hyväksymispäivä arvosana

arvostelija

Palveluiden valinta

Pasi Tuominen

Helsinki 02.03.2009

HELSINGIN YLIOPISTO
Tietojenkäsittelytieteen laitos

HELSINGIN YLIOPISTO □ HELSINGFORS UNIVERSITET – UNIVERSITY OF HELSINKI□

Tiedekunta □ Fakultet – Faculty	Laitos □ Institution □ Department
Matemaattis-luonnontieteellinen tiedekunta	Tietojenkäsittelytieteen laitos

Tekijä□□□Författare □ Author

Pasi Tuominen

Työn nimi□□ Arbetets titel □ Title

Palveluiden valinta

Oppiaine □ Läroämne □ Subject

Tietojenkäsittelytiede

Työn laji□□ Arbetets art □ Level	Aika□□ Datum □ Month and year	Sivumäärä□□ Sidoantal □ Number of pages
Seminaari	2.3.2009	8 sivua

Tiivistelmä□□ Referat □ Abstract

Tässä seminaarityössä esitellään web-palveluiden etsimiseen ja valitsemiseen liittyviä ongelmia.

Avainsanat – Nyckelord □ Keywords

Palvelut, valinta, UDDI, SOA

Säilytyspaikka □ Förvaringställe □ Where deposited

Muita tietoja □ Övriga uppgifter □ Additional information

Sisältö

1 Johdanto

Palvelusuuntautuneessa arkkitehtuurissa on kolme osapuolta; palvelun tuottaja, palvelun käyttäjä ja palveluiden välittäjä. Palveluiden välittäjän tehtävänä on parantaa ohjelmistojen uudelleenkäytettävyyttä: uutta ohjelmistoa, voidaan kysyä palveluiden välittäjältä onko tiettyjä tarvittuja toiminnallisuuksia jo olemassa. Mikäli toivottuja palveluita löytyy, voidaan niitä käyttää ja siten ohjelmiston toteuttamisen vaatima työmäärä vähenee.

Perusongelmana on, että kun palveluita on tuotettu paljon, niiden seasta alkaa olla vaikea löytää juuri sitä tai niitä palveluita joita tarvitsisi toiminnan toteuttamiseksi. Mikäli tarpeisiin sopivaa palvelua ei löydy, niin joudutaan kehittämään palvelun toteuttava komponentti, mahdollisesti vain sen takia, että jo toteutettua palvelua ei löydetty. Tämän vuoksi palveluiden löytäminen on ensiarvoisen tärkeää.

Tässä paperissa käsitellään yleisesti palveluiden hakemismenetelmien vaatimuksia, mm. infrastruktuuri ja prosessimielessä. Lisäksi esitellään eniten käytetyssä palveluhakemistoimplementaatiossa, UDDI:ssa, olevaa palveluiden hakumenetelmää ja tehdään katsaus muutamaan palveluiden löytämistä parantavaan tutkimukseen.

2 Palveluiden etsintä ja valinta

Palveluiden etsintä ja valinta on prosessi, jossa pyritään löytämään jokin tietty palvelu tai sen tarjoaja kolmannen osapuolen, palveluiden välittäjän, avulla. Yleisesti ottaen prosessi sisältää seuraavat vaiheet. Palvelun toteuttaja julkaisee palvelunsa välittäjälle, joka tallentaa palvelun tiedot. Tämän jälkeen palvelun etsijä voi kysyä välittäjältä tietääkö se jotain palvelua, jota etsitään. Välittäjän täytyy sitten päätellä etsijän kysymyksestä millaisia palveluita etsijä kaipaa, ja palauttaa joukko palveluita jotka vastaavat parhaiten etsittyä palvelua.

Lisäksi on tarvetta dynaamisille hakurakenteille, jotka ovat aina ajan tasalla tarjoten mahdollisimman laajan ja tehokkaan palveluiden valintamahdollisuuden. Hakumekanismien pitäisi tarjota erilaisia hakumahdollisuuksia sekä toteutusvaiheelle että suoritusaikana. Toteutusvaiheessa kehittäjä voi selata palveluita selvittääkseen millaisia palveluita on olemassa, kun taas ohjelma suoritusaikanaan voi olla

kiinnostunut kaikista palveluista, jotka toteuttavat jonkin tietyn rajapinnan.

