

WebServices LLynch endion ASP 2.3



Inhaltsverzeichnis

1. Einleitung	1
2. Operationen der Webservice-Schnittstelle	2
2.1. profileList-Operation	2
2.2. subscription-Operation	3
2.3. contactUpdate-Operation	4
2.4. unsubscription-Operation	4
2.5. importAndSendOut-Operation	5
2.6. progress-Operation	5
2.7. report-Operation	5
3. Anhang	6
3.1. Empfohlene Webservice-Frameworks	6
3.2. WSDL für die Webservices	6
Stichwortverzeichnis	22

Kapitel 1. Einleitung

Das WebServices-Handbuch für *LLynch endion ASP* erläutert die Operationen der Webservice-Schnittstelle und gibt Hilfestellungen bei der Integration in Ihre Anwendungen.

Es wird im Folgenden davon ausgegangen, dass die Konzepte und grundlegenden Funktionen von *LLynch endion ASP* bekannt sind. Sollte dies nicht der Fall sein, so finden Sie weitere Informationen hierzu im Benutzerhandbuch von Llynch endion ASP [<http://help.endion.biz/help2.3/index.jsp>].

Bei weitere Fragen oder Anregungen setzen Sie sich gerne mit uns in Verbindung:

NMMN - New Media Markets & Networks GmbH

Langbehnstraße 6

22761 Hamburg

Tel: 0 40 – 28 41 18-0

Fax: 0 40 – 28 41 18-999

E-Mail: software@llynch.net

<http://www.llynch.net>

Kapitel 2. Operationen der WebService-Schnittstelle

In diesem Kapitel werden alle Webservice-Operationen von *LLynch endion ASP* detailliert erläutert.

LLynch endion ASP setzt voraus das der Webservice-Client mit dem WS-Security-Standard [<http://www.oasis-open.org/committees/wss/>] (im Speziellen mit UsernameToken) umgehen kann. Dies ist Voraussetzung, um eine Authentifikation mit der Webservice-Schnittstelle von *LLynch endion ASP* herzustellen.

Exemplarisch wird an dieser Stelle einmalig ein SOAP-Header inklusive UsernameToken aufgeführt. In den folgenden Beispielen wird dieser Teil der SOAP-Nachricht weggelassen, um Ihnen die Übersicht zu erleichtern.



Beispiel-SOAP-Header für die UsernameToken-Authentifizierung

```
<soapenv:Envelope
  xmlns:sch="http://www.llynch.net/endion/schemas"
  xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/">
  <soapenv:Header>
    <wsse:Security soapenv:mustUnderstand="1"
      xmlns:wsse="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss- ...
      wssecurity-secext-1.0.xsd">
      <wsse:UsernameToken wsu:Id="UsernameToken-30298281"
        xmlns:wsu="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss- ...
        wssecurity-utility-1.0.xsd">
        <wsse:Username>username</wsse:Username>
        <wsse:Password Type="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-username- ...
        token-profile-1.0#PasswordText">
          password
        </wsse:Password>
        <wsse:Nonce>658vZckQHgKwPZMIqPQgpQ==</wsse:Nonce>
        <wsu:Created>2008-08-29T15:44:08.056Z</wsu:Created>
      </wsse:UsernameToken>
    </wsse:Security>
  </soapenv:Header>

  <!-- Hier folgt der soapenv:Body... -->

</soapenv:Envelope>
```

2.1. profileList-Operation

Die „profileList“-Operation gibt eine Liste aller in *LLynch endion ASP* gespeicherten Profile zurück. Zudem kann das Ergebnis durch Angabe eines bestimmten Profiltyps weiter eingeschränkt werden.

Zur Einschränkung des Suchergebnisses wird das optionale Element „type“ verwendet. Dieser kann die Werte „Newsletter“, „Interest“ oder „TargetGroup“ annehmen. Wenn das Element gesetzt ist werden lediglich die Profile zurückgegeben, die dem angegebene Profiltyp entsprechen, ansonsten erhalten Sie eine Liste **aller** bekannten Profile.



