

# Web-sovelluspalveluiden teknisiä suosituksia SerAPI-projektissa

	SerAPI-projekti
	Yhteys henkilö: <a href="mailto:Marko.Sormunen@uku.fi">Marko.Sormunen@uku.fi</a>
	Versio 1.0
	viimeksi päivitetty: 18.11.2005

1	Johdanto .....	4
1.1	Dokumentin tarkoitus.....	4
2	Web-sovelluspalvelun toiminta ja rakentuminen.....	5
2.1	Web-sovelluspalvelustandardit ja -määritykset .....	5
2.2	Web-sovelluspalvelujen sanomaliikenteestä.....	8
2.2.1	HTTP:n tilattomuus.....	8
2.2.2	HTTP:n yksisuuntainen kutsumekanismi .....	8
2.2.3	HTTP-sanomien suojaaminen .....	9
3	SOAP-viestit .....	9
3.1	EchoService-palvelun SOAP-kutsu.....	9
3.2	EchoService-palvelun SOAP-vastaus.....	11
3.3	SOAP-viestien laajennukset.....	12
4	SOAP-liittymien kuvaaminen WSDL:n avulla .....	12
4.1	Mitä WSDL merkitsee .....	12
4.2	EchoService-palvelu WSDL-esityksenä.....	15
4.2.4	Service-elementti .....	17
4.2.5	Binding-elementti .....	18
4.2.6	PortType-elementti .....	18
4.2.7	Message-elementit .....	19
4.2.8	Types-elementti .....	19
4.3	WSDL:n käytön suositukset .....	21
4.4	WSDL-esityksen laatiminen.....	22
4.5	Liitetiedostot.....	23
4.6	XML-skeeman käyttö .....	25
4.7	WS-I -yhteensopivuuden tarkistaminen.....	27
5	Kehittyvät WS-määritykset .....	27
6	Viitattut määritykset ja dokumentit .....	29
	Liite 1: EchoService-palvelun referenssitoteutus .....	30
	Liite 2: EchoService-palvelun WSDL-esityksen testiraportti .....	32

<b>Versio</b>	<b>Päivämäärä</b>	<b>Laatija(t)</b>	<b>Kommentti</b>
1.0	18.11.2005	Marko Sormunen	Versio 1.0 julkaistu.

# 1 Johdanto

## 1.1 Dokumentin tarkoitus

Tämä dokumentti on kirjoitettu osaksi Kuopion yliopiston SerAPI-projektin teknisiä selvityksiä. Sen tarkoitus on olla mahdollisimman selkeä kuvaus **SerAPI-projektin puitteissa** tapahtuvasta web-sovelluspalvelujen kirjoittamisesta ja julkaisemisesta **Web Services Description Language** (WSDL)-kielellä ja näiden sovelluspalvelujen käyttämisestä **Simple Object Access Protocol** (SOAP)-kehyksen avulla.

Dokumentin sisältö on **tiivis yhteenveto** World Wide Web Consortiumin (W3C) web-palveluihin liittyvistä **ydinstandardeista** ja niiden **käytöstä**, tarkoituksena muodostaa selkeä kuva yhteen toimivista web-sovelluspalveluista. Tässä yhteydessä web-sovelluspalvelulla tarkoitetaan julkista rajapintaa tarjoavaa sovelluspalvelutoteutusta, jota voidaan kutsua Remote Procedure Call (RPC) –tyylisesti HTTP:n avulla siirtyvillä SOAP-viesteillä.

Dokumentissa ei pyritä antamaan kattavaa kuvaa W3C:n web-sovelluspalveluja käsittelevien standardien monipuolisista käyttömahdollisuuksista vaan kertomaan tarkasti yksi riittävän hyväksi havaittu toimintatapa niiden hyödyntämisestä. Suuri osa tästä rajauksesta johtuu Web Services Interoperability Organizationin (WS-I) antamista lisämääreistä, joiden tarkoitus on helpottaa web-sovelluspalvelujen yhteistoiminnallisuuden toteuttamista rajaamalla standardien soveltamismahdollisuuksia. Web-sovelluspalvelujen rakentamista ja käyttöä pyritään lähestymään tiiviiden mutta kattavien esimerkkien avulla. Näiden esimerkkien tavoitteena on antaa lukijalle eväät luoda omia web-palveluita WSDL-esityskieltä käyttäen.

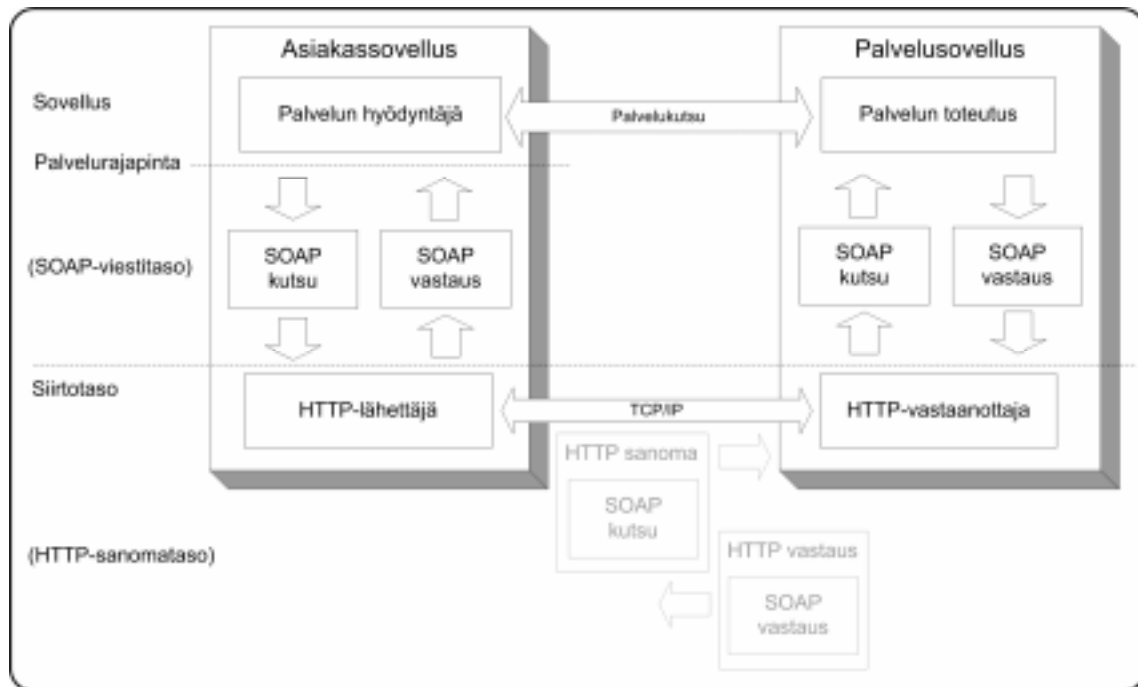
Dokumentin tavoitteet ovat:

- esittää yksi web-sovelluspalvelujen käyttötapa mahdollisimman laajan yhteistoiminnallisuuden takaamiseksi eri SOAP-viestintää käyttävien osapuolien välillä.
- esittää palvelujen rajapintoja selkeillä WSDL-määrittelyillä ja kuvata yksityiskohtaisesti, mitä nämä määrittelyt pitävät sisällään.
- esittää web-sovelluspalvelujen kehittämisessä käytettävä perusrunko, jota voidaan tarpeen vaatiessa laajentaa muilla kehittyvillä standardeilla, kuten viestiturvallisuus ja reititys.
- tukea jo olemassa olevia WSDL/SOAP-standardeja laajentavia terveydenhuolto-sektorin määrittelyksiä, kuten Clinical Document Architecture (CDA).

## 2 Web-sovelluspalvelun toiminta ja rakentuminen

Tässä kappaleessa kuvataan yleisesti SOAP-viestiliikenteeseen liittyviä peruststandardeja ja mihin niitä sovelletaan. Kuvassa 1 on esitelty palvelusovelluksen ja sitä hyödyntävän asiakasprosessin välistä perustason liikennettä käsitetasolla.

Web-sovelluspalveluiden tekninen arkkitehtuuri sisältää ainakin osat **HTTP-lähetäjä** ja **HTTP-vastaanottaja**, jotka vastaavat SOAP-viestien eli **SOAP-kutsujen** ja – **vastausten** siirtämisestä HTTP-protokollan avulla. Edellisen lisäksi arkkitehtuuri yleensä pitää sisällään toiminnallisuutta, jonka tehtävänä on ottaa pois sovelluskehittäjän käsistä SOAP-viesteihin liittyvä XML-tekstidatan käsittely. Käytännössä **HTTP-sanomataso** ja **SOAP-viestitaso** piilotetaan täysin ja web-sovelluspalveluita käytetään asiakasprosessista käsin aivan kuten normaaleja paikallisia ohjelmakirjastoja.



