

Web-sovelluspalveluiden tekniset määrittymiset

Versio 1.0

	SerAPI-projekti	
	Yhteyshenkilö	Marko Sormunen (Marko.Sormunen@uku.fi)
	Dokumentin tila	Valmis
	Päiväys	31.8.2007

Sisällysluettelo

1	Johdanto	6
2	Palveluiden perusmäärittelykielet	8
2.1	SOAP	8
2.2	WSDL	8
2.3	Perusmäärittelykielten hyödyntäminen	9
3	Palveluiden yhteentoimivuus	10
3.1	WS-I Basic Profile	10
3.2	WS-I Basic Security Profile	10
3.3	WS-I Basic Attachments Profile	10
3.4	WS-I Sample Application	10
3.5	WS-I Usage Scenarios	11
3.6	WS-I -testityökalut	11
4	Web-sovelluspalvelujen turvallisuuden perusmäärittelyt	12
4.1	HTTPS	12
4.2	XML-Signature	12
4.3	XML Encryption	13
4.4	WS-Security	13
4.5	SAML	14
5	Palveluarkkitehtuurin yhteistoimintatekniikat	16
5.1	Sanomatason yhteistoiminta	16
5.1.1	WS-ReliableMessaging	16
5.1.2	WS-ReliableMessaging -prototyypitoteutus	19
5.1.3	WS-Trust	20
5.1.4	WS-Addressing	21
5.1.5	WS-SecureConversation	21
5.1.6	WS-Federation	21
5.2	Palveluiden käytäntöjen esittäminen ja mukauttaminen	22
5.2.1	WS-Policy	22
5.2.2	WS-SecurityPolicy	22
5.2.3	WS-PolicyAttachment	23
5.3	Yhteistoimintatekniikat ja tietoturvallisuus	23
5.4	WS-määrittelysten yhteentoimivuudesta	23
5.5	Palvelutason yhteistoiminta	24
5.5.1	Liiketoimintaprosessien tekninen kuvaus	24
5.5.2	Orkestroinnin mallintaminen	25
5.5.3	Palveluhakemistojen käytöstä	26
6	Tekninen infrastruktuuri	28
6.1	SOAP-sanoman kulku päättepisteiden välillä	28
6.2	Web-sovelluspalveluiden suoritusympäristö	30
6.3	SOAP-sovelluspalveluiden käyttö ilman SOAP-välineistöä	31
7	Eri soveltamistapojen teknisiä ratkaisuja	32
7.1	Toteutusrunkojen generointi WSDL:n pohjalta	32
7.2	HL7v3 WS Transport Profile	32
7.3	HL7 Finland Open CDA-adapterikäytännöt	32
7.4	ebXML	33
8	Teknisiä suunnittelukäytäntöjä	35

8.1	Web-sovelluspalvelun kutsumekanismit.....	35
8.1.1	Yksisuuntainen palvelukutsu	35
8.1.2	Kaksisuuntainen synkroninen palvelukutsu.....	35
8.1.3	Kaksisuuntainen asynkroninen takaisinkutsu	35
8.2	Yksinkertaisen SOAP-sovelluspalvelun laatiminen	35
8.3	Web-sovelluspalvelujen yksikkötestaaminen	36
8.4	Prosessien tekninen mallintaminen.....	36
8.5	Liiketoiminnallisen yhteistoiminnan tekninen mallintaminen.....	36
8.6	Esimerkki web-sovelluspalveluiden yhteistoiminnasta - CCOW.....	36
9	Yhteenveto	38
10	Lähteet.....	40

Liite 1: WS-I profiilien standardit

Liite 2: SAML

Versiohistoria

Versio:	Pvm:	Laatijat:	Selitys:
1.0	31.8.2007	Marko Sormunen Juha Mykkänen Heli Luostarinen Mikko Saesmaa	- Versio 1.0

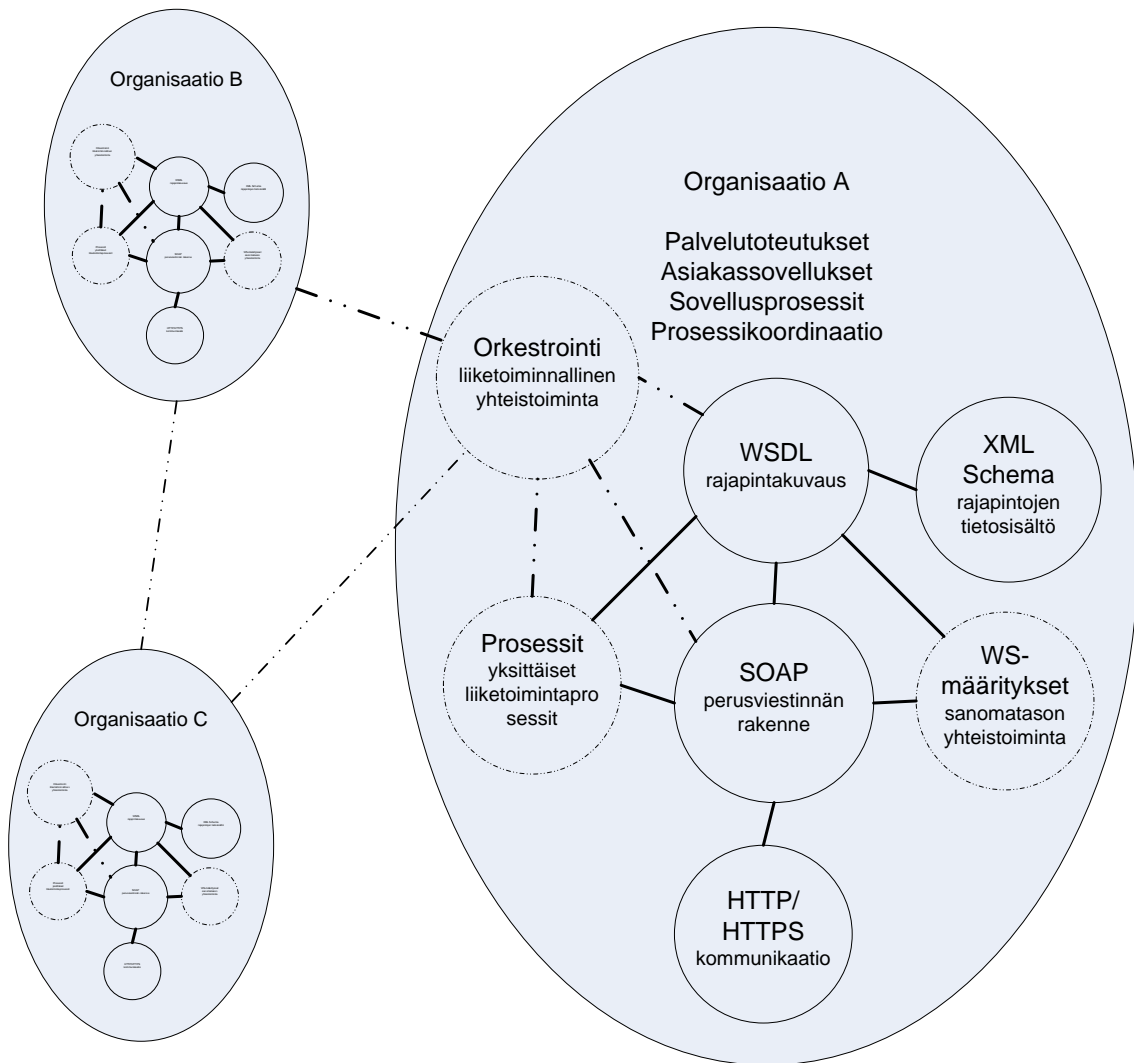
Esipuhe

Tämä työ liittyy SerAPI-hankkeeseen (Palveluarkkitehtuuri ja Web-sovelluspalvelut Terveydenhuollon Ohjelmistotuotannossa ja -integraatiossa), jossa tutkitaan ja kehitetään web-sovelluspalvelujen ja palvelupohjaisen arkkitehtuurin hyödyntämistä terveydenhuollon tietojärjestelmätarpeisiin ja sovellusintegraatioon ja uusiin sekä olemassa oleviin ohjelmistotuotteisiin. Hanketta rahoittavat Tekes (päätos nro 40437/04) sekä joukko yrityksiä ja sairaanhoitopiirejä.

1 Johdanto

Web Services-soveltamisoppaan kolmannessa osassa keskitytään web-sovelluspalveluiden teknisiin määrittymiin ja niiden hyödyntämiseen rakennettaessa palvelupohjaista sovellusarkkitehtuuria (SOA, *Service-Oriented Architecture*).

Dokumentissa keskitytään tärkeimpiin teknisiin määrittymiin, joita sovelluskehitysvälineistöjen toimittajat ovat edistäneet viime vuosina. Tarkoitus on lähestyä aihepiiriä antamalla suosituksia ja viittaamalla esimerkkeihin teknisten määrittymien käytöstä. Kuvassa 1 on esitelty tässä dokumentissa tarkemmin läpi käytäviä käsitteitä ja teknisiä määrittymiä.



Kuva 1. SOA-arkkitehtuurin tekninen koostumus

Kuvassa 1 esitetyistä teknisistä määrittymistä ovat vakiintuneet web services-perusstandardeiksi WSDL, SOAP, XML HTTP/HTTPS. Näiden avulla päästään jo toteuttamaan kutsu-vastaus -

periaatteella toimivia SOAP-sovelluspalveluita, jotka mahdollistavat usein jo tarpeeksi toiminnallisuutta varsinkin yhden organisaation sisäiseen käyttöön.

Organisaatioiden väliseen viestintään ja monimutkaisempien toimintaprosessien esittäminen on kuitenkin ollut näillä perusstandardeilla vaikeaa, ja on vaatinut hyvin epästandardeja ratkaisuita. Tässä dokumentissa esitellään myös SOAP-perusstandardien lisäksi tärkeimpiä sanomatason yhteistoimintaa kuvaavia WS-määrytyksiä.

Yksittäisten liiketoimintaprosessien kuvaamista käsitellään yleisellä tasolla. Lisäksi SerAPI on tuottanut selvityksen yhdestä prosessien tekniseen mallintamiseen tarkoitettua kielestä (SERAPIBPEL).

Organisaatioiden välisen liiketoiminnallisen yhteistoiminnan (orkestrointi) kuvaamiseen ei ole vielä vakiintunut kattavaa standardia. Aihetta käsitellään kuitenkin yleisesti. Lisäksi SerAPI on tuottanut selvityksen yhdestä orkestroinnin esittämiseen tarkoitettua kielestä (SERAPIWCDL).

2 Palveluiden perusmäärittyskielet

Tässä kappaleessa on kuvattu lyhyillä yhteenvedoilla web-sovelluspalveluiden perusmäärittyskielet. Luetelluista kielistä löytyy runsaasti esimerkkejä tästä dokumentista. Lisäksi luetellut kielet on kuvattu esimerkkien kanssa SerAPI:n tekemässä taustaselvityksessä ”Web-sovelluspalveluiden teknisiä suosituksia” (WSSER).

2.1 SOAP

SOAP (Simple Object Access Protocol) kuvaa web-sovelluspalvelun ja sitä kutsuvan asiakasprosessin välillä liikkuvien sanomien rakennetta, toisin sanoen, mitä XML-rakennneosia SOAP-sanomissa pitää olla, jotta ne ovat oikein muodostettuja.

Sovelluspalvelun ymmärtämien SOAP-sanomien tietosisältö ja XML-kielinen muoto määrätään web-sovelluspalvelun rakennetta kuvaavassa WSDL-dokumentissa.

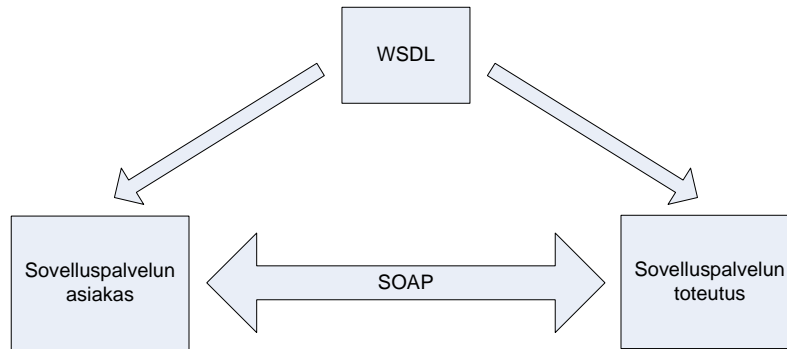
SOAP määrää myös sanomien rakenteen ja jakaa sen otsikko- ja runko-osioihin. Käytännössä otsikkotiedot sisältävät metatietoa SOAP-sanomasta, kuten WS-määrittysten mukaisia XML-elementtejä, ja runko-osio itse sanomassa välitetyn XML-tietosisällön.

Nykyaikaiset sovelluskehitysvälineet mahdollistavat SOAP-sanomaliikenteen abstraktion sovelluspalvelujen WSDL-esityksen avulla, kuten kappaleessa 2.2 on esitetty. Tällöin SOAP-sanomaliikenne ei näy välttämättä sovelluskehittäjälle mitenkään.

2.2 WSDL

WSDL (Web Service Description Language) kuvaa, miten sovelluspalveluja voidaan esittää XML-muotoisen WSDL-dokumentin avulla alustasta ja tiedonsiirrosta riippumatta. WSDL esittää sovelluspalvelun rajapintoja ja tietosisältöä abstraktilla tasolla XML-skeeman (XML Schema) avulla. Käytännössä WSDL-dokumentissa lisäksi sidotaan (binding) sovelluspalvelu käyttämään SOAP:ia sanomatason siirtoformaattina ja HTTP:tä tiedonsiirto-protokollana.

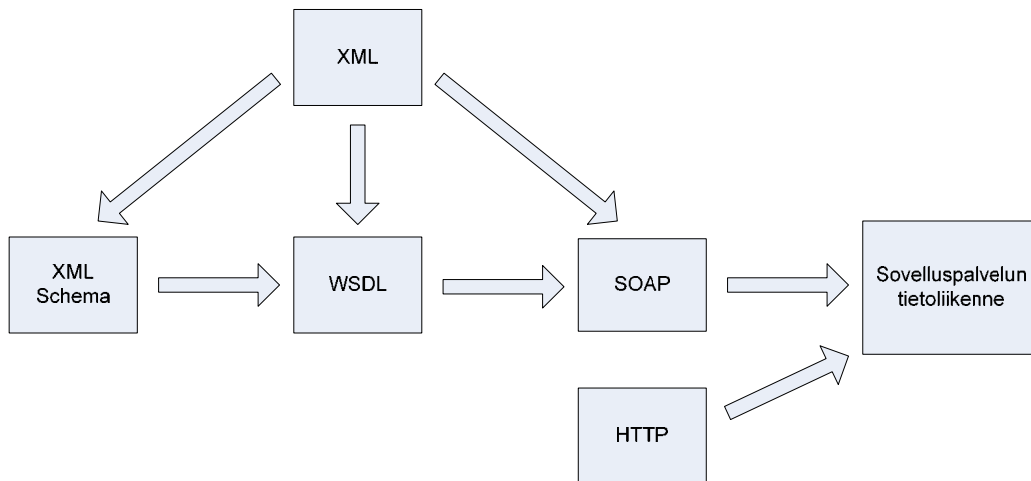
Web services-välineistö voi käyttää WSDL-esitystä joko sovelluspalvelun tai asiakassovelluksen rungon generoimiseen jollakin ohjelmointikielellä. Tällöin on mahdollista toteuttaa tai kutsua sovelluspalvelua normaalien API-tyyppisten ohjelmointirajapintojen avulla. SOAP-välineistöt tarjoavat myös mahdollisuuden generoida automaattisesti WSDL-esityksen luokasta, tai luokan rajapinnan kuvaavasta luokasta. Näin generoidun WSDL:n avulla voidaan ottaa sovelluspalvelu nopeasti käyttöön, koska WSDL-esitys sisältää kaiken tarvittavan informaation sovelluspalvelun etäkäyttöön. Tämä tilanne on esitetty kuvassa 2; WSDL-esitys pitää sisällään kaiken tarvittavan tiedon SOAP-sanomaliikenteen toteuttamiseksi asiakkaan ja toteutuksen välillä.



Kuva 2. Sovelluspalvelun toteutuksen, asiakkaan ja WSDL:n looginen suhde

2.3 Perusmäärittelykielten hyödyntäminen

Kuvassa 3 on esitetty looginen yhteys SOAP-perusmäärittelykielten ja sovelluspalvelun tietoliikenteen välillä. Kuvassa nuolet osoittavat määrittystä hyödyntävään osapuoleen.



Kuva 3. SOAP-perusmäärittelykielet ja sovelluspalvelun tietoliikenne

Periaatteessa SOAP-sanomien rakentamiseen riittää HTTP:n lähettämisen ja vastaanottamisen mahdollistava komponentti. Tällöin voidaan saada aikaan mahdollisimman joustavia SOAP-sanomia, sillä niitä voidaan koostaa suoraan merkkijonotasolla. Tämä kuitenkin on kömpelöä ja työlästä, ja käytännössä minimitasona voidaan pitää XML-tiedon ja SOAP-sanomien käsittelyä DOM-tasolla (Document Object Model), jossa XML-elementtejä käsitellään ohjelmointikirjastojen avulla.

3 Palveluiden yhteentoimivuus

Palveluiden yhteentoimivuuden parantamiseksi tässä dokumentissa **suositellaan** noudattamaan Web Services Interoperability (WS-I) -organisaation määrittymiä SOAP-sovelluspalveluiden sanomatasolle sekä WSDL-esitysten ymmärtämisen ja käyttöönoton parantamiseksi.

WS-I ei pyri kehittämään uutta toiminnallisuutta lisääviä määrittymiä, vaan parantamaan olemassa olevien määrittymien yhteentoiminnallisuuden kehittämiseen. Seuraavissa kappaleissa käydään läpi lyhyesti näistä määrittymistä sovellusohjelmoijille tärkeimpiä.

Tässä dokumentissa kuvattujen WS-I profiilien lisäksi WS-I kehittää ainakin luotettavaa SOAP-viestintää toteuttamista selkeyttää ohjeistusta.

Liitteessä 1 on lueteltu tarkemmin eri WS-I:n profiilien yhteen niputtamia standardeja.

3.1 WS-I Basic Profile

WS-I Basic Profile (WSBP) on WS-I:n julkaisema perusmäärittymä SOAP-sovelluspalveluiden sanomatason yhteentoimivuuden parantamiseksi. Lisäksi WS-BP esittää lisävaatimuksia ja rajoituksia helpottamaan sovelluspalveluiden WSDL-esitysten yksikäsitteistä generointia ja ymmärtämistä

3.2 WS-I Basic Security Profile

WS-I Basic Security Profile (WSIBSP) pyrkii selkeyttämään XML-elementtien allekirjoitukseen (XML-Security, kappale 4.2) ja salaukseen (XML Encryption, kappale 4.3) perustuvan SOAP-sanomien turvallisuusmäärittymien (WS-Security, kappale 4.4) esittämien rakenteiden käyttöä. WS-I BSP:tä **suositellaan** noudattamaan mahdollisimman tarkasti allekirjoitettaessa tai salattaessa yksittäisiä XML-rakenteita SOAP-sanoman sisällä.

3.3 WS-I Basic Attachments Profile

WS-I Basic Attachments Profile (WSIATT) kuvaa, miten MIME-tyyppisiä liitetiedostoja voidaan määrittellä web-sovelluspalveluiden kutsujen yhteydessä WSDL-esityksessä, ja siirtää SOAP-sanomien kanssa samassa HTTP-viestissä. Profiilia **voidaan** käyttää, jos liitetiedostojen tarjoamaa toiminnallisuutta tarvitaan SOAP-sanomiin. Käytännössä MIME-tyyppinen liitetiedostojen esitystapa on sama kuin mitä käytetään esimerkiksi sähköpostin avulla lähetettävien liitetiedostojen yhteydessä.

3.4 WS-I Sample Application

WS-I on edellisissä kappaleissa mainittujen määrittymien lisäksi laatinut useita käyttöprofiileja ja sovelluskehitystä tukevia dokumentteja kuvitellun hajautetun esimerkkisovelluksen ympärille (WSI-SAPP). Sovelluksessa on toteutettu kuvitteellinen sovelluspalveluketju tavaran valmistajasta varaston kautta tuotteen myyjään.

Sovellus niputtaa yhteen WS-I:n kehittämät määrittymiset ja toimii esimerkkinä niiden soveltamisesta. Sovelluksen tarkoitus ei sinällään ole olla esimerkkinä järkevästi toteutetusta sovelluspalveluarkkitehtuurista, vaan sen painopiste on palveluketjun eri osien yhteistoiminnan kuvaamisella.

WS-I tarjoaa esimerkkisovelluksen arkkitehtuurin lisäksi usean eri valmistajan rakentaman esimerkkisovelluksen, sekä käyttötapaesimerkkejä kuvitteellisen sovelluspalveluketjun toiminnasta WSDL-esityksestä SOAP-sanomatasolle (kappale 3.5). Lisäksi esimerkkisovelluksen dokumentaation kautta kuvataan, miten WS-I:n ohjeita ja määrittymiä noudattavan sovelluksen SOAP-viestintä toteutetaan turvallisesti.

3.5 WS-I Usage Scenarios

WS-I Usage Scenarios -määrittymis (WSIUSE) konkretisoi WS-I Basic Profile -määrittymistä reaali maailman soveltamistilanteissa tarjoamalla määrittymiä ja esimerkkejä matalalla sanomatasolla erilaisista SOAP-sovelluspalvelun käyttötilanteista. Lisäksi WS-I US esittää, miten kuvata näitä käyttötilanteita sovelluspalvelun WSDL-esityksessä.

3.6 WS-I -testityökalut

Sovelluspalvelutoteutuksen WS-I:n määrittymien mukaisuuden toteaminen olisi erittäin työlästä ilman WS-I:n tarjoamia automaattisia testityökaluja (WSITEST). Niiden avulla voidaan todentaa sovelluspalvelun WSDL-esityksen WS-I Basic Profilen mukaisuus teknisellä tasolla. Määrittymien mukaisuus tarkastetaan yksityiskohtaisen testilistan perusteella tutkimalla WSDL-esitystä ja siihen liittyviä skeematiedostoja.

Testityökalut toimivat sovelluskehittäjän näkökulmasta ”musta laatikko” -periaatteella siten, että sovelluspalvelun WSDL-esitys annetaan parametrina ja lopputuloksena saadaan yksityiskohtainen selvitys WS-I Basic Profilen mukaisuudesta. Selvitys voidaan liittää sovelluspalvelun dokumentaation joukkoon lausuntona WS-I:n määrittymien mukaisuudesta. Itse testausraportti ei takaa muuta kuin WSDL-esityksen ja sen mukaan laadittujen SOAP-sanomien WS-I:n mukaisuuden, minkä pitäisi taata mahdollisimman laaja tuki WSDL-esitykselle eri sovelluskehitysvälinevalmistajien taholta. Testausraportti ei kerro mitään WSDL:n laadusta tai sovelluspalvelun toiminnan kuvaamisesta järkevällä tavalla: esimerkiksi WSDL-esityksen viittaamista XML-skeematiedostoista tarkistetaan vain skeeman oikeamuotoisuus.

4 Web-sovelluspalvelujen turvallisuuden perusmäärittelyt

4.1 HTTPS

HTTPS-pohjaista salausta voidaan pitää *de facto*-standardina, kun halutaan salata SOAP-sanomat tiedonsiirtotasolla. HTTPS:n etuina voidaan pitää erittäin vakiintunutta standardisoitumista, ja sen voidaan todeta löytyvän sisäänrakennettuna kaikissa sovelluskehittimissä ja sovelluskehittämisalustoissa.