Beispiel für die profileList-Operation mit einem Filter für Newsletter Profile

```
<soapenv:Envelope
  xmlns:sch="http://www.llynch.net/endion/schemas"
  xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/">
  <soapenv:Body>
    <sch:profileListRequest>
      <sch:type>Newsletter</sch:type>
    </sch:profileListRequest>
  </soapenv:Body>
</soapenv:Envelope>
```

2.2. subscription-Operation

Die „subscription“-Operation stellt eine Zuordnung zwischen einem Kontakt und einem Profil her. Die Operation kann somit verwendet werden, um Kontakte für Profile (beispielsweise Newsletter-Versand-Profil) anzumelden. Ist das ausgewählte Profil auf Double-Optin-Modus eingestellt, wird diese Zuordnung mit dem Status „ausgewählt“ angelegt. Ist das Profil auf den Single-Optin-Modus eingestellt, wird der Status „bestätigt“ verwendet.

Innerhalb des „subscriptionRequest“-Elementes müssen jeweils ein „profileId“-Element und ein „email“-Element enthalten sein. Zusätzlich können beliebig viele „property“-Elemente aufgeführt werden. Die „property“-Elemente sind Name-Wert-Paare mit deren Hilfe zusätzliche Kontakteigenschaften gesetzt werden können.

Gültige Namen für „property“-Elemente sind:

- prefix_id
- firstname
- lastname
- personExternalReference
- street
- number
- zip
- city
- company
- companyExternalReference
- userdefined1
- userdefined2
- userdefined3
- userdefined4
- userdefined5
- Alle Benutzerfeldernamen, die in *LLynch endion ASP* konfiguriert sind



Beispiel für die subscription-Operation

```

<soapenv:Envelope
  xmlns:sch="http://www.llynch.net/endion/schemas"
  xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/">
  <soapenv:Body>
    <sch:subscriptionRequest>
      <!-- Anmelden einer Person für das Profil 100 -->
      <sch:profileId>100</sch:profileId>
      <sch:email>someone@example.com</sch:email>
      <sch:property>
        <sch:name>firstname</sch:name>
        <sch:value>Peter</sch:value>
      </sch:property>
    </sch:subscriptionRequest>
  </soapenv:Body>
</soapenv:Envelope>

```

2.3. contactUpdate-Operation

Die „contactUpdate“-Operation aktualisiert die Kontaktdaten eines ausgewählten Kontaktes in *LLynch endion ASP*.

Innerhalb des „contactUpdateRequest“-Elementes muss ein „email“-Element enthalten sein. Zusätzlich können - analog zur „subscription“-Operation - beliebig viele „property“-Elemente aufgeführt werden. Die „property“-Elemente sind Name-Wert-Paare über welche die zusätzlichen Kontakteigenschaften gesetzt werden können.

Die gültigen Namen für „property“-Elemente entsprechen denen der „subscription“-Operation. (Siehe Abschnitt 2.2)



Beispiel für die contactUpdate-Operation

```

<soapenv:Envelope
  xmlns:sch="http://www.llynch.net/endion/schemas"
  xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/">
  <soapenv:Body>
    <sch:contactUpdateRequest>
      <sch:email>someone@example.com</sch:email>
      <sch:property>
        <sch:name>firstname</sch:name>
        <sch:value>Peter</sch:value>
      </sch:property>
    </sch:contactUpdateRequest>
  </soapenv:Body>
</soapenv:Envelope>

```

2.4. unsubscription-Operation

Die „unsubscription“-Operation setzt den Status einer Kontakt-Profil-Zuordnung auf „gelöscht“. Die Operation kann beispielsweise dazu verwendet werden, um einen Kontakt von einem Newsletter abzumelden.

**Beispiel für die unsubscription-Operation**

```
<soapenv:Envelope
  xmlns:sch="http://www.llynch.net/endion/schemas"
  xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/">
  <soapenv:Body>
    <sch:unsubscriptionRequest>
      <sch:profileId>100</sch:profileId>
      <sch:email>someone@example.com</sch:email>
    </sch:unsubscriptionRequest>
  </soapenv:Body>
</soapenv:Envelope>
```

2.5. importAndSendOut-Operation



Diese Operation ist EXPERIMENTELL und kann sich in den folgenden *LLynch endion ASP* Versionen ändern.

