Peppi - Koulutuksen suunnittelijan ja opettajan palvelut

Tekninen vaatimusmäärittely

Metropolia TAMK TAMPEREEN AMMATTIKORKEAKOULU edu IX

Versiohistoria

Versio	Päiväys	Tekijä	Selite
0.1	9.12.2010	Jaakko Rannila	Runko
0.2	13.12.2010	Projektiryhmä	1. päivän tuotos
0.3	14.12.2010	Projektiryhmä	2. päivän tuotos
0.4	1.2.2011	Jaakko Rannila	Stilisointi ja korjaukset
0.5	3.2.2011	Jaakko Rannila	Muotoilut
1.0	6.3.2011	Jaakko Rannila	Ohjausryhmälle lähetetty versio

Sisällysluettelo

1 Johdanto

Tässä dokumentissa kuvataan koulutuksen suunnittelijan ja opettajan palvelukokonaisuuden sovellusarkkitehtuuri ja tekniset vaatimukset.

2 Termit ja määritelmät

Termi	Selite
Palvelu	Joukko kiinteästi yhteenliittyviä toimintoja, esim. opintosuunnitelman hakupalvelu
Palvelualusta	Tietojärjestelmä joka liittää palvelut yhteen.
Järjestelmä	Palvelualusta ja siihen kiinteästi liittyvät muut tietojärjestelmät

Termi	Selite
ESB	Enterprise Service Bus, palveluväylä
JBI	Java Business integration, JSR 208 ja JSR 312-spesifikaatioissa kuvattu standardi helpottamaan palveluiden ja tietojärjestelmien integrointia
SE	JBI-spesifikaatiossa mainittu Service Engine -komponentti, joka voidaan liittää palveluväylään tietoa prosessoivaksi komponentiksi, esim.

	BPEL-komponentti, XSLT-komponentti
BC	JBI-spesifikaatiossa mainittu Binding Component -komponentti, jonka avulla palveluväylä liitetään tietyn protokollan avulla rajapintaan
BPEL	Business Process Execution Language, liiketoimintaprosessien kuvaamiseen tarkoitettu XML-pohjainen kieli
SCA	Service Component Architecture, palveluiden toteuttamiseen ja koostamiseen tarkoitettu suhteellisen nuori standardi
OSGi	Standardi modulaarista alustaa varten
WS	Web Service, verkossa oleva palvelun tarjoama rajapinta
SOAP	XML-pohjainen kieli ja protokolla
EJB	Enterprise Service Bean, transaktionaalinen komponenttipohjainen tekniikka palveluiden toteuttamiseen
JSON	Yksinkertainen tiedonsiirtomuoto
REST/RESTful rajapinta	Rajapinta joka hyödyntää HTTP protokollaa

3 Arkkitehtuuriperiaatteet

Tavoitteena on uudistaa korkeakoulujen sovellusarkkitehtuuria, joka on nykyisin muodostunut joukosta monoliittisia järjestelmiä. Monoliittisille järjestelmille on tyypillistä, että ohjelman toiminnallisuutta voidaan hyödyntää vain järjestelmän oman käyttöliittymän kautta. Jos monoliittisen järjestelmän tietoa hyödyntää muissa järjestelmissä, edellyttää se haastavia ja kalliita integraatio projekteja, mitkä aiheuttavat usein samojen tietojen päällekkäistä tallentamista useisiin järjestelmiin, mikä aiheuttaa helposti myös tiedon eheydellisiä ongelmia.

Uudistuksen ydintavoitteena teknologiamielessä on nimenomaan saada aikaiseksi sen tapainen sovellusarkkitehtuuri, jossa ei muodosteta monoliittisia järjestelmiä vaan atomisia palveluita, joita koostamalla saadaan aikaan sähköisiä palveluita. Tällöin palvelut voidaan kohdistaa paremmin oikeille käyttäjille ja tietojen integroimiseen on oma alusta, joka nimenomaan siihen tarkoitukseen kehitetty.

Arkkitehtuuriperiaatteena pidetään seuraavia periaatteita. Kehyksen johon palveluita rakennetaan ja integroidaan tulee olla avoimen lähdekoodin ratkaisu, jolloin tarvittaessa voidaan mennä lähdekooditasolle. Tämä siitä syystä, että tuotettavat palvelut ja integraatiot voivat osoittautua haastaviksi ja tällöin tärkeää, että pystytään tutkimaan kehyksen toimintalogiikkaa. Myös sen takia, että lähdekoodi antaa mahdollisuuden löytää ongelmien todelliset syyt. Tämän lisäksi kehykselle pitää löytyä riittävää tukea tarvittaessa ja kehyksen käytöstä pitää löytyä referenssejä. Lisäksi valittavan kehyksen tulee selviytyä Peppi projektissa toteutetuista Proof of Consept (POC) / Teknisen alustan koestusvaiheesta.