Palveluiden haun täytyy olla myös riittävän nopeaa. Jossain tilanteissa liian hitaasti löytynyt palvelu on sama asia kuin se, ettei palvelua olisi löytynyt laisinkaan.

Suurin ongelma palveluiden valintamekanismien toteuttamisessa on se, että palvelut ovat erittäin monimuotoisia. Toistaiseksi ei ole olemassa oikein mitään menetelmää, jolla tietokoneet voisivat ymmärtää, mitä palvelu tekee.

Palveluiden etsimismekanismeja on useanlaisia: rekistereitä, indeksejä, katalogeja sekä myös vertaisverkkoa hyödyntäviä ratkaisuja.

3 UDDI - Palveluhakemisto nyt

UDDI (Universal Description, Discovery and Integration) on OASIS järjestön ylläpitämä palveluhakemistoimplementaatio. UDDI on ehtinyt edetä jo kolmanteen versioonsa vuodesta 2000. UDDI on keskitetty rekisteri, jossa palvelut on kategorisoitu tiettyihin kategorioihin. UDDI:n suosio johtuu siitä, että se on ollut jo pitkään olemassa. Lisäksi sitä ennen ei ole ollut oikein mitään tapaa koota palveluita yhteen. [GPS04].

UDDI:ssa on SOAP rajapinnat, joiden avulla voidaan tehdä hakuja ja julkaista tietoa rekisterissä. Rajapinnan kautta pääsee selaamaan UDDI:iin rekisteröityjen palveluiden WSDL:llä määriteltyjä rajapintoja.

3.1 Palvelun etsiminen UDDI:ssa

UDDI:ssa palvelut ovat järjestettyinä erilaisiin kategorioihin ja niitä etsitään hakusanoilla. Hakuja voidaan rajata monella tavalla, esimerkiksi jos on kiinnostunut vain tietyn yrityksen palveluista, voi hakea tämän yrityksen nimellä. On myös mahdollista etsiä vain tiettyyn kategoriaan kuuluvia palveluita, kuten tietyn alan tai alueen palveluita.

Kategoriat aiheuttavat kuitenkin erään ongelman: palvelun tuottajan pitää päättää mihin kategorioihin oma palvelu kuuluu. Vastaavasti palvelua etsivän käyttäjän pitää keksiä mihin kategoriaan palvelun tuottaja on ajatellut palvelun kuuluvan löytääkseen sen.

UDDI:ssa ei myöskään ole mitään mekanismia, jolla voisi löytää keskenään samankaltaiset palvelut, vaan käyttäjä luultavasti tyytyy ensimmäiseen löytämään

palveluun, joka ei välttämättä ole paras mahdollinen. Esimerkiksi jos käyttäjä etsii palvelun nimen mukaan ”CreateOrder” palvelua, niin hän voi olla myös kiinnostunut ”OrderGeneration” palvelusta. Etenkin jos jälkimmäinen on edullisempi tai luotettavampi.

Tämän vuoksi palveluita löytyy joko liian paljon, liian vähän tai ne eivät vastaa haluttuja toimintoja, joten etsiminen vie paljon aikaa ja on turhauttavaa [HZ07].

Lisäksi WSDL:llä ei määritellä palveluiden semantiikkaa, joten palveluiden validointiin tarvitaan käyttäjä. UDDI:ssa on kuitenkin mahdollista liittää monta palvelua samaan WSDL dokumenttiin. Esimerkiksi lentoyhtiöillä voi olla toteutettuna saman rajapinnan kautta toimiva lippujen hinnan kysely, tällöin on helppo toteuttaa ohjelmisto joka kysyy kaikkien mukana olevien lentoyhtiöiden lippujen hinnat. Toisenlainen tilanne voisi olla sellainen, jossa useampi palveluntarjoaja tarjoaa samaa palvelua saman rajapinnan kautta. Tällöin tietokone pystyy automaattisesti käyttämään kunakin hetkenä sellaisen tarjoajan palveluita, jotka ovat toiminnassa.