Die „importAndSendOut“-Operation ermöglicht es, in einem Schritt Kontaktdaten zu importieren, einen Newsletter anzulegen und diesen zu versenden.



Um diesen Webservice zu nutzen, muss der Webservice-Client die Kontaktdaten und Newsletterdefinitionen per SOAP-Attachment mit MTOM-Erweiterung anhängen können.

2.6. progress-Operation



Diese Operation ist EXPERIMENTELL und kann sich in den folgenden *LLynch endion ASP* Versionen ändern.

Die „progress“-Operation gibt eine Liste mit den Fortschrittsinformationen der aktiven Hintergrundprozesse aus *LLynch endion ASP* zurück. Diese Liste kann auf bestimmte Hintergrundprozesse eingeschränkt werden.

2.7. report-Operation



Diese Operation ist EXPERIMENTELL und kann sich in den folgenden *LLynch endion ASP* Versionen ändern.

Die „report“-Operation gibt ein XML-Dokument zurück, welches die statistischen Daten zu einem bestimmten Newsletter enthält.



Um diesen Webservice zu nutzen, muss der Webservice-Client die XML-Datei per SOAP-Attachment mit MTOM-Erweiterung empfangen können.

Kapitel 3. Anhang

In diesem Kapitel sind zusätzliche Informationen und Referenzen aufgelistet, die die Implementation der *LLynch endion ASP-WebServices* vereinfachen und unterstützen können.

3.1. Empfohlene Webservice-Frameworks

Es wird empfohlen, eins der im Folgenden genannten Webservice-Frameworks zur Implementation eines Clients für die *LLynch endion ASP-WebServices* zu verwenden.

- Spring Web Services [<http://www.springframework.org/spring-ws>]
- WSO2 Web Services Framework for PHP [<http://www.wso2.org/projects/wsf/php>]