3.1 Yleiset periaatteet ja vaatimukset

Peppi toteutetaan palvelupohjaisena järjestelmänä, jonka sovellusintegraatiot ja businesslogiikkaa toteutetaan selkeästi erotettavassa palvelukerroksessa ja jonka käyttöliittymä logiikka rakennetaan tämän palvelukerroksen päälle. Tarkoitus on myös, että kaiken mitä voidaan tehdä käyttöliittymällä, voidaan tehdä suoraan palvelukerroksesta. Näin käyttöliittymä voidaan täysin vaihtaa menettämättä mitään alkuperäisen käyttöliittymän toiminnallisuutta. Tämä on hyödyllistä kun koostamme uusia käyttöliittymiä monesta eri palvelusta, missä Pepin palvelukerros on vain yksi osa palveluluista.

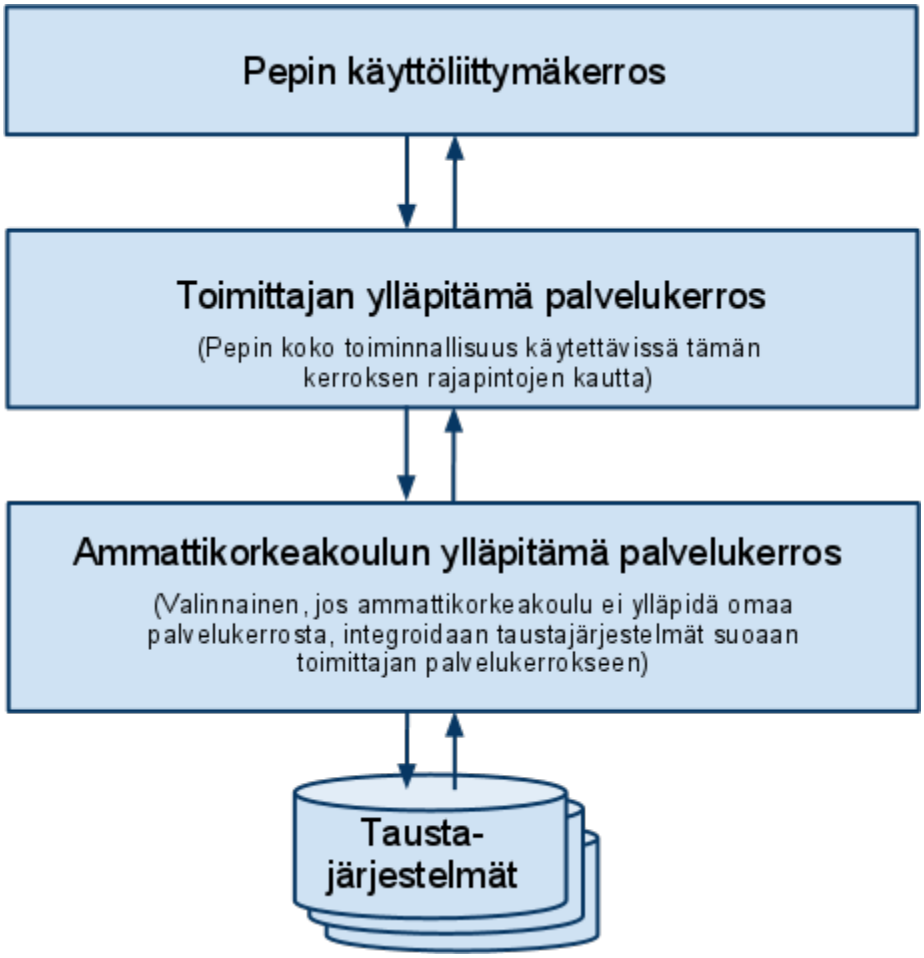
Palvelukerroksen palvelut tulee toteuttaa sillä tasolla, että käyttöliittymäkerroksessa ei tarvitse enää käyttää transaktioita. Näin saadaan aikaiseksi selkeä työnjako palvelukerroksen ja käyttöliittymän välille. Käyttöliittymä voidaan rakentaa yksinkertaisemmaksi ja sellaisten rajapintojen päälle, mitkä eivät tue transaktionaalisuutta.

