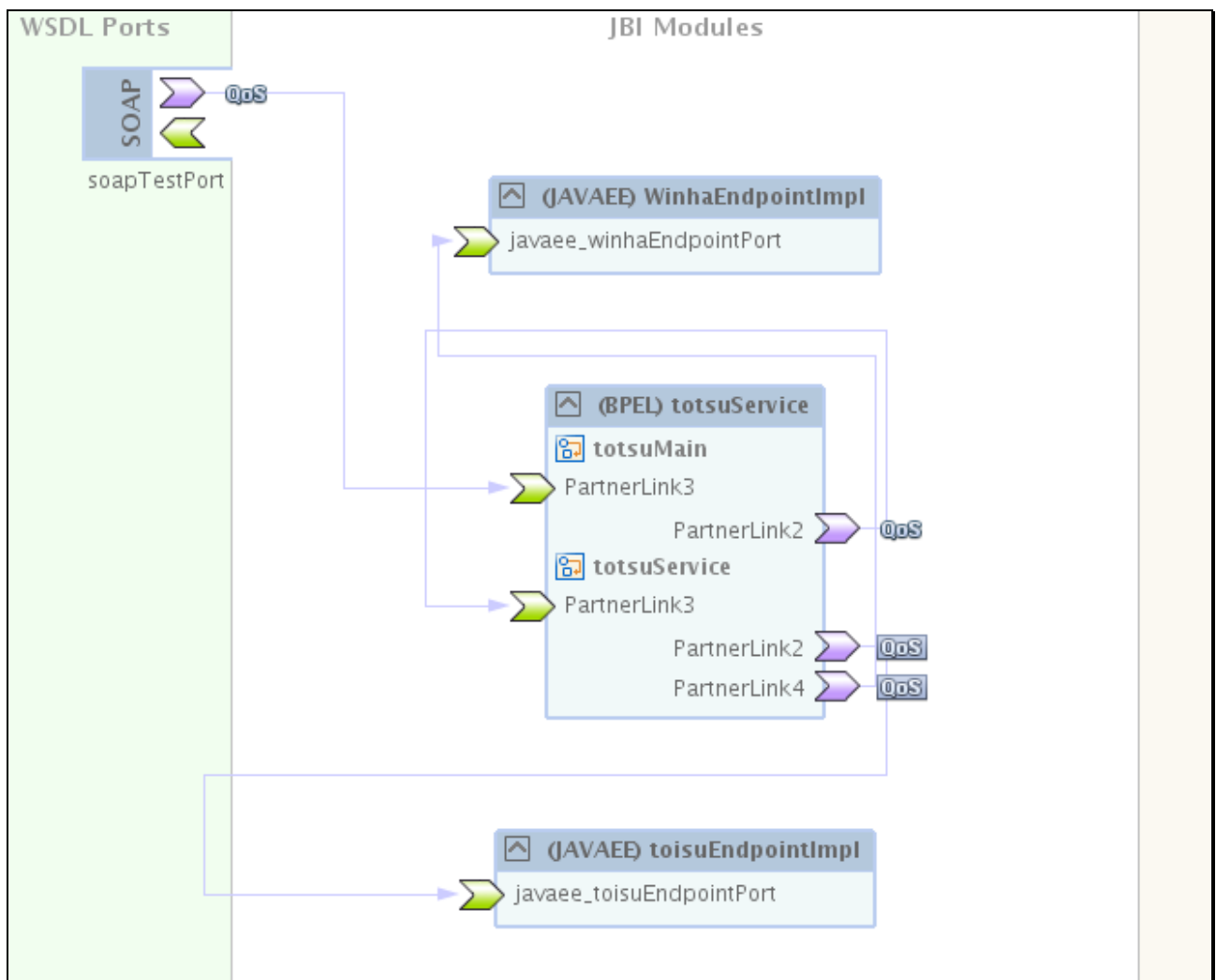
Lisätään lähetettävään viestiin merkkijono "throw_exception", jolloin Winha-palvelun pitäisi heittää Runtime-poikkeus

Tarkastetaan sovelluspalvelimen loki, jossa kerrotaan transaktion epäonnistuneen

Tarkastetaan että tietoa ei löydy Toisu-palvelun tietokannasta

Testin tulos

Transaktiotesti onnistui, mutta vaikka uudessa 2.2 versiossa voi konfiguroida tiedon(XAtransaction attribute), jonka mukaan transaktioita luodaan, täytyy sisään tulevan viestin olla joko liitettynä transaktioon tai BPEL -prosessi täytyy käynnistää toisen BPEL -prosessin sisältä. Lisäksi BPEL -moottori tulee olla konfiguroitu persistoituun tilaan sekä palvelurajapintojen täytyy olla transaktionaalisia. Transaktion peruuntumisen jälkeen lokiin tuli myös virheviestejä liittyen BPEL -moottorin tietokantayhteyteen. Kyseisessä versiossa liikaa ongelmia transaktioiden toimintaan saamiseksi, jotta sitä voisi suositella tuotantokäyttöön.



Yhteenveto

PoC- projektin aikana tehtiin integraatioita OpenESB tuotteella. Projektin aikana käytettiin useita protokollasidonnaisia komponentteja, joilla testattiin tuotteen integroitavuutta. Lähtökohta oli hakea vastauksia aiemmin esitettyihin skenaarioihin ja integraatio-ongelmiin. Projektin aikataulua häiritsi huonosti toteutettu Winha- rajapinta, jonka kytkeminen OpenESB:hen osoittautui odotettua suuremmaksi haasteeksi. Lisäksi uuden version työkalut havaittiin liian bugisiksi tuotantokäyttöä ajatellen ja projektin aikana toteutui yksi skenaarioissa mainituista uhkakuvista, tuotteen tulevaisuus on kyseenalainen OpenESB tuotteen tekijän SUNin myyminen Oraclelle. Toimittajasidonnaisuus huomioitiin alusta alkaen ja testattaviksi tuotteiksi valittiin JBI- standardia noudattavat tuotteet. Komponenttien siirrettävyyteen liittyy kuitenkin nykyisillä versioilla riskejä, kirjastoristiriidat, joten käytännössä standardi ei tällä hetkellä takaa toimivuutta.

Lukuun ottamatta OpenESB- tuotteeseen liittyviä ongelmia, SOA- konsepti todettiin toimivaksi ja on toteutettavissa. Todettiin että ESB- tuotteen ja yleisen tietorakenteita kuvaavan skeeman avulla päästään eroon järjestelmien välisistä riippuvuuksista.

Lyhyt yhteenveto tuloksista

- SOA- konsepti on toimiva ja toteutettavissa.
- palvelurajapinnan hyödyntäminen kannattaa aloittaa järjestelmien välisten integraatioiden toteuttamisesta
- pitkäkestoisiin prosesseihin liittyvä liiketoimintalogiikan toteuttaminen vaatii lisäselvitystä
- lopullisen toteutusalueen valinnassa kannattaa vielä odottaa, mutta tuotteen tulee kuitenkin tukea JBI- standardia
- vaihtoehtoisena ratkaisumallina voi olla myös SCA (Service Component Architecture) specifikaatiota tukevat tuotteet, joita PoC:ssa ei testattu

Ehdotus konkreettisista jatkotoimenpiteistä

- Logican toimittamien rajapintojen uusiminen standardin mukaisiksi
- Rajapinta käyttäjäkoulun käynnistäminen Metropoliasa projektipäälliköille, projektiin osallistuville sekä päättäjille
- Skeematyötä ja siihen liittyvää impletointia tulee jatkaa ja laajentaa (myös osallistumalla kansallisiin hankkeisiin)