4 Manuaalinen ja automaattinen valinta

Palveluiden hakuun ja valintaan pitäisi olla mahdollisuus sekä toteutusaikana, että ohjelman suoritusaikana. Erityisen ongelmallista on, miten järjestelmä voisi suoritusaikanaan etsiä palveluita, arvioida niiden sopivuutta ja valita sopivimman palvelun tilanteen vaatimaan tehtävään. Epäilemättä palvelun validoinnin ja luotettavuuden määrittely ei onnistu automaattisesti. Lisäksi on syytä kyseenalaistaa tällaista toiminnallisuutta käyttävän ohjelmiston luotettavuus, sillä se luottaa tietoon, jonka tuottaa kolmas osapuoli, joka voi vaihtua suoritusaikana.

Ratkaisut tuntuvat tällä hetkellä olevan lähinnä käyttäjän tukemiseksi. Siinäkin on paljon parantamisen varaa, sillä koneen pitäisi ymmärtää mitä käyttäjä etsii, eikä etsiä vain käyttäjän antamia avainsanoja.

Mielenkiintoista on myös se, että palveluiden tarjoajat voivat vaihdella. Teoriassa on mahdollista, että kehitysvaiheessa on selailtu saatavilla olevia palveluita, ja päätetty toteuttaa ohjelmisto siten, että se käyttää joitain palveluita. Myöhemmin voidaan sitten huomata, että palveluita ei enää ole julkisesti saatavilla. Tai mitä jos sovellus etsii käyttämänsä palvelut automaattisesti, mutta ei löydä mitään sopivaa.

Tarvitaan selvästi sopimuksia palveluiden toiminnasta. Tämä aiheuttaa vielä lisää ongelmia palveluiden etsimiseen ja valintaan, sillä pitäisi pystyä jollain tavalla etsimään sellaisia palveluita, jotka lupaavat tiettyä jatkuvuutta ja luotettavuutta.

Esimerkiksi Günay ja Yolum [GY08] esittivät menetelmän, jossa haku määritellään lineaarisella temporaalilogiikalla. Muodostettu looginen lauseke kirjoitetaan sitten XML muotoiseksi hauksi. Tällöin voidaan määritellä tarkasti millaiset ehdot palvelun pitää täyttää. Vaikka sinänsä vaikuttaa hienolta, että haku voidaan rajata erittäin tarkoin ehdoin, niin käytännössä kuitenkin loogisten lausekkeiden miettiminen ja niiden kirjoittaminen XML muotoiseksi hauksi kuulostaa varsin raskaalta toimenpiteeltä pelkän palvelun hakemisen toteuttamiseksi. Miten toimitaan jos haku on liian rajattu, eikä yhtään vastausta löydy? Idea vaikuttaa vielä turhauttavammalta kuin yksittäisten avainsanojen avulla hakeminen.

5 Keskitetty ja hajautettu hakemisto

UDDI implementaatio on esimerkki keskitetystä palvelurekisteristä. Ongelmallista näissä on se, että rekisterin ylläpitäjä voi vaikuttaa siihen, mitä haut tuottavat. Keskitetyt järjestelmät eivät myöskään skaalaudu hyvin, jolloin ne voivat muodostaa pullonkauloja. UDDI:ssa on mahdollista vaihtaa hakemistodataa eri UDDI rekistereiden välillä, mutta palveluiden ja rekisterien määrän kasvaessa kopioinnista tulee epäkäytännöllistä.

Vertaisverkot tarjoaa infrastruktuurin palvelujen etsintään hajautetussa, itsestään organisoituvassa, ympäristössä. Kukin verkon solmu toimii tiedon reitittäjän roolin lisäksi palveluiden tarjoajana. Tai oikeastaan ne tarjoavat tiedon siitä, miten palvelua pääsee käyttämään. Tietynlaisten vertaisverkkojen on todettu skaalautuvan hyvin [SMLN+03]. Lisäksi vertaisverkoissa mikään taho ei voi vaikuttaa siihen, mitä palveluita verkosta löydetään.

6 Tehokkaampi palveluiden hakumenetelmä

Tässä kappaleessa esitellään tutkimus, joka pyrkii parantamaan palveluiden löydettävyyttä. Tutkimuksen näkökulmana on käyttäjän palveleminen tarkemmilla tuloksilla, eli tässä ei pyritä toteuttamaan automaattista palveluiden valintaa.