Kuva 1. Web-sovelluspalveluiden tekninen arkkitehtuuri

### 2.1 Web-sovelluspalvelustandardit ja -määritykset

Dokumentissa esitetty käytäntö web-sovelluspalvelun kuvaamisesta WSDL:n avulla ja kuvatun web-palvelun käyttö SOAP:in avulla pohjautuu seuraaviin teknisiin määrityksiin ja standardeihin:

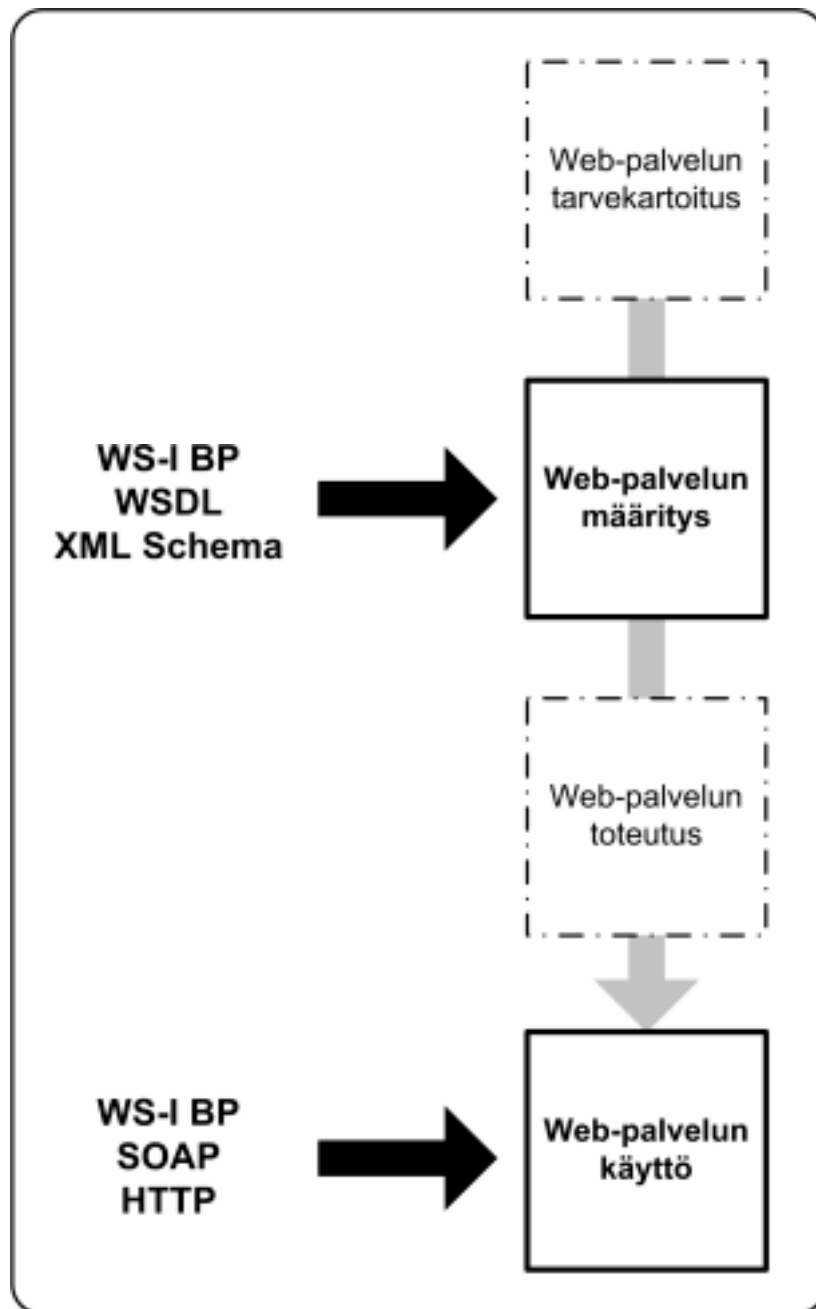
- **Extensible Markup Language (XML)**, version 1.0.

- **XML Schema** (XML S). XML-skeeman avulla kuvataan XML-dokumentteja. Skeeman avulla on esimerkiksi mahdollista tarkistaa, että XML-dokumentti on oikean muotoinen. XML-skeemaa käytetään WSDL-kielessä kuvaamaan web-sovelluspalvelujen rajapintoja eli minkälaisia SOAP-viestejä ne ymmärtävät.
- **Simple Object Access Protocol** (SOAP), version 1.1. SOAP-standardi kuvaa web-sovelluspalvelun ja sitä kutsuvan asiakasprosessin välillä liikkuvien viestien rakennetta, toisin sanoen, mitä osia (XML-elementtejä) SOAP-viesteissä pitää olla, jotta ne ovat oikein muodostettuja. SOAP-standardin uusin versio on 1.2, mutta WS-I Basic Profile määrää käytettäväksi version 1.1.

SOAP-viestien tietosisältö määräytyy web-sovelluspalvelun rakennetta kuvaavassa WSDL-dokumentissa; SOAP-standardi kuvaa XML-rakenteisen kehyksen, jolla tietosisältö välitetään. SOAP-standardia on käsitelty kappaleessa 3.

- **Web Services Description Language** (WSDL) version 1.1. WSDL-standardi kuvaa, miten web-sovelluspalveluja voidaan esittää XML-muotoisen WSDL-dokumentin avulla alustasta ja tiedonsiirrosta riippumatta. Käytännössä WSDL-dokumentit sidotaan kuitenkin usein miten käyttämään SOAP:ia viestitasolla ja HTTP:tä tiedonsiirtoprotokollana. Tämä tapahtuu laajentamalla web-sovelluspalvelun WSDL-esitystä erityisellä WSDL-SOAP –sidonnalla. WSDL-standardia on käsitelty kappaleessa 4. Myös WSDL-standardin uusin versio on 1.2, mutta WS-I Basic Profile määrää käytettäväksi versiota 1.1
- **WS-I Basic Profile** (WS-I BP), version 1.1. Basic Profile -määrityksessä määrätään WSDL-dokumenteille ja SOAP-viesteille lisärajoituksia, jotta yhteistoiminnallisten web-palvelujen rakentaminen olisi helpompaa. Lisäksi määrittäminen tarkentaa useita tulkinnanvaraisia kohtia WSDL- ja SOAP-standardeissa.
- **Hypertext Transfer Protocol** (HTTP), version 1.1. HTTP on ylivoimaisesti käytetyin tiedonsiirtoprotokolla SOAP-viestien siirtämisessä ja sen käyttöä vaatii myös WS-I Basic Profile.

Edellä mainittujen standardien lisäksi web-sovelluspalveluihin liittyy monia ylemmän tason määrityksiä, jotka käsittelevät palveluiden yhteistoiminnallisuutta. Näistä on kerrottu lisää kappaleessa 4.6. Kuva 2 esittää tässä mainittujen protokollien käyttöä web-sovelluspalvelujen eri rakennusvaiheissa.



**Kuva 2.** Standardien ja määrittysten soveltamiskohdat

## 2.2 Web-sovelluspalvelujen sanomaliikenteestä

WS-I Basic Profile määrää web-sovelluspalvelut käyttämään SOAP-viestien siirtoprotokollana HTTP:tä. Tässä kappaleessa käydään läpi joitakin HTTP:n käyttöön liittyviä piirteitä.

### 2.2.1 HTTP:n tilattomuus

SOAP-viestintää käyttävät palvelut ovat toteutukseltaan tilattomia, kuten on myös HTTP-tiedonsiirtoprotokolla; Web-sovelluspalvelun **tilattoman** toteutuksen periaatteena on, että se ei voi yhdistää samaan prosessiin kuuluvia palvelukutsuja toisiinsa. Jos palvelukutsujen välinen looginen yhteys on välitettävä **tilalliselle** web-sovelluspalvelulle, yhteys on yksilöitävä tunnisteella ja sisällytettävä jollain tavalla jokaiseen HTTP-sanomaan. Tämä tunniste voi olla esimerkiksi "sessioavain", joka esimerkiksi tunnistaa usean palvelukutsun kuuluvan samaan laajempaan prosessiin. Sessioavaimen lisäämiseksi voidaan erottaa kolme eri lähestymistapaa:

1. Tunniste voidaan lisätä suoraan SOAP-viestiin osana itse tietosisältöä. Esimerkiksi "sessioavain" voidaan liittää jokaiseen SOAP-viestiin siten, että se on osa web-sovelluspalvelun jokaista kutsuttavaa operaatiota.
2. Toinen vaihtoehto on sisällyttää "sessioavain" osaksi SOAP-viestien otsikkotietoja (kts. kappale 3.3), jolloin "sessioavaimen" ei tarvitse näkyä web-sovelluspalvelun WSDL-esityksessä, jossa web-sovelluspalvelun kutsuttavat operaatiot kuvataan. WS-ReliableMessaging (WS-RM) –määritys sisältää tähän toiminnallisuuteen tarvittavat XML-elementit. WS-ReliableMessaging tarjoaa myös laajemman ja kokonaisvaltaisen ratkaisun SOAP-viestien luotettavan siirron ja vastaanottamisen varmistamiseksi.
3. Kolmantena vaihtoehtona "sessioavain" voidaan myös liittää vielä alemmalla tasolla SOAP-viestin kuljettavaan HTTP-sanomaan HTTP-evästeeksi (cookie). WS-I Basic Profile sisältää ohjeistuksen tämän toteuttamiseksi, mutta SOAP-toteutusten ei tarvitse tukea tätä toiminnallisuutta.

Näistä kolmesta vaihtoehdosta vaihtoehto 2 on monipuolisin ja pisimmälle standardisoitu, mutta samalla myös monimutkaisin; Vaihtoehto 3 on helpoin toteuttaa. Selkeää, jokaiseen tilanteeseen sopiva ratkaisua on vaikea tarjota.

### 2.2.2 HTTP:n yksisuuntainen kutsumekanismi

Web-sovelluspalvelujen luonteeseen kuuluu myös yksisuuntainen viestinvaihto: Palvelu vastaa saamaansa kutsuun, mutta ei itse osaa aloittaa kutsu-vastaus –tapahtumaa.

Tästä etuna on palveluiden teknisen toteutuksen yksinkertaisuus. Haittapuolena on vaikeus mallintaa monimutkaisia liiketoimintaprosesseja, jotka eivät välttämättä toimi tällä periaatteella.