3.2. WSDL für die WebServices

Das WSDL-Dokument der Webservice-Schnittstelle ist unter der URL <http://esb.llynch.net:8193/endionService?wsdl> verfügbar. Der Vollständigkeit halber wird das Dokument aber auch an dieser Stelle aufgeführt.

```
<?xml version="1.0" encoding="UTF-8"?>
<wsdl:definitions
  xmlns:sch="http://www.llynch.net/endion/schemas"
  xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/"
  xmlns:tns="http://www.llynch.net/endion/definitions"
  xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/"
  targetNamespace="http://www.llynch.net/endion/definitions">
  <wsdl:types>
    <schema
      xmlns="http://www.w3.org/2001/XMLSchema"
      xmlns:tns="http://www.llynch.net/endion/schemas"
      xmlns:xmime="http://www.w3.org/2005/05/xmlmime"
      elementFormDefault="qualified"
      targetNamespace="http://www.llynch.net/endion/schemas">

    <simpleType name="clientId">
      <annotation>
        <documentation>
          LLynch endion ASP Client-ID
        </documentation>
      </annotation>

      <restriction base="integer"/>
    </simpleType>

    <simpleType name="profileId">
      <annotation>
        <documentation>
          LLynch endion ASP Profile-ID
        </documentation>
      </annotation>
      <restriction base="integer"/>
    </simpleType>

    <simpleType name="email">
```

```
<annotation>
  <documentation>
    E-Mail-Address
  </documentation>
</annotation>
<restriction base="string"/>
</simpleType>

<complexType name="property">
  <annotation>
    <documentation>
      Key-Value pair
    </documentation>
  </annotation>
  <sequence>
    <element name="name" type="string"/>
    <element name="value" type="string"/>
  </sequence>
</complexType>

<complexType name="response">
  <annotation>
    <documentation>
      General response base type.
    </documentation>
  </annotation>
  <sequence>

    <element name="code" type="integer">
      <annotation>
        <documentation>
          Error code for computational error checking.
        </documentation>
      </annotation>
    </element>
    <element name="message" type="string">
      <annotation>

        <documentation>
          Error description for display usage.
        </documentation>
      </annotation>
    </element>
  </sequence>
</complexType>

<complexType name="profile">
  <annotation>

    <documentation>
      Profile descriptor (currently profile type is missing).
    </documentation>
  </annotation>
  <sequence>
    <element name="id" type="integer"/>
    <element name="name" type="string"/>
  </sequence>
```

```
</complexType>

<complexType name="profileImport">
  <annotation>
    <documentation>
      Profile description for import. Contains contact data in
      csv format as SOAP-Attachment and a mapping description
      to link database fields with csv columns.
    </documentation>
  </annotation>
  <sequence>
    <element name="contacts" type="base64Binary"
      xmlns:expectedContentTypes="application/octet-stream">
      <annotation>

        <documentation>
          SOAP-attached CSV-file containing the contact data
          to import in this profile.
        </documentation>
      </annotation>
    </element>
    <choice maxOccurs="unbounded" minOccurs="0">
      <element name="mapping">
        <annotation>
          <documentation>

            Defines a database field to csv column mapping.
          </documentation>
        </annotation>
        <complexType>
          <attribute name="field" type="string">
            <annotation>
              <documentation>
                Database field.
              </documentation>
            </annotation>
          </attribute>
          <attribute name="column" type="integer">
            <annotation>
              <documentation>
                Zero indexed csv column.
              </documentation>
            </annotation>
          </attribute>
        </complexType>
      </element>
    </choice>
  </sequence>
  <attribute name="name" type="string">
    <annotation>
      <documentation>
        Name of the profile to import.
      </documentation>
    </annotation>
  </attribute>
</complexType>
```

```

<attribute default="false" name="update" type="boolean">
  <annotation>
    <documentation>
      True to update existing contact data, false to
      import only new contact data.
    </documentation>
  </annotation>
</attribute>
<attribute default="UTF-8" name="encoding" type="string">
  <annotation>
    <documentation>
      CSV-file encoding.
    </documentation>
  </annotation>
</attribute>
<attribute default=";" name="separator" type="string">
  <annotation>
    <documentation>
      CVS-file separator character.
    </documentation>
  </annotation>
</attribute>
<attribute name="emailColumn" type="integer">
  <annotation>
    <documentation>
      Column number of the email column in the CVS-file.
    </documentation>
  </annotation>
</attribute>
</complexType>

<complexType name="newsletterImport">
  <annotation>
    <documentation>
      Newsletter description/data to import.
    </documentation>
  </annotation>
  <sequence>

    <element name="name" type="string">
      <annotation>
        <documentation>
          Name of the newsletter to import.
        </documentation>
      </annotation>
    </element>
    <element name="from" type="string">
      <annotation>
        <documentation>
          E-Mail-Address used as sender address for the
          imported newsletter (has to be configured already
          in LLynch endion ASP).
        </documentation>
      </annotation>
    </element>
  </sequence>
</complexType>

```

```

</element>
<element name="subject" type="string">
  <annotation>
    <documentation>
      Newsletter subject line.
    </documentation>
  </annotation>
</element>
<element name="html" type="base64Binary"
  xmlns:expectedContentTypes="text/plain">
  <annotation>
    <documentation>
      SOAP-attached newsletter html-file for the html part.
    </documentation>
  </annotation>
</element>

<element name="text" type="base64Binary"
  xmlns:expectedContentTypes="text/plain">
  <annotation>
    <documentation>
      SOAP-attached newsletter text-file for the text part.
    </documentation>
  </annotation>
</element>

<choice maxOccurs="unbounded" minOccurs="0">
  <element name="tracking">
    <annotation>
      <documentation>
        Linktracking configuration for the imported
        newsletter.
      </documentation>
    </annotation>
    <complexType>
      <attribute name="type">
        <annotation>
          <documentation>
            Type of the link to configure. Could be
            'html' or 'text'.
          </documentation>
        </annotation>
        <simpleType>
          <restriction base="string">
            <enumeration value="html"/>
            <enumeration value="text"/>
          </restriction>
        </simpleType>
      </attribute>
      <attribute name="number" type="integer">
        <annotation>
          <documentation>
            Zero-indexed number of the link to track.
            A separate index is used for html and
  