HTTPS rajoittuu kuitenkin suojaamaan ainoastaan kahden internetissä sijaitsevan ja tunnistetun päätepisteen (fyysisen koneen) välisen yhteyden. Erimerkiksi käytettäessä kuvan 5 esittämää välittäjäpalvelinta SOAP-sanomaa käsitellään välillä suojaamattomassa muodossa, ja se suojataan uudelleen lähetettäessä lopulliselle päätepisteelle. Seuraavissa kappaleissa keskitytään niihin turvallisuusstandardeihin, joiden avulla voidaan salata SOAP-sanomia sanomatasolla.

4.2 XML-Signature

XML-Signature -standardi (XMLSIG) määrittelee, miten XML-elementtejä voidaan allekirjoittaa. Allekirjoittamisen tarkoitus on suojata SOAP-sanomaa siten, että sanoman allekirjoitetusta tarkistettaessa mahdollinen allekirjoitetun sanoman tai allekirjoituksen muuttaminen huomataan.

Allekirjoitusten esittämistä varten XML-Signature esittelee erillisen Signature-elementin, joka sisältää kaiken allekirjoituksen myöhempään tarkistamiseen vaadittavan tiedon. Signature-elementti voidaan lisätä alkuperäiseen allekirjoitetun XML-elementin sisältävään rakenteeseen, tai esittää erillisenä.

Itse allekirjoittamistapahtuma tai allekirjoituksen tarkistaminen voidaan hoitaa erillisten ohjelmakirjastojen avulla, sillä allekirjoituksessa käytettävät algoritmien toteutukset (tiivisteiden laskeminen ja niiden salaaminen) toimivat musta laatikko -periaatteella. Käytännössä näiden algoritmien käyttäminen sovelluspalveluliikenteen turvaamiseen vaatii XML-Signature -määrittelyn ainakin pääosittaista ymmärtämistä, sillä esimerkiksi edellä mainitun Signature-elementin rakenne vaihtelee allekirjoituksen käyttötarkoituksen mukaan.

XML-Signature avulla voidaan XML-sanomiin lisätä sanoman mukana kulkevaa tietoa allekirjoituksesta ja allekirjoittajasta. Usein kuitenkin riittää, jos XML-sanomaa lähetettäessä julkisessa verkossa voidaan lähettäjän tai vastaanottajan fyysisen laitteen (palvelin tai asiakaskone) identiteetti voidaan varmentaa. Tähän tarkoitukseen HTTPS riittää hyvin, ja se voidaan ottaa käyttöön näkyvästi sanomatason alapuolella sanoman kuljetustasolla.

XML-Signature ei esitä, miten allekirjoitusten laatimisessa tai varmentamisessa tarvittavat sertifikaatit tai turvallisuusavaimet toimitetaan osapuolille. Käytännössä allekirjoituksessa tarvittavien sertifikaattien ja avainten hallinnointi on osa laajempaa organisaation ratkaistavaa PKI-kokonaisuutta (Public Key Infrastructure).

4.3 XML Encryption

XML Encryption -standardi (XMLENC) määrittää, miten yksittäisiä XML-elementtejä voidaan salata XML-sanoman sisällä. Alkuperäinen XML-elementti voidaan korvata uudella salatulla rakenteella, ja XML-sanomaan lisätä metatietoa, jolla salattu rakenne voidaan purkaa takaisin alkuperäiseksi XML-elementiksi.

Lisäksi XML Encryption sisältää rakenteita, joilla voidaan siirtää XML-elementtien allekirjoituksessa laatimisessa tai varmentamisessa tarvittavia turvallisuusavaimia voidaan siirtää XML-sanomien sisällä turvallisesti.

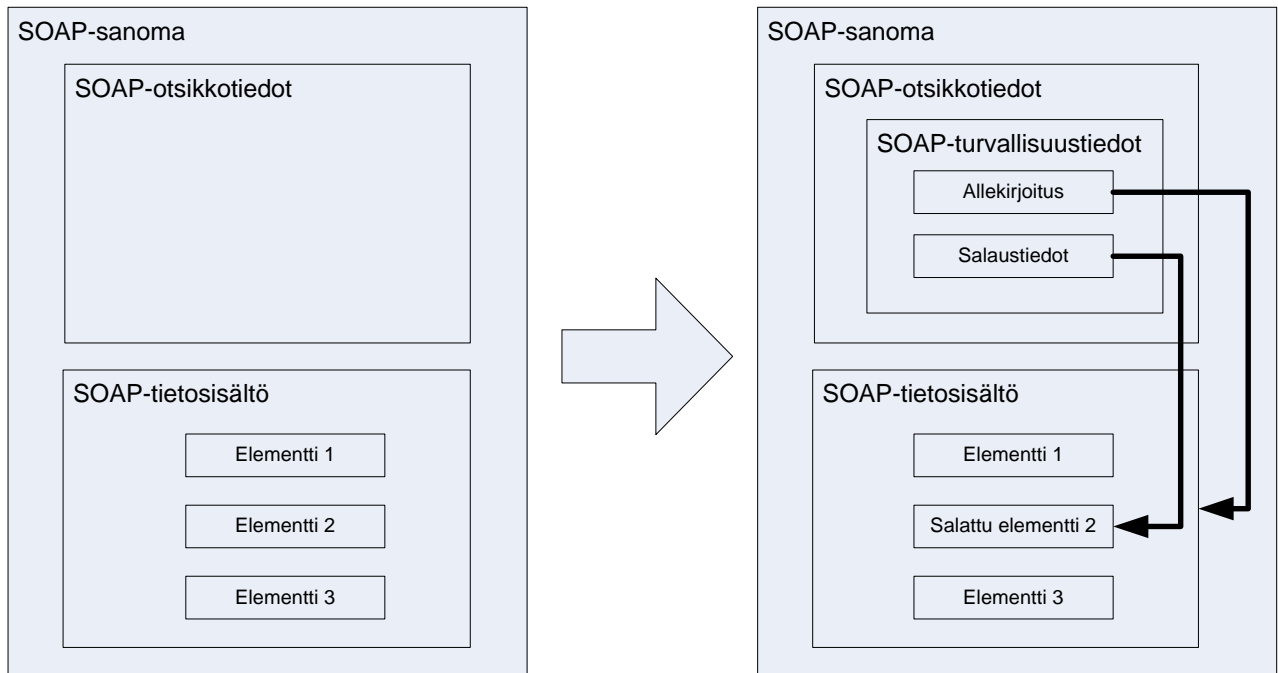
Käytännössä XML Encryption-standardin käytön pitää olla aina hyvin perusteltua, sillä yleensä salausta tarvitaan siirrettäessä XML-sanomia julkisessa verkossa. Tällöin koko sanomaa suojahtaessa HTTPS on ylivoimaisesti toimivin vaihtoehto, koska se voidaan ottaa käyttöön näkymättömästi sanomatason alapuolella sanoman kuljetustasolla.

XML Encryption -määrittymisen käyttö on perustellumpaa, jos tarvitaan XML-sanoman mukana kulkevaa salausta, jonka täytyy säilyä myös tiedon siirtotason ulkopuolella. Tällöin salaus säilyy koko ajan myös käytettäessä kuvan 5 mukaista välittäjäpalvelintä. Tällaista tietoa on SOAP-sanomien otsikkotiedoissa siirrettävät väliaikaiset avaimet tai muu salaustieto: Esimerkiksi kappaleessa 5.1.2 kuvattu WS-Trust -määrittymis pohjautuu XML Encryption -standardin käyttöön turvallisuuskontekstia laadittaessa kahden SOAP-päätepisteen välillä.

4.4 WS-Security

WS-Security -määrittymis (WSSEC) yhdistää XML-Signature - ja XML Encryption-standardeja, jotta niitä voitaisiin käyttää turvaamaan SOAP-sanomien avulla kommunikoivia sovelluspalveluja. Käytännössä tämä tarkoittaa SOAP-sanomien salaukseen ja allekirjoitukseen liittyvien metatietojen esittämistä sanoman otsikko-osassa siten, että turvallisuustieto ei vaikuta itse sanoman tietosisältöön. Salaamisen yhteydessä salattava SOAP-sanoman osa tosin luonnollisesti korvataan salatulla tiedolla.

SOAP-sanoma tai sen osia voidaan myös allekirjoittaa useaan kertaan, ja nämä allekirjoitukset voidaan tarkistaa toisistaan riippumattomasti. Samalla tavalla SOAP-sanoma tai sen osia voidaan salata tarpeen mukaan. Seuraavassa kuvassa 4 esitetään SOAP-sanoman allekirjoittamisessa ja osittaisessa salaamisessa tapahtuvat muutokset ja lisäykset (turvallisuustiedot) itse sanomaan. Kuvan esimerkissä SOAP-turvallisuustietoihin lisätään SOAP-tietosisältöön liittyvä allekirjoitus. Lisäksi osa SOAP-tietosisällöstä salataan.



Kuva 4. SOAP-sanoman turvallisuustiedot

Kuvan 4 SOAP-turvallisuustietoihin on sisällytettävä tarvittavat tiedot, jonka avulla SOAP-tietosisällön allekirjoitus voidaan tarkistaa, ja salattu tietoelementti purkaa. Nämä tiedot pitävät sisällään käytettyjen algoritmien nimet, ja viittaukset tarkistuksessa ja purkamisessa tarvittaviin turvallisuusavaimiin.

WS-Security -määrittely ei ota kantaa, miten SOAP-sanomaliikenteen salauksessa tarvittavien tietojen (sertifikaatit ja avaimet) jaetaan tai miten niitä hallinnoidaan organisaatio-tasolla.

WS-Securityyn tutustuminen ja käyttöönotto kannattaa aloittaa kappaleessa 3.2 esitellyn WS-I Basic Security Profilen kautta, sillä siinä kuvataan kaikki yleisimmät WS-Securityn käyttötapaukset, ja miten esittää SOAP-sanomien turvallisuusmetatietoa yhtenäisellä tavalla.

4.5 SAML

Security Assertion Markup Language -kielen (SAML) avulla voidaan XML-kielisiin dokumentteihin tai SOAP-sanomiin lisätä johonkin henkilöön tai muuhun subjettiin liittyvää tietoa, esimerkiksi liittyen henkilön onnistuneeseen varmentamiseen tai valtuuttamiseen. Nimestään huolimatta SAML siis soveltuu hyvin muunkin kuin turvallisuuteen liittyvän tiedon esittämiseen. Liitteessä 2 esitellään tarkemmin SAMLin toiminnallisuutta

SAML ei itsessään sisällä käyttäjän tai muiden osapuolien tunnistamismekanismeja, tai käyttöoikeuksien varmentamiseen tarkoitettuja vuorovaikutuksia, vaan se määrittää raamit ja elementit, joiden avulla sovellusprosessit voivat vaihtaa turvallisuuteen ja käyttöoikeuksiin liittyviä jo aikaisemmin oikeaksi todennettuja tietoja (*assertions*), esimerkiksi tieto käyttäjän onnistuneesta tunnistuksesta. Välittämällä tämä tieto SAMLin mukaisilla XML-elementeillä voidaan toteuttaa esimerkiksi kertakirjautuminen.

SAML on OASIS-järjestön standardi.

5 Palveluarkkitehtuurin yhteistoimintatekniikat

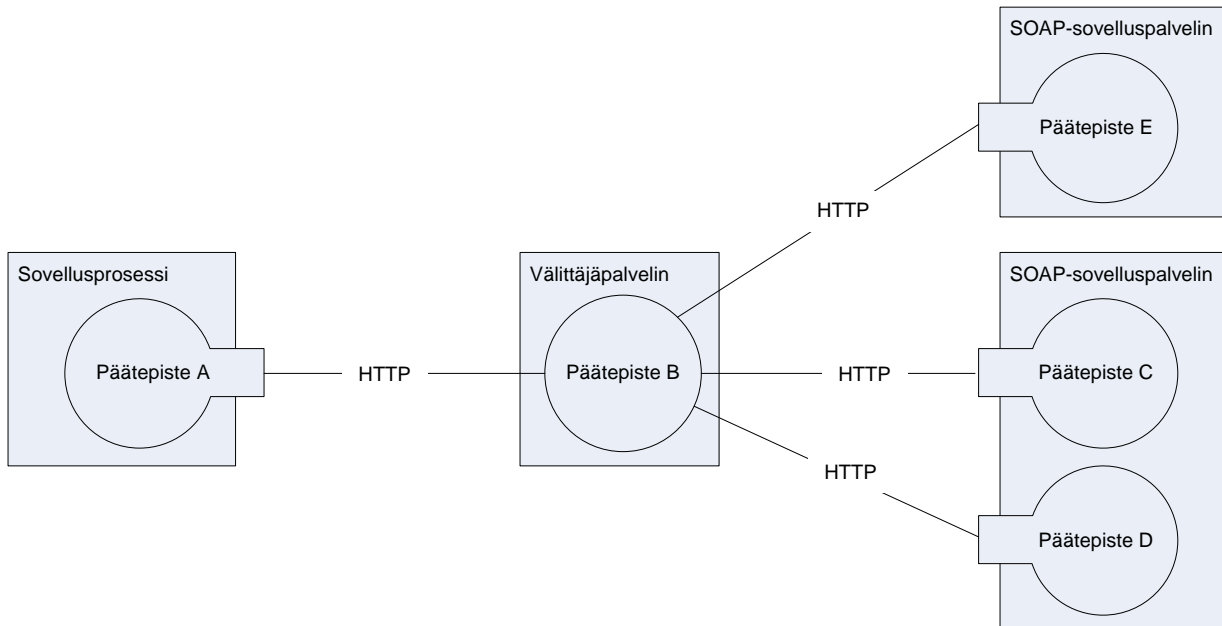
5.1 Sanomatason yhteistoiminta

Tässä luvussa käydään läpi vakiintuneimpia WS-määrittymiä, joilla voidaan lisätä SOAP-sanomatasolla sovelluspalvelujen yhteistoiminnallisuutta ja tulkittavuutta parantavaa metatietoa.

Luvussa esitellyistä WS-määrittymistä ainoastaan WS-ReliableMessaging ja WS-Trust tarjoavat toiminnallista määrittystä itsenäisten SOAP-sanomien muodossa; Muut WS-määrittymät sisältävät ainoastaan SOAP-sanomaotsikon sisällä kulkevaa metatietoa. Huomattavaa WS-määrittymissä on, että ne eivät XML:n rakenteisesta luonteesta johtuen ole sidottu SOAP-sanomarakenteisiin. Tämä mahdollistaa WS-määrittymien käytön esimerkiksi eri SOAP- ja WSDL-kielten versioiden kanssa.

Tässä luvussa esitetyt WS-määrittymät perustuvat kahden määritellyn SOAP-päätepisteen väliseen liikenteeseen. Käytännössä SOAP-päätepiste tarkoittaa useimmiten joko SOAP-asiakaprosessia tai SOAP-sovelluspalvelua, mutta päätepiste voi toimia myös pelkkänä sanoman ohjaajana. Kuvassa 4 on esitetty tilanne, jossa sovellusliikenne tapahtuu usean päätepisteen välillä.

Välittäjäpalvelin voi toimia myös väliaikaisena ”puskuripalvelimena” sovelluspalvelimien välissä siten, että esimerkiksi kuvassa 5 päätepisteet B ei välittäisi itse sanomia perille, vaan päätepisteet A, C, D ja E itse kyselisivät tietyn aikavälein niille tarkoitettuja sanomia.



Kuva 5. Usean päätepisteen välinen sovellusliikenne

5.1.1 WS-ReliableMessaging

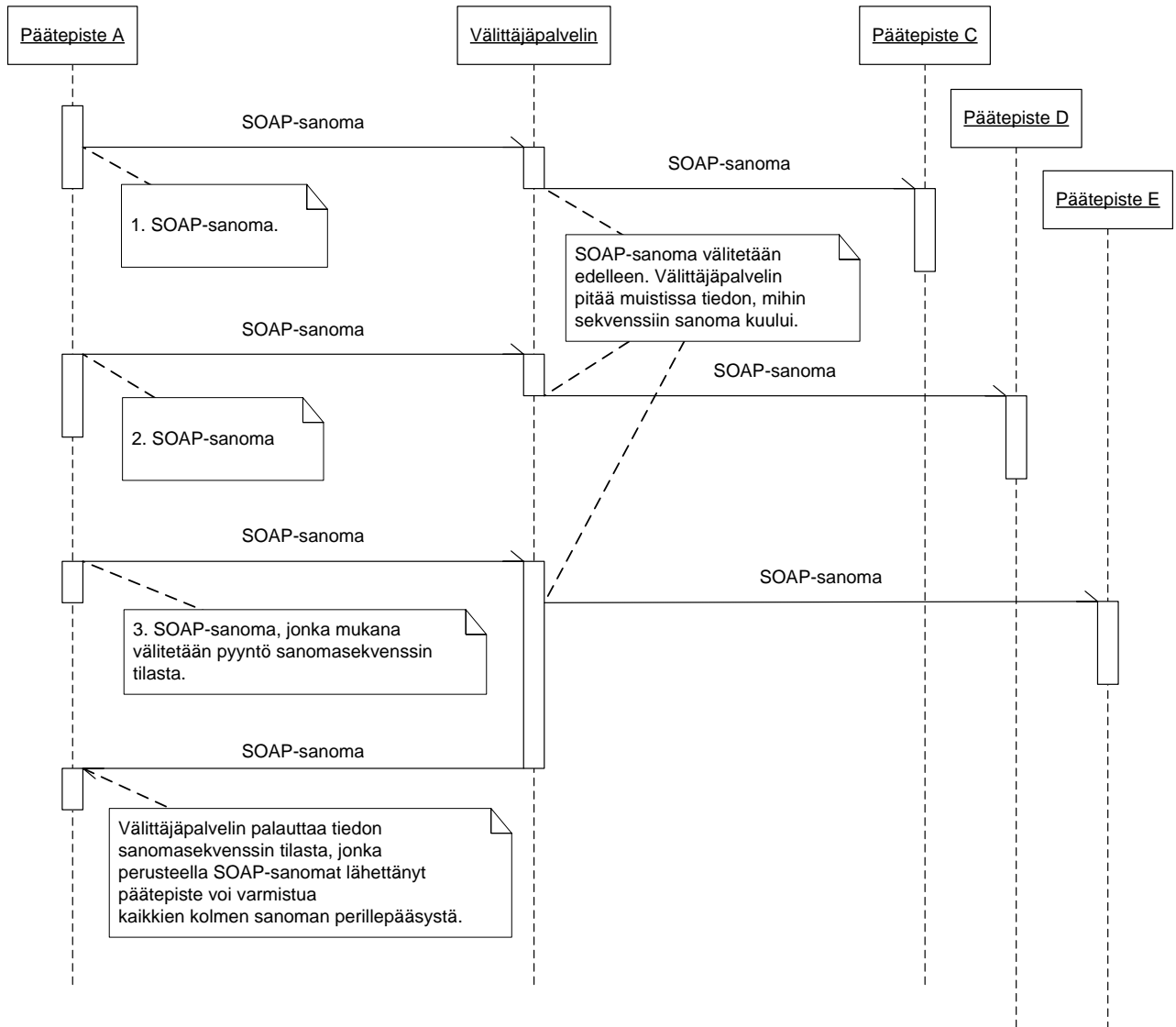
WS-ReliableMessaging (WSRM) on yksi tärkeimmistä WS-määrittymistä, sillä sen avulla voidaan toteuttaa siirtoprotokollasta riippumaton luotettava ja asynkroninen SOAP-viestintä. Tämä tapahtuu sopimalla ensin kahden SOAP-päätepisteen välillä yhteinen *sanomasekvenssi*, johon sitten voidaan

yhdistää SOAP-sanomia sekvenssitunnuksen avulla, siten että esimerkiksi SOAP-sanomien lähettäjä voi halutessaan kysyä toteutukselta sekvenssiin kuuluvien SOAP-sanomien perille pääsystä.

WS-RM ei ota huomioon fyysisestä tietoliikennekatkosta tai muusta vastaavasta virhetilanteesta aiheutuvaa häiriötä. Käytännössä sen avulla siis voidaan seurata, ovatko SOAP-sanomat menneet oikeaan osoitteeseen, ja ovatko ne prosessoitu oikeassa järjestyksessä.

Kuitenkin esimerkiksi käytettäessä HTTP-siirtoprotokollaa WS-RM:n käyttö on useimmiten normaalissa SOAP-päätepisteiden välisessä sanomaliikenteessä turhaa, sillä esimerkiksi jo SOAP-sanomien mukana kulkevista HTTP-statuskoodeista voidaan päätellä, onko SOAP-sanoma mennyt perille vai ei. WS-RM:n avulla kuitenkin voidaan usean SOAP-sanoman sitominen loogisesti yhteen siten, että yhteys säilyy SOAP-sanomatasolla sekvenssitunnuksen avulla. WS-RM myös mahdollistaa todellisten asynkronisten SOAP-sovelluspalveluiden toteuttamisen, koska sen avulla voidaan varmistaa lähetettyjen ja vastaanotettujen SOAP-sanomien käsittely oikeassa järjestyksessä.

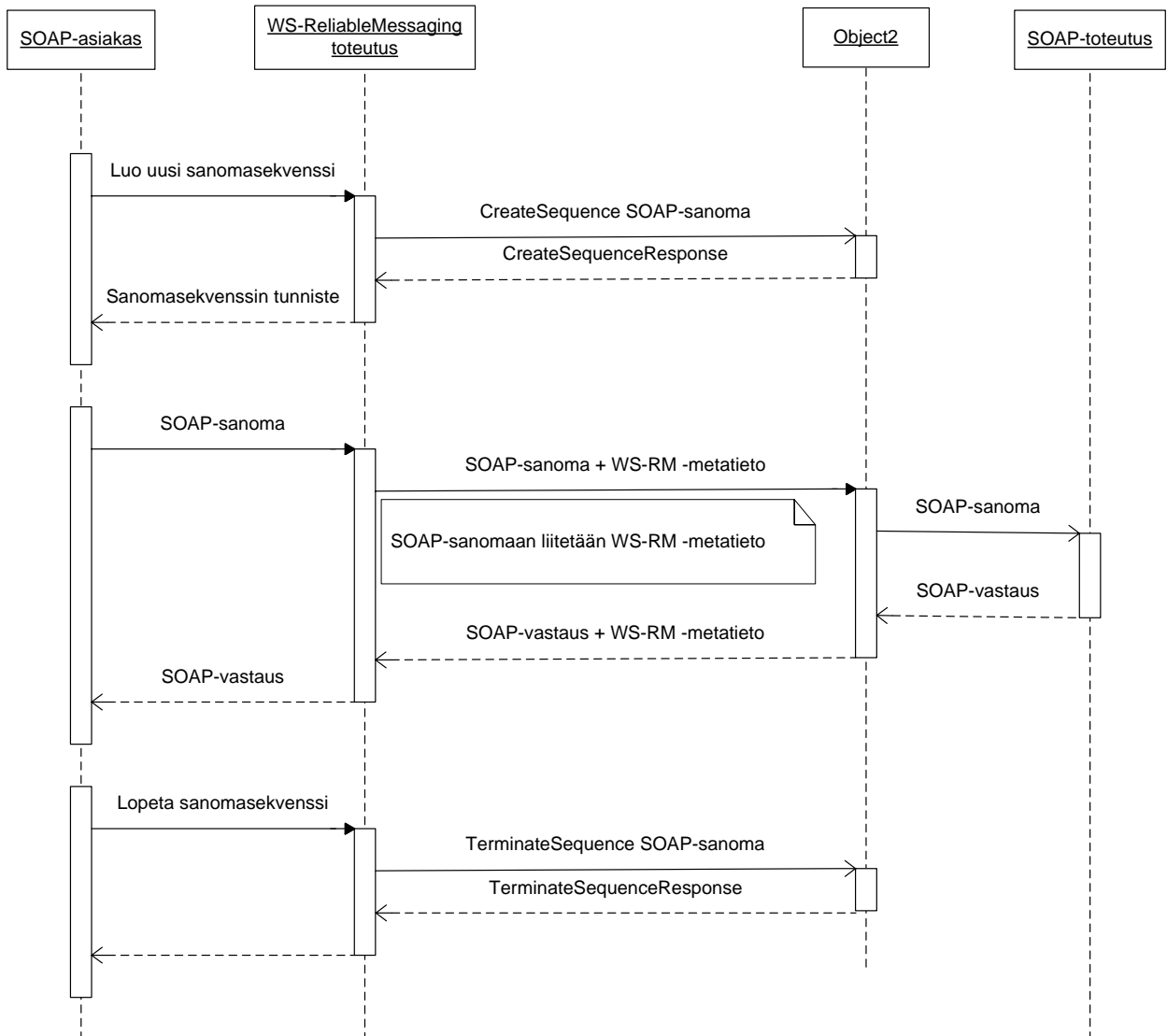
Yksi WS-ReliableMessaging -määrittelyksen käyttökohteista on SOAP-sanomaliikenteessä käytettävien välityspalvelimien asynkronisen liikenteen turvaaminen (kuva 6). Tällöin SOAP-sanoman lähettävä päätepiste voi pyytää välityspalvelinta palauttamaan varmistuksen perille menneestä sanomasta. Tämä tilanne on esitetty seuraavassa sekvenssikaaviossa.