SOAP-viesti siirretään HTTP-sanoman sisällä sen Body-osiossa. Käytännössä tiedonsiirtoprotokolla-taso ei näy edes sovellusohjelmoijalle web-sovelluspalveluja rakennettaessa, sillä kaikki kehitysympäristöt sisältävät nykyään kattavan tuen HTTP:n lähettämiseen ja vastaanottamiseen.

### 2.2.3 HTTP-sanomien suojaaminen

HTTP-protokollan suurena etuna on mahdollisuus tarvittaessa käyttää HTTPS:ää yhteyskanavan salaamiseksi. HTTPS:n käyttö on suositeltavaa, jos web-sovelluspalvelun ja sitä käyttävän asiakasprosessin välinen tiedonsiirto on suojattava; HTTPS rajoittuu tiedonsiirron suojaamiseen HTTP-lähettäjän ja HTTP-vastaanottajan välillä (kts. Kuva 1). Lisäksi HTTPS:n avulla voidaan lähettäjän ja vastaanottajan identiteetit varmentaa sertifikaattien avulla.

HTTP:n korvaaminen HTTPS:llä onnistuu helposti, koska HTTPS salaa koko siirrettävän sanoman täysin piilossa ylemmältä SOAP-viesti –tasolta.

## 3 SOAP-viestit

Tässä kappaleessa esitellään HTTP-sanomien sisällä kulkevia SOAP-viestejä ja kerrotaan niiden koostumuksesta tarkemmin. Web-sovelluspalveluesimerkiksi on rakennettu yksinkertainen **EchoService** –palvelu, jonka rajapinta sisältää ainoastaan yhden echoText-operaation. Operaatio ottaa vastaan yhden tekstimuotoisen parametrin ja palauttaa myös yhden tekstimuotoisen paluuarvon.

### 3.1 EchoService-palvelun SOAP-kutsu

Esimerkki 1 sisältää esimerkki-HTTP -sanoman, joka pitää sisällään EchoService-palvelun echoText-kutsun SOAP-operaation. Esimerkissä on merkittynä kaikki HTTP-sanoman mukana menevä tieto. Tästä itse SOAP-kutsu on pelkästään XML-osa, joka on kuvattuna **soapenv:Envelope** –elementillä.

```
POST http://localhost/EchoService/Service/EchoService HTTP/1.0
Content-Type: text/xml; charset=utf-8
Accept: application/soap+xml, application/dime, multipart/related, text/*
User-Agent: Axis/1.2RC2
Host: localhost
Cache-Control: no-cache
Pragma: no-cache
SOAPAction: "urn:serapi:SOAPEXAMPLE#echoText"
Content-Length: 302

<soapenv:Envelope>
  xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
  xmlns:xsd="http://www.w3.org/2001/XMLSchema"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
    <soapenv:Body>
      <echoText xmlns="urn:serapi:SOAPEXAMPLE">
        <text>kal kuteksti!!!</text>
      </echoText>
    </soapenv:Body>
  </soapenv:Envelope>
```

### Esimerkki 1. EchoText-SOAP-kutsu

SOAP-kutsusta voidaan erottaa seuraavat osat:

- Juurielementti **Envelope**, joka pitää sisällään koko SOAP-kutsun.
- Nimialuemäärittely `soapenv`, jota SOAP-kutsun pakolliset elementit käyttävät.
- Elementti **Body**, joka pitää sisällään SOAP-kutsun tietosisällön.
- Elementti **echoText**, joka kertoo web-sovelluspalvelun kutsun nimen. EchoText-elementin yhteydessä on myös määritelty web-palvelukutsun oma nimialue. EchoText-elementti ja sen sisältö on kuvattu web-sovelluspalvelun WSDL-esityksessä kappaleessa 4.2.8.
- Elementti **text**, joka pitää sisällään echoText-kutsulle annettavan tekstimuotoisen parametrin.

Esimerkin SOAP-kutsussa huomattavaa on sen täsmällinen esitysmuoto. Body-elementin sisällä oleva echoText-elementti tulkitaan yksiselitteisesti web-palvelukutsun nimeksi. Lisäksi echoText-elementin sisällä oleva text-elementti tulkitaan kutsun parametriksi.

## 3.2 EchoService-palvelun SOAP-vastaus

Esimerkki 2 sisältää esimerkki-HTTP -vastauksen, joka on palautettu esimerkin 1 SOAP-kutsuun. Esimerkissä on lueteltuna kaikki HTTP-vastauksen mukana palautettava tieto.

Tästä itse SOAP-vastausta on pelkästään XML-osa, joka on kuvattuna

**soapenv:Envelope** -elementillä.

```
HTTP/1.1 200 OK
Date: Wed, 12 Jan 2005 08:25:00 GMT
Server: Oracle Application Server Containers for J2EE 10g (9.0.4.0.0)
Connection: Close
Content-Type: text/xml; charset=utf-8

<soapenv:Envelope
  xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
  xmlns:xsd="http://www.w3.org/2001/XMLSchema"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
  <soapenv:Body>
    <echoTextResponse xmlns="urn:serapi:SOAPEXAMPLE">
      <echoedText>Echoed text: kai kuteksti!!!</echoedText>
    </echoTextResponse>
  </soapenv:Body>
</soapenv:Envelope>
```

### Esimerkki 2. EchoText-kutsun palauttama SOAP-vastaus

SOAP-vastauksesta voidaan erotella seuraavat osat:

- Juurielementti **Envelope**, joka pitää sisällään koko SOAP-vastauksen.
- Nimialuemäärittäminen `soapenv`, jota SOAP-vastauksen pakolliset elementit käyttävät.
- Elementti **Body**, joka pitää sisällään SOAP-vastauksen tietosisällön.
- Elementti **echoTextResponse**, joka kertoo SOAP-vastauksen pitävän sisällään vastauksen `echoText`-kutsuun ja itse vastauselementti **echoedText**.  
EchoTextResponse-elementin yhteydessä on myös määritelty web-palveluvastauksen nimialue. EchoTextResponse-elementti ja sen sisältö on kuvattu web-sovelluspalvelun WSDL-esityksessä kappaleessa 4.2.8.
- Elementti **echoedText**, joka pitää sisällään `echoText`-kutsun toteutuksen laatiman merkkimuotoisen vastauksen.

Esimerkin SOAP-vastaus on pitkälti samanmuotoinen kuin alkuperäinen SOAP-kutsu. Body-elementin sisällä oleva `echoTextResponse`-elementti tulkitaan yksiselitteisesti liittyvän alkuperäiseen web-palvelukutsuun ja `echoedText`-elementin sisällä oleva teksti on web-palvelukutsun paluarvo.

### 3.3 SOAP-viestien laajennukset

SOAP-viestien laajentaminen onnistuu helposti, koska ne ovat XML-dokumentteja ja laajennukset yleensä saadaan lisättyä sanomiin ilman, että olemassa olevia sanomaelementtejä tarvitsee muuttaa. Esimerkissä 3 on muokattuna esimerkki 2:n SOAP-viesti, johon on lisätty aikaleima (wsu:Timestamp-elementti).

```
<soapenv:Envelope
  xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/">
  <soapenv:Header>
    <wsu:Timestamp
      xmlns:wsu="http://schemas.xmlsoap.org/ws/2002/07/utility">
      <wsu:Creation>2004-12-10T08:55:25+0200</wsu:Creation>
    </wsu:Timestamp>
  </soapenv:Header>
  <soapenv:Body>
    <echoTextResponse xmlns="urn:serapi:SOAPExample">
      <echoedText>Echoed text: kai kuteksti!!!</echoedText>
    </echoTextResponse>
  </soapenv:Body>
</soapenv:Envelope>
```

#### Esimerkki 3. EchoText-kutsun palauttama SOAP-vastaus aikaleiman kanssa

Aikaleiman lisäämiseksi SOAP-vastaukseen on lisätty myös **Header**-elementti jota kutsutaan SOAP-viestin otsikkotiedoiksi. Metatieto, kuten tässä yhteydessä aikaleima, sijoitetaan aina Header-elementin sisälle. Tämä sen vuoksi, että SOAP-viestiä tulkitsevan osapuolen ei välttämättä tarvitse huomioida Header-elementin sisältöä, kuten tässä tapauksessa aikaleimaa. Header-elementin muokkaaminen ei vaikuta millään tavalla itse sanoman sisältöön eli Body-elementin sisältöön.

Otsikkotietoihin lisättävä laajennusta ei tarvitse web-sovelluspalvelun WSDL-esityksessä. SOAP-viestien otsikkotiedot voidaan kuitenkin kuvata WSDL:ssä samalla tavalla kuin SOAP-viestien Body-osiotkin.