```

```
    for text links.
  </documentation>
</annotation>
</attribute>
</complexType>

</element>
</choice>
</sequence>
</complexType>

<complexType name="progressInfo">
  <annotation>
    <documentation>
      A general progress info of a task currently running
      in LLynch endion ASP.
    </documentation>
  </annotation>
  <sequence>
    <element name="id" type="string">
      <annotation>
        <documentation>
          Task-ID this update is related to.
        </documentation>
      </annotation>
    </element>

    <element name="update" type="string">
      <annotation>
        <documentation>
          A JSON encoded update structure containing
          at least a percentage value and a text description.
        </documentation>
      </annotation>
    </element>
  </sequence>
</complexType>

<element name="profileListRequest">
  <annotation>
    <documentation>
      Parameters for the profile-list operation.
    </documentation>
  </annotation>
  <complexType>
    <sequence>
      <element maxOccurs="1" minOccurs="0" name="type">

        <annotation>
          <documentation>
            The type of profiles to list. This could be
            'Newsletter', 'Interest' or 'TargetGroup'.
            Also the type element is optional. If no type
            is specified all profiles are returned.
          </documentation>
        </annotation>
      </simpleType>
```

```
<restriction base="string">
  <enumeration value="Newsletter"/>
  <enumeration value="Interest"/>

  <enumeration value="TargetGroup"/>
</restriction>
</simpleType>
</element>
</sequence>
</complexType>
</element>
<element name="profileListResponse">
  <annotation>

  <documentation>
    Result for the profile-list operation.
    Extends the general response type so the elements
    'code' and 'message' are available.
  </documentation>
</annotation>
<complexType>
  <complexContent>
    <extension base="tns:response">
      <sequence>
        <annotation>

        <documentation>
          A list of matching profile elements.
        </documentation>
        </annotation>
        <element maxOccurs="unbounded" minOccurs="0"
          name="profile" type="tns:profile"/>
      </sequence>
    </extension>
  </complexContent>
</complexType>

</element>

<element name="subscriptionRequest">
  <annotation>
  <documentation>
    Parameters for the subscription operation.
  </documentation>
</annotation>
<complexType>
  <sequence>

  <element name="profileId" type="tns:profileId">
    <annotation>
    <documentation>
      Profile-ID of the profile to subscribe to.
      This ID could be looked up using the profileList
      operation.
    </documentation>
    </annotation>
  </element>
  <element name="email" type="tns:email">
```

```
<annotation>

  <documentation>
    E-Mail of the contact to subscribe.
  </documentation>
</annotation>
</element>
<choice maxOccurs="unbounded" minOccurs="0">
  <annotation>
    <documentation>
      Key-Value pairs of contact informations to be
      stored with subscription.
    </documentation>

    </annotation>
  <element name="property" type="tns:property"/>
</choice>
</sequence>
</complexType>
</element>
<element name="subscriptionResponse" type="tns:response">
  <annotation>
    <documentation>

      Result for the subscription operation.
      Extends the general response type so the elements
      'code' and 'message' are available.
    </documentation>
  </annotation>
</element>

<element name="contactUpdateRequest">
  <annotation>
    <documentation>
      Parameters for the contactUpdate operation.
    </documentation>

  </annotation>
  <complexType>
    <sequence>
      <element name="email" type="tns:email">
        <annotation>
          <documentation>
            E-Mail of the contact to update.
          </documentation>
        </annotation>

      </element>
    </sequence>
  </complexType>
</element>
<choice maxOccurs="unbounded" minOccurs="0">
  <annotation>
    <documentation>
      Key-Value pairs of contact informations to be
      updated.
    </documentation>
  </annotation>
  <element name="property" type="tns:property"/>
</choice>
```

```
</sequence>
</complexType>
</element>
<element name="contactUpdateResponse" type="tns:response">
  <annotation>
    <documentation>
      Result for the contactUpdate operation.
      Extends the general response type so the elements
      'code' and 'message' are available.
    </documentation>
  </annotation>
</element>

<element name="unsubscribeRequest">
  <annotation>
    <documentation>
      Parameters for the unsubscribe operation.
    </documentation>
  </annotation>
  <complexType>
    <sequence>

      <element name="profileId" type="tns:profileId">
        <annotation>
          <documentation>
            Profile-ID of the profile to unsubscribe from.
            This ID could be looked up using the profileList
            operation.
          </documentation>
        </annotation>
      </element>
      <element name="email" type="tns:email">
        <annotation>

          <documentation>
            E-Mail of the contact to unsubscribe.
          </documentation>
        </annotation>
      </element>
    </sequence>
  </complexType>
</element>
<element name="unsubscribeResponse" type="tns:response">
  <annotation>
    <documentation>
      Result for the unsubscribe operation.
      Extends the general response type so the elements
      'code' and 'message' are available.
    </documentation>
  </annotation>
</element>

<element name="importAndSendOutRequest">
  <annotation>
    <documentation>
```