Kuva 6. Asynkroninen SOAP-sanomaliikenne välittäjäpalvelimen kanssa

SOAP-sanoman kuuluminen sanomasekvenssiin ilmoitetaan lisäämällä tietty metatieto sanoman otsikkotietoihin. Samoin kyselyt ja vastaukset viestisekvenssin tilasta toimitetaan otsikkotietojen yhteydessä. SOAP-sanomia lähittävä päätepiste voi lopettaa sekvenssin halutessaan, ja pyytää kuitausta toiselta päätepisteeltä perille saapuneista sanomista. Kuvassa 7 on esitetty WS-ReliableMessagingin ja SOAP-sovellusviestinnän yhteistoimintaa.

WS-ReliableMessaging -määrittymästä on julkaistu vasta alustava versio, mutta sitä voidaan käyttää, jos sen tarjoamaa toiminnallisuutta tarvitaan. WS-ReliableMessaging -määrittymän uutta versiota ollaan myös standardisoimassa OASIS-järjestön kautta.



Kuva 7. WS-ReliableMessaging ja synkroninen SOAP-sanomaliikenne

5.1.2 WS-ReliableMessaging -prototyyppitoteutus

SerAPI on toteuttanut WS-ReliableMessaging -määrittymästä prototyyppitoteutuksen (WSRMP), jonka dokumentaation yhteydessä on nähtävissä esimerkkisanomia WS-määrittysten käytöstä. Proto-

tyyppitoteutus myös käyttää hyväkseen muita tässä luvussa mainittuja WS-määrittymiä. Prototyyppi käyttää hyväkseen myös muita luvussa 5 kuvattuja WS-määrittymiä.

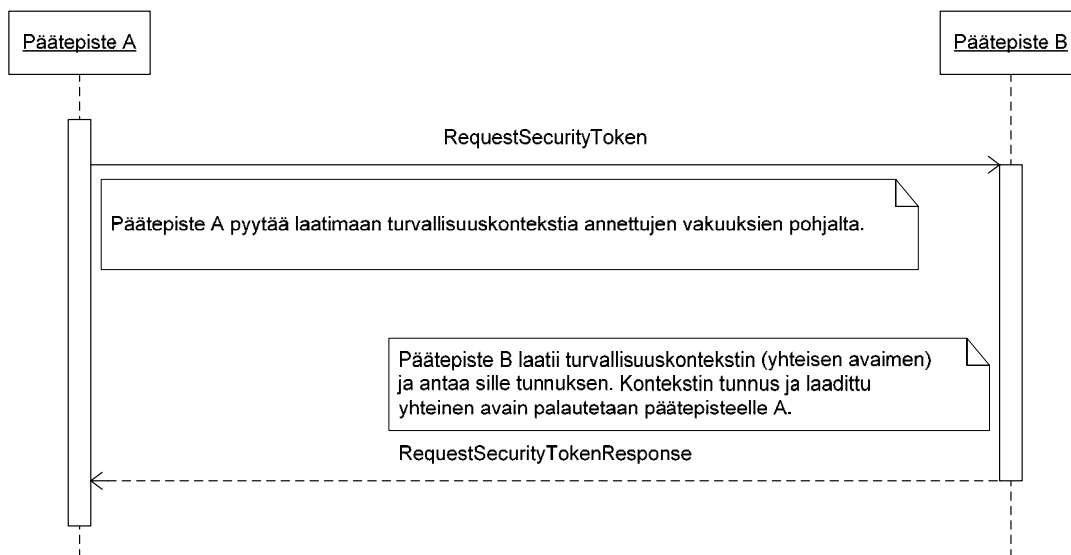
5.1.3 WS-Trust

WS-Trust -määrittymisen (WST) avulla kaksi SOAP-päätepistettä voivat sopia keskenään turvallisuuskontekstia. Yhteisen turvallisuuskontekstin tunniste liitetään kummassakin päätepiisteessä esimerkiksi jaettuun salaiseen avaimen. Tunnisteen esittämisessä suositellaan käytettävän WS-SecureConversation -määrittymistä (kts kappale 5.1.5).

SOAP-päätepiiste voi halutessaan käyttää turvallisuuskontekstia esimerkiksi SOAP-sanoman salaamiseen jaetun salausavaimen avulla ja liittää salaisuuteen liitetyn turvallisuuskontekstin tunnuksen mukaan SOAP-sanomaan. Sanoman vastaanottava päätepiiste pystyy purkamaan sanoman salauksen turvallisuuskontekstin tunnuksen avulla, yhdistämällä sen uudelleen samaan jaettuun salausavaimen.

SerAPI:n laatima WS-ReliableMessaging -prototyyppi (WSRMP) käyttää hyväkseen WS-Trust -määrittymistä pystyttämällä turvallisuuskontekstin, jossa samaan sanomasekvenssiin kuuluvat SOAP-sanomat lähetetään kontekstin sisällä.

Kuvassa 8 on esitetty yksi WS-Trust -määrittymisen esittämä tilanne, jossa turvallisuuskonteksti neuvotellaan jaetun salaisen avaimen avulla. Kuvassa mainitut vakuudet voivat olla esimerkiksi käyttäjän tai työaseman tunnistava sertifikaatti, tai salasana/tunnus -pari. Vakuudet siirretään salattuina XML Encryption -määrittymisen mukaisesti.



Kuva 8. WS-Trust ja turvallisuuskontekstin laatiminen

Kuvassa 8 oletetaan, että päätepiisteen A pystyy salaamaan kuvassa mainitut vakuudet ja päätepiiste B pystyy varmentamaan niiden lähettäjäksi päätepiisteen A. Lisäksi päätepiisteen A pitää pystyä varmentamaan päätepiiste B:n palauttaman yhteisen avaimen alkuperä. Käytännössä tämä voidaan toteuttaa laatimalla etukäteen kummallekin päätepiisteeseen asennettu avainpari, jonka avulla esi-

merkiksi pääte piste A salaa vakuudet ja pääte piste B purkaa ne. WS-Trust -määrittymis esittää myös muita tapoja luottamuksen varmentamiseksi.

WS-Trust -määrittymisestä on julkistettu vasta alustava versio, mutta sitä voidaan käyttää pohjana jos sen tarjoamaa toiminnallisuutta tarvitaan.

5.1.4 WS-Addressing

WS-Addressing -määrittymistä (WSA) hyödynnetään tai vaaditaan käytettävän melkein jokaisessa muussa WS-määrittymässä, sillä sen avulla voidaan standardilla tavalla esittää SOAP-pääte pisteitä. WS-Addressingin avulla SOAP-sanoman otsikotietoihin voidaan liittää metatietoa sanoman lähettäjä- ja vastaanottaja-pääte pisteistä. Vastaanottaja tarkoittaa tässä yhteydessä lopullista SOAP-pääte pistettä, ei esimerkiksi mahdollisia välityspalvelimia. Etuna tästä on, että SOAP-sanoman osoitetiedot siirtyvät sanoman sisällä, ja ne voidaan tulkita myöhemmin tarvittaessa.

Huomattavaa on, että WS-Addressing -määrittymisen mukaisilla osoitetiedoilla merkitty SOAP-sanoma voidaan lähettää eri fyysiseen osoitteeseen (esimerkiksi välityspalvelimelle) reititettäväksi, kuin mitä osoitetiedot antavat ymmärtää. Reititystä hoitava välityspalvelin on vastuussa WS-Addressing -tiedon tulkitsemisesta ja SOAP-sanoman lähettämisestä oikeaan lopulliseen osoitteeseen oikealla tietoliikenneprotokollalla.

Kappaleessa 5.1.1 esitetyssä kuvassa 6 välittäjä palvelin osaa ohjata SOAP-sanomat niiden oikeille pääte pisteille sanomien otsikotiedoissa olevien WS-Addressing -määrittymien mukaisten metatietojen perusteella.

WS-Addressingin avulla siis kaikki tiedot SOAP-sanoman lähettämisestä oikeaan osoitteeseen saadaan eroteltua tietoliikennekerroksesta. Tämä mahdollistaa SOAP-sanoman lähettämisen tarvittaessa eri tietoliikenneprotokollilla, esim. HTTP tai SMTP.

WS-Addressing sisältää myös tietoelementit, joilla SOAP-sanoma voi viitata edelliseen samaan prosessiin liittyvään SOAP-sanomaan. Tämä avulla voidaan toteuttaa yksinkertainen asynkroninen web-sovelluspalvelu, joka osaa lähettää SOAP-sanomavastauksen myöhemmin oikealle SOAP-pääte pisteelle.

WS-Addressing -määrittymis on W3C:n suositus (W3C Recommendation).

5.1.5 WS-SecureConversation

WS-SecureConversation -määrittymistä (WSC) käyttää hyväkseen muun muassa WS-Trust. Sen avulla SOAP-sanomaan voidaan lisätä tieto SOAP-sanoman kuulumisesta tiettyyn turvallisuuskontekstiin, joka on aikaisemmin esimerkiksi neuvoteltu käyttämällä WS-Trust -määrittymistä. Käytännössä WS-SecureConversation esittää SOAP-sanomaan lisättävän yhteisen kontekstin XML-rakenteen ja miten WS-SecureConversation toimii yhteen WS-Trust -määrittymisen kanssa.

WS-SecureConversation -määrittymisestä on julkistettu vasta alustava versio, mutta sitä voidaan käyttää pohjana, jos sen tarjoamaa toiminnallisuutta tarvitaan.

5.1.6 WS-Federation

WS-Federation -määrittymisen (WSF) avulla ennalta määritetyt toimialueet voivat yhdistää henkilöiden tai muiden instanssien turvallisuuteen tai valtuuksiin liittyviä tietoja. Toimialueet voivat olla

konkreettisia fyysisiä rakenteita (liikeyritykset tai sairaalat) tai puhtaasti toiminnallisia alueita kuten käyttäjän tunnistaminen tai valtuuksien myöntäminen ja hallinnointi. WS-Federation -määrittymisen avulla voidaan esimerkiksi välittää luotettavasti tieto käyttäjän sisäänkirjautumisesta tai valtuuksista kahden toimialueen välillä.

WS-Federation -määrittymisessä käytetään hyväksyen tässä dokumentissa esitettyjä määrittymiä WS-Trust, WS-SecurityPolicy ja WS-Security.

WS-Federation -määrittymisestä on julkistettu vasta alustava versio, mutta sitä voidaan käyttää pohjana jos sen tarjoamaa toiminnallisuutta tarvitaan.

5.2 Palveluiden käytäntöjen esittäminen ja mukauttaminen

5.2.1 WS-Policy

WS-Policy -määrittymisen (WSP) avulla voidaan ilmaista tietoa käytännöistä kahden SOAP-päätepisteen välillä. Tällaista ovat esimerkiksi tietoa web-sovelluspalvelun laadusta, kuten SOAP-sanomien käsittelyn nopeudesta. Lisäksi SOAP-päätepiste voi ilmaista, millaisia turvallisuusvarmenteita se olettaa päätepiestettä kutsuvan asiakasprosessin lähettävän, tai miten ne on laadittava. WS-Policy -määrittymisen avulla voitaisiin myös toteuttaa toiminnan ohjausta, niiltä osin kuin sen katsotaan olevan osa SOAP-sovelluspalvelun käyttöpolitiikkaa.

WS-Policy -määrittymisessä ei sisällä ohjeita, miten SOAP-päätepiestettä voidaan kysyä sen hyväksymiä käytäntöjä, tai miten SOAP-päätepieste voi itse julkistaa nämä käytännöt (kts. kappale 5.2.3). SOAP-päätepiesteiden käytännöt (esim. käyttöpolitiikka) pitäisi kuitenkin olla sanomaliikenteeseen osallistuvien SOAP-päätepiesteiden käytettävissä ennen varsinaista sovellustason SOAP-sanomien lähettämistä, jotta niistä olisi hyötyä.

Käytännössä WS-Policy -määrittymisen avulla voidaan siis esimerkiksi mukauttaa SOAP-päätepiesteiden toimintaa sen hetkiseen toimintaympäristöön. Tämä tosin voi vaatia hyvin paljon lisätoiminnallisuutta SOAP-sovelluspalvelun toteutukseen, ja käytännössä on useimmiten liian työlästä verrattuna saavutettuihin etuihin.

WS-Policy -määrittymisestä on julkistettu vasta alustava versio, mutta sitä voidaan käyttää pohjana, jos sen tarjoamaa toiminnallisuutta tarvitaan.

5.2.2 WS-SecurityPolicy

WS-SecurityPolicy -määrittymisessä (WSSP) laajentaa WS-Policy -määrittymistä esittämällä, miten se toimii muiden web-sovelluspalvelujen turvallisuutta käsittelevien WS-määrittymien kanssa yhteen (WS-Trust, WS-SecureConversation ja WS-Security). Esimerkiksi WS-Trust -määrittymistä toteuttava SOAP-päätepieste voi WS-SecurityPolicy -määrittymisen avulla ilmaista tarkempaa käyttötietoa, miten tai kenen uuden turvallisuuskontekstin halutaan laadittavan. WS-SecurityPolicy -määrittymisen avulla SOAP-päätepieste voi myös ilmaista, miten se haluaa vastaanottamansa SOAP-sanomat salattavan tai allekirjoitettavan.

WS-Policy -määrittymisessä ei sisällä ohjeita, miten SOAP-päätepiesteeltä voidaan kysyä sen hyväksymiä käytäntöjä, tai miten SOAP-päätepieste voi itse julkistaa nämä käytännöt (kts. kappale 5.2.3).

WS-SecurityPolicy -määrytyksestä on julkistettu vasta alustava versio, mutta sitä voidaan käyttää pohjana, jos sen tarjoamaa toiminnallisuutta tarvitaan.

5.2.3 WS-PolicyAttachment

W3C-konsortio on esittänyt alustavasti WS-PolicyAttachment -määrytystä (WSPA), jonka avulla WS-Policy -määrytyksen mukaisia XML-elementtejä voidaan liittää osaksi SOAP-sovelluspalvelun WSDL-määrytystä.

WS-PolicyAttachment - ja WS-SecurityPolicy -määrytysten avulla voidaan esimerkiksi SOAP-sovelluspalvelun WSDL-esityksessä esittää, mitkä osat SOAP-sanomista pitää allekirjoittaa tai salata.

WS-PolicyAttachment -määrytyksestä on julkistettu vasta alustava versio, mutta sitä voidaan käyttää pohjana, jos sen tarjoamaa toiminnallisuutta tarvitaan.

5.3 Yhteistoimintatekniikat ja tietoturvallisuus

Suurin osa tässä dokumentissa esitetyistä web-sovelluspalveluihin tai palveluarkkitehtuurin rakentamiseen liittyvistä WS-määrytyksistä on tarkoitettu käyttäjäturvallisuuteen tai käyttäjien valtuuttamiseen liittyvään yhteistoiminnallisuuden edistämiseen. Käytännössä käyttöönotettujen WS-määrytysten vaikutus täytyy tällöin olla laajempi kuin sanomatason yhteistoiminnallisuuden parantaminen.

WS-määrytysten käyttöönottoa hidastaa niiden soveltamisen vaikeus; esimerkiksi WS-ReliableMessaging -määrytyksen tarjoamat mahdollisuudet pitäisi ottaa huomioon jo alusta alkaen suunniteltaessa SOA-pohjaista palveluarkkitehtuuria, toisin kuin esimerkiksi HTTPS:n käyttöönotto. Tällöin ongelmaksi voi kokonaisuuden hahmottamisen vaikeus, sillä eri WS-määrytykset kuitenkin toimivat eri sovellusarkkitehtuuritasoilla.

Riittävän kattavaa tietoturvallisuutta suunniteltaessa pitäisi siis ottaa huomioon yksittäisten WS-määrytysten tarjoamat toiminnallisuudet, ja suunnitella niiden pohjalta kokonaisvaltainen ja yhtenäinen tietoturva-arkkitehtuuri, joka toimii myös osana palveluarkkitehtuuria.

5.4 WS-määrytysten yhteentoimivuudesta

WS-määrytysten tuki eri sovelluskehitysvälineistössä on vaihtelevaa. Tämä sen takia, että useat WS-määrytykset ovat edelleen kehityksen alla, ja usein määrytysten toiminnallisuus tai tekninen toteuttaminen ei ole määritelty kovin tarkasti. Niitä on tämän takia vaikea tukea sovelluskehitysvälineistöissä aidosti siten, että saavutettaisiin yhteistoiminnallisuuden parantumista.

WS-määrytyksiä käyttöönotettaessa kannattaa myös ottaa huomioon, että niitä hyödyntävät SOAP-sovelluspalvelut saattavat toimia suoraan yhteen ainoastaan, jos sovelluspalvelut toimivat saman infra-valmistajan palvelinympäristössä. Tämä johtuu siitä, että saman WS-määrytyksen toteuttaminen voi johtaa lähtökohdista ja tarkoituksesta riippuen hyvin erilaiseen lopputulokseen. Lisäksi toteutukset voivat toimia semanttisesti eri tavoilla myös hyödyntäessään WS-määrytysten XML-elementtejä.

WS-määrittäykset tarjoavat myös niin hienostunutta lisätoiminnallisuutta, että niiden yleinen hyötykäyttö SOAP-viestinnässä on vaikeaa. WS-määrittäyksiä kannattaakin ottaa käyttöön vain silloin kuin se oikeasti on perusteltua ja tarjoaa ehdottomasti tarvittavaa lisätoiminnallisuutta.

WS-määrittäykset eivät suoranaisesti monimuotoisuutensa takia paranna SOAP-päätepisteiden tai korkeammalla tasolla web-sovelluspalvelujen yhteentoimivuutta, vaan ne helpottavat asteittaista siirtymistä kohti SOA-arkkitehtuuria. Tämä johtuu siitä, että SOAP-sanomien otsikkotietoihin voidaan lisätä tarvittaessa laajennuksia ilman, että sanomien alkuperäinen rakenne tai merkitys muuttuu.

5.5 Palvelutason yhteistoiminta

Tässä kappaleessa esitellään korkeamman prosessi-tason yhteistoiminnallisuutta parantavia lähestymistapoja SOA-arkkitehtuuria kohti siirryttäessä ja tätä suoraan tukevia määrittäyksiä.

SOA-arkkitehtuurien kantavana ajatuksena on yksittäisten palvelujen koostaminen suuremmaksi kokonaisuudeksi joko yhdistämällä palveluja prosesseiksi, tai ohjaamalla palveluja ulkoisesti orkestroimalla. Palvelujen koostamista tai orkestrointia voidaan tehdä erikseen siihen määritellyllä kielellä, tai määrittää ja rakentaa palvelujen yhteistoiminta suoraan asiakassovelluksen koodissa.

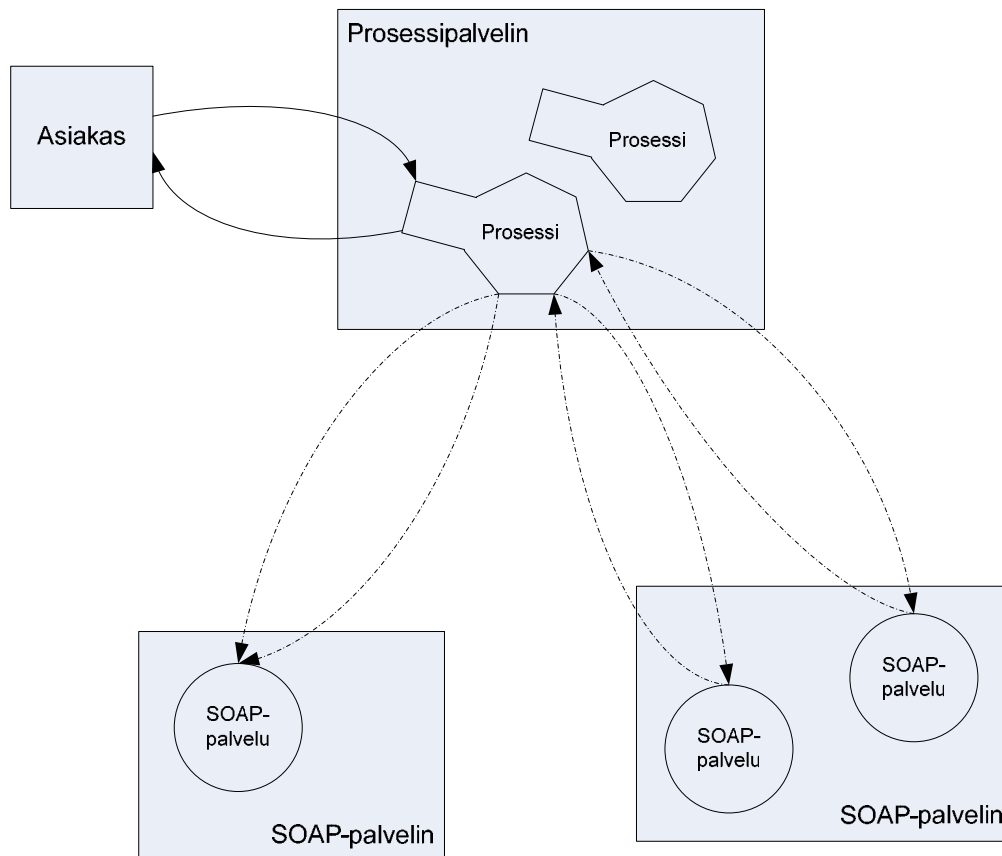
5.5.1 Liiketoimintaprosessien tekninen kuvaus

Liiketoimintaprosessin tekninen kuvaus tarkoittaa useamman web-sovelluspalvelukutsun niputtaminen yhdeksi prosessiksi, ja prosessin kuvausta siihen erikseen tarkoitettulla kielellä. Tällöin liiketoimintaprosessi voidaan irrottaa muusta sovelluslogiikasta, jolloin sen automatisointi ja seuraaminen on helpompaa. Lisäksi liiketoimintaprosessi voi sisältää muuta web-sovelluspalvelukutsujen ohjaamiseen tarkoitettua yksinkertaista ohjauslogiikkaa.