## 4 SOAP-liittymien kuvaaminen WSDL:n avulla

Tässä kappaleessa on kuvattu WSDL:n etuja ja miten kappaleessa 3 kuvattu

**EchoService**-palvelun rajapinta sekä käyttö kuvataan WSDL:n avulla.

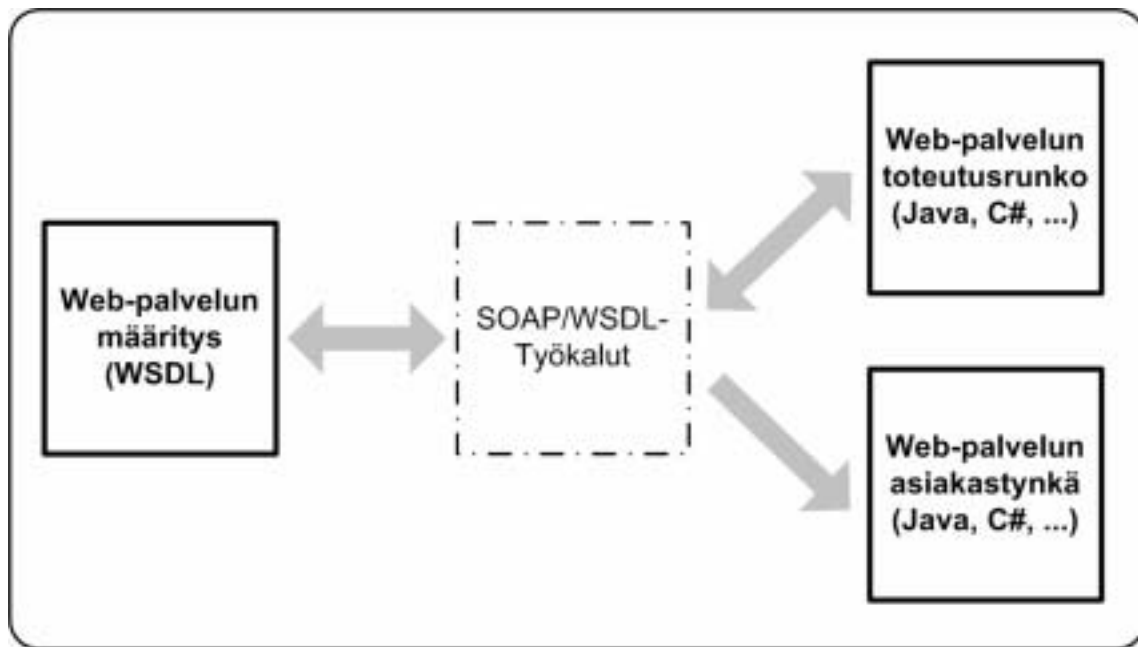
Esimerkkisovelluksen WSDL-esityksen osat käydään tarkasti läpi kappaleessa 4.2.

### 4.1 Mitä WSDL merkitsee

WSDL (Web Services Description Language) on XML-pohjainen kieli, jonka avulla voidaan kuvata abstrakti rajapinta. Yleensä WSDL:ää sovelletaan kuvaamaan nimenomaan ohjelmointirajapinta-tyyppistä toiminnallisuutta kapseloivaa konkreettista rajapintaa, joka kommunikoi SOAP-viestien ja HTTP-sanomien avulla. WSDL kuvaa ainoastaan

rajapintaa ja sen kutsumekanismia, ei sen toteutuksen muita sisäisiä ja teknisiä ratkaisuja tai toteutuksen laatua.

WSDL:ää voidaan käyttää pohjimmiltaan kahteen tarkoitukseen: Joko kuvaamaan olemassa olevan web-sovelluspalvelun rajapinta tai automaattisesti generoimaan web-palvelun perustoteutus ("runko", engl. "skeleton") tai asiakasliittymä ("tynkä", engl. "stub" tai "proxy") jollekin tietylle ohjelmointikielelle. Web-sovelluspalvelu voidaan toteuttaa ohjelmoimalla palvelun toiminnallisuus rungon ympärille; Toteutettu web-sovelluspalvelu voidaan ottaa käyttöön asiakasprosessissa kutsumalla sen palvelukutsuja tyngän avulla.



**Kuva 3.** WSDL-esityksen käyttötarkoitukset

WSDL-kielinen esitys web-palvelun rajapinnasta on riittävän perinpohjainen ja tarkka, että sen pohjalta voidaan:

- generoida automaattisesti yksinkertainen käyttöliittymä rajapintatoteutukselle.
- ilmaista täsmällisesti WSDL:n kuvaaman web-sovelluspalvelun osoite URL-muodossa.
- ilmaista täsmällisesti web-sovelluspalvelun käsittelemien SOAP-viestien muoto.

WSDL-kielillä voidaan esittää erittäin monimutkaisia liittymärakenteita. Huomattavaa on, että liittymän WSDL-esityksen pitää kuitenkin olla muodostettu tietyllä tavalla, jotta siitä voidaan generoida runko tai tynkä. Tämä johtuu siitä, että WSDL-esityksen kuvaaman web-palvelun liittymät ja sanomien tietosisältö pitää olla myös esitettävissä normaalien

ohjelmointikielten puitteissa ilmaistuilla rajapinta-rakenteilla, kuten funktioina, parametreina ja paluuarvoina.

Vastaavasti web-sovelluspalvelun jollakin ohjelmointikielellä kuvatusta rajapinnasta voidaan generoida WSDL-esitys. Käytännössä vaikka tämä vaihe on yleensä työkalujen avulla automaattinen, WSDL:ä joutuu jälkeinpäin muuttamaan käsin. Tämä sen vuoksi, että WSDL-esitykseen joudutaan lisäämään itse esimerkiksi web-palvelun URL-osoite.

WSDL-esityksen avulla voidaan myös web-palvelulle määrittää muita liittymiä (kutsupiste) kuin HTTP/SOAP. Tämä on mahdollista, koska web-palvelu voidaan samassa WSDL-esityksessä sitoa useaan eri tiedonsiirtomuotoon, kuten HTTP/SOAP, SMTP/SOAP tai FTP/SOAP. Käytännössä kuitenkin muiden tapojen kuin HTTP/SOAP käyttö on harvinaista ja niille ei ole määritelty WS-I Basic Profilen kaltaista yhteistoiminnallisuuden kehystä.

## 4.2 EchoService-palvelu WSDL-esityksenä

Seuraavassa esimerkissä on esitetty **EchoService**-palvelun WSDL-kielinen esitys kokonaan. WSDL:n eri osien XML-elementit on jaettu useaan nimialueeseen sen mukaan, mistä standardista tai määrittäjästä ne ovat peräisin:

- XML-skeemaelementit (xsd-nimialue)
- WSDL-elementit (oletusnimialue)
- WSDL:n ja SOAP:in yhdistävät elementit (wsdl soap-nimialue)

Lisäksi EchoService-palvelulla on oma nimialueensa, johon palvelu määrittellään WSDL-esityksessä (serapi -nimialue).

WSDL-esitys määrittelee vain SOAP-viestien Body-osion sisällön, mutta sillä on mahdollista myös kuvata Header-osion sisältöä. Tätä mahdollisuutta ei kuitenkaan yleensä hyödynnetä, joten se on jätetty myös tämän dokumentin ulkopuolelle.

```
<?xml version="1.0" encoding="UTF-8"?>
<definitions targetNamespace="urn:serapi:SOAPExample"
xml ns="http://schemas.xmlsoap.org/wsdl/"
xml ns:serapi="urn:serapi:SOAPExample"
xml ns:wsdl soap="http://schemas.xmlsoap.org/wsdl/soap/"
xml ns:xsd="http://www.w3.org/2001/XMLSchema">
<types>
<schema elementFormDefault="qualified"
targetNamespace="urn:serapi:SOAPExample"
xml ns="http://www.w3.org/2001/XMLSchema">
<element name="echoText">
<complexType>
<sequence>
<element name="text" type="xsd:string"/>
</sequence>
</complexType>
</element>
<element name="echoTextResponse">
<complexType>
<sequence>
<element name="echoedText" type="xsd:string"/>
</sequence>
</complexType>
</element>
<element name="noEchoTextProvided"/>
</schema>
</types>

<message name="echoTextRequestMessage">
<part element="serapi:echoText" name="parameters"/>
</message>
<message name="echoTextResponseMessage">
<part element="serapi:echoTextResponse" name="parameters"/>
</message>
<message name="noEchoTextProvided">
<part element="serapi:noEchoTextProvided" name="fault"/>
</message>
```

```

<portType name="EchoServiceOperations">
  <operation name="echoText">
    <input message="serapi:echoTextRequestMessage"/>
    <output message="serapi:echoTextResponseMessage"/>
    <fault name="noEchoTextProvided" message="serapi:noEchoTextProvided"/>
  </operation>
</portType>

<binding name="EchoServiceSOAPBinding" type="serapi:EchoServiceOperations">
  <wsdl:soap:binding style="document"
    transport="http://schemas.xmlsoap.org/soap/http"/>
  <operation name="echoText">
    <wsdl:soap:operation soapAction="urn:serapi:SOAPEXAMPLE#echoText"/>
    <input>
      <wsdl:soap:body use="literal"/>
    </input>
    <output>
      <wsdl:soap:body use="literal"/>
    </output>
    <fault name="noEchoTextProvided">
      <wsdl:soap:fault name="noEchoTextProvided" use="literal"/>
    </fault>
  </operation>
</binding>

<service name="EchoService">
  <port binding="serapi:EchoServiceSOAPBinding" name="EchoServiceSOAPPort">
    <wsdl:soap:address
      location="http://localhost/EchoService/services/EchoService"/>
  </port>
</service>
</definitions>

```

#### Esimerkki 4. EchoService-palvelun koko WSDL-esitys

## 4.2.4 Service-elementti

Esimerkki 5 esittää EchoService-palvelun WSDL-esityksen service-elementin, joka kuvastaa web-palvelun kutsupisteitä eli niitä osoitteita, joiden kautta web-palvelu ottaa vastaan palvelukutsuja. Osoitteet ilmaistaan **address**-elementillä ja annetaan URL-muodossa SOAP-viestien yhteydessä. Käytännössä URL:ia voi joutua muokkaamaan jälkeinpäin oikeaan osoitteeseen, kun web-sovelluspalvelu otetaan käyttöön. Yhdellä web-sovelluspalvelulla voi olla useita kutsupisteitä, joita voidaan kuvata jokaista omalla port-elementillä, tosin yleensä kutsupisteitä on vain yksi. **Port**-elementti myös määrää, mitä sidontaa kutsupiste käyttää. Tämä osoitetaan **binding**-elementin avulla. Port-elementissä myös määrätään kutsupisteen (eli web-palvelun) nimi **name**-attribuutilla. Jotkut SOAP-toteutukset asettavat oletuksena Web-sovelluspalvelun nimeksi port-elementissä ilmoitetun nimen service-elementissä ilmoitetun sijasta.

```
<service name="EchoService">
  <port binding="serapi:EchoServiceSOAPBinding" name="EchoServiceSOAPPort">
    <wsdl:soap:address
      location="http://localhost/EchoService/service/EchoService"/>
  </port>
</service>
```