Parameters for the importAndSendOut operation.

```
</documentation>
</annotation>
<complexType>
  <sequence>
    <element maxOccurs="1" minOccurs="1" name="profileData"
      type="tns:profileImport">
      <annotation>
        <documentation>
          Profile description which is to be imported.
        </documentation>
      </annotation>
    </element>
    <choice maxOccurs="1" minOccurs="1">
      <element name="newsletterData" type="tns:newsletterImport">
        <annotation>
          <documentation>
            Used to create a new newsletter after contact
            import and send this one.
          </documentation>
        </annotation>
      </element>
      <element name="newsletterId" type="integer">
        <annotation>
          <documentation>
            Used to send a predefined newsletter using
            its id after contact import.
          </documentation>
        </annotation>
      </element>
    </choice>
  </sequence>
</complexType>
</element>
<element name="importAndSendOutResponse">
  <annotation>
    <documentation>
      Result for the importAndSendOut operation.
      Extends the general response type so the elements
      'code' and 'message' are available.
    </documentation>
  </annotation>
</element>
<complexType>
  <complexContent>
    <extension base="tns:response">
      <sequence>
        <element name="progressId" type="string">
          <annotation>
            <documentation>
              ID of a LLynch endio ASP task to lookup
              the progress of the operation after
              invocation.
            </documentation>
          </annotation>
        </element>
      </sequence>
    </extension>
  </complexContent>
</complexType>
```

```

    </annotation>
  </element>
  <element name="newsletterId" type="string">
    <annotation>
      <documentation>
        ID of the created newsletter to lookup
        statistics after the operation invocation.
      </documentation>
    </annotation>
  </element>

  </sequence>
</extension>
</complexContent>
</complexType>
</element>

<element name="progressRequest">
  <annotation>
    <documentation>

      Parameters for the progress operation.
    </documentation>
  </annotation>
  <complexType>
    <sequence maxOccurs="1" minOccurs="0">
      <element name="id" type="string">
        <annotation>
          <documentation>
            ID of the task to lookup progress for.
            If no ID is given, then all current tasks
            will report their progress.
          </documentation>
        </annotation>
      </element>
    </sequence>
  </complexType>
</element>
<element name="progressResponse">
  <annotation>
    <documentation>
      Result for the progress operation.
      Extends the general response type so the elements
      'code' and 'message' are available.
    </documentation>
  </annotation>
  <complexType>
    <complexContent>
      <extension base="tns:response">
        <sequence maxOccurs="unbounded" minOccurs="0">
          <annotation>
            <documentation>
              A progress information for each requested
              LLynch endion ASP task.
            </documentation>
          </annotation>
        </sequence>
      </extension>
    </complexContent>
  </complexType>