Prosessin määrittämiseksi voidaan käyttää siihen tarkoitettua kieltä, kuten BPEL (kts kappale 8.4). Tällöin prosessi voidaan eristää muusta sovelluslogiikasta ja suorittaa siihen tarkoitettussa sovelluspalvelimessa. Lisäksi palveluprosessia voidaan tarvittaessa muuttaa helpommin ja sen suorittamista ja tehokkuutta voidaan seurata paremmin. Kuvassa 9 on esitetty kaavakuva prosessista, sen asiakkaasta ja prosessin sovelluslogiikan toteuttavista sovelluspalveluista.

Prosessia suorittava sovelluspalvelin voi hoitaa myös prosessin kommunikaation ulkopuolisten SOAP-sovelluspalvelujen kanssa, ja käytännössä myös prosessi voi näkyä ulospäin kutsuttavana normaalina SOAP-sovelluspalveluna, joka voi toimia tarpeen mukaan synkronisesti tai asynkronisesti.

Palveluprosessi voidaan myös perinteisellä tavalla rakentaa suoraan sovelluskoodin sekaan siten, että ulkopuolisia SOAP-sovelluspalveluja kutsutaan samalla tavalla, kuin muitakin etäohjelmakutsuja. Tällöin palveluprosessin muuttaminen voi edellyttää suuria muutoksia itse kutsuvaan sovellukseen, ja palveluprosessin erottaminen muusta sovelluslogiikasta on paljon vaikeampaa.



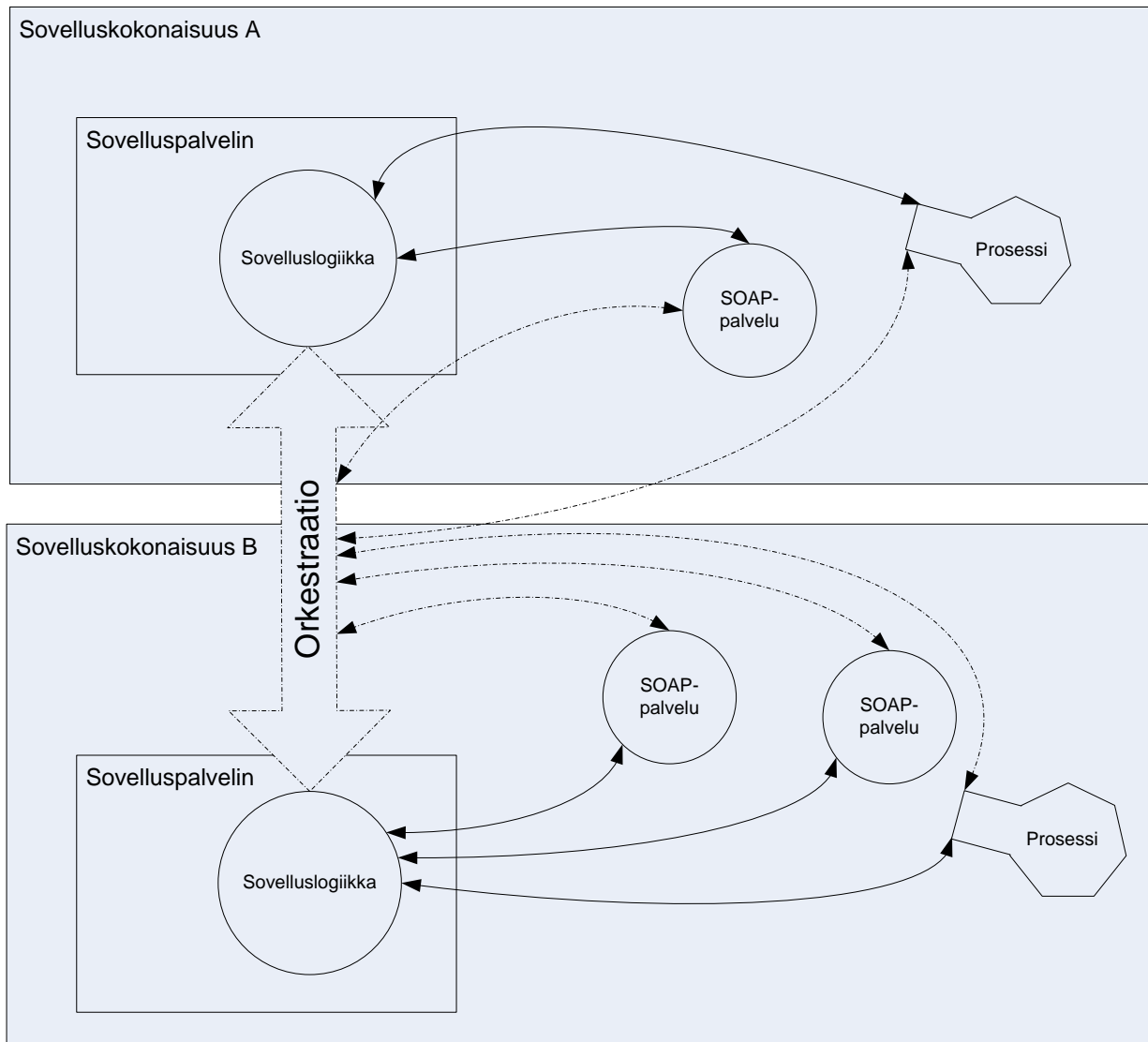
Kuva 9. Prosessi, prosessiin liittyvät palvelut ja prosessin asiakas

5.5.2 Orkestroinnin mallintaminen

Orkestroinnilla tarkoitetaan liiketoiminnallisen tason mahdollisimman pitkälle automatisoitua yhteistoimintaa, jossa useampi palveluprosessi koordinoidaan suuremman toiminnallisen tavoitteen saavuttamiseksi. Orkestroinnin avulla voidaan käytännössä yhdistää useampi palveluprosessi yhdeksi loogiseksi kokonaisuudeksi.

Orkestroinnin avulla pyritään myös eriyttämään sovellusprosessit muusta sovelluslogiikasta, jolloin sovellusprosessien välisen kommunikaation automatisointi ja seuraaminen on helpompaa.

Kuvassa 10 on esitetty kaavakuva toiminnasta, jossa kahden sovelluksen toiminnallisuutta on yhdistetty osan toiminnallisuudestaan yhdeksi kokonaisuudeksi. Sovellukset tarjoavat yhteisen orkestraatio-mallinnuksen kautta osiaan (SOAP-palveluja ja prosesseja) yhteistoiminnallisuuden käyttöön. Orkestraatio-mallinnus kuvaa osien roolit ja yhteistoiminnan säännöt.



Kuva 10. Esimerkki orkestroinnista ja sovellusten yhteistoiminnallidesta

Käytännössä orkestroinnin avulla useampaa yhteen liittyvää palveluprosessia voidaan tarkastella, hallita ja muokata yhden orkestraatio-määrittymisen kautta. Orkestraatio myös helpottaa liiketoiminnallisten tavoitteiden saavuttamista, sillä orkestraation osapuolet (esimerkiksi liikeyritykset) voivat paremmin sopia yhteisistä pelisäännöistä. Tämä tapahtuu helpoiten käyttämällä liiketoiminnallisen yhteistoiminnan määrittämiseen tarkoitettua kieltä, kuten kappaleessa 8.5 esitettyä WS-CDL (WSDL).

5.5.3 Palveluhakemistojen käytöstä

Kappaleissa 5.5.1 ja 5.5.2 esitellyt prosessien ja orkestraation mallintamisten kuvat esittävät impliittisesti, että sovelluspalveluja hyödyntävät osapuolet kutsuvat suoraan sovelluspalveluja etukäteen määrättyissä verkko-osoitteissa, esim. URL:ien avulla. Toinen mahdollisuus on abstraktoida sovelluspalvelujen fyysiset verkko-osoitteet palveluhakemistojen avulla. Tällöin sovelluspalveluja hyödyntävä osapuoli voi hakemistoa tutkimalla käyttää sopivaa sovelluspalvelua dynaamisesti. Tä-

mä on mahdollista vain, jos useampi samaan palveluhakemistoon rekisteröitynyt sovelluspalvelu tarjoaa saman palvelukokonaisuuden.

Palveluhakemiston määrittely ja rakentaminen voidaan toteuttaa esimerkiksi käyttämällä UDDI-standardeja (UDDI).

Käytännössä palveluhakemistojen ja dynaamisen yhteistoiminnan käyttöönottoa on kuitenkin vaikea suositella tai puolustella terveydenhuollon tietoteknisissä ympäristöissä, sillä tarvittavien sovelluspalvelujen tulee olla joka tapauksessa käytettävissä koko ajan, ja samankaltaisten sovelluspalvelujen rakentamista tai käyttöönottoa on vaikea perustella. Palveluhakemistojen levinneisyys ei myöskään ole vastannut niihin asetettuja odotuksia muilla toimialoilla.

Monesti sovelluspalvelujen käyttöönotto myös vaatii korkeamman tason organisaatioiden välisiä sopimuksia sovelluspalvelun tarjoaman toiminnallisuuden hyödyntämisestä, jolloin sovelluspalvelun käyttöönotto organisaatiossa johtaa kuitenkin kahden osapuolen välisiin sopimuksiin. Tällöin sovelluspalvelun suora hyödyntäminen ilman abstraktiokerrosta on perusteltua.

Kappaleessa 5.1 on esitetty toinen ratkaisu dynaamiseen yhteistoiminnan aikaansaamiseen erillisen välityspalvelun avulla. Välityspalvelun käytöllä saadaan aikaan samanlaisia lopputuloksia kuin palveluhakemistojen käytöllä, koska välityspalvelu ohjaa SOAP-sanomat lopulliseen fyysiseen verkkoosoitteeseen. Palveluhakemistoja käytettäessä alkuperäisen SOAP-sanoman lähettävä päätepiste lähettää sanoman suoraan lopulliselle kohdepalvelulle, kunhan sen osoite on ensin saatu selville.

6 Tekninen infrastruktuuri

6.1 SOAP-sanoman kulku päätepisteiden välillä

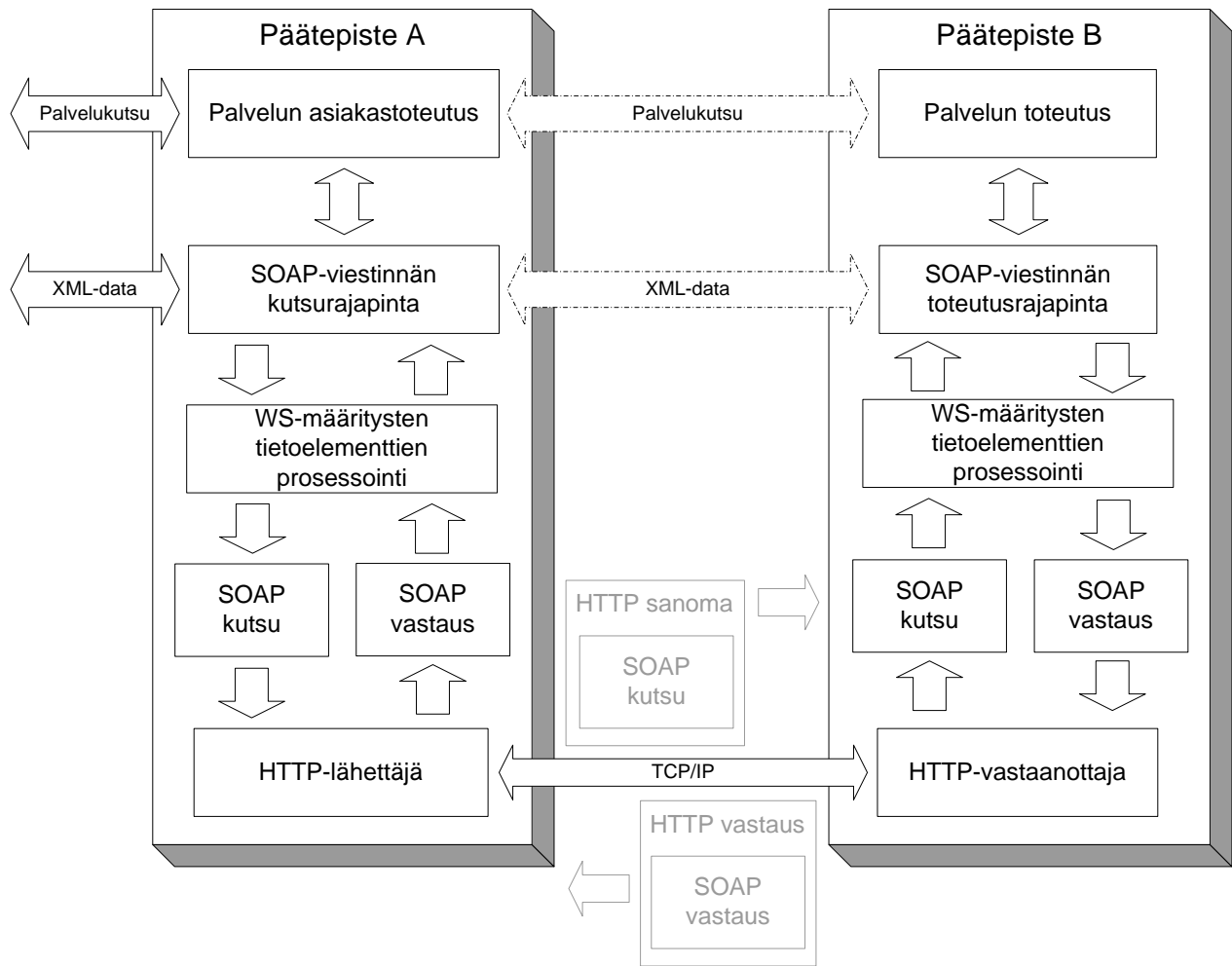
HTTP-protokollan avulla SOAP-sanoman lähettävän päätepisteen toiminta pitää sisällään:

- Aliohjelmakutsun, joka sisältää tarvittavat parametrit vastaavan SOAP-kutsun laatimiseen. Aliohjelmakutsu voi olla SOAP-välineistöstä riippuen normaalin aliohjelmakutsun muotoinen. Vaihtoehtoisesti asiakasprosessi voi koostaa lähetettävän SOAP-sanoman itse ja tarjota sitä SOAP-välineistön kautta lähetettäväksi XML-datana. SOAP-välineistö voi myös lisätä tarpeen mukaan WS-määritysten mukaisia tietoelementtejä SOAP-sanomaan.
- SOAP-kutsu lähetetään ennalta määrättyyn SOAP-päätepisteeseen prosessoitavaksi. SOAP-välineistöstä ja halutusta prosessointitavasta riippuen lähettäjä voi jäädä odottamaan vastausta (synkroninen prosessointi) tai ottaa vastaan vastauksen myöhemmin (asynkroninen prosessointi).

HTTP-protokollan avulla SOAP-sanoman vastaanottavan päätepisteen toiminta pitää sisällään:

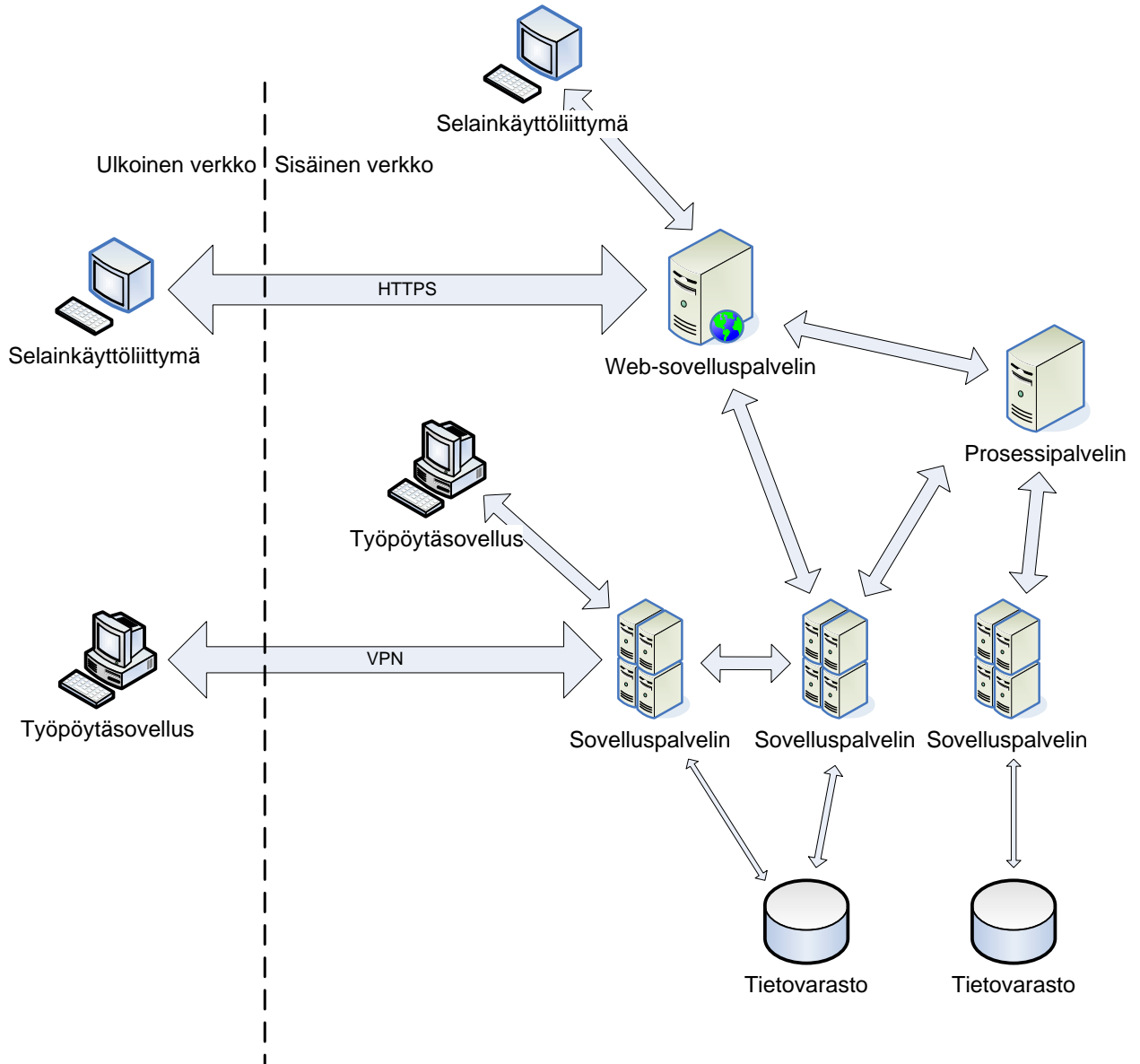
- SOAP-sanoma vastaanotetaan ja ohjataan SOAP-sovelluspalvelun toteutusrajapinnan kautta asianmukaiselle sovelluspalvelutoteutukselle. Vastaanottamisen yhteydessä voidaan myös tulkita SOAP-sanoman mahdollisia WS-tietoelementtejä. SOAP-sanoma voidaan esimerkiksi uudelleen reitittää toiseen osoitteeseen tai merkitä saapuneeksi, jos SOAP-sanoma on merkitty kuuluvaksi turvallisuuskontekstiin.
- Muunnetaan sovelluspalvelutoteutuksen palauttama vastaus SOAP-sanomamuotoon ja palautetaan se alkuperäiselle lähettäjälle SOAP-vastauksena. Paluusanoma voidaan lähettää joko alkuperäisen HTTP-viestin kuittausviestissä (synkroninen prosessointi) tai erillisen HTTP-viestin sisällä (asynkroninen prosessointi). SOAP-välineistö voi myös tarvittaessa lisätä WS-määritysten mukaisia tietoelementtejä paluusanomaan.

Kuvassa 11 on esitetty läpileikkauskuva SOAP-sovelluspalvelutoteutuksen perusosista. Web-sovelluspalvelun WSDL-esityksestä voidaan generoida kaikki muu, paitsi itse palvelulogiikan sisältävä palvelun toteutus. Tämä tosin edellyttää, että web-sovelluspalvelun kutsurajapinta on suhteellisen helposti esitettävissä perinteisten ohjelmointikielten semantiikalla. Muussa tapauksessa asiakasprosessi voi koostaa SOAP-sanoman itse ja tarjota sitä SOAP-välineistölle siirrettäväksi XML-datana.



Kuva 11. SOAP-sanomaliikenteen perusosat

6.2 Web-sovelluspalveluiden suoritusympäristö



Kuva 12. Web-sovelluspalveluiden suoritusympäristö

Kuvassa 12 on esitetty esimerkkinä kuvitteellinen web-sovelluspalveluiden fyysinen suoritusympäristön SOA-pohjainen arkkitehtuuri. Käytännössä suoritusympäristöissä web-sovelluspalvelimet ja muut sovelluspalvelimet usein yhdistetään yhdeksi fyysiseksi palvelimeksi, joka tarjoaa ajoympäristön yhdelle tai useammalle sovellukselle. Lisäksi työpöytäsovellukset vielä usein kommunikoivat suoraan tietovaraston kanssa, jos työpöytä-sovellus sisältää itse tarvittavan sovelluslogiikan.

Kuvassa on esitetty kahtia jako ulkoisen ja sisäisen verkon välillä. Ulkoisesta verkosta (julkinen internet) tulevat yhteydet on suojattava erikseen asianmukaisella tavalla, joko käyttäen HTTPS-suojauksia (selainkäyttöliittymät) tai VPN-putken (Virtual Private Network) käyttöön perustuvaa suojauksia. Sisäisestä verkosta sovelluspalvelimille tulevat yhteydet voidaan myös tarvittaessa suoja-

ta, mutta usein sisäisen verkon suojaamiseksi riittää IP-pohjainen verkon fyysisten laitteiden tunnistaminen.

Kuvan 11 arkkitehtuurissa prosessipalvelin voidaan myös käsittää sovelluspalvelimeksi, joka suorittaa prosessinmallinnuskielellä (esim. BPEL, kts kappale 8.4) esitettyjä prosesseja. Tällöin myös itse prosesseja voidaan kutsua samalla tavalla kuin normaaleja web-sovelluspalveluita SOAP-sanomien avulla.

Web-sovelluspalvelujen korkeampaa yhteistoimintaa, orkestraatiota, on vaikea sijoittaa SOA-arkkitehtuuriin täsmällisesti. Tämä johtuu siitä, että orkestraatiota ei ohjata tai suoriteta yhdessä palvelimessa, vaan jokainen orkestraatioon osallistuva web-sovelluspalvelu vastaa itse oman panoksensa tarjoamisesta orkestraatioon ja suorittamisen siirtymisen toiselle web-sovelluspalvelulle tarvittaessa. Tämä tarkoittaa, että jokaisen orkestraatioon osallistuvan web-sovelluspalvelun pitää osata tulkita orkestraation sisältämiä ohjeita yhteistoiminnallisuuden saavuttamiseksi, jos yhteistoiminta on esitetty esimerkiksi WS-CDL (kappale 8.5) -kielen avulla.