**Esimerkki 5.** EchoService-palvelun WSDL-esityksen service-elementti

## 4.2.5 Binding-elementti

Esimerkki 6 esittää EchoService-palvelun WSDL-esityksen binding-elementtiä, joka määrittää web-palvelun kutsupisteen ymmärtämien palvelukutsujen protokollan. Binding-elementti määrittää, että EchoServicePort-kutsupiste ymmärtää **document**-tyylisiä SOAP-viestejä, jotka siirretään HTTP:n avulla. EchoService-palvelun echoText-kutsu ja – vastaus sisältävät yhden body-osion, jonka parametrit siirretään SOAP-viestissä ilman tyyppi-informaatiota eli **literal**-tyyppisinä. Body-osion sijasta echoText-kutsu voi palauttaa myös SOAP-virheen (**fault**) nimeltään noEchoTextProvided. WSDL:ssä määritellyt SOAP-virheet ovat aina sovellustason virheitä; Web-sovelluspalvelun toteutus voi myös palauttaa SOAP-virheen, vaikka niitä ei olisi määritelty WSDL-esityksessä.

```
<binding name="EchoServiceSOAPBinding" type="serapi:EchoServiceOperations">
  <wsdl:soap:binding style="document"
    transport="http://schemas.xmlsoap.org/soap/http"/>
  <operation name="echoText">
    <wsdl:soap:operation soapAction="urn:serapi:SOAPExample#echoText"/>
    <input>
      <wsdl:soap:body use="literal"/>
    </input>
    <output>
      <wsdl:soap:body use="literal"/>
    </output>
    <fault name="noEchoTextProvided">
      <wsdl:soap:fault name="noEchoTextProvided" use="literal"/>
    </fault>
  </operation>
</binding>
```

**Esimerkki 6.** EchoService-palvelun WSDL-esityksen binding-elementti

## 4.2.6 PortType-elementti

Esimerkki 7 sisältää esimerkin portType-elementistä, joka esittää abstraktin yhteenvedon jokaisesta web-sovelluspalvelun operaatiosta (**operation**). Operation-elementti esittää yhtä web-sovelluspalvelun tarjoamaa kutsua ja viittaa sen parametrina saamaansa SOAP-kutsuun, palauttamaansa SOAP-vastaukseen ja mahdollisiin virheisiin. Näihin kaikkiin osiin viitataan **message**-attribuutin avulla.

```
<portType name="EchoServiceOperations">
  <operation name="echoText">
    <input message="serapi:echoTextRequestMessage"/>
    <output message="serapi:echoTextResponseMessage"/>
    <fault name="noEchoTextProvided" message="serapi:noEchoTextProvided"/>
  </operation>
</portType>
```

**Esimerkki 7.** EchoService-palvelun WSDL-esityksen portType-elementti

## 4.2.7 Message-elementit

Esimerkki 8 esittää yhteenvedon EchoService-palvelun echoText-palvelukutsun SOAP-viestin, vastauksen ja mahdollisen virheen viestien osista, joita esittää **part**-elementit. Jokainen part-elementti viittaa **element**-attribuutilla XML-skeemaan, joka kertoo osan elementtien sisällön. Jos part-elementin name-attribuutin arvo on "parameters", kuten esimerkissä 8, element-attribuuttin avulla viitattu elementti tulkitaan API-tyyppiseksi web-sovelluspalvelukutsuksi tai -palveluvastaukseksi. Jos element-attribuutti viittaa esimerkiksi viestinä siirrettävän XML-dokumentin skeemaan, name-attribuutin arvoksi pitää laittaa esimerkiksi "body".

```
<message name="echoTextRequestMessage">
  <part element="serapi:echoText" name="parameters"/>
</message>
<message name="echoTextResponseMessage">
  <part element="serapi:echoTextResponse" name="parameters"/>
</message>
<message name="noEchoTextProvided">
  <part element="serapi:noEchoTextProvided" name="fault"/>
</message>
```

**Esimerkki 8.** EchoService-palvelun WSDL-esityksen message-elementit

## 4.2.8 Types-elementti

Esimerkki 9 esittää EchoService-palvelun echoText-palvelukutsun parametrin, paluuarvon ja mahdollisen virheen sisällön. EchoText-palvelukutsu ottaa vain yhden **text**-nimisen parametrin, jonka tyyppi on **xsd:string** ja palauttaa samoin yhden echoedText-nimisen paluuarvon. Mahdollinen virhe (**noEchoTextProvided**) ei sisällä mitään sovelluskohtaisia elementtejä, vaan se on normaali SOAP-virhe. Kuitenkin itse virhe-elementti on sisällytettävä skeemaan.

```
<types>
  <schema elementFormDefault="qualified"
    targetNamespace="urn:serapi:SOAPExample"
    xmlns="http://www.w3.org/2001/XMLSchema">
    <element name="echoText">
      <complexType>
        <sequence>
          <element name="text" type="xsd:string"/>
        </sequence>
      </complexType>
    </element>
    <element name="echoTextResponse">
      <complexType>
        <sequence>
          <element name="echoedText" type="xsd:string"/>
        </sequence>
      </complexType>
    </element>
    <element name="noEchoTextProvided"/>
  </schema>
</types>
```

**Esimerkki 9.** EchoService-palvelun WSDL-esityksen types-elementit

### 4.3 WSDL:n käytön suositukset

Web-sovelluspalvelu, kuten tässä dokumentissa esitetty EchoService-palvelu, voidaan kuvata usealla eri tavalla WSDL:n avulla. Ongelmana onkin, että vaikka nämä kaikki kuvaukset olisivat oikeita WSDL-kielen puitteissa, käytännössä niiden tulkitseminen on osoittautunut vaikeaksi SOAP-välineistöjen keskuudessa. WS-I Basic Profile rajoittaa web-sovelluspalvelun kuvaamiseen WSDL-kielen avulla enää kaksi tapaa: Dokumentti- ja RPC-tyyliset WSDL-esitykset. Näistä kahdesta tässä dokumentissa suositellaan käytettävän document-tapaa, ja sitä käyttää myös EchoService-palvelun WSDL-esitys.

Dokumentti- ja RPC-tyylit eivät juuri eroa toisistaan ja niiden kuvaamien web-palveluiden käyttämät SOAP-viestit ovat lisäksi identtisiä. Periaatteessa document-tyylisillä WSDL-esityksillä voidaan kuvata minkä tahansa XML-skeeman lähettämistä web-palvelulle, kun taas RPC-tyylinen WSDL rajoittuu enemmän perinteisempään RPC-viestintään. Käytännössä kuitenkin WS-I Basic Profile yhdistää nämä kaksi lähestymistapaa lähes samaksi.

Seuraavassa listassa on esitetty joitakin lisäsuosituksia yhteistoiminnallisuuden parantamiseksi ja WSDL-esitysten luettavuuden parantamiseksi:

- WSDL-esityksen tyyli binding-elementissä on oltava **document** (kts. kappale 4.2.5).
- SOAP-kutsuelementin nimen **on oltava** sama binding-, operation-, message- ja types-elementtien sisällä. Esimerkiksi EchoService-palvelun yhteydessä WSDL-esityksessä kutsun nimi on näissä elementeissä "EchoText" (kts. kappaleet 4.2.5, 4.2.6, 4.2.7, 4.2.8).
- SOAP -virhe-elementin nimen **on oltava** sama binding-, operation-, message- ja types-elementtien sisällä (kts. kappaleet 4.2.5, 4.2.6, 4.2.7 ja 4.2.8).
- WSDL-esityksen selkeyden takia SOAP-viestien message-elementit kannattaa nimetä selkeästi eri tavalla kuin SOAP-elementit (kts. kappaleet 4.2.6 ja 4.2.7). Tämä ei kuitenkaan koske SOAP-virheitä esittäviä message-elementtejä.
- SOAP-vastuselementin nimen **on oltava** sisällä esitettynä muodossa "xxxResponse". Esimerkiksi EchoText-palvelukutsun SOAP-vastuselementin nimi on "EchoTextResponse" (kts. kappaleet 4.2.7 ja 4.2.8).
- WSDL-esitys voi sisältää documentation-elementtejä, jotka sisältävät selkokielistä ja vapaamuotoista dokumentaatiota. Näitä elementtejä voi sijoittaa minkä tahansa WSDL:n osaelementin sisään. Skeamaelementtien dokumentoinnissa on noudatettava XML Schema-määrittelyssä esitettyä dokumentaatiotapaa, jossa documentation-elementti on sisällytettävä erillisen annotation-elementtiin. Documentation-elementin

tulkitaan dokumentoivan sitä elementtiä, jonka ensimmäinen (ja mielellään ainoa) lapsielementti se on. Esimerkissä 10 on esitetty erilaisia tapoja lisätä dokumentaatiota WSDL-esitykseen.

```
<types>
  <schema elementFormDefault="qualified"
    targetNamespace="urn:serapi:SOAPEXAMPLE"
    xmlns="http://www.w3.org/2001/XMLSchema">
    <element name="echoText">
      <complexType>
        <sequence>
          <element name="text" type="xsd:string">
            <annotation>
              <documentation>Text-elementti on merkkijono, jonka palvelu kaluttaa takaisin</documentation>
            </annotation>
          </element>
        </sequence>
      </complexType>
    </element>
  </schema>
</types>
.
.
.
<message xmlns="http://schemas.xmlsoap.org/wsdl/" name="echoTextRequestMessage">
  <part element="serapi:echoText" name="parameters">
    <documentation>echoText on EchoService-sovellyspalvelun alnoa kutsu</documentation>
  </part>
</message>
```

**Esimerkki 10.** WSDL-esityksen dokumentaatio

## 4.4 WSDL-esityksen laatiminen

Web-sovelluspalvelun WSDL-esityksen laatimiseen on kaksi tapaa, jotka tosin eivät ole toisiaan poissulkevia: Esityksen voi tehdä joko käsin tai käyttää jotain WSDL-generaattoria, joka muodostaa esityksen valmiista luokasta tutkimalla sen julkistamia metodien nimiä sekä niiden paluuarvoja ja parametreja. Usein kuitenkin näiden generaattorien tekemää WSDL-koodia joudutaan muokkaamaan käsin joko ulkoasusyistä tai se ei esimerkiksi ole sisällöltään sitä, mitä haluttiin. Joka tapauksessa WSDL-generaattorien tekemä esitys kannattaa ajaa testityökalujen (WS-I TT) läpi WS-I-yhteensopivuuden varmistamiseksi kappaleen 4.7 esimerkin mukaisesti.