Palvelukerroksen tulee tarjota mahdollisuus asynkroniseen viestinvälitykseen siten, että sanoma välitetään määränpäähän vähintään kerran.

Palvelut tulee tehdä tilattomiksi. Tilan tallennukseen liittyvät asiat hoidetaan käyttöliittymässä, workflowssa tai prosessi yms. moottorissa. Näin palvelut saadaan yksinkertaisemmiksi, vikasietoisemmiksi ja skaalatutuvammiksi.

Ammattikorkeakoulun kannattaa tarjota rajapinnat taustajärjestelmiin palveluina, jotta riippuvuudet taustajärjestelmän rajapinnan yksityiskohdista jää ammattikorkeakoulun oman sovelluskehityksen vastuulle vähentäen toimittajariippuvuutta. Ammattikorkeakoulu voi toteuttaa palvelut esim. oman palvelukerroksen avulla, jolloin ammattikorkeakoulun palvelukerroskokonaisuus jakautuu kahteen osaan: Siihen mikä on toimittajan ylläpitämä ja siihen mikä on oman ylläpidon piirissä. Valitun tai vallittujen palvelukerroksien olisi tästä syystä tarkoituksenmukaista pystyä integroitumaan löyhästi yhteen esim. jollain viestinvälitykseen liittyvällä protokollalla.

Järjestelmä tulee suunnitella ja toteuttaa siten että se on hyvin skaalautuva ja klusteroitavissa. Tämä tulee ottaa eritoten huomioon tietokantojen vaatimuksissa klusteroinnin suhteen. Näin varmistetaan ettei kuorman kasvaessa, klusteroinnin vaatima työmäärä kasva ylivoimaiseksi.



3.2 Noudatettavat standardit

Toteutettavan teknisen viitekehyksen pitää tukea yleisesti hyväksyttyjä standardeja joihin voidaan tukeutua. (erityisesti: SOAP, WSDL, JAX-WS, Rest. Eduksi katsotaan, jos tukee yleisiä standardeja kuten JBI, OSGi, SCA)

3.3 Autorisointi ja auktorisointi

Koska molemmilla ammattikorkeakouluilla on omat taustajärjestelmät identiteetin hallintaan (tunnukset, ryhmät, roolit, organisaatiorakenne), autorisointiin ja auktorisointiin tulee tehdä oma abstrakti palvelukokonaisuus. Tämän palvelukokonaisuuden rajapinnoista tehdään tarvittaessa omat toteutukset molemmille ammattikorkeakouluille. Mikään muu toteutettavan järjestelmän palvelu ei saa sisältää suoraa riippuvuutta ammattikorkeakoulun identiteetin hallinnan taustajärjestelmiin.

Autorisointi täytyy tehdä jo palvelutasolla. Sitä ei voi jättää pelkästään näkymäkerrokselle. Tämä keventää näkymäkerroksen taakkaa sekä vähentää autorisoinnin moninkertaista toteuttamista.

Tarvittavia palveluita ovat ainakin:

- Identiteettipalvelu
- ryhmäpalvelu
- Roolipalvelu
- oikeuspalvelu

3.4 Rajapinnat ja liittymät muihin järjestelmiin

Yleisinä periaatteille Integroitaville taustajärjestelmille asetamme, että niiden on tuettava

1) Ohjelmallisia rajapintoja kuten esim. soap, rest, jms, activeMQ ynnä muita vastaavanlaisia rajapintoja. Esimerkkeinä rajapinnoista mitä emme tämän nojalla tue on:

a) Tekstitiedostoihin perustuvia rajapintoja, koska niillä ei voi muodostaa pyyntö/vastaus-tyyppisiä palveluita eikä toteuttaa virheenkäsittelyä.

b) Suoraan tietokantatauluihin liittyviä integraatiota, koska ne luovat riippuvuuksia fyysisiin tauluihin. Integraatio on kuitenkin, mikäli muita vaihtoehtoja ei ole, mahdollista tehdä tietokantanäkymien avulla, mikäli rajapinta ei edellytä kantaan kirjoittamista. Jos rajapinta edellyttää tietokantaan kirjoittamista, täytyy rajapinnan toteuttamiseen käyttää tietokantaproseduuria.