6.3 SOAP-sovelluspalveluiden käyttö ilman SOAP-välineistöä

Koska SOAP-sanomat koostuvat XML-elementeistä, niiden koostaminen on täysin mahdollista ilman erillistä SOAP- tai XML-välineistöä. Käytännössä SOAP-sovelluspalvelun kutsumiseen riittää HTTP-lähetäjä (kts kuva 8), jolle syötetään XML-muotoinen merkkijono vastaanottavan SOAP-sovelluspalvelun prosessoitavaksi. Tätä lähestymistapaa käyttävää sovellusliittymää kutsutaan myös REST-liittymäksi (Representational State Transfer). REST-lähestymistapaa seuraten yksinkertainen SOAP-sovelluspalvelu voi tarjota myös pelkkään HTTP:n käyttöön perustuvaa liittymää, jossa sovelluspalvelun parametrit annetaan suoraan sovelluspalvelun URL:in yhteydessä.

HTTP-lähetäjän pitää osata muodostaa TCP/IP -yhteys SOAP-sovelluspalvelimeen ja muotoilla lähetyksessä tarvittavat HTTP-otsikkotiedot. Lisäksi lähetäjän pitää osata lukea SOAP-sovelluspalvelun palauttama vastaus ja palauttaa se kutsuvalle asiakasprosessille. Käytännössä jokainen nykyään käytetty sovelluskehitysympäristö sisältää HTTP/HTTPS-komponentin, joka hoitaa tämän.

Samalla tavalla SOAP-sovelluspalvelin voidaan toteuttaa HTTP-vastaanottajan ympärille ilman kolmannen osapuolen toteuttamaan SOAP-välineistöä.

Asynkronisten SOAP-sovelluspalveluiden käyttö ilman SOAP-välineistöä voi olla vaikeaa, sillä asiakasprosessin pitää pystyä odottamaan myöhemmin palauttavaa SOAP-sovelluspalvelun vastaus-ta. Lisäksi asiakasprosessin pitää vastauksen saamisen jälkeen pystyä synkronisoimaan toiminta esimerkiksi käyttöliittymän kanssa ilman loppukäyttäjän toimintaa häiritsemättä.

7 Eri soveltamistapojen teknisiä ratkaisuja

7.1 Toteutusrunkojen generointi WSDL:n pohjalta

SOAP-välineistön pitäisi tarjota mahdollisuus palvelu- ja asiakastoteutuksen laatimiseen siten, että sovellusohjelmoija voi käsitellä SOAP-kutsu- ja -vastaussanomaa suoraan XML-oliomuodossa (niin sanottu Document Object Model (DOM) -muoto). Monimuotoista XML-dataa (esimerkiksi HL7v3-sanomia tai CDA R2-dokumenttia) on helpompi käsitellä DOM-muodossa, koska tämän lähestymistavan avulla SOAP-sanomia käsitellään yhtenäisen ja monipuolisen API-tyyppisen rajapinnan kautta.

Vaihtoehtoisesti SOAP-välineistö voi generoida täydellisen luokkahierarkian WSDL:n pohjalta. Tällöin SOAP-sovelluspalvelun toteuttaminen tapahtuu hyvin pitkälti samalla tavalla, kuin jos sovelluspalvelu toteutettaisiin paikallisena ohjelmakirjastona. Tämä lähestymistapa soveltuu hyvin pienimuotoisempien API-tyyppisten sovelluspalveluiden rakentamiseen, sillä niiden tällöin SOAP-sanomien luokkahierarkia ei kasva hyvin suureksi.

7.2 HL7v3 WS Transport Profile

HL7v3:n toimialakohtaiset (esim. potilashallinto ja ajanvaraus) sanomamäärittymiset on jaettu sovellusrooleihin, jotka esittävät jonkin toiminnallisen kokonaisuuden vaatimat sanomavuorovaikutukset. Sovellusroolit on järjestetty pareiksi siten, että esimerkiksi *ajanvarauksen pyytäjä*-sovellusroolin vastakappale on *ajanvarauksen varmistaja*-sovellusrooli.

HL7v3:ssä on erotettu tarkasti toimialakohtaiset sanomamäärittymiset siirtotason määrittymistä. Tämän ansiosta HL7v3 on määrittänyt erillisiä siirtoprofiileja HL7v3-sanomien siirtämiselle kahden osapuolen välillä.

Siirtoprofiileista HL7v3 Web Services Transport Profile (HL7TP) esittää, miten HL7v3-sanomia siirretään SOAP-kehyksen sisällä, ja miten HL7v3:n toimialakohtaiset sovellusroolit esitetään WSDL-kuvausten avulla. Yhdessä WSDL-esityksessä kuvataan, mitä HL7v3-vuorovaikutuksia (SOAP-sanomia) web-sovelluspalveluksi toteutettava yksi sovellusrooli ottaa vastaan.

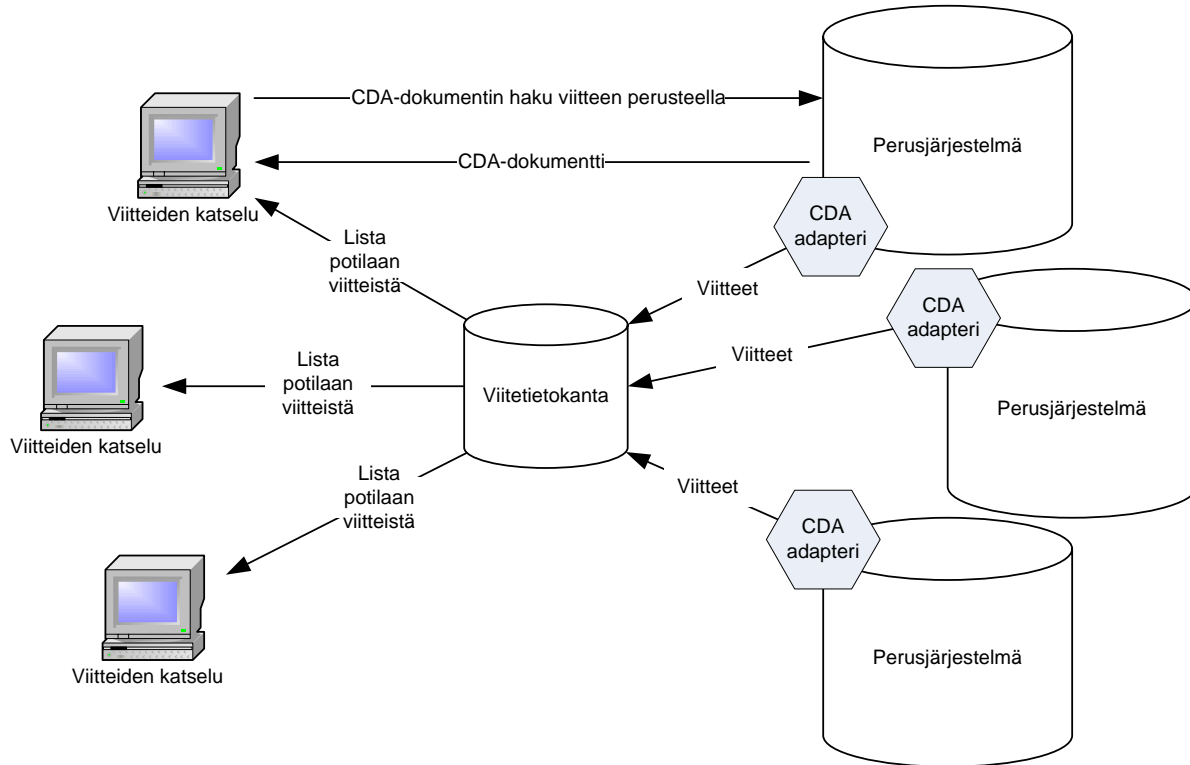
HL7v3 Transport Profile esittää, että HL7v3-määrittymiä toteuttavat ja käyttävät osapuolet noudattaisivat WS-I Basic Profile 1.0:n mukaisia SOAP-sanomia. Lisäksi profiili esittää, miten vuorovaikutuksia voidaan ketjuttaa peräkkäin WS-Addressing -määrittymisen avulla. Tarvittaessa profiilissa suositellaan myös käytettävän WS-ReliableMessaging -määrittymistä, jos sanomien välityksessä tarvitaan HTTP:tä luotettavampaa viestien välityskanavaa.

HL7 Finland on valmistanut suomalaisten terveydenhuoltoalan organisaatioiden ja sovelluskehitysyritysten käytettäväksi erillisen HL7v3-implementointiooppaan (HL7V3FI).

7.3 HL7 Finland Open CDA-adapterikäytännöt

HL7 Finlandin Open CDA-hankkeen puitteissa kehitetyt Open CDA-adapterimäärittymiset (HL7FICDA) käyttävät SOAP-sanomakehystä CDA R2-dokumenttien (Clinical Document Architecture) ja niiden viitteiden siirtämiseen.

Adapterit mahdollistavat potilaskertomus-lomakkeiden viitteiden siirtämisen standardisoitua SOAP-sanomia käyttäen keskitettyyn tietovarastoon, josta niitä voidaan hakea myöhemmin tarkasteltaviksi. Viitteen perusteella voidaan halutessa noutaa viitattu CDA R2-dokumentti (esimerkiksi potilaskertomus-lomake) alkuperäisestä perusjärjestelmästä (kuva 13). Viitteiden selaamiseen ja viitattujen dokumenttien katseluun on rakennettu myös erillinen web-sovellus.



Kuva 13. CDA-adapterien tuottamien viitteiden käyttö

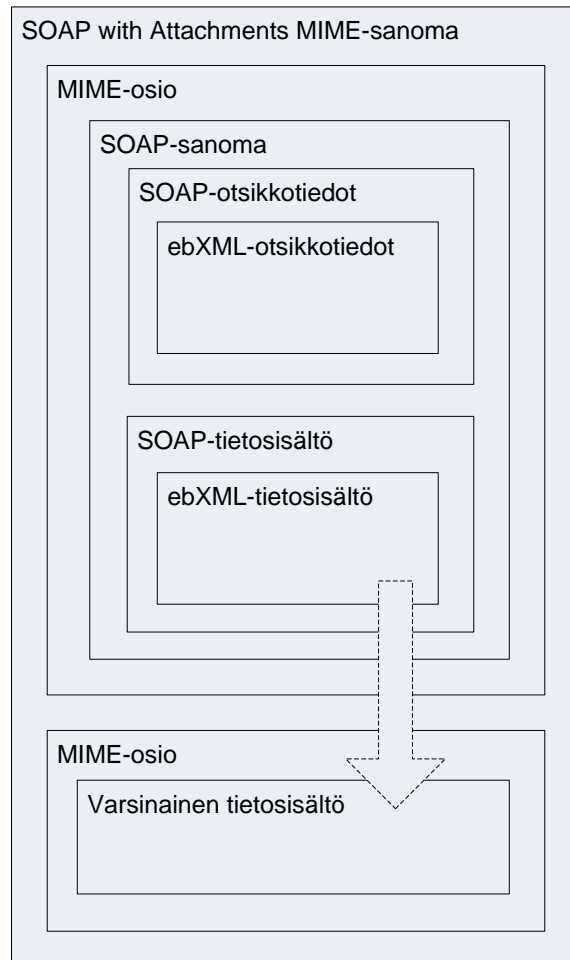
CDA-adapterien tuottamissa SOAP-sanomissa otsikkotiedoissa ilmaistaan adapterin perusjärjestelmän tunniste ja kohdejärjestelmän tunniste (viitetietokanta). Lisäksi otsikkotietoihin sisällytetään muuta metatietoa, kuten aikaleima ja sanoman tunniste. Näiden metatietojen esittämisessä ei kuitenkaan ole käytetty WS-määrittymiä, vaan tietojen esittäminen tapahtuu ebXML-tyylisillä elementeillä (kts kappale 7.4). Itse CDA-dokumentin viite esitetään SOAP-sanoman runko-osiossa.

Open CDA-adapterimäärittymiset sisältävät myös SOAP-sanomamäärittymiset, joiden avulla voidaan hakea viitattu CDA-dokumentti katselua varten. CDA-dokumentti siirretään SOAP-sanoman runko-osion sisällä. Lisäksi adapterimäärittymiset esittävät kuittausanomat, joilla siirrettyjen viitteiden ja katseluun haettujen CDA-dokumenttien siirtosanoman kuitataan.

7.4 ebXML

ebXML (EBXML) on OASIS-konsortion kehittämä XML-kieleen pohjautuva standardi, joka on tarkoitettu kommunikointikieleksi elektroniseen kaupankäyntiin. ebXML käsittää myös laajemmassa mittakaavassa kokonaisen arkkitehtuurin, jolla yritykset ja sovellukset voivat tarjota tai etsiä palveluita yhteisistä ebXML-rekistereistä.

ebXML-kieliset sanomat siirretään SOAP-kääreen sisällä, mutta lisäksi ne koodataan MIME Multipart-tyyppisiksi sanomiksi. Käytännössä tämä tarkoittaa, että ebXML-kieliset sanomat koodataan samalla tavalla kuin sähköposti ja sen avulla lähetettävät liitetiedostot. Kuvassa 14 on näkyvissä ebXML-kielisen sanoman rakenne.



Kuva 14. ebXML-sanoman rakenne

ebXML:n määrittämistä otsikkotiedoista löytyy muun muassa hyvin saman sisältöiset määrittymiset, kuin mitä WS-Addressing (kts kappale 5.1.4) tarjoaa. ebXML:ää ei kuitenkaan pidetä varsinaisena operatiivisten web-sovelluspalvelujen toteuttamiskielenä, vaan enemmänkin yritysten väliseen tietoliikenteeseen sopivana (esim. tietojen synkronointi).

8 Teknisiä suunnittelukäytäntöjä

8.1 Web-sovelluspalvelun kutsumekanismit

8.1.1 Yksisuuntainen palvelukutsu

Yksisuuntainen palvelukutsu on kahden SOAP-päätepisteen välinen kutsu, jossa SOAP-sanoman lähetetään vastaanottajalle, ja vastaanottaja ei palauta SOAP-muotoista vastausta, ainoastaan tyhjän HTTP-kuittausviestin.

Lähetetty SOAP-sanoma voi kuitenkin sisältää otsikkotietoja, joiden avulla web-sovelluspalvelu pystyy lähettämään myöhemmin SOAP-vastaussanoman alkuperäisen sanoman lähettäjälle. Tämä voidaan toteuttaa WS-Addressing -määrittelyksen avulla (kts kappale 5.1.4). Tällöin web-sovelluspalvelu toimii itse yksisuuntaisen palvelukutsun lähettäjänä, mutta se tulkitaan alkuperäisen SOAP-sanoman asynkroniseksi SOAP-vastaussanomaksi.

8.1.2 Kaksisuuntainen synkroninen palvelukutsu

Kaksisuuntainen synkroninen palvelukutsu voidaan käsittää toimivan hyvin samalla tavalla kuin perinteisten ohjelmointikielten funktio-kutsut: SOAP-kutsussa annettujen parametrien pohjalta palautetaan SOAP-vastaussanoma, joka siirretään HTTP-kuittausviestin sisällä.

Kaksisuuntainen palvelukutsu on yleisin SOAP-sovelluspalveluiden kutsumekanismi. Tällöin SOAP-palvelukutsulle on määriteltä XML-muotoisena sekä kutsu- että vastaussanomaa.

8.1.3 Kaksisuuntainen asynkroninen takaisinkutsu

Kaksisuuntaisessa asynkronisessa palvelukutsussa kutsun vastaanottaja palauttaa välittömästi tyhjän HTTP-kuittausviestin. Lopullinen SOAP-paluuviesti lähetetään erillisessä HTTP-viestissä, kun prosessointi on saatu päätökseen. Asynkroninen takaisinkutsu on teknisesti vaikeampi toteuttaa kuin synkroninen liikenne, sillä myös alkuperäisen SOAP-palvelukutsun lähettäjän on pystyttävä ottamaan vastaan ja prosessoimaan SOAP-vastaussanoma myöhemmin.

8.2 Yksinkertaisen SOAP-sovelluspalvelun laatiminen

Service API on laatinut erillisen dokumentin WSDL:n hyödyntämisestä web-sovelluspalveluja laadittaessa ja julkistettaessa (WSSERAPI).

SOAP-sovelluspalvelun laatiminen voi tapahtua periaatteessa kolmella tavalla:

- Julkistetaan olemassa oleva paikallinen valmiiksi rakennettu ohjelmakirjasto WSDL:n avulla SOAP-sovelluspalveluna
- Luodaan palvelusovelluksen runko olemassa olevan kolmannen osapuolen tekemän WSDL-esityksen pohjalta. Tämän jälkeen toteutetaan palvelusovelluksen sovelluslogiikka samalla tavalla kuin paikallisen ohjelmakirjaston rakentaminen.

- Laaditaan SOAP-sovelluspalvelu ilman WSDL-esitystä, jos sellaista ei esimerkiksi ole saatavilla tai sovelluspalvelun rajapinta on kuvattu jotain muuta tekniikkaa käyttäen.

Käytännössä WSDL:n käyttö on suositeltavaa, koska siitä generoidut rajapinnat tekevät aina varmasti oikean muotoisia SOAP-sanomia. Tämä johtuu siitä, että sovellusohjelmoija ei itse pääse vaikuttamaan suoraan SOAP-sanomien muotoiluun.

Useimmat nykyaikaiset sovelluskehittimet tukevat kaikkia edellä mainittuja tapoja. Käytännössä sovellusohjelmoija voi yleensä keskittyä sovelluslogiikan kirjoittamiseen ja sovelluskehittin hoitaa WSDL/SOAP-tekniikoihin liittyvät tekniset yksityiskohdat automaattisesti ja näkymättömästi.

8.3 Web-sovelluspalvelujen yksikkötestaaminen

Web-sovelluspalvelujen yksikkötestaaminen tarkoittaa yhden web-sovelluspalvelun yhden metodin testaamista kerrallaan. Etuna tässä on testitapausten suorittamisen ja lopputuloksen tarkistamisen yksinkertaisuus. Haittana on testien keskittyminen niin pieneen kokonaisuuden osaseen, että yksittäisestä testitapauksesta saatavan tuloksen perusteella ei voida sanoa koko arkkitehtuurin tai muun laajemman prosessin toiminnasta välttämättä mitään. Yksikkötestaamisesta koskee myös samat ongelmat kuin muutakin testaamista: ”Oikeiden” testitapausten laatiminen tai testitapausten näennäinen satunnaisuus voi olla hyvin vaikea toteuttaa.

Yksikkötestaamisessa on myös hyvin vaikea testata sovelluspalvelun toimintaa pitkän ajan kuluessa, esimerkiksi mahdollisten muistivuotojen varalta.

Web-sovelluspalvelujen yksikkötestaamista kuitenkin suositellaan, sillä testitapausten huolellinen rakentaminen voi helpottaa myös samalla kokonaisvaltaisen SOA-arkkitehtuurin laatimista. Lisäksi tällöin yksikkötestaaminen soveltuu myöhemmin web-sovelluspalvelujen regressiotestaamiseen.

8.4 Prosessien tekninen mallintaminen

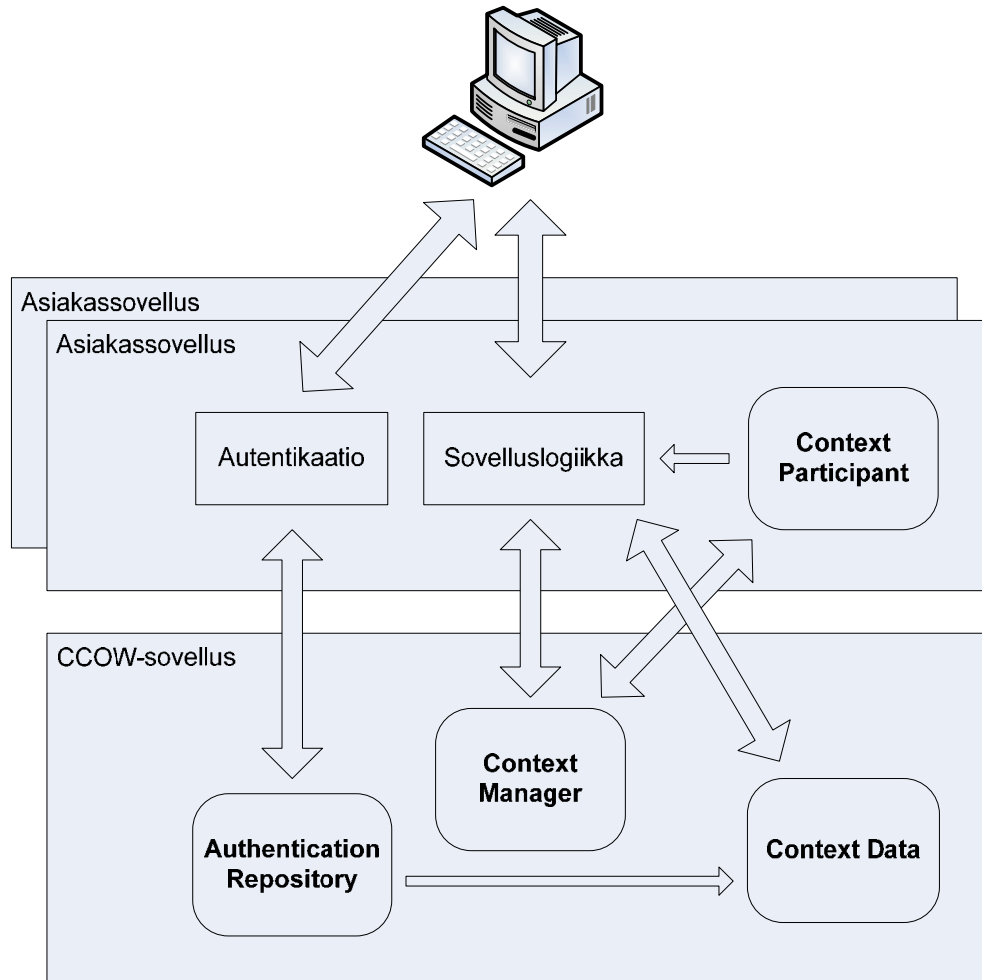
SerAPI on julkaissut dokumentin prosessien tekniseen mallintamiseen ja suorittamiseen tarkoitettua BPEL-määrittymiskielestä (SERAPIBPEL).

8.5 Liiketoiminnallisen yhteistoiminnan tekninen mallintaminen

SerAPI on julkaissut dokumentin orkestraation tekniseen mallintamiseen tarkoitettua WSCDL-kielestä (SERAPIWSCDL). WS-CDL -kieltä ei ole tarkoitettu suoritettavaksi kieleksi, vaan sen avulla voidaan kuvata orkestraatio ja sen osapuolten roolit ja vastuut.

8.6 Esimerkki web-sovelluspalveluiden yhteistoiminnasta - CCOW

SerAPI on toteuttanut osana web-sovelluspalvelujen yhteistoiminnan tutkimusta esimerkkitoteutuksen Clinical Context Object Workgroup -standardista. Standardi ei sisällä teknistä määrittystä WSDL/SOAP -toteutukselle, mutta WSDL-rajapinnat toteutetuille CCOW-rajapinnoille on laadittu osana esimerkkitoteutusta (SERAPICCOW). Kuva 15 esittää CCOW-toteutuksen yleisarkkitehtuuria.



Kuva 15. CCOW-toteutuksen yleisarkkitehtuuri

CCOW-toteutus ja siihen liittyvät esimerkkisovellukset täyttävät SOA-arkkitehtuurin tunnusmerkit: CCOW-rakenneosat (AuthenticationRepository, ContextManager, ContextData ja ContextParticipation) on toteutettu itsenäisinä web-sovelluspalveluina, joita kuvataan abstraktilla tasolla WSDL-määrittysten avulla.