Käytännössä jokainen enterprise-tason sovelluskehitin sisältää WSDL-generaattorin, jolla voidaan julkistaa valmis ohjelmakoodi (yleensä luokka) web-palveluksi ja jota kannattaa hyödyntää. Lisäksi jotkut sovelluskehittimet tukevat jo suoraan WS-I:n määrittämiä eli niiden kehittämä WSDL on WS-I-yhteensopivaa.

## 4.5 Liitetiedostot

Liitetiedostojen käyttö ei ole kovin olennaista sovellusrajapinta-tyylisiä web-sovelluspalveluja laadittaessa, mutta tämä kappale on kuitenkin mukana tässä dokumentissa, koska liitetiedostojen käytölle voi löytyä perusteita jatkossa.

WSDL-esityksessä voidaan myös web-palvelun kutsuihin liittää MIME-tyyppisiä liitetiedostoja joko parametreiksi tai paluuarvoiksi. Liitetiedostojen käyttö on määritelty dokumentissa **WS-I Attachments Profile 1.0** (WS-I AP). Seuraavassa esimerkissä on esimerkkinä EchoService-palvelun WSDL-esitys, jonka echoText-kutsun vastaukseen on liitetty itsenäinen binäärimuotoinen dokumentti. Dokumentti on merkitty **base64binary**-tyyppiseksi liitteeksi.

```
<?xml version="1.0" encoding="UTF-8"?>
<definitions targetNamespace="urn:serapi:SOAPExample"
  xmlns="http://schemas.xmlsoap.org/wsdl/"
  xmlns:impl="urn:serapi:SOAPExample"
  xmlns:wsoap="http://schemas.xmlsoap.org/wsdl/soap/"
  xmlns:mime="http://schemas.xmlsoap.org/wsdl/mime/"
  xmlns:xsd="http://www.w3.org/2001/XMLSchema">
  <types>
    <schemaelementformDefault="qualified"
      targetNamespace="urn:serapi:SOAPExample"
      xmlns="http://www.w3.org/2001/XMLSchema">
      <element name="echoText">
        <complexType>
          <sequence>
            <element name="text" type="xsd:string"/>
          </sequence>
        </complexType>
      </element>

      <element name="echoTextResponse">
        <complexType>
          <sequence>
            <element name="echoedText" type="xsd:string"/>
          </sequence>
        </complexType>
      </element>

      <element name="noEchoTextProvided"/>
    </schema>
  </types>

  <message name="echoTextRequestMessage">
    <part element="serapi:echoText" name="parameters"/>
  </message>
  <message name="echoTextResponseMessage">
    <part element="serapi:echoTextResponse" name="parameters"/>
    <part type="xsd:base64Binary" name="hello_picture"/>
  </message>
  <message name="noEchoTextProvided">
    <part element="serapi:noEchoTextProvided" name="fault"/>
  </message>

  <portType name="EchoServiceOperations">
    <operation name="echoText">
      <input message="serapi:echoTextRequestMessage"/>
      <output message="serapi:echoTextResponseMessage"/>
      <fault name="noEchoTextProvided" message="serapi:noEchoTextProvided"/>
    </operation>
  </portType>
</definitions>
```

```

</operati on>
</portType>

<bi ndi ng name="EchoServi ceSOAPBi ndi ng" type="serapi : EchoServi ceOperati ons">
  <wsdl soap: bi ndi ng styl e="document"
    transport="http: //schemas. xml soap. org/soap/http"/>
  <operati on name="echoText">
    <wsdl soap: operati on soapActi on="urn: serapi : SOAPExamp l e#echoText"/>
    <i nput>
      <wsdl soap: body use="l i teral "/>
    </i nput>
    <output>
      <mi me: mul ti partRel ated>
        <mi me: part>
          <wsdl soap: body parts="parameters" use="l i teral "/>
        </mi me: part>
        <mi me: part>
          <mi me: content part="hel lo_pi cture" type="i mage/png"/>
        </mi me: part>
      </mi me: mul ti partRel ated>
    </output>
    <faul t name="noEchoTextProvi ded">
      <wsdl soap: faul t name="noEchoTextProvi ded" use="l i teral "/>
    </faul t>
  </operati on>
</bi ndi ng>

<servi ce name="EchoServi ce">
  <port bi ndi ng="serapi : EchoServi ceSOAPBi ndi ng" name="EchoServi cePort">
    <wsdl soap: address
      l ocati on="http: //l ocal host/EchoServi ce/servi ces/EchoServi cePort"/>
  </port>
</servi ce>
</defi ni ti ons>

```

### Esimerkki 11. EchoService-palvelun WSDL-esitys MIME-liitetiedoston kanssa

Esimerkki 12 esittää HTTP-vastausta, joka pitää sisällään EchoText-kutsun SOAP-vastauksen ja liitetiedoston (kuvan, joka on PNG-formaatissa).

```

HTTP/1.1 200 OK
Date: Wed, 12 Jan 2005 09:46:08 GMT
Server: Oracle Application Server Containers for J2EE 10g (9.0.4.0.0)
Connection: Close
Content-Type: multi part/related; type="text/xml"; start="<EAFC46324A98790D3A762D268DDF2590>";
boundary="1105523169353"

105523169353
Content-Type: text/xml; charset=UTF-8
Content-Transfer-Encoding: binary
Content-Id: <EAFC46324A98790D3A762D268DDF2590>

<soapenv: Envel ope
xml ns: soapenv="http: //schemas. xml soap. org/soap/envel ope/"
xml ns: xsd="http: //www. w3. org/2001/XMLSchema"
xml ns: xsi="http: //www. w3. org/2001/XMLSchema-i nstance">
  <soapenv: Body>
    <echoTextResponse xml ns="urn: serapi : SOAPExamp l e">
      <echoedText>Echoed text: kai kuteksti !!!</echoedText>
      <hel lo_pi cture href="ci d: 5F08194DF078D566EE3AA8AC9580E2D1"/>
    </echoTextResponse>
  </soapenv: Body>
</soapenv: Envel ope>
1105523169353
Content-Type: i mage/png
Content-Transfer-Encoding: bi nary

```

```
Content-Id: <5F08194DF078D566EE3AA8AC9580E2D1>
i VB0RwOKGgoAAAANSuHEUgAAAI AAAABACAYAAADS1n9/AAABRWI DQ1BQaG90b3Nob3AgSUND
.
.
AAi g00QwwgFAAI OmgBEOAAJoNAGMcAAQKJYI QDgAADADqe4mfcm+xAAAAAEI FTkSuQmCC
-----_Part_0_20890333.1105523169353--
```

**Esimerkki 12.** HTTP-vastaus, joka sisältää EchoText-kutsun SOAP-vastauksen ja liitetiedoston (yksinkertaistettu)

## 4.6 XML-skeeman käyttö

SOAP-viestien tietosisältö siis kuvataan WSDL-esityksessä XML-skeeman avulla. Tässä kappaleessa annetaan ohjeita, mitä osia skeemasta kannattaa käyttää tietyn tyyppisen rakenteen esittämiseksi.

Esitettävä rakenne	Esimerkki XML-skeemasta ja vastaavasta XML:stä
Valikoima elementtejä, ts. <i>array</i> . Esimerkissä on kuvattu valikoimaa, johon kuuluu 0-10 merkkijonotyyppistä elementtiä.	<pre>&lt;xsd:schema   targetNamespace="urn:serapi:WSDLExamples"   xmlns:xsd="http://www.w3.org/2001/XMLSchema"&gt;   &lt;xsd:element name="merkkiJono" minOccurs="0"     maxOccurs="10" type="xsd:string"/&gt; &lt;/xsd:schema&gt;</pre> <p>Skeemaa vastaava XML-esimerkki:</p> <pre>&lt;merkkiJono&gt;merkki jono 1&lt;/merkkiJono&gt; &lt;merkkiJono&gt;merkki jono 2&lt;/merkkiJono&gt; &lt;merkkiJono&gt;merkki jono 3&lt;/merkkiJono&gt; &lt;merkkiJono&gt;merkki jono 4&lt;/merkkiJono&gt;</pre>
Viittaus toisessa skeemassa määriteltyyn tyyppiin käyttäen <i>type</i> -attribuuttia, jonka avulla viittaaminen toisessa skeemassa määritettyyn tietotyyppiin tai elementtiin on SOAP-välineissä tuetuin tapa.	<pre>&lt;xsd:schema   targetNamespace="urn:serapi:WSDLExamples"   xmlns:xsd="http://www.w3.org/2001/XMLSchema"&gt;   xmlns:type="urn:serapi:WSDLExamplesTypes"&gt;   &lt;xsd:element name="patient" type="type:Patient"/&gt; &lt;/xsd:schema&gt;</pre> <pre>&lt;xsd:schema   targetNamespace="urn:serapi:WSDLExamplesTypes"   xmlns:xsd="http://www.w3.org/2001/XMLSchema"&gt;   &lt;xsd:complexType name="Patient"&gt;     &lt;xsd:sequence&gt;       &lt;xsd:element name="SSN" type="xsd:string"/&gt;       &lt;xsd:element name="firstName" type="xsd:string"/&gt;       &lt;xsd:element name="lastName" type="xsd:string"/&gt;     &lt;/xsd:sequence&gt;   &lt;/xsd:complexType&gt; &lt;/xsd:schema&gt;</pre> <p>Skeemaa vastaava XML-esimerkki:</p> <pre>&lt;patient xmlns="urn:serapi:WSDLExamplesTypes"&gt;   &lt;SSN&gt;291274-XXXX&lt;/SSN&gt;   &lt;firstName&gt;Marko&lt;/firstName&gt;   &lt;lastName&gt;Sormunen&lt;/lastName&gt; &lt;/patient&gt;</pre>