6.1 WSQBE

Crasso, Zunino ja Campo (2008) esittävät WSQBE (Web Service Query-By-Example)-nimisen menetelmän [CZC08], jolla etsitään palveluita siten, että palveluita etsivä taho tuottaa sen palvelun rajapintamäärittelyn, jonka haluaa ja käyttää sitä haun parametrina. Haun rajapintamäärittelystä muodostetaan vektori, jota verrataan rekisteröidyistä palveluista tehtyyn vektorien joukkoon. Vektoreiden välisestä kulmasta voidaan päätellä niiden samankaltaisuus.

Sen sijaan että käyttäjä voi kirjoittaa haluamansa palvelun rajapintamäärittelyn, voi hän kirjoittaa kuvauksen myös luonnollisella kielellä. Molemmissa tapauksessa näitä hakuja käsitellään tietyillä tavoilla joiden seurauksena syötteistä muodostetaan joukko sanoja, joista puolestaan muodostetaan hakuvektori.

Hakuprosessin algoritmi on seuraavanlainen:

1: **procedure** DISCOVER(*example*)
2: String[] *stems*← PULLOUTSTEMS(*example*)
3: double[] *vector* ← CREATEVECTOR(*stems*)
4: Category[] *category*← CLASSIFY(*vector*)
5: **for all** *service* ∈ *category*[0] **do**
6: **if** COSSIM(*vector*, *service*) >*threshold* **then**
7: APPEND(*service*,*candidates*)
8: **return** *candidates*

Aluksi syötteestä muodostetaan joukko sanoja. Sanajoukossa pyritään yhdistämään erilaisia nimeämistapoja, esimerkiksi siten että get_quote ja getQuote käsitellään samanlaisina. Tämän jälkeen sanajoukosta poistetaan vielä stop-wordit ja sanojen taivutuspäätteet, jolloin saadaan sanojen vartalot (stems). Näistä sanoista muodostetaan vektori, joka pyritään luokittelemaan sen komponenteista pääteltyyn kategoriaan. Vektorien luokittelussa oletetaan, että palvelun kategoria riippuu palvelun tekstikuvauksista ja sen metodin tunnisteista. Lopuksi hakuvektoria verrataan kaikkiin saman kategorian palveluista samalla tavalla tehtyihin vektoreihin. Riittävän samankaltaiset vektorit muodostavat siten vastausten joukon.

WSQBE pyrkii minimoimaan palvelun etsijän vaivan antamalla etsijälle mahdollisuuden toteuttaa haku luonnollisella kielellä, tai esimerkiksi Java:lla

määritellyn rajapinnan avulla, joka muunnetaan WSDL dokumentiksi Java2WSDL:llä. Tästä etuna se, ettei kehittäjän tarvitse opetella uutta kieltä tehdäkseen hakuja. Olettaen tietysti että Java on jo tuttu.

WSQBE käyttää TF-IDF (Term Frequency – Inverse Document Frequency) tai TF:ää sanojen painottamiseen. Menetelmissä siis usein dokumentissa esiintyvä sana on tärkeämpi kuin harvoin esiintyvä, ja mitä harvinaisempi sana on kaikissa dokumenteissa (IDF) niin sitä oleellisempi se on niille dokumenteille, joissa se esiintyy. Tästä seuraa se, että kun lisätään uusi dokumentti, niin sanojen IDF muuttuu ja se pitää laskea uudelleen, mikä taas tarkoittaa sitä, että palveluiden kuvauksista luotujen vektoreiden komponentit muuttuvat, jonka vuoksi koko vektoriavaruus pitää laskea uudelleen. TF on yksinkertaisempi heuristiikka, jossa ei ole tätä ongelmaa, mutta ei myöskään niin tarkkoja hakutuloksia.

Crasson, Zuninon ja Campon näkemyksen mukaan suurin ongelma WSQBE:ssä on, että se olettaa laajan luokittelun olevan olemassa. Nimittäin jos WSQBE:ssä halutaan luoda uusi luokka palveluille, niin kaikki olemassa olevat palvelut joudutaan luokittelemaan uudelleen. Toinen ongelma on, että vaikka menetelmä yrittää yhdistellä erilaisia syntakseja, niin haku on siitä huolimatta syntaktinen. Löydettyjen palveluiden tarkkuus riippuu siis niiden julkaisijoiden käyttämästä syntaksista ja tavoista luoda kuvauksia.