CCOW-toteutuksen sovelluspalvelut eivät ole tilattomia, vaan niiden yhteistoiminta ja palvelujen tilan säilyvyys perustuu asiakasprosessille annettaviin kuponkeihin, eikä esimerkiksi kappaleessa 5.1.4 kuvatun WS-Addressing -määrittymisen käyttöön. Kontekstikupongin avulla esimerkiksi asiakasprosessi voi asettaa nimi/arvo -pareja yhteiseen kontekstiin, ja kuponki tunnistaa tällöin voimassa olevan kontekstiin liittyvän transaktion. SerAPI:n CCOW-esimerkkitoiteutuksen kotisivu sisältää tarkemmat sekvenssikaaviot palvelujen toiminnasta.

9 Yhteenveto

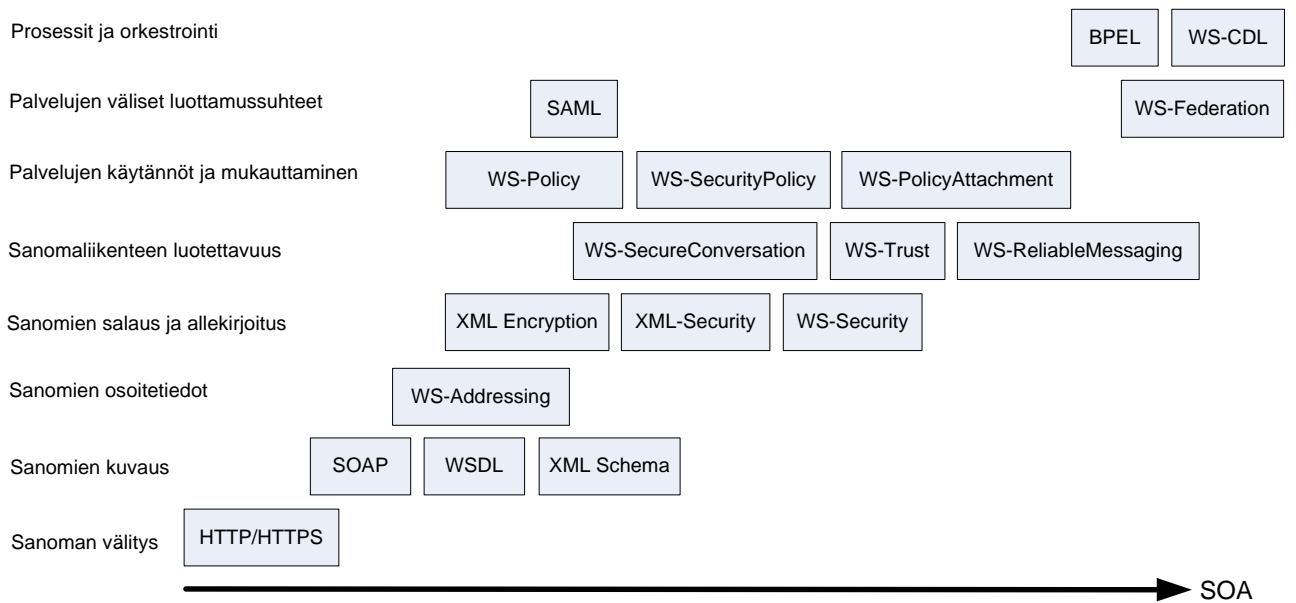
Sovelluskehittäjien tarkoitus on yleisesti piilottaa mekaaniset työvaiheet ja helpottaa sovelluksien osien yhteen liittämistä rajapintojen abstraktion kautta. SOA-pohjaisten arkkitehtuuriratkaisujen laatimiseen soveltuvissa sovelluskehittäjissä tämä tehtävä on erityisen vaikea, sillä:

- SOA-ratkaisujen laatimiseen käytettävät rajapinnat ovat jo valmiiksi hyvin abstrakteja. Tästä ovat esimerkkeinä tässä dokumentissa viitatu BPEL-kieli ja eri WS-määrittymiset.
- Mekaanisia työvaiheita on usealla eri tasolla sovellusarkkitehtuurissa. Esimerkiksi SOAP-sanomien koostaminen matalalla tasolla ja prosessien yhteistoiminnan ohjaaminen korkealla tasolla.
- Yhtenäisten täsmällisten standardien puuttuminen ja jatkuva kehitys olemassa olevien kesken vaikeuttaa huomattavasti täsmällisen yhteistoiminnallisuuden aikaansaamista sovelluskehitys välineiden avulla.

Yhteisenä tekijänä SOA-arkkitehtuurin määrittämisessä on XML-kieli, jolla kuvataan prosesseja, orkestraatiota ja sanomatason yhteistoimintaa. XML on myös yhteinen tekijä SOA-arkkitehtuuria rakennettaessa sovelluskehittäjien avulla. Käytännössä esimerkiksi määriteltäessä liiketoimintaprosessia sovelluskehittäjä voi tarjota graafisen suunnittelu ympäristön prosessin määrittämiseen. Lopputuloksena on BPEL-kielinen prosessikuvaus, joka on luonnollisesti XML-muotoinen.

Kehitysvälineet voivat myös sisältää ohjattuja toimintoja esimerkiksi WS-määrittymisten käyttöönottoon sovelluspalvelujen välillä. Tällöin ohjattu toiminto voi laatia kaiken tarvittavan ohjelmakoodin WS-määrittymisen vaatiman toiminnallisuuden saavuttamiseksi, tai pelkästään ottaa käyttöön ulkopuolisen ohjelmakirjaston joka sisältää toiminnallisuuden. Käytännössä WS-määrittymisten tai prosessi- ja orkestraatiomäärittymiskielten yleis-luontoisuudesta johtuen todellisen yhteistoiminnallisuuden aikaansaaminen eri valmistajien toteutusten välillä vaatii erittäin todennäköisesti tapauskohtaista räätälöintiä, varsinkin siirryttäessä kohti entistä heterogeenisempiä ympäristöjä.

Kuvassa 16 on yritetty mahduttaa samaan kuvaan tässä dokumentissa käsiteltyjä ja viitattuja standardeja ja määrittymiä, joita sovellettaessa voidaan katsoa siirryttävän kohti SOA-pohjaisia sovellusarkkitehtuureja. Käytännössä kuvan esittämä yhteenveto on varsin pelkistetty ja ei esimerkiksi kerro yksittäisten määrittymisten hyödyntämisestä usealla tasolla.



Kuva 16. Kohti SOA-arkkitehtuureja

10Lähteet

SERAPIBPEL	SerAPI. <i>BPEL-katsaus V1.0</i> http://www.uku.fi/tike/his/serapi/yhteys/selvitykset/BPEL_katsaus.doc Maaliskuu 2005.
SAML	OASIS. <i>SAML V2.0 OASIS Standard specification set.</i> http://docs.oasis-open.org/security/saml/v2.0/saml-2.0-os.zip Toukokuu 2007.
UDDI	OASIS. <i>UDDI Specifications</i> http://www.oasis-open.org/committees/uddi-spec/doc/tcpspecs.htm Toukokuu 2007.
CCOW	HL7 Australia. <i>CCOW Resources Page</i> http://www.hl7.org.au/CCOW.htm Marraskuu 2006.
SERAPICCOW	SerAPI. <i>CCOW example implementation using WSDL/SOAP</i> http://kettinki.uku.fi/CCOW Marraskuu 2006.
EBXML	OASIS. ebXML web site Marraskuu 2006. http://www.ebxml.org/
HL7FICDA	Health Level Seven Finland. <i>Avoimet Rajapinnat v 1.1.2</i> Helmikuu 2003. http://virtual.vtt.fi/virtual/hl7/cda/02-avoimet-rajapinnat/cda-adapteri-v1.12.zip
HL7TP	Health Level Seven, Inc. <i>HL7v3 Standard: Transport Specification - Web Services Profile, Release 2.</i> September 2006 Ballot. Elokuu 2006. http://www.hl7.org/v3ballot/html/infrastructure/transport/transport-wsprofiles.htm
HL7V3FI	HL7 Finland. <i>HL7 Finlandin suositus V3-viestien käyttöönottoon</i> Open CDA 2006. Helmikuu 2006. http://virtual.vtt.fi/virtual/hl7/cda/05-open-cda2006/v3-messaging.zip
WSIATT	WS-I. <i>Attachments Profile.</i> WS-I Final Material. Huhtikuu 2006. http://www.ws-i.org/Profiles/AttachmentsProfile-1.0.html
WSISATT	WS-I. <i>Deliverables from the Sample Applications Working Group</i> Toukokuu 2006 http://www.ws-i.org/deliverables/workinggroup.aspx?wg=sampleapps
WSSER	SerAPI. <i>Web-sovelluspalveluiden teknisiä suosituksia.</i> Marraskuu 2005. http://www.uku.fi/tike/his/serapi/mater/WS-suositukset_v1.doc
WS-I	<i>Web Services Interoperability Organization</i> http://www.ws-i.org

WSIBP	WS-I. <i>WS-I Basic Profile, version 1.1.</i> WS-I Basic Profile Final Material. Huhtikuu 2006. http://www.ws-i.org/Profiles/BasicProfile-1.1.html
WSIBSP	WS-I. <i>WS-I Basic Security Profile, version 1.0.</i> Final Material. Maaliskuu 2007. http://www.ws-i.org/Profiles/BasicSecurityProfile-1.0.html
WSIUUSE	WS-I. <i>WS-I Usage Scenarios, version 1.01</i> Final Specification. Joulukuu 2003 http://www.ws-i.org/SampleApplications/SupplyChainManagement/2003-12/UsageScenarios-1.01.pdf
WSISERAPI	<i>Yhteenveto WS-I:n kehittämistä määrittymisistä.</i> Kesäkuu 2006 http://www.uku.fi/tike/his/serapi/yhteys/selvitykset/WS-I_Yhteenveto_v1.2.doc
XMLSIG	W3C. <i>XML-Signature Syntax and Processing.</i> W3C Recommendation. Helmikuu 2002 http://www.w3.org/TR/xmlsig-core/
WSSEC	OASIS. <i>Web Services Security: SOAP Message Security 1.1 (WS-Security 2004)</i> OASIS Standard Specification. Helmikuu 2006 http://www.oasis-open.org/committees/download.php/16790/wss-v1.1-spec-os-SOAPMessageSecurity.pdf
WSC	OpenNetwork, Layer 7, Computer Associates, Microsoft, Verisign, IBM et al. <i>Web Services Secure Conversation (WS-SecureConversation)</i> Revised public draft. Toukokuu 2004 http://specs.xmlsoap.org/ws/2004/04/sc/ws-secureconversation.pdf
SERAPIWSCDL	SerAPI. <i>WS-CDL, versio 0.9</i> http://www.uku.fi/tike/his/serapi/yhteys/selvitykset/WS-CDL-katsaus.doc
WSA	W3C. <i>Web Services Addressing 1.0 - Core</i> W3C Recommendation. May 2006 http://www.w3.org/TR/ws-addr-core/
WSF	BEA, BMC Software, CA Inc., IBM et al. <i>Web Services Federation Language (WS-Federation)</i> Version 1.1. Joulukuu 2006 http://download.boulder.ibm.com/ibmdl/pub/software/dw/specs/ws-fed/WS-Federation-V1-1B.pdf
WSP	VeriSign, Microsoft, Sonic Software, IBM, SAP. <i>Web Services Policy Framework (WS-Policy)</i> Public draft release. Syyskuu 2004 http://specs.xmlsoap.org/ws/2004/09/policy/ws-policy.pdf
WSPA	W3C. <i>Web Services Policy 1.2 - Attachment (WS-PolicyAttachment)</i> W3C Member Submission. Huhtikuu 2006 http://www.w3.org/Submission/WS-PolicyAttachment/
WSRM	BEA, Microsoft, IBM, TIBCO. <i>Web Services Reliable Messaging Protocol (WS-ReliableMessaging)</i>

	Helmikuu 2005 http://specs.xmlsoap.org/ws/2005/02/rm/ws-reliablemessaging.pdf
WSRMP	SerAPI. <i>WS-ReliableMessaging -prototyypitoteutus</i> SerAPI-projektin tuotoksia. Kesäkuu 2006. http://www.uku.fi/tike/his/serapi/yhteys/selvitykset/wsrmp/index.html
WSSP	Microsoft, VeriSign, IBM, RSA. <i>Web Services Security Policy Language (WS-SecurityPolicy)</i> Public consultation draft release. Heinäkuu 2005. http://specs.xmlsoap.org/ws/2005/07/securitypolicy/ws-securitypolicy.pdf
WST	OpenNetwork, Layer 7, Computer Associates, Microsoft, VeriSign, IBM et al. <i>Web Services Trust Language (WS-Trust)</i> Initial public draft release. Helmikuu 2005 http://specs.xmlsoap.org/ws/2005/02/trust/WS-Trust.pdf

Liite 1: WS-I profiilien standardit

Web Services Interoperability Organization (WS-I) on avoin organisaatio, joka on sitoutunut edistämään web-sovelluspalveluiden yhteentoimivuutta (interoperability) eri alustoilla, käyttöjärjestelmissä ja ohjelmointikielillä toteutetuissa ympäristöissä. Organisaatio tarjoaa ohjeistusta, suositeltavia käytäntöjä ja tukea yhteentoimivien web-sovelluspalveluiden kehittämiseksi. (WS-I 2006a) WS-I ei ensisijaisesti pyri kehittämään web-sovelluspalveluiden standardeja, vaan auttamaan ja ohjeistamaan eri yritysten ja organisaatioiden työskentelyä niiden hyväksikäyttämisessä.

Organisaatio on perustettu helmikuussa 2002, jonka jälkeen siihen on liittynyt yli 170 yritystä (WS-I 2004a). WS-I:n jäseniin kuuluu kaiken kokoisia ohjelmistotoimittajia, yritysasiakkaita ja monia muita web-sovelluspalveluista kiinnostuneita (WS-I 2006b).

WS-I:ssä työskennellään työryhmissä, joita ovat:

- Basic Profile Working Group (WS-I 2006i)
- Basic Security Profile Working Group (WS-I 2004d)
- Requirements Gathering Working Group (WS-I 2005b)
- Sample Applications Working Group (WS-I 2004f)
- Testing Tools Working Group (WS-I 2004e)
- XML Schema Work Plan Working Group (WS-I 2004g)

WS-I tuottaa mm. web-sovelluspalveluihin liittyviä profiileita (ks. luvut 2-5), testaustyökaluja (ks. luku 6) ja esimerkkitoimituksia (ks. luku 7). Profiilit asettavat kohteelleen vaatimuksia, tarkennuksia ja suosituksia, joiden perusteella voidaan osoittaa kohteen yhdenmukaisuus profiiliin kanssa (WS-I 2003d).

1 Basic Profile

Basic Profile kuvaa, kuinka web-sovelluspalvelumäärittelyksiä tulisi käyttää yhdessä yhteentoimivien web-sovelluspalvelujen kehittämiseksi. (WS-I 2004h)

Version 1.1 (WS-I 2006c) luomiseksi Basic Profile 1.0 (WS-I 2004b) on rakennettu uudelleen siten, että kaikki kirjekuoren (envelope) esittämiseen ja esitysmuotoon liittyvät vaatimukset on siirretty omaan Simple SOAP Binding Profile 1.0 -profiiliin (WS-I 2004c). Uuden rakenteen ansiosta Basic Profile 1.1 on helposti liitettävissä mihin tahansa kirjekuoren esittämisen määrittelevään profiiliin, mukaan lukien Simple SOAP Binding Profile 1.0 ja Attachments Profile 1.0 (WS-I 2006d). (WS-I 2004h)

Basic Profile 1.1 ja Simple SOAP Binding Profile 1.0 yhdessä korvaavat Basic Profile 1.0:n. Attachments Profile 1.0 lisää SOAP Messages with Attachments -tuen ja sitä on tarkoitus käyttää yhdessä Basic Profile 1.1:n kanssa. (WS-I 2006c)

Basic Profile 1.1:een on sisällytetty viitteinä seuraavien määrittelysten vaatimukset, lukuun ottamatta profiilissa korvattuja vaatimuksia: (WS-I 2006c)

- Simple Object Access Protocol (SOAP) 1.1 (W3C 2000b)
- RFC2616: Hypertext Transfer Protocol -- HTTP/1.1 (IETF 1999b)

- RFC2965: HTTP State Management Mechanism (IETF 2000a)
- Extensible Markup Language (XML) 1.0 (Second Edition) (W3C 2000a)
- Namespaces in XML 1.0 (W3C 1999)
- XML Schema Part 1: Structures (W3C 2001b)
- XML Schema Part 2: Datatypes (W3C 2001c)
- Web Services Description Language (WSDL) 1.1 (W3C 2001a)
- UDDI Version 2.04 API Specification, Dated 19 July 2002 (OASIS 2002b)
- UDDI Version 2.03 Data Structure Reference, Dated 19 July 2002 (OASIS 2002a)
- UDDI Version 2 XML Schema (OASIS 2001)
- RFC2818: HTTP Over TLS (IETF 2000b)
- RFC2246: The TLS Protocol Version 1.0 (IETF 1999a)
- The SSL Protocol Version 3.0 (Freier ym. 1996)
- RFC2459: Internet X.509 Public Key Infrastructure Certificate and CRL Profile (IETF 1999c)

2 Simple SOAP Binding Profile

SOAP 1.1 määrittelee XML-rakenteen lähetettävälle viesteille, joita kutsutaan kirjekuoriksi (envelope). Simple SOAP Binding Profile 1.0 valtuuttaa rakenteen ja asettaa tietyt rajoitteet sen käyttöön. (WS-I 2004h)

Simple SOAP Binding Profile 1.0 perustuu Basic Profile 1.0:n vaatimuksiin, jotka liittyvät kirjekuoren (envelope) esittämiseen ja esitysmuotoon viestissä. Nämä vaatimukset on eriytetty Basic Profile 1.1:sta, jotta muita profiileja voitaisiin käyttää sen kanssa. (WS-I 2004c)

Simple SOAP Binding Profile 1.0 ja Basic Profile 1.1 yhdessä korvaavat Basic Profile 1.0:n. (WS-I 2004c)

Simple SOAP Binding Profile 1.0:aan on sisällytetty viitteinä seuraavien määrittelysten vaatimukset, lukuun ottamatta profiilissa korvattuja vaatimuksia: (WS-I 2004c)

- Simple Object Access Protocol (SOAP) 1.1 (W3C 2000b)
- Extensible Markup Language (XML) 1.0 (Second Edition) (W3C 2000a)
- Namespaces in XML 1.0 (W3C 1999)
- RFC2616: Hypertext Transfer Protocol -- HTTP/1.1 (IETF 1999b)
- WSDL 1.1 (W3C 2001a), Section 3

3 Attachments Profile

Attachments Profile 1.0 täydentää Basic Profile 1.1:n lisäämällä tuen SOAP Messages with Attachments -määrittelykseen perustuville SOAP-viestien lisäosille, kuten MPEG- ja JPEG-tiedostoille tai lääketieteellisille kuville. (WS-I 2006d)

SOAP Messages with Attachments määrittelee MIME multipart/related -rakenteen lisäosien paketoimille SOAP-viesteihin. (WS-I 2006d)

Tämä profiili lisää SOAP Messages with Attachments ja MIME bindings -tuen ja sitä on tarkoitus käyttää yhdessä Basic Profile 1.1:n kanssa. (WS-I 2006d)

Attachments Profile 1.0:aan on sisällytetty viitteinä seuraavien määrittymisten vaatimukset, lukuun ottamatta profiilissa korvattuja vaatimuksia: (WS-I 2006d)

- SOAP Messages with Attachments (W3C 2000c)
- Extensible Markup Language (XML) 1.0 (Second Edition) (W3C 2000a)
- Namespaces in XML 1.0 (W3C 1999)
- RFC2557 MIME Encapsulation of Aggregate Documents, such as HTML (MHTML) (IETF 1999d)
- RFC2045 Multipurpose Internet Mail Extensions (MIME) Part One: Format of Internet Message Bodies (IETF 1996a)
- RFC2046 Multipurpose Internet Mail Extensions (MIME) Part Two: Media Types (IETF 1996b)
- RFC2392 Content-ID and Message-ID Uniform Resource Locators (IETF 1998)
- WSDL 1.1 (W3C 2001a), Section 5.0

4 Basic Security Profile

Basic Security Profile 1.0 -määrittymis (WS-I 2006e) on vielä luonnosasteella. Se tarjoaa ohjeita WS-Security:n käyttöön sekä erilaisia security token -muotoja (security token esittää joukon vaatimuksia). (WS-I 2006j)

Profiiliin on sisällytetty viitteinä seuraavien määrittymisten vaatimukset, lukuun ottamatta profiilissa korvattuja vaatimuksia: (WS-I 2006e)

- RFC 2818: HTTP over TLS (IETF 2000b)
- RFC 2246: The TLS Protocol Version 1.0 (IETF 1999a)
- The SSL Protocol Version 3.0 (Freier ym. 1996)
- Web Services Security: SOAP Message Security 1.0 (WS-Security 2004), OASIS Standard 200401, March 2004 (OASIS 2004a)
- Web Services Security: SOAP Message Security 1.0 (WS-Security 2004), Errata 1.0 Committee Draft 200512, December 2005 (OASIS 2005a)
- Basic Profile Version 1.0 (BP1.0) (WS-I 2004b)
- Basic Profile Version 1.0 Errata (WS-I 2005c)
- Basic Profile Version 1.1 (BP1.1) (WS-I 2006c)
- Simple SOAP Binding Profile Version 1.0 (SSBP1.0) (WS-I 2004c)
- XML-Signature Syntax and Processing (W3C 2002a)
- XML Encryption Syntax and Processing (W3C 2002b)
- Web Services Security: UsernameToken Profile 1.0, OASIS Standard 200401, March 2004 (OASIS 2004b)
- Web Services Security: UsernameToken Profile 1.0, Errata 1.0 Committee Draft 200401, September 2004 (OASIS 2004d)
- Web Services Security: X.509 Certificate Token Profile, OASIS Standard 200401, March 2004 (OASIS 2004c)
- Web Services Security: X.509 Token Profile 1.0, Errata 1.0 Committee Draft 200512, December 2005 (OASIS 2005b)
- RFC2459: Internet X.509 Public Key Infrastructure Certificate and CRL Profile (IETF 1999c)
- Information technology "Open Systems Interconnection" The Directory: Public-key and attribute certificate frameworks Technical Corrigendum 1 (ITU 2001)

- Web Services Security: Rights Expression Language (REL) Token Profile 1.0, OASIS Standard: 19 December 2004 (OASIS 2004e)
- Web Services Security: Kerberos Token Profile 1.1, OASIS Standard Specification, 1 February 2006 (OASIS 2006c)
- Web Services Security: SAML Token Profile 1.0, OASIS Standard, 01 Dec. 2004 (OASIS 2004f)
- Attachments Profile Version 1.0 (API.0) (WS-I 2006d)
- Web Services Security: SOAP Messages with Attachments (SwA) Profile 1.1, OASIS Standard, 1 February 2006 (OASIS 2006d)

5 Reliable Secure Profile

Reliable Secure Profile 1.0 -määrittelyksen (WS-I 2006f) kehitys on aloitettu WS-I:ssä. Se tulee tarjoamaan ohjeistuksen OASIS WS-ReliableMessaging 1.1:n (OASIS 2006a) ja OASIS WS-SecureConversation 1.3:n (OASIS 2006b) sekä normatiivisesti viitattujen määrittelysten (esim. IETF:n RFC:t) käyttöön. (WS-I 2006f)

Määrittely tulee liittyvän kokonaisuuteen Basic Profile 1.1:n, Attachments Profile 1.0:n, Simple SOAP Binding Profile 1.0:n sekä Basic Security Profile 1.0:n ja 1.1:n kanssa. Jatkossa myös Basic Profile 1.2 ja 2.0 tulevat liittyvän vastaavasti kokonaisuuteen. (WS-I 2006f)

6 Testityökalut

WS-I:n testaustyökalut (Javalle ja C#:lle) on suunniteltu auttamaan sovelluskehittäjiä määriteltäessä, onko heidän sovelluksensa profiilin ohjeiden mukaisia. (WS-I 2006g) Ne testaavat web-sovelluspalvelutoteutuksia mustalaatikkotestausmenetelmällä. Kohteena on web-sovelluspalvelun ja hyödyntäjäsovellusten välinen vuorovaikutus. (WS-I 2003a)

WS-I testaustyökalut toimivat SOAP-viestinvälitykseen pohjautuvan sovelluskehityksen apuvälineinä. Ne ottavat parametrina WSDL-määrittelyksen ja generoivat testiraportin (HTML-sivun) sen perusteella. Lisäksi niiden avulla voidaan validoida suorituksen aikaisia SOAP-sanomia palvelun ollessa toiminnassa ja etsiä Basic Profilen mukaisia SOAP-palveluita ulkopuolisesta UDDI-rekisteristä ja testata myös niiden WS-I -yhdenmukaisuus. Testiraporttia voi käyttää esimerkiksi vakuutena WS-I:n mukaisesta yhteistoiminnallisuudesta, tosin WS-I irtisanoutuu kaikesta vastuusta testityökalujen käytöstä seuranneista virheistä tai väärinkäytöksistä. Testiraportti pitää sisällään yleisesti tiedon hyväksymisestä tai hylkäämisestä. Jälkimmäisessä tapauksessa liitetään mukaan myös sanallinen kuvaus hylkäämisen perusteista ja missä kohdassa WSDL-tiedostoa virhe löytyi. (WS-I 2003a)

Työkalujen toimintaperiaatteeseen on vaikuttanut niiden integroituvuus sovelluskehittäjiin: Testityökalujen Java-versio voidaan ottaa käyttöön esimerkiksi Oracle JDeveloper 10g -kehitysympäristöön siten, että minkä tahansa projektiin liitetyn WSDL-tiedoston WS-I:n mukainen muotoilu ja yhteentoimivuus voidaan tarkistaa suoraan suunnittelu-ympäristöstä käsin. Testityökalujen C#-versio puolestaan voidaan integroida Microsoft Visual Studio .NET 2003:een (Tools → External Tools → Add).