<p>XML-skeeman sisäänrakennettujen tietotyyppien käytössä kannattaa käyttää lukujen esittämiseen <i>short</i>-tyyppiä, jos esitettävä luku on 16-bittinen. Muuten lukua esitetään automaattisesti rajoittamattomana lukuna.</p>	<pre>&lt;xsd: schema   targetNamespace="urn: serapi : WSDLExampI eTypes"   xmlns: xsd="http: //www. w3. org/2001/XMLSchema" &gt;   &lt;xsd: el ement name="16-bi tti nen_luku" type="xsd: short" /&gt;   &lt;xsd: el ement name="64-bi tti nen_Luku" type="xsd: long" /&gt;   &lt;xsd: el ement name="rajo i ttamaton_luku"     type="xsd: i nteger" /&gt; &lt;/xsd: schema&gt;</pre>
<p>Perustietotyyppien rajoittaminen XML-skeemassa ei ole yleensä tuettu WSDL-välineissä. Tämän takia tiedon tarkistaminen kannattaa tehdä myös sovelluksen sisällä. WSDL:ään sisältyvät XML-skeeman avulla voidaan kuitenkin tarkistaa erikseen, ovatko SOAP-viestit sen mukaisia.</p>	<pre>&lt;xsd: schema   targetNamespace="urn: serapi : WSDLExampI eTypes"   xmlns: xsd="http: //www. w3. org/2001/XMLSchema" &gt;   &lt;xsd: el ement name="maaKoodi " &gt;   &lt;xsd: si mpl eType&gt;   &lt;xsd: restri cti on base="xsd: stri ng" /&gt;   &lt;pattern val ue=" [A-Z] {3}" /&gt;   &lt;/xsd: restri cti on&gt;   &lt;/xsd: si mpl eType&gt; &lt;/xsd: el ement&gt; &lt;/xsd: schema&gt;</pre> <p>Ylläolevan skeeman mukaisesti maaKoodi on merkkijono, joka muodostuu aina kolmesta isosta kirjaimesta, esim. FIN. Kuitenkin harva WSDL-työkalu huomioisi rajoituksia, joten maaKoodi-elementin oikeellisuus kannattaa tarkistaa myös sovelluskoodissa.</p>
<p>Attribuuttien käyttö skeemassa on yleensä tuettu, mutta attribuuttien määrittämisessä on oltava tarkkana. Oheisessa esimerkissä on Patient-elementille määritetty SSN-attribuutti</p>	<pre>&lt;xsd: compl exType name=" Pati ent"   xmlns: xsd="http: //www. w3. org/2001/XMLSchema" &gt;   &lt;xsd: sequen ce&gt;   &lt;xsd: el ement name="fi rstName" type="xsd: stri ng" /&gt;   &lt;xsd: el ement name="l astName" type="xsd: stri ng" /&gt;   &lt;/xsd: sequen ce&gt;   &lt;xsd: attri bute name="SSN" type="xsd: ID" use="requi red" /&gt; &lt;/xsd: compl exType&gt;</pre>

<p>Mahdollisesti tyhjien elementtien esittämisessä WSDL:ssä voidaan käyttää minOccurs-attribuutin lisäksi myös nillable-attribuuttia, mutta sitä ei suositella käytettäväksi array-tyyppisten rakenteiden yhteydessä.</p>	<pre>&lt;xsd:complexType name="Patient" xml ns:xsd="http://www.w3.org/2001/XMLSchema"&gt; &lt;xsd:sequence&gt; &lt;xsd:element name="firstName" type="xsd:string"/&gt; &lt;xsd:element nillable="true" name="middleName" type="xsd:string"/&gt; &lt;xsd:element minOccurs="0" name="middleName" type="xsd:string"/&gt; &lt;xsd:element name="lastName" type="xsd:string"/&gt; &lt;/xsd:sequence&gt; &lt;/xsd:complexType&gt;</pre> <p>Skeemaa vastaava XML-esimerkki:</p> <pre>&lt;patient xml ns="urn:serapi:WSDLExampleTypes" xml ns:xsi="http://www.w3.org/2001/XMLSchema-instance"&gt; &lt;firstName&gt;Marko&lt;/firstName&gt; &lt;middleName xsi:nil="true"/&gt; &lt;lastName&gt;Sormunen&lt;/lastName&gt; &lt;/patient&gt;</pre>
---	---

**Taulukko 1.** Ohjeita XML-skeeman käytöstä WSDL-esityksessä

## 4.7 WS-I –yhteensopivuuden tarkistaminen

WS-I tarjoaa testityökaluja, joilla WSDL-esityksen WS-I –yhteensopivuus voidaan tarkistaa (WS-I TT). Työkalut ovat saatavilla Java- ja C#-toteutuksina ja ne tarkistavat, onko niille parametrina annettu WSDL-esitys WS-I Basic Profile 1.1 –määrityksen mukainen. Tuloksena saatu HTML-muotoinen testiraportti voidaan esimerkiksi esittää todisteena, että WSDL noudattaa WS-I:n määrityksiä. Liitteessä 1 on esimerkki testiraportti tarkistetusta EchoService-palvelun WSDL-esityksestä.

Käytännössä WSDL-esitys kannattaa testata kaikilla alustoilla, joilla siitä generoitavan web-palvelun on tarkoitus toimia, joko palvelutoteutuksena tai osana asiakasprosessia. Tämä sen vuoksi, että kaikki WSDL-työkalut eivät tue WS-I –yhteensopivuuden sisältämiä ominaisuuksia.

## 5 Kehittyvät WS-määritykset

Tässä kappaleessa on lyhyesti lueteltu erilaisia SOAP/WSDL-standardien päälle rakentuvia määrityksiä. Näistä useat ovat vielä kehityksen alla ja ne sisältävät myös monesti päällekkäistä toiminnallisuutta. Tämä johtuu siitä, että näitä WS-määrityksiä kehittävät useat eri yhteisöt ja yrityskonsortiot. Tässä yhteydessä on lueteltu ainoastaan niitä määrityksiä ja standardeja, joita suositellaan käytettävän. Taulukko 2 sisältää listan näistä suosituksista.

Määritys	Määrityksen kuvaus
WS-I Basic	WS-I:n kehittämä määritys, joka asettaa yhteistoiminnallisuuden

Profile (WS-I BP)	vaatimia rajoitteita SOAP- ja WSDL-standardien käytölle ja selkeyttää useita tulkinnanvaraisia kohtia niissä. WS-I BP ei ota kantaa, mitä muita WS-standardeja ja -määrittäjiä pitäisi soveltaa SOAP-viesteissä.
WS-I Simple SOAP Binding Profile (WS-I SSBP)	WS-I:n SOAP-välineistöjen kehittäjille tarkoitettu määrittäjä, jolla pyritään parantamaan matalalla tasolla SOAP-viestien lähettämisen ja vastaanottamisen yhteistoiminnallisuutta. Tämä määrittäjä ei ole tarkoitettu web-sovelluspalveluiden tekijöille, paitsi jos sovelluspalvelua kutsutaan rakentamalla SOAP-viesti itse ilman WSDL-esityksestä generoitua tynkää.
WS-I Basic Security Profile (WS-I BSP)	WS-I Basic Security Profile Working Groupin kehittämä määrittäjä, joka yhdistää W3C:n standardit <i>WS-Security</i> , <i>WS-Encryption</i> ja <i>Exclusive XML Canonicalization</i> . WS-I SP:n avulla SOAP-viestejä voidaan varmentaa allekirjoitusten avulla ja niiden sisältö voidaan myös salata.
WS-I Attachments Profile (WS-I AP)	WS-I:n kehittämä määrittäjä, jonka avulla web-palvelun WSDL-esityksessä voidaan palvelun kutsuihin liittää liitetiedostoja.
WS-Addressing (WS-A)	BEA, IBM, Microsoft, SAP ja Sun Microsystems ovat esittäneet virallisen esityksen WS-Addressing –määrittäjästä W3C:lle standardoitavaksi. WS-Addressing –määrittäjän avulla SOAP-viestiin voidaan liittää tietoa sanoman lähettäjistä ja vastaanottajasta sekä sanoman kohteen osoite (esim. URL).
WS- ReliableMessagi ng (WS-RM)	Microsoftin, IBM:n ja Tibco Softwaren ajama määrittäjä luotettavan SOAP-viestinvälityksen toteuttamiseksi. Määrittäjä ehdotetaan standardoitavaksi OASIS:en kautta.
Universal Description, Discovery and Integration (UDDI)	OASIS:n kehittämä määrittäjä, jonka avulla web-palvelu voidaan julkaista UDDI-rekisterissä ja tarvittaessa hakea käyttöön UDDI-rekisterin sisältämän web-palvelun kuvauksen perusteella.