2) Hallittua virheenkäsittelyä, mikä edellyttää:

a) pyytävä osapuoli saa tiedon virheestä siten, että syy voidaan jäljittää tiettyyn tapahtumaan ja/tai korruptoituneeseen tietoon.

b) Virheviestin käsitteleminen ei vaadi "parsimista" esim. merkkijonon monimutkaista tulkitsemista.

c) Virheviesti ja kuvaus erotetaan selkeästi siten, että virheviesti on aina sama

liittyen samanlaiseen virheeseen, mutta selitteen teksti voi vaihdella.

3.4.1 Henkilötiedot

Järjestelmän tulee pystyä hakea tiedot ulkopuolisista järjestelmistä. Henkilötietojen lähdejärjestelmänä käytetään organisaatioiden IAM- tai IDM-järjestelmiä.

Metropoliassa IAM-järjestelmänä on Amme. Ammeen ja Pepin integraatiossa hyödynnetään Tuubi2 projektissa syntynyttä skeemaa, joka tehtiin Tuubin ja Ammeen integraatiota varten.

Tamkin osalta järjestelmä on muutoksen alla, joten vielä ei voida määritellä tapahtuuko haku yhdestä (IDM tai vastaava) tai useammasta järjestelmästä (LDAP ja OHA yms.).

Henkilöihin liittyvää tietoa voidaan laajentaa oman profiilin avulla, joka tallentuu Peppi kokonaisuuteen (ks. toiminnallinen määrittely 6.1.3).

3.4.2 Organisaatiotieto

Järjestelmäkokonaisuuden tulee tietää mitä ovat organisaation yksiköt ja ketkä ovat yksiköiden päälliköitä/esimiehiä. Kokonaisuuden pitää myös tietää henkilön asema organisaatiorakenteessa.

Organisaatiorakenne Peppi kokonaisuuden käyttöön tulee ensisijaisesti pystyä hakemaan taustajärjestelmistä, jos tämän tyyppisiä taustajärjestelmiä on organisaatioilla olemassa. Jos tätä tietoa ei ole taustajärjestelmissä, tulee organisaatiorakenne tieto tallentaa johonkin, esimerkiksi Peppi kokonaisuuden alle. Tätä tietoa voi tällöin hyödyntää myös muut palvelut, jotka tarvitsevat yksikkörakenteen ja esimies-alais rakenteen.

3.4.3 Kustannustunnisteet (kustannuspaikkahierarkia, toiminnot, projektinumerot)

Peppi kokonaisuudessa käytettäviä yleisiä tunniste ja nimistö tietoja, kuten kustannuspaikkahierarkia, toiminnot ja projektinumerot, haetaan tietojen lähdejärjestelmistä käyttämällä Peppi kokonaisuuden palveluväylää.

Haettavaa tietoa tulisi käsitellä siten, että Peppi kokonaisuudella on reaaliaikaisesti tunnistetiedot käytettävissä. Tunnistetiedoilla tulee olla voimassaoloaika, joilla voidaan ohjata tunnisteiden käyttöä Peppi kokonaisuudessa.

3.5 Kokonaiskuva

3.5.1 Palvelut

Peppi-projektissa toteutettavat palvelut tarjoavat rajapinnat muiden palveluiden sekä käyttöliittymien käytettäväksi. Rajapinnat suojataan ja versioidaan siten, että moduulin päivitys ei riko rajapintaan kytkeytyviä käyttöliittymiä tai integraatioita. Palvelut voivat käyttää tallennusratkaisuna tietokantaa tai jotain muuta tallennusratkaisua, esimerkiksi materiaalin säilömiseen voidaan käyttää Alfresco-järjestelmää. Tallennusratkaisu on moduulin sisäinen asia eikä näy palvelun ulkopuolelle.

3.5.2 Integraatiot ulkoisiin järjestelmiin

Palvelut keskustelevat ulkopuolisten järjestelmien kanssa alustan tarjoamien komponenttien avulla. Ulkopuolisia järjestelmiä ovat esimerkiksi Winha ja HR-järjestelmä.

3.5.3 Käyttöliittymät

Peppi-projektissa toteutetaan opettajan ja suunnittelijan työpöytä. Näkymät hakevat tietoa palvelukerrokselta julkaistujen rajapintojen avulla. Näkymät suodattavat toimintoja käyttäjän roolien perusteella. Peppi-palveluiden julkisiin rajapintoihin on mahdollista kytkeä muita asiakasohjelmia, esimerkiksi mobiili käyttöliittymä herätenäkymään.