WS-I Testing Tools 1.0 (WS-I 2003a) on suunniteltu Basic Profile 1.0:n testaukseen. Interoperability Testing Tools 1.1 (WS-I 2005a) on suunniteltu Basic Profile 1.1:n ja Simple SOAP Binding Profile 1.0:n testaukseen. (WS-I 2006g)

6.1 Java-testityökalujen käyttö

WS-I:n Java-testityökalut on saatavilla yhdessä paketissa (WS-I 2003a). Monet Java-sovelluskehitysympäristöt tarjoavat mahdollisuuden integroida ne osaksi kehitysympäristöä, ja tämä helpottaakin niiden käyttöä huomattavasti. Tässä kappaleessa esitetään, miten välineistöjä voi käyttää ilman erillistä sovelluskehitysympäristöä. Esimerkkejä voi käyttää niin Windows- kuin Unix/Linux -pohjaisissa järjestelmissä. Esimerkit olettavat, että WS-I -testityökalupaketti on purettu paikalliselle kovalevyille.

WS-I -testityökalut konfiguroidaan erillisellä XML-tiedostolla, jossa viitataan tarkasteltavaan WSDL:ään ja mitä osia siitä halutaan tarkemmin WS-I -yhteensopivuusraporttiin. Seuraavassa listauksessa on esitelty esimerkkinä testityökalujen ymmärtämä XML-konfiguraatitiedosto.

```
<?xml version="1.0" encoding="UTF-8"?>
<wsi:configuration name="Sample Basic Profile Analyzer Configuration" xmlns:wsi="http://www.ws-i.org/testing/2004/07/analyzerConfig/">
  <wsi:description>
    Esimerkki konfiguraatitiedosto WS-I -testityökalujen käytöstä / Marko Sormunen
  </wsi:description>
  <wsi:verbose>false</wsi:verbose>
  <wsi:assertionResults type="all" messageEntry="true" failureMessage="true"/>

  <!-- Miin WS-I yhteentoimivuusraportti laaditaan -->
  <wsi:reportFile replace="true" location="report.xml">
    <wsi:addStyleSheet href="common/xsl/report.xsl" type="text/xsl"/>
  </wsi:reportFile>

  <!-- Mistä tiedostosta haetaan testimateriaali -->
  <wsi:testAssertionsFile>
    common/profiles/BasicProfile_1.1_TAD.xml
  </wsi:testAssertionsFile>

  <!-- Mistä materiaalista WS-I -yhteentoimivuusraportti laaditaan -->
  <wsi:wSDLReference>
    <!-- WSDL:stä testattava port-elementti -->
    <wsi:wSDLElement type="port" parentElementName="WSDLn_esiittämä_sovelluspalvelu"
      namespace="urn:prefix:WSDL:n_targetNamespace">
      APRServiceSOAPPort
    </wsi:wSDLElement>
    <!-- Viittaus itse testattavaan WSDL:ään -->
    <wsi:wSDLURI>
      wsd/ testattava_tiedosto.wsd
    </wsi:wSDLURI>
  </wsi:wSDLReference>
</wsi:configuration>
```

WS-I -testityökalu voidaan käynnistää seuraavalla komennolla:

```
java -cp java/lib/wsi-test-tools.jar;java/lib/wsd4j.jar;java/lib/xercesImpl.jar;java/lib/uddi4j.jar org.wsi.test.analyzer.BasicProfileAnalyzer -config analyzerConfig.xml
```

Komentoa ajettaessa oletetaan että XML-konfiguraatitiedosto on analyzerConfig.xml -nimisessä tiedostossa, ja komento ajetaan WS-I -testityökalujen juurihakemistosta.

11 Esimerkkisovellukset

Esimerkkisovellus esittelee korkean tason yhteentoimivan esimerkin jakeluketjun hallinnan (Supply Chain Management, SCM) sovelluksesta Basic Profile 1.0:n mukaisten web-sovelluspalveluiden käyttöä esittelevien käyttötapauksien avulla. (WS-I 2006h)

Mallinnettavassa sovelluksessa jälleenmyyjä tarjoaa elektroniikkahyödykkeitä kuluttajille (tyypillinen B2C- eli business to consumer -malli). Hoitaakseen tilaukset jälleenmyyjän täytyy hoitaa varastojen käyttöasteita. Kun jonkin tuotteen määrä laskee tietyn rajan alle, jälleenmyyjän täytyy tilata tuotetta lisää valmistajalta (tyypillinen B2B-malli). Hoitaakseen jälleenmyyjän tilauksen valmistajan täytyy tuottaa loppunutta hyödykettä. Tosielämässä valmistaja tilaisi hyödykkeen osat tavaran-toimittajilta, mutta sovelluksen yksinkertaistamiseksi tämä oletetaan manuaaliseksi prosessiksi, joka hoidetaan faksilla. (WS-I 2003c)

Sovelluksen käyttötapauksista kerrotaan luvussa 8.1, arkkitehtuurista luvussa 8.2 ja käyttöskenaariot luvussa 8.3. Luvussa 8.4 luetellaan yritykset, jotka ovat toteuttaneet SCM-esimerkkisovelluksia.

11.1 Supply Chain Management Use Cases

Supply Chain Management Use Cases 1.0 (WS-I 2003c) esittelee SCM-esimerkkisovelluksen käyttötapaukset. Käyttötapauksien tarkoituksena ei ole esittää tosielämän liiketoimintasovellusta, vaan niiden tarkoituksena on havainnollistaa web-sovelluspalvelua. (WS-I 2006k)

Käyttötapaukset ovat: (WS-I 2003c)

- Hyödykkeiden ostaminen (Purchase Goods)
- Hyödykkeiden paikantaminen (Source Goods)
- Varaston täydentäminen (Replenish Stock)
- Loppuneiden hyödykkeiden toimittaminen (Supply Finished Goods)
- Loppuneiden hyödykkeiden valmistaminen (Manufacture Finished Goods)
- Demon konfigurointi ja suoritus (Configure & Run Demo)
- Tapahtumien kirjaaminen (Log Events)
- Tapahtumien katselu (View Events)

11.2 Supply Chain Management Sample Architecture

Supply Chain Management Sample Application Architecture 1.01 on toteutettu Supply Chain Management Use Cases 1.0:n pohjalta. Se esittää tarkasti SCM-esimerkkisovelluksen teknisen mallin ja toteutuksen. Sen ei ole tarkoitus esittää kaikki tosielämän SCM-sunnittelun ja –toteutuksen erityispiirteitä, vaan esitellä, kuinka Basic Profile 1.0:n mukaisia web-sovelluspalveluja voisi suunnitella, toteuttaa ja ottaa käyttöön. (WS-I 2003b)

11.3 Sample Architecture Usage Scenarios

WS-I Usage Scenarios kuvaa yleiset viestimallit web-sovelluspalveluiden viestienvaihtoon, joita käytetään WS-I:n esimerkkisovelluksien perustana. (WS-I 2006k)

WS-I Usage Scenarios 1.01 esittelee Basic Profile 1.0:n kanssa käytettävät sovellusalueesta riippumattomat skenaariot: one-way, synchronous request/response ja basic callback. Näiden skenaarioiden tarkoituksena on tarjota riittävästi tietoa WS-I:n mukaisten web-sovelluspalveluiden luomiseksi. WS-I:n käyttötapaukset (WS-I Use Cases) käyttävät skenaarioita sovellusten korkean tason määrittelysten mallintamiseen. (WS-I 2003e)

11.4 Sample Application Implementations

Basic Profile 1.0:n mukaisia SCM-esimerkkisovelluksia ovat tehneet seuraavat yritykset: (WS-I 2006h)

- BEA Systems
- Bowstreet
- Corillian
- IBM
- Microsoft
- Nokia
- Novell
- Oracle
- Quovadx
- SAP
- Sun Microsystems

Lähteet

- Freier ym. 1996 Freier A, Karlton P, Kocher P. The SSL Protocol Version 3.0. 18.11.1996.
<http://wp.netscape.com/eng/ssl3/draft302.txt>
- IETF 1996a Freed N, Borenstein N. RFC2045: Multipurpose Internet Mail Extensions (MIME) Part One: Format of Internet Message Bodies. 1996.
<http://www.ietf.org/rfc/rfc2045.txt>
- IETF 1996b Freed N, Borenstein N. RFC2046: Multipurpose Internet Mail Extensions (MIME) Part Two: Media Types. 1996.
<http://www.ietf.org/rfc/rfc2046.txt>
- IETF 1998 Levinson E. RFC2392: Content-ID and Message-ID Uniform Resource Locators. 1998.
<http://www.ietf.org/rfc/rfc2392.txt>
- IETF 1999a Dierks T, Allen C. RFC2246: The TLS Protocol Version 1.0. 1999.
<http://www.ietf.org/rfc/rfc2246.txt>
- IETF 1999b Fielding R, Gettys J, Mogul J, Frystyk H, Masinter L, Leach P, Berners-Lee T. RFC2616: Hypertext Transfer Protocol -- HTTP/1.1. 1999.
<http://www.ietf.org/rfc/rfc2616.txt>
- IETF 1999c Housley R, Ford W, Polk W, Solo D. RFC2459: Internet X.509 Public Key Infrastructure Certificate and CRL Profile. 1999.
<http://www.ietf.org/rfc/rfc2459.txt>
- IETF 1999d Palme J, Hopmann A, Shelness N. RFC2557: MIME Encapsulation of Aggregate Documents, such as HTML (MHTML). 1999.
<http://www.ietf.org/rfc/rfc2557.txt>
- IETF 2000a Kristol D, Montulli L. RFC2965: HTTP State Management Mechanism. 2000.
<http://www.ietf.org/rfc/rfc2965.txt>
- IETF 2000b Rescorla E. RFC2818: HTTP Over TLS. 2000.
<http://www.ietf.org/rfc/rfc2818.txt>
- ITU 2001 International Telecommunication Union (ITU). Information technology - Open Systems Interconnection - The Directory: Public-key and attribute certificate frameworks, Technical Corrigendum 1, Recommendation X.509 (2000) - Corrigendum 1. 10/2001.
http://www.itu.int/dms_pubrec/itu-t/rec/x/T-REC-X.509-200110-I!Cor1!PDF-E.pdf
- OASIS 2001 OASIS. UDDI Version 2 XML Schema. 2001.
http://uddi.org/schema/uddi_v2.xsd
- OASIS 2002a OASIS. UDDI Version 2.03 Data Structure Reference. 19.7.2002.
<http://uddi.org/pubs/DataStructure-V2.03-Published-20020719.htm>

- OASIS 2002b OASIS. UDDI Version 2.04 API Specification. 19.7.2002.
<http://uddi.org/pubs/ProgrammersAPI-V2.04-Published-20020719.htm>
- OASIS 2004a OASIS. Web Services Security: SOAP Message Security 1.0. 15.3.2004.
<http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-soap-message-security-1.0.pdf>
- OASIS 2004b OASIS. Web Services Security: Username Token Profile 1.0. 15.3.2004.
<http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-username-token-profile-1.0.pdf>
- OASIS 2004c OASIS. Web Services Security: X.509 Certificate Token Profile. 15.3.2004.
<http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-x509-token-profile-1.0.pdf>
- OASIS 2004d OASIS. Web Services Security: UsernameToken Profile 1.0, Errata 1.0, Committee Draft 200401. 09/2004.
<http://www.oasis-open.org/committees/download.php/11143/oasis-200401-wss-username-token-profile-1.0-errata-003.pdf>
- OASIS 2004e OASIS. Web Services Security: Rights Expression Language (REL) Token Profile, OASIS Standard. 19.12.2004.
<http://docs.oasis-open.org/wss/oasis-wss-rel-token-profile-1.0.pdf>
- OASIS 2004f OASIS. Web Services Security: SAML Token Profile, OASIS Standard. 1.12.2004.
<http://docs.oasis-open.org/wss/oasis-wss-saml-token-profile-1.0.pdf>
- OASIS 2005a OASIS. Web Services Security: SOAP Message Security 1.0 (WS-Security 2004), Errata 1.0, Committee Draft 200512. 12/2005.
<http://www.oasis-open.org/committees/download.php/16792/oasis-200512-wss-soap-message-security-1.0-errata-005.pdf>
- OASIS 2005b OASIS. Web Services Security: X.509 Token Profile 1.0, Errata 1.0, Committee Draft 200512. 12/2005.
<http://www.oasis-open.org/committees/download.php/16796/oasis-200512x509-token-profile-1.0-errata-005.pdf>
- OASIS 2006a OASIS. WS-Reliable Messaging 1.1, Committee Draft 03. 14.3.2006.
<http://docs.oasis-open.org/ws-rx/wsrn/200602/wsrn-1.1-spec-cd-03.pdf>
- OASIS 2006b OASIS. WS-SecureConversation 1.3. Viitattu 13.6.2006.
http://www.oasis-open.org/committees/tc_home.php?wg_abbrev=ws-sx
- OASIS 2006c OASIS. Web Services Security: Kerberos Token Profile 1.1, OASIS Standard Specification. 1.2.2006.
<http://www.oasis-open.org/committees/download.php/16788/wss-v1.1-spec-os-KerberosTokenProfile.pdf>
- OASIS 2006d OASIS. Web Services Security: SOAP Messages with Attachments (SwA) Profile 1.1, OASIS Standard. 1.2.2006.
<http://www.oasis-open.org/committees/download.php/16672/wss-v1.1-spec-os-SwAProfile.pdf>

- W3C 1999 W3C. Namespaces in XML 1.0. 14.1.1999.
<http://www.w3.org/TR/REC-xml-names/>
- W3C 2000a W3C. Extensible Markup Language (XML) 1.0 (Second Edition), W3C Recommendation. 6.10.2000.
<http://www.w3.org/TR/2000/REC-xml-20001006>
- W3C 2000b W3C. Simple Object Access Protocol (SOAP) 1.1, W3C Note. 8.5.2000.
<http://www.w3.org/TR/2000/NOTE-SOAP-20000508/>
- W3C 2000c W3C. SOAP Messages with Attachments, W3C Note. 11.12.2000.
<http://www.w3.org/TR/2000/NOTE-SOAP-attachments-20001211>
- W3C 2001a W3C. Web Services Description Language (WSDL) 1.1, W3C Note. 15.3.2001.
<http://www.w3.org/TR/2001/NOTE-wsdl-20010315>
- W3C 2001b W3C. XML Schema Part 1: Structures, W3C Recommendation. 2.5.2001.
<http://www.w3.org/TR/2001/REC-xmlschema-1-20010502/>
- W3C 2001c W3C. XML Schema Part 2: Datatypes, W3C Recommendation. 2.5.2001.
<http://www.w3.org/TR/2001/REC-xmlschema-2-20010502/>
- W3C 2002a W3C. XML-Signature Syntax and Processing, W3C Recommendation. 12.2.2002.
<http://www.w3.org/TR/2002/REC-xmlsig-core-20020212/>
- W3C 2002b W3C. XML Encryption Syntax and Processing, W3C Recommendation. 10.12.2002.
<http://www.w3.org/TR/2002/REC-xmlenc-core-20021210/>
- WS-I 2003a WS-I. Testing Tools 1.0. 15.9.2003.
(C#) http://www.ws-i.org/Testing/Tools/2004/01/WSI_Test_CS_01.00.01.0008_bin.zip
(Java) http://www.ws-i.org/Testing/Tools/2004/01/WSI_Test_Java_01.00.01_bin.zip
- WS-I 2003b WS-I. Supply Chain Management Sample Application Architecture Version 1.01, Final Specification. 9.12.2003.
<http://www.ws-i.org/SampleApplications/SupplyChainManagement/2003-12/SCMArchitecture1.01.pdf>
- WS-I 2003c WS-I. Supply Chain Management Use Case Model Version 1.0, Final Specification. 1.12.2003.
<http://www.ws-i.org/SampleApplications/SupplyChainManagement/2003-12/SCMUseCases1.0.pdf>
- WS-I 2003d WS-I. WS-I Profile Conformance Framework Version 1.0, Working Group Draft. 4.11.2003.
<http://www.ws-i.org/schemas/conformanceClaim/ProfileConformance-1.01-WGD.pdf>
- WS-I 2003e WS-I. WS-I Usage Scenarios Version 1.01, Final Specification. 9.12.2003.
<http://www.ws-i.org/SampleApplications/SupplyChainManagement/2003-12/UsageScenarios-1.01.pdf>

- WS-I 2004a WS-I. WS-I Promotes Profiles to Final Material Status. 24.8.2004.
<http://www.ws-i.org/press/pressrelease.aspx?pr=20040824a>
- WS-I 2004b WS-I. Basic Profile Version 1.0, Final Material. 16.4.2004.
<http://www.ws-i.org/Profiles/BasicProfile-1.0-2004-04-16.html>
- WS-I 2004c WS-I. Simple SOAP Binding Profile, Version 1.0, Final Material. 24.8.2004.
<http://www.ws-i.org/Profiles/SimpleSoapBindingProfile-1.0-2004-08-24.html>
- WS-I 2004d WS-I. Working Group Charter: Basic Security Profile, Version 1.8. 19.7.2004.
http://www.ws-i.org/docs/charters/WSBasic_SecurityProfile_Charter1-8.pdf
- WS-I 2004e WS-I. Working Group Charter: Web Services Profile Testing Tools and Material, Version 1.5. 18.2.2004.
http://www.ws-i.org/docs/charters/Test_Charter1-5.pdf
- WS-I 2004f WS-I. Working Group Charter: Web Services Sample Applications. 7.6.2004.
http://www.ws-i.org/docs/charters/WSBasic_Sample_Applications_Charter2-2.pdf
- WS-I 2004g WS-I. Working Group Charter: XML Schema Work Plan, Version 1.1, WS-I Administrative Document. 24.9.2004.
http://www.ws-i.org/docs/charters/Schema_Plan_Charter1-0.pdf
- WS-I 2005a WS-I. Interoperability Testing Tools 1.1. 13.6.2005.
(C#) http://www.ws-i.org/Testing/Tools/2005/06/WSI_Test_CS_Final_1.1.zip
(Java) http://www.ws-i.org/Testing/Tools/2005/06/WSI_Test_Java_Final_1.1.zip
- WS-I 2005b WS-I. Working Group Charter: WS-I Requirements Gathering, WS-I Administrative Document. 8.9.2005.
http://www.ws-i.org/docs/charters/Requirements_Charter1-3.pdf
- WS-I 2005c WS-I. Basic Profile Version 1.0 Errata, Board Approval Draft, Revision: 1.37. 25.10.2005.
<http://www.ws-i.org/Profiles/BasicProfile-1.0-errata-2005-10-25.html>
- WS-I 2006a Web Services Interoperability Organization (WS-I). Viitattu 13.6.2006.
<http://www.ws-i.org/>
- WS-I 2006b WS-I. About Us. Viitattu 13.6.2004.
<http://www.ws-i.org/about/Default.aspx>
- WS-I 2006c WS-I. Basic Profile Version 1.1, Final Material. 10.4.2006.
<http://www.ws-i.org/Profiles/BasicProfile-1.1-2006-04-10.html>
- WS-I 2006d WS-I. Attachments Profile Version 1.0, Final Material. 20.4.2006.
<http://www.ws-i.org/Profiles/AttachmentsProfile-1.0-2006-04-20.html>
- WS-I 2006e WS-I. Basic Security Profile Version 1.0, Working Group Draft. 29.3.2006.
<http://www.ws-i.org/Profiles/BasicSecurityProfile-1.0-2006-03-29.html>
- WS-I 2006f WS-I. Working Group Charter: Reliable Secure Profile 1.0. 21.4.2006.
http://www.ws-i.org/docs/charters/RSP_Charter1-0.pdf
- WS-I 2006g WS-I. Testing Tools. Viitattu 24.5.2006.
<http://www.ws-i.org/deliverables/workinggroup.aspx?wg=testingtools>

- WS-I 2006h WS-I. Sample Applications. Viitattu 24.5.2006.
<http://www.ws-i.org/implementation.aspx>
- WS-I 2006i WS-I. Working Group Charter: Basic Profile 1.2 and 2.0. 30.5.2006.
http://www.ws-i.org/docs/charters/WSBasic_Profile_Charter2-1.pdf
- WS-I 2006j WS-I. Deliverables from the Basic Security Profile Working Group. Viitattu 20.6.2006.
<http://www.ws-i.org/deliverables/workinggroup.aspx?wg=basicsecurity>
- WS-I 2006k WS-I. Deliverables from the Sample Applications Working Group. Viitattu 20.6.2006.
<http://www.ws-i.org/deliverables/workinggroup.aspx?wg=sampleapps>

Liite 2: SAML

Security Assertion Markup Language (SAML) on OASIS-konsortion Security Services Technical Committee -komitean (SSTC) XML pohjainen standardi, jonka avulla voidaan välittää turvallisuustietoja eri entiteettien välillä. SAMLiin kuuluu turvallisuustietojen lisäksi pyyntö/vastaus-protokolla ja säännöt siitä, kuinka näitä tietoja välitetään standardeilla kuljetusprotokollilla. Turvallisuustiedot välitetään *vakuuksina* (assertions), jotka ovat XML:ää. Ne sisältävät yhden tai useamman *lauseen* (statement), joka kohdistuu johonkin subjettiin (henkilö tai ohjelma). Lauseita on kolmenlaista:

- Tunnistus (authentication)
- Ominaisuus (attribute)
- Valtuutuspäätös (authorization decision)

Tunnistuksessa on tieto jonkin subjektin tunnistamisesta tiettyyn aikaan tietyllä menetelmällä. Ominaisuudella voidaan vakuuteen liittää yksityiskohtaista tietoa subjektista, kuten esimerkiksi tieto siitä, että käyttäjä kuuluu tiettyyn etuoikeutettuun luokkaan. Valtuutuspäätös kertoo mitä subjekti on oikeutettu tekemään, esim. ostamaan jokin tietty tuote. SAML antaa myös luoda omia vakuus- ja lausetyyppejä. Vakuudet voidaan digitaalisesti allekirjoittaa.