6.2 Semantiikan käyttämisestä hauissa

Avainsanahaun hallitseva ongelma on se, ettei sen avulla voi löytää kahta samankaltaista palvelua, joilla on erilaiset kuvaukset: kaksi eri WSDL kuvausta voi kuvata saman palvelun, mutta eri sanoin.

Ontologioilla voidaan mallintaa palveluissa esiintyvien käsitteiden merkitystä, joiden suhteita voidaan hyödyntää semanttisen haun toteuttamiseksi. Tällöin siis sanojen etsimisen sijaan siirrytään sanojen merkityksien etsimiseen ja löytämiseen. Palveluiden semanttisten kuvauksien toteuttamiseen on useita kieliä, esimerkiksi: DAML-S, OWL-S tai WSDL-S. [GPS04]

Siitä huolimatta, että ontologioiden tekeminen on kallista, aikaa vievää, kurjaa ja virhealtista [CZC08] niin vaikuttaa siltä että semanttiset lähestymistavat ovat suosittuja tutkimuskohteita tällä hetkellä. Tätä kehityssuuntaa on vähän ihmeteltävä, sillä vaikka

semanttiset menetelmät vaikuttavat oikein tehokkailta haun toteuttamisen kannalta, niin kukaan ei kuitenkaan halua toteuttaa ontologioita, joihin nämä menetelmät pohjautuvat. Lienee kuitenkin syytä mainita, että monet tutkimukset käyttävät pääsääntöisesti jonkinlaisia hybridejä yhdistellen semanttista hakua muihin hakumenetelmiin.

7 Yhteenveto

Tässä paperissa tehtiin varsin yleinen katsaus erilaisiin palveluiden etsintä- ja valintamenetelmiin. Pohdittiin hieman automaattisen palveluiden valinnan ongelmia, kuten miten palveluihin voi luottaa, josta taas seuraa hakemisen kannalta ongelmia.

Kerrottiin keskitetyn ja hajautetun hakupalvelun ominaisuuksista.

Lopuksi esiteltiin tutkimus WSQBE menetelmästä, jossa muodostettiin haku luomalla esimerkki halutun palvelun rajapintakuvauksesta ja puhuttiin hieman semantiikasta palveluiden haussa.

Lähteet

CZC08 Crasso, M., Zunino, A. ja Campo, M., Query by example for web services. SAC '08: Proceedings of the 2008 ACM symposium on Applied computing, New York, NY, USA, 2008, ACM, sivut 2376-2380.

GPS04 Garofalakis, J., Panagis, Y. ja Sakkopoulos, E., Web service discovery mechanisms: looking for a needle in a haystack. International Workshop on Web Engineering, 2004.

GY08 Günay, A. ja Yolum, P., Semantic matchmaking of web services using model checking. AAMAS '08: Proceedings of the 7th international joint conference on Autonomous agents and multiagent systems, Richland, SC, Portugal, 2008, International Foundation for Autonomous Agents and Multiagent Systems, sivut 273-280.

HZ07 Hao, Y. ja Zhang, Y., Web services discovery based on schema matching. ACSC '07: Proceedings of the thirtieth Australasian conference on Computer science, Darlinghurst, Australia, Australia, 2007, Australian Computer Society, Inc., sivut 107-113.

SMLN+03 Stoica, I., Morris, R., Liben-Nowell, D., Karger, D. R., Kaashoek, M. F., Dabek, F. ja Balakrishnan, H., Chord: a scalable peer-to-peer lookup protocol for internet applications. IEEE/ACM Trans. Netw., 11,1(2003), sivut 17-32.

TZC+07 Tsai, W. T., Zhou, X., Chen, Y., Xiao, B., Paul, R. A. ja Chu, W., Roadmap to a full service broker in service-oriented architecture. ICEBE '07: Proceedings of the IEEE International Conference on e-Business Engineering, Washington, DC, USA, 2007, IEEE Computer Society, sivut 657-660.