```

```
</annotation>
<element name="progress" type="tns:progressInfo"/>
</sequence>
</extension>
</complexContent>
</complexType>
</element>

<element name="reportRequest">

<annotation>
<documentation>
Parameters for the report operation.
</documentation>
</annotation>
<complexType>
<sequence maxOccurs="1" minOccurs="1">
<element name="newsletterId" type="integer">
<annotation>

<documentation>
ID of the newsletter to fetch a report for.
</documentation>
</annotation>
</element>
</sequence>
</complexType>
</element>
<element name="reportResponse">

<annotation>
<documentation>
Result for the report operation.
Extends the general response type so the elements
'code' and 'message' are available.
</documentation>
</annotation>
<complexType>
<complexContent>
<extension base="tns:response">
<sequence maxOccurs="1" minOccurs="0">

<element name="report" type="base64Binary"
xmime:expectedContentTypes="text/xml">
<annotation>
<documentation>
SOAP-attached XML report document.
</documentation>
</annotation>
</element>
</sequence>
</extension>

</complexContent>
</complexType>
</element>

</schema>
```

```
</wsdl:types>
<wsdl:message name="contactUpdateRequest">
  <wsdl:part element="sch:contactUpdateRequest" name="contactUpdateRequest">
    </wsdl:part>
  </wsdl:message>

<wsdl:message name="contactUpdateResponse">
  <wsdl:part element="sch:contactUpdateResponse" name="contactUpdateResponse">
    </wsdl:part>
  </wsdl:message>
<wsdl:message name="progressResponse">
  <wsdl:part element="sch:progressResponse" name="progressResponse">
    </wsdl:part>
  </wsdl:message>
<wsdl:message name="profileListResponse">
  <wsdl:part element="sch:profileListResponse" name="profileListResponse">
    </wsdl:part>
  </wsdl:message>
<wsdl:message name="unsubscribeRequest">
  <wsdl:part element="sch:unsubscribeRequest" name="unsubscribeRequest">
    </wsdl:part>
  </wsdl:message>
<wsdl:message name="reportResponse">
  <wsdl:part element="sch:reportResponse" name="reportResponse">
    </wsdl:part>
  </wsdl:message>
<wsdl:message name="progressRequest">
  <wsdl:part element="sch:progressRequest" name="progressRequest">
    </wsdl:part>
  </wsdl:message>
<wsdl:message name="subscriptionRequest">
  <wsdl:part element="sch:subscriptionRequest" name="subscriptionRequest">
    </wsdl:part>
  </wsdl:message>
<wsdl:message name="subscriptionResponse">
  <wsdl:part element="sch:subscriptionResponse" name="subscriptionResponse">
    </wsdl:part>
  </wsdl:message>
<wsdl:message name="reportRequest">
  <wsdl:part element="sch:reportRequest" name="reportRequest">
    </wsdl:part>
  </wsdl:message>

<wsdl:message name="unsubscribeResponse">
  <wsdl:part element="sch:unsubscribeResponse"
    name="unsubscribeResponse">
    </wsdl:part>
  </wsdl:message>
<wsdl:message name="importAndSendOutRequest">
  <wsdl:part element="sch:importAndSendOutRequest"
    name="importAndSendOutRequest">
    </wsdl:part>
  </wsdl:message>
<wsdl:message name="importAndSendOutResponse">
```

```
<wsdl:part element="sch:importAndSendOutResponse"
  name="importAndSendOutResponse">
</wsdl:part>
</wsdl:message>
<wsdl:message name="profileListRequest">
  <wsdl:part element="sch:profileListRequest" name="profileListRequest">
</wsdl:part>
</wsdl:message>
<wsdl:portType name="endion">
  <wsdl:operation name="progress">

    <wsdl:input message="tns:progressRequest" name="progressRequest">
</wsdl:input>
    <wsdl:output message="tns:progressResponse" name="progressResponse">
</wsdl:output>
</wsdl:operation>
<wsdl:operation name="profileList">
  <wsdl:input message="tns:profileListRequest" name="profileListRequest">
</wsdl:input>
  <wsdl:output message="tns:profileListResponse" name="profileListResponse">