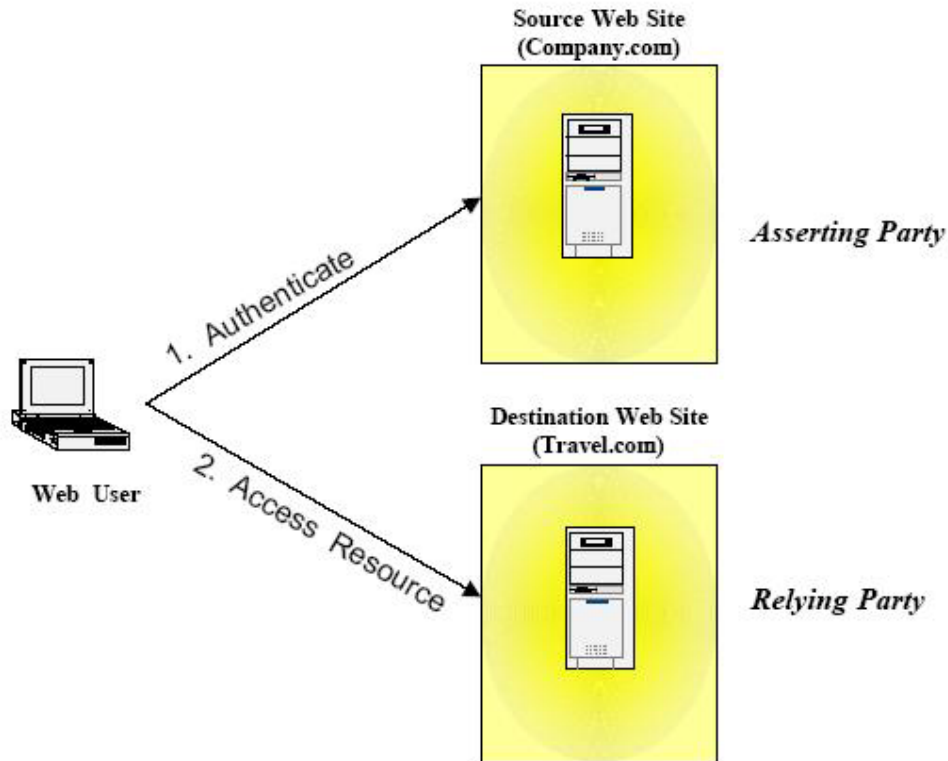
SAMLin idea on, että jostain subjektista (henkilö, järjestelmä) voidaan jonkun luotettavan tahon toimesta antaa vakuuksia, joihin muut verkossa olevat entiteetit pystyvät luottamaan. Tämä tuo mukanaan kaksi käsitettä: vakuuksia antava taho (asserting party) sekä vakuuksista riippuvainen taho (relying party). SAML määrittelee useita mekanismeja luottamuksen luomiseksi vakuuksia antavan tahon ja niistä riippuvaisen tahon välille.

1 Käyttötapaukset

SSTC kehitti eri käyttötapauksia, jolla rajattiin SAMLin vaatimuslistaa. SAMLin näkökulmasta tärkein näistä käyttötapauksista on Web Single Sign-On (SSO). Tämän ongelman ratkaiseminen onkin SAMLin pääasiallinen tavoite.

1.1 Single sign-on

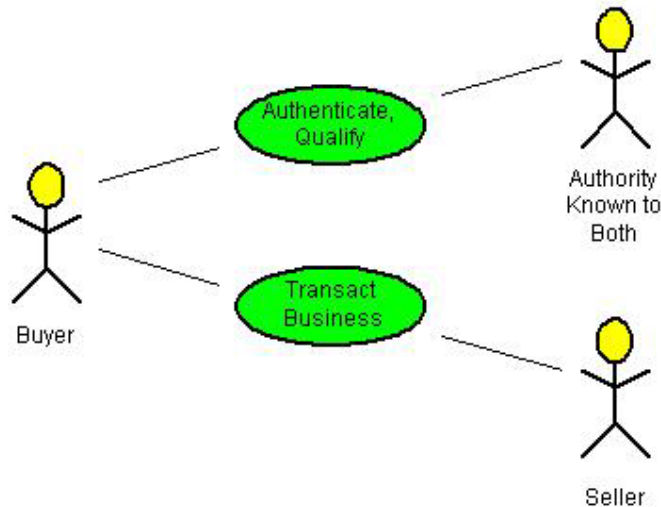
SAML Single sign-on (SSO) avulla käyttäjä voi kirjautua jollekin tietylle verkkoalueelle ja päästä tämän kirjautumisen avulla muillekin verkkoalueille. Tarkemmin sanottuna käyttäjä ei mene alkuperäisen verkkoalueen kautta muille verkkoalueille vaan kirjautumista hoitava verkkoalue lähettää muille vakuuksia. Kuvassa 1. on esimerkkinä tapaus, jossa Company-yhtiön työntekijä pääsee käyttämään Travel-yhtiön suojattuja verkkosivuja kirjaututtuaan ensin oman yhtiönsä verkkoalueelle. Tässä Company-yhtiö on vakuuksia antava taho (asserting party) ja Travel-yhtiö vakuuksista riippuvainen taho (relying party).



Kuva 1: Esimerkki SSO käyttötapauksesta [Hug04].

1.2 Distributed transaction

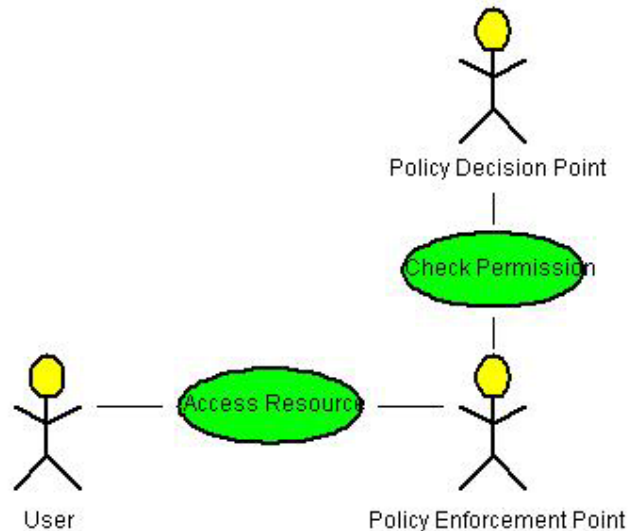
Hajautetussa transaktiossa käyttäjä voi tehdä liiketoimia verkon yli jossain liiketoimipaikassa kirjaututtuaan ensin esimerkiksi oman yhtiönsä sivuilla. SAML antaa mahdollisuuden rajata käyttäjän tekemää transaktiota esimerkiksi sisältämällä vakuuteen lauseen, jossa rajataan käyttäjälle myönnetty summa rahaa.



Kuva 2: Hajautettu transaktio [Mal01].

1.3 Authorization service

Tässä käyttötapauksessa SAMLia voidaan käyttää tarkistamaan käyttäjän valtuudet. Aliluvun 2.2 esimerkkiä mukaillen ostaja voisi ottaa myyjään (kuvassa Policy Enforcement Point) suoraan yhteyttä ilman ennalta tehtyä kirjautumista muualle. Kun ostaja sitten haluaisi ostaa jotain, myyjä voisi SAMLin avulla tarkistaa ostajan valtuudet ottamalla yhteyttä luotettuun kolmanteen osapuoleen (kuvassa Policy Decision Point), joka päättää mitä valtuuksia ostajalle annetaan. Tämä kolmas osapuoli voi olla esimerkiksi ostajan työnantaja.



Kuva 3: Valtuuksien tarkastaminen [Mal01].

2 Vakuudet

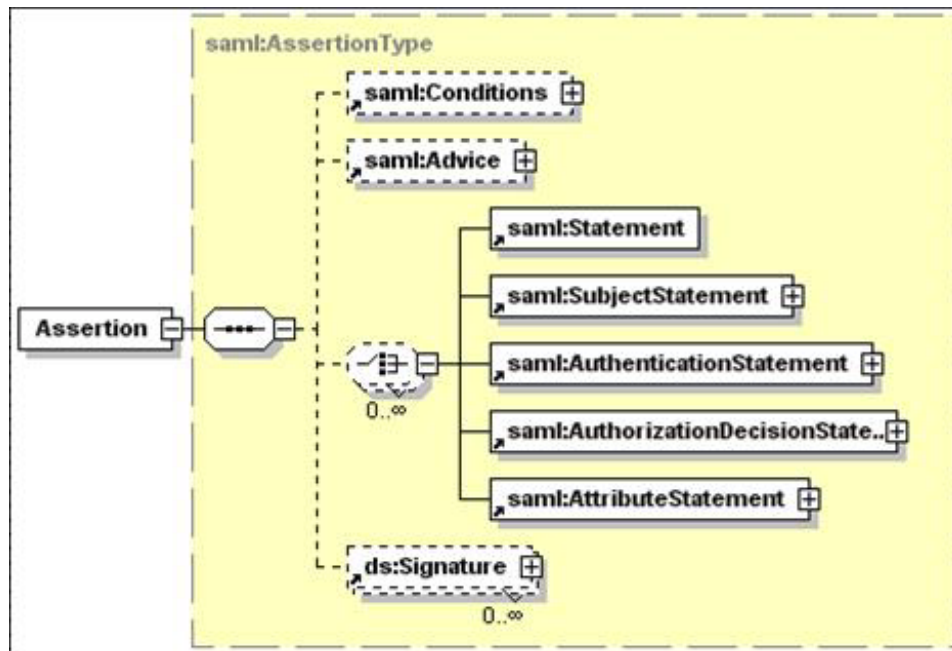
Vakuudet ovat jonkun tahon antamaa faktatietoa jostain subjektista. Vakuudet koostuvat yhdestä tai useammasta lauseesta (statement), joita on kolmea erilaista: tunnistus, ominaisuus ja valtuutus päätös.

2.1 Lauseiden yhteiset tiedot

Kaikilla lauseilla ovat seuraavat yhteiset tiedot:

- Myöntäjän ID ja myöntöajankohdan aikaleima
- Vakuuden ID
- Vakuuden kohteena olevan subjektin nimi ja hallinta-alueen tunnus. Lisäksi voi olla jokin subjektin tunnistava asia, esim. julkinen avain.
- Yksi tai useampi tila, jossa vakuus on validi.
- Ohjeita

Myöntäjän ID:n lisäksi on tärkeää, että vakuuteen laitetaan myöntöajankohdasta aikaleima ja aika, jolloin vakuus on validi. Tämän lisäksi vakuudella voi olla muitakin tiloja. Ohjeita-kohtaan voi laittaa esim. tietoa kuinka vakuus on tehty. Kuvassa 4. näkyy vakuuden rakenne.



Kuva 4: Vakuuden (assertion) rakenne [Mal01].

Seuraavassa kuvassa (Kuva 5.) on esimerkki kaikille lauseille yhteisistä tiedoista.

```

<saml:Assertion
  MajorVersion="1" MinorVersion="0"
  AssertionID="128.9.167.32.12345678"
  Issuer="Smith Corporation"
  IssueInstant="2001-12-03T10:02:00Z">
  <saml:Conditions
    NotBefore="2001-12-03T10:00:00Z"
    NotOnOrAfter="2001-12-03T10:05:00Z">
    <saml:AudienceRestrictionCondition>
      <saml:Audience>...URI...</saml:Audience>
    </saml:AudienceRestrictionCondition>
  </saml:Conditions>
  <saml:Advice>
    ...a variety of elements can go here...
  </saml:Advice>
  ...statements go here...
</saml:Assertion>

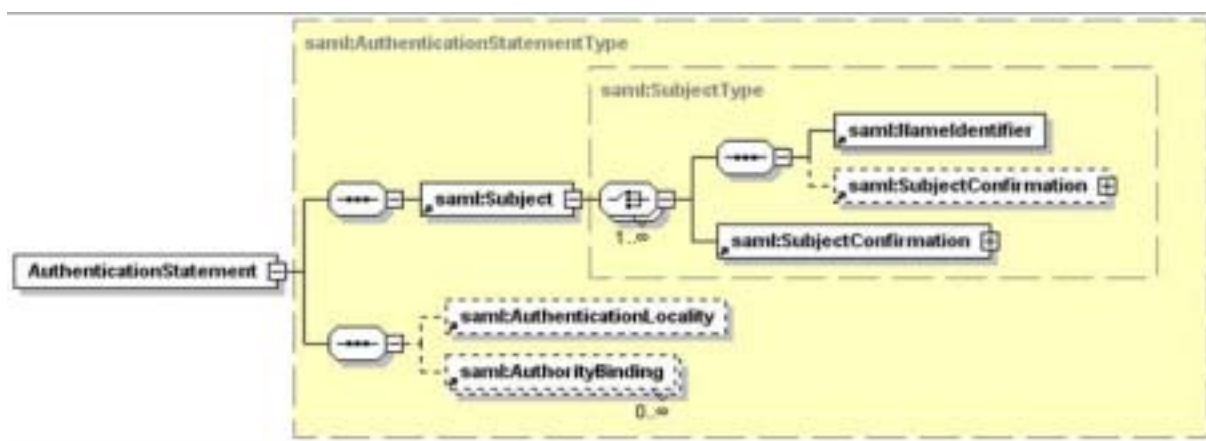
```

Kuva 5: Esimerkki yhteisistä tiedoista [Mal01].

Esimerkin saml-nimiavaruus on tarkoitettu vakuuksien määrittelemiseen. SAML ei ota kantaa yksilöiviin tunnistuksiin, joten esimerkiksi vakuuden tunnisteen (AssertionID) pitäminen yksiselitteisenä jää käyttöönotajan vastuulle. Esimerkissä näkyy myös tiloja, jotka määrittelevät milloin vakuus on validi (esim. aikaväli).

2.2 Tunnistus

Tunnistus-lauseessa vakuuden antaja vakuuttaa, että henkilö tai järjestelmä x on tunnistettu menetelmällä y ajankohtana z. Suunniteltu erityisesti SSO-tapauksia silmälläpitäen. Kuvassa 6. näkyy tunnistuksen rakenne.



Kuva 6: AuthenticationStatement-elementtityypin rakenne [Mal01].

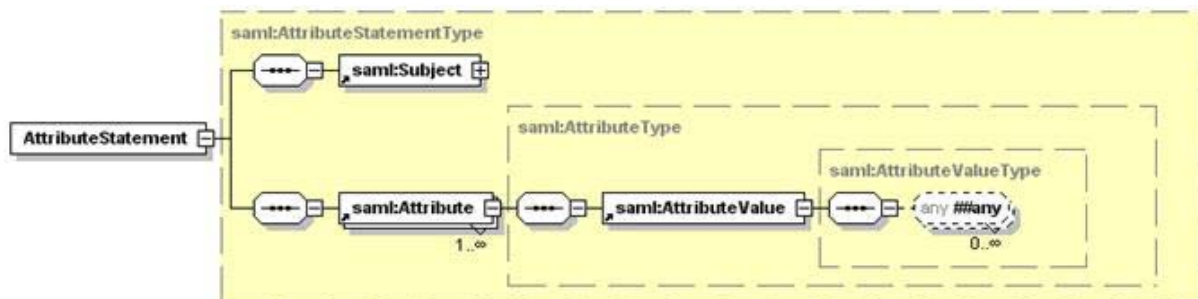
SubjectConfirmation-elementtityypin avulla varmistetaan, että paikalliseen järjestelmään yrittävä ja vakuudessa viitattu subjekti ovat sama. Vakuushan pyydetään vakuuksia antavalta taholta sen jälkeen kun subjekti on ottanut yhteyttä. SAML v1.1 ei ota tarkemmin kantaa subjektin valtuuksiin (credential passing). SAML-protokollalla voi kuitenkin kysyä valtuutusta päästä jokin subjekti johonkin tiettyyn suojattuun resurssiin, aiheesta lisää luvussa 2.4. Kuvassa 7 näkyy esimerkki tunnistamisesta.

```
<saml:Assertion ..>
  <saml:AuthenticationStatement
    AuthenticationMethod="password"
    AuthenticationInstant="2001-12-03T10:02:00Z">
    <saml:Subject>
      <saml:NameIdentifier
        SecurityDomain="smithco.com"
        Name="joeuser" />
      <saml:ConfirmationMethod>
        http://...core-25/sender-vouches
      </saml:ConfirmationMethod>
    </saml:Subject>
  </saml:AuthenticationStatement>
</saml:Assertion>
```

Kuva 7: Esimerkki tunnistuksesta [Mal01].

2.3 Ominaisuus

Hajautetun transaktion (Luku 2.2) tai valtuuksien tarkastamisen (Luku 2.3) yhteydessä voidaan vakuuden sisällä välittää käyttäjistä tietoja, SAML-kielillä ominaisuuksia. Näitä ominaisuuksia voisivat olla esimerkiksi käyttäjän rooli ja hänen saldorajansa. Ominaisuus-lauseessa voi siis välittää tietoja seuraavaan tyyliin: vakuuksia antava taho vakuuttaa, että käyttäjällä on ominaisuudet A, B ja C arvoilla "a", "b" ja "c". Kuvassa 8. näkyy AttributeStatement-elementtityypin rakenne sekä Kuvassa 9. esimerkki kahden attribuutin välittämisestä vakuuden sisällä.



Kuva 8: AttributeStatement-elementtityypin rakenne [Mal01].

```

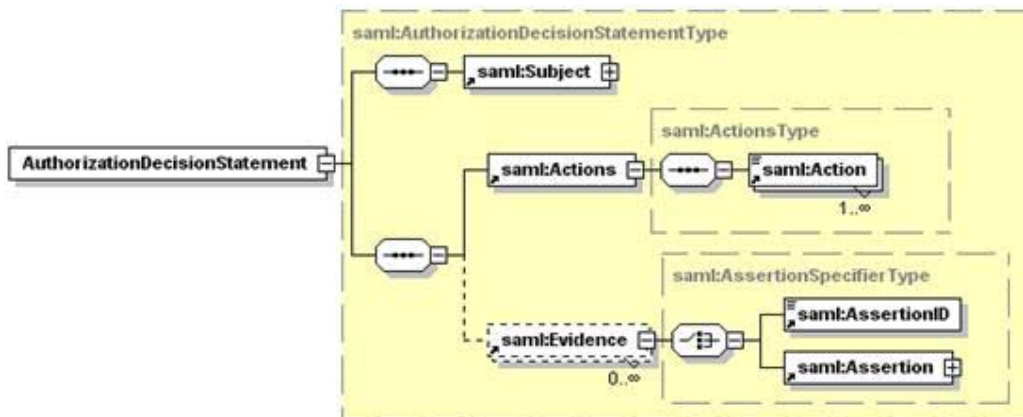
<saml:Assertion ...>
  <saml:AttributeStatement>
    <saml:Subject>...</saml:Subject>
    <saml:Attribute
      AttributeName="PaidStatus"
      AttributeNamespace="http://smithco.com">
      <saml:AttributeValue>
        PaidUp
      </saml:AttributeValue>
    </saml:Attribute>
    <saml:Attribute
      AttributeName="CreditLimit"
      AttributeNamespace="http://smithco.com">
      <saml:AttributeValue>
        <my:amount currency="USD">500.00
        </my:amount>
      </saml:AttributeValue>
    </saml:Attribute>
  </saml:AttributeStatement>
</saml:Assertion>

```

Kuva 9: Esimerkki attribuuttien välittämisestä [Mal01].

2.4 Valtuutus päätös

Vakuuksia antava taho päättää antaako valtuuden subjektille S päästä tietyllä tavalla A johonkin resurssiin R ottaen huomioon todisteen E. Tässä subjekti voi olla joko käyttäjä tai ohjelma. Resurssi voi olla esimerkiksi nettisivu tai web service. Valtuutus päätös on hyödyllinen hajautetussa transaktiossa ja valtuuksien antamisessa. Kuvassa 10. näkyy AuthorizationDecisionStatement-elementtityypin rakenne ja Kuvassa 11. esimerkki sen käytöstä.



Kuva 10: AuthorizationDecisionStatement-elementtityypin rakenne [Mal01].

```

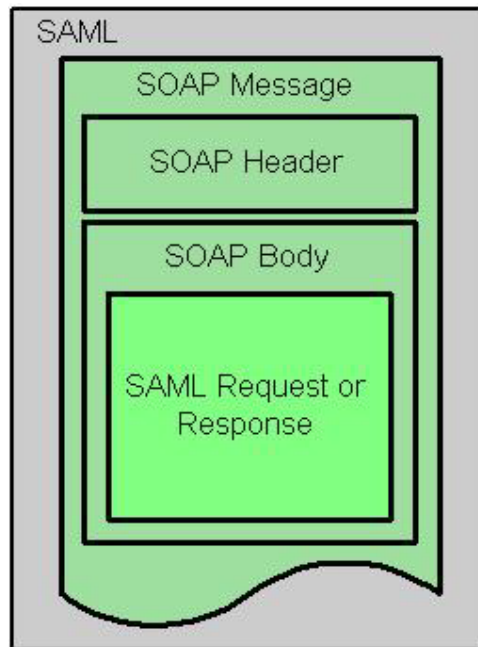
<saml:Assertion ..>
  <saml:AuthorizationStatement
    Decision="Permit"
    Resource="http://jonesco.com/rpt_12345.htm">
    <saml:Subject>...</saml:Subject>
    <saml:Actions
      ActionNamespace="http://...core-25/rwcdc">
      <saml:Action>Read</saml:Action>
    </saml:Actions>
  </saml:AuthorizationStatement>
</saml:Assertion>

```

Kuva 11: Esimerkki valtuutus päätöksestä [Mal01].

3 Sidokset ja profiilit

Sidokset määrittelevät kuinka SAML viestejä välitetään verkkoa pitkin. SAML perussidos on SOAP-over-HTTP eli SAML viesti sisällytetään SOAP viestiin ja siirretään HTTP-protokollaa hyväksikäyttäen, Kuva 12. Muitakin sidoksia on, esimerkiksi "raw HTTP".



Kuva 12: SOAP-over-HTTP sidos [Mal01].

Profiilit ovat tavallaan kuin templaatteja, jotka kertovat kuinka käyttää SAML kieltä johonkin tiettyyn tarkoitukseen. Profiileja on esimerkiksi SOAP Profile, joka kertoo kuinka SAML kielellä vaakuutetaan jokin SOAP viestin body-osa.

Lähteet

[Hug04] J. Hughes et al., *Technical Overview of the OASIS Security Assertion Markup Language (SAML) V1.1*. OASIS, May 2004.

<http://www.oasis-open.org/committees/download.php/6837/sstc-saml-tech-overview-1.1-cd.pdf>, viitattu 17.8.2005

[Mal01] E. Maler, *SAML basics - a technical introduction to the Security Assertion Markup Language*, Sun Microsystems Inc., March 2001. <http://xml.coverpages.org/Maler-saml-basics.ppt>, viitattu 3.11.2005