**Taulukko 2.** Luettelo yleisimmistä web-sovelluspalvelu –määrittäjäistä

## 6 Viitattut määriykset ja dokumentit

HTTP	W3C. <i>Hypertext Transfer Protocol, version 1.1.</i> W3C Network Working Group RFC 2616. Kesäkuu 1999. <a href="http://www.w3.org/Protocols/rfc2616/rfc2616.html">http://www.w3.org/Protocols/rfc2616/rfc2616.html</a>
HTTPS	Rescorla, E. <i>RFC 2818 – HTTP over TLS.</i> Network Working Group. toukokuu 2000. <a href="http://www.fags.org/rfcs/rfc2818.html">http://www.fags.org/rfcs/rfc2818.html</a>
SOAP	W3C. <i>Simple Object Access Protocol (SOAP) 1.1</i> W3C Note. Toukokuu 2000. <a href="http://www.w3.org/TR/2000/NOTE-SOAP-20000508/">http://www.w3.org/TR/2000/NOTE-SOAP-20000508/</a>
UDDI	OASIS. <i>UDDI Version 3.02.</i> UDDI Spec Technical Committee Draft. Lokakuu 2004. <a href="http://uddi.org/pubs/uddi-v3.0.2-20041019.htm">http://uddi.org/pubs/uddi-v3.0.2-20041019.htm</a>
URN	Moats, A. <i>RFC 2141 – URN Syntax.</i> Network Working Group. Toukokuu 1997. <a href="http://www.fags.org/rfcs/rfc2141.html">http://www.fags.org/rfcs/rfc2141.html</a>
W3C	<i>World Wide Web Consortium</i> <a href="http://www.w3.org">http://www.w3.org</a>
WS-A	W3C. <i>Web Services Addressing (WS-Addressing).</i> W3C Member Submission. Elokuu 2004. <a href="http://www.w3.org/Submission/ws-addressing/">http://www.w3.org/Submission/ws-addressing/</a>
WSDL	W3C. <i>Web Services Description Language (WSDL) Version 1.2.</i> W3C Working Draft 3. Maaliskuu 2003 <a href="http://www.w3.org/TR/2003/WD-wsdl12-20030303/">http://www.w3.org/TR/2003/WD-wsdl12-20030303/</a>
WSI	<i>Web Services Interoperability Organization</i> <a href="http://www.ws-i.org">http://www.ws-i.org</a>
WS-I AP	WS-I. <i>WS-I Attachments Profile, version 1.0.</i> WS-I Attachments Profile Final Material. Elokuu 2004. <a href="http://www.ws-i.org/Profiles/AttachmentsProfile-1.0.html">http://www.ws-i.org/Profiles/AttachmentsProfile-1.0.html</a>
WS-I BP	WS-I. <i>WS-I Basic Profile, version 1.1.</i> WS-I Basic Profile Final Material. Elokuu 2004. <a href="http://www.ws-i.org/Profiles/BasicProfile-1.1.html">http://www.ws-i.org/Profiles/BasicProfile-1.1.html</a>
WS-I BSP	WS-I. <i>WS-I Basic Security Profile, version 1.0.</i> Working Group Draft. Joulukuu 2004. <a href="http://www.ws-i.org/Profiles/BasicSecurityProfile-1.0.html">http://www.ws-i.org/Profiles/BasicSecurityProfile-1.0.html</a>
WS-I SSBP	WS-I. <i>WS-I Simple SOAP Binding Profile Version 1.0.</i> Final Material. Elokuu 2004. <a href="http://www.ws-i.org/Profiles/SimpleSoapBindingProfile-1.0.html">http://www.ws-i.org/Profiles/SimpleSoapBindingProfile-1.0.html</a>
WS-I TT	WS-I. <i>Interoperability Testing Tools 1.1</i> WS-I Basic Profile 1.1 Test Tools. Joulukuu 2004. <a href="http://www.ws-i.org/deliverables/workinggroup.aspx?wg=testingtools">http://www.ws-i.org/deliverables/workinggroup.aspx?wg=testingtools</a>
WS-RM	Microsoft, Bea, IBM, Tibco Software. <i>WS-ReliableMessaging.</i> Kesäkuu 2005. <a href="http://msdn.microsoft.com/ws/2005/02/ws-reliablemessaging/">http://msdn.microsoft.com/ws/2005/02/ws-reliablemessaging/</a>
XML	W3C. <i>Extensible Markup Language (XML) 1.0 (Third Edition)</i> W3C Recommendation. Helmikuu 2004 <a href="http://www.w3.org/TR/REC-xml/">http://www.w3.org/TR/REC-xml/</a>
XML S	W3C. <i>XML Schema Specifications and Development</i> W3C Recommendation. Lokakuu 2004. <a href="http://www.w3c.org/XML/Schema#dev">http://www.w3c.org/XML/Schema#dev</a>

## Liite 1: EchoService-palvelun referenssitoteutus

Tässä kappaleessa on lyhyesti kuvattu EchoService-palvelun toteutus yhdellä SOAP-välineistöllä. Esimerkeissä on käyty läpi sekä asiakassovelluksen että lyhyen web-palvelutoteutuksen rakentaminen hyödyntämällä välineistön toiminnallisuutta mahdollisimman pitkälle.

Seuraavassa on listattu Kuopion yliopiston kettinki-palvelimella sijaitsevan EchoService-palvelun referenssitoteutukseen liittyviä web-osoitteita:

- EchoService-palvelun juurihakemisto osoitteessa <http://kettinki.uku.fi/EchoService/>.
- EchoService-palvelun WSDL-esitys on nähtävissä osoitteessa <http://kettinki.uku.fi/EchoService/services/EchoService?WSDL>.
- EchoService-palvelun kutsupiste on osoitteessa <http://kettinki.uku.fi/EchoService/services/EchoService>.

Referenssitoteutuksessa käytetty SOAP-välineistö on ilmainen Open Source-pohjainen SOAP-toteutus **Apache Axis 1.2RC3**. Se on saatavilla osoitteesta <http://ws.apache.org/axis/>. Apache Axis toimii yhdessä lähes minkä tahansa Java-sovelluspalvelimen kanssa. Tässä kappaleessa ei kuvata tarkasti kaikkia työvaiheita EchoService-palvelun toteuttamiseksi, vaan esitetään päävaiheet rungon ja tyngän luomiseksi.

### Web-palvelun rungon ja tyngän generointi WSDL:n pohjalta

EchoService-palvelun rungon ja tyngän generointi onnistuu Apache Axis-välineistöön kuuluvan `wsdl2java`-työkalun avulla. Esimerkki 13 sisältää komentoriviltä ajettavan komennon, jolla tämä tapahtuu.

```
j java -cp axis.jar;commons-logging.jar;commons-discovery.jar;wsdl4j.jar;saaj.jar;jaxrpc.jar
org.apache.axis.wsdl.WSDL2Java -s -S false
http://kettinki.uku.fi/EchoService/services/EchoService?WSDL
```

**Esimerkki 13.** EchoService-palvelun rungon ja tyngän generointi WSDL-esityksestä

Oletuksena `wsdl2java`-työkalu generoi EchoService-palvelun kutsumiseen tarkoitetun tyngän. Lisäämällä vivut `"-s -S false"` `wsdl2java` generoi myös palvelutoteutuksen rungon. Työkalu kehittää myös useita apuluokkia toteuttamisen tueksi. Näiden konkreettista käyttöä esitellään seuraavissa esimerkeissä.

Esimerkki 14 sisältää EchoService-palvelun referenssitoteutuksen listauksen.

```

/**
 * EchoServiceBidingImpl.java
 *
 * This file was auto-generated from WSDL
 * by the Apache Axis 1.2RC3 Nov 16, 2004 (12:19:44 EST) WSDL2Java emitter.
 */

package SOAPExample.serapi;

public class EchoServiceBidingImpl implements SOAPExample.serapi.EchoServiceOperations {
    public String echoText(String param) throws java.rmi.RemoteException {
        if (param == null || param.equals("")) {
            NoEchoTextProvided ex = new NoEchoTextProvided();
            ex.setFaultString("Echoed text seems to be an empty value, which is not allowed.");
            throw ex;
        }
        return "Echoed text: " + param;
    }
}

```

#### Esimerkki 14. EchoService-palvelun referenssitoteutus

Esimerkki 15 sisältää asiakassovelluksen, joka kutsuu esimerkin 13 referenssitoteutusta. Asiakassovellus käyttää wsdl2java-työkalun generoimia tynkäluokkia EchoOperations ja EchoServiceLocator.

```

package SOAPExample.serapi;

public class EchoTextClient {
    public EchoTextClient() {
        try {
            EchoServiceOperations echoService = new EchoServiceLocator().getEchoServiceSOAPPort();
            System.out.println(echoService.echoText("Kaiuta tämä teksti!!!"));
        } catch (Exception ex) {
            ex.printStackTrace();
        }
    }
    public static void main(String[] args) {
        EchoTextClient echoTextClient = new EchoTextClient();
    }
}

```

#### Esimerkki 15. EchoService-palvelua käyttävä asiakastoteutus

## **Liite 2: EchoService-palvelun WSDL-esityksen testiraportti**

Valitettavasti WS-I –testityökalujen generoima testiraportti on liian pitkä liitettäväksi tähän yhteyteen. EchoService-sovelluspalvelun WS-I testiraportti on nähtävissä osoitteessa <http://kettinki.uku.fi/EchoService/wsi-report.html>.