</wsdl:output>
</wsdl:operation>
<wsdl:operation name="subscription">
  <wsdl:input message="tns:subscriptionRequest" name="subscriptionRequest">
</wsdl:input>
  <wsdl:output message="tns:subscriptionResponse"
    name="subscriptionResponse">
</wsdl:output>
</wsdl:operation>
<wsdl:operation name="unsubscribe">

  <wsdl:input message="tns:unsubscribeRequest"
    name="unsubscribeRequest">
</wsdl:input>
  <wsdl:output message="tns:unsubscribeResponse"
    name="unsubscribeResponse">
</wsdl:output>
</wsdl:operation>
<wsdl:operation name="report">
  <wsdl:input message="tns:reportRequest" name="reportRequest">
</wsdl:input>
  <wsdl:output message="tns:reportResponse" name="reportResponse">

</wsdl:output>
</wsdl:operation>
<wsdl:operation name="importAndSendOut">
  <wsdl:input message="tns:importAndSendOutRequest"
    name="importAndSendOutRequest">
</wsdl:input>
  <wsdl:output message="tns:importAndSendOutResponse"
    name="importAndSendOutResponse">
</wsdl:output>
</wsdl:operation>
<wsdl:operation name="contactUpdate">

  <wsdl:input message="tns:contactUpdateRequest"
    name="contactUpdateRequest">
```

```
</wsdl:input>
  <wsdl:output message="tns:contactUpdateResponse"
    name="contactUpdateResponse">
  </wsdl:output>
</wsdl:operation>
</wsdl:portType>
<wsdl:binding name="endionSoap11" type="tns:endion">
  <soap:binding style="document"
    transport="http://schemas.xmlsoap.org/soap/http"/>
  <wsdl:operation name="progress">

    <soap:operation soapAction=""/>
    <wsdl:input name="progressRequest">
      <soap:body use="literal"/>
    </wsdl:input>
    <wsdl:output name="progressResponse">
      <soap:body use="literal"/>
    </wsdl:output>
  </wsdl:operation>
  <wsdl:operation name="profileList">

    <soap:operation soapAction=""/>
    <wsdl:input name="profileListRequest">
      <soap:body use="literal"/>
    </wsdl:input>
    <wsdl:output name="profileListResponse">
      <soap:body use="literal"/>
    </wsdl:output>
  </wsdl:operation>
  <wsdl:operation name="subscription">

    <soap:operation soapAction=""/>
    <wsdl:input name="subscriptionRequest">
      <soap:body use="literal"/>
    </wsdl:input>
    <wsdl:output name="subscriptionResponse">
      <soap:body use="literal"/>
    </wsdl:output>
  </wsdl:operation>
  <wsdl:operation name="unsubscribe">

    <soap:operation soapAction=""/>
    <wsdl:input name="unsubscribeRequest">
      <soap:body use="literal"/>
    </wsdl:input>
    <wsdl:output name="unsubscribeResponse">
      <soap:body use="literal"/>
    </wsdl:output>
  </wsdl:operation>
  <wsdl:operation name="report">

    <soap:operation soapAction=""/>
    <wsdl:input name="reportRequest">
      <soap:body use="literal"/>
    </wsdl:input>
    <wsdl:output name="reportResponse">
      <soap:body use="literal"/>
    </wsdl:output>
```

```
</wsdl:operation>
<wsdl:operation name="importAndSendOut">

  <soap:operation soapAction=""/>
  <wsdl:input name="importAndSendOutRequest">
    <soap:body use="literal"/>
  </wsdl:input>
  <wsdl:output name="importAndSendOutResponse">
    <soap:body use="literal"/>
  </wsdl:output>
</wsdl:operation>
<wsdl:operation name="contactUpdate">

  <soap:operation soapAction=""/>
  <wsdl:input name="contactUpdateRequest">
    <soap:body use="literal"/>
  </wsdl:input>
  <wsdl:output name="contactUpdateResponse">
    <soap:body use="literal"/>
  </wsdl:output>
</wsdl:operation>
</wsdl:binding>

<wsdl:service name="endionService">
  <wsdl:port binding="tns:endionSoap11" name="endionSoap11">
    <soap:address location="http://esb.llynch.net:8193/endionService"/>
  </wsdl:port>
</wsdl:service>
</wsdl:definitions>
```

Stichwortverzeichnis