



TNF

Technisch-Naturwissenschaftliche
Fakultät

WebServices

SICHERHEIT IN APPLIKATIONSPROTOKOLLEN

Eingereicht von:

Robert Emmer (0656105), Peter Wurm (0655445)

Angefertigt am:

Institut für Informationsverarbeitung und Mikroprozessortechnik

Beurteilung:

DI. Dr. Peter René Dietmüller

Linz, 18.11.2011

Inhaltsverzeichnis

1	Web Services.....	2
1.1	Einführung	2
1.2	Wichtige Merkmale	2
1.3	Beispielszenario.....	3
2	UDDI.....	4
2.1	Funktionsweise	4
2.2	Verzeichnis Kategorien	5
2.3	Informationen.....	5
2.4	Architektur	6
2.5	UDDI Register.....	6
2.6	Sicherheit.....	7
2.6.1	Allgemein.....	7
2.6.2	tModel.....	8
2.6.3	Identifikation	8
2.6.4	Authentifikation	9
2.6.5	Vertraulichkeit.....	9
3	WSDL	10
3.1	Was ist WSDL?.....	10
3.2	WSDL Dokumentenstruktur	10
3.3	Arten von Funktionen.....	11
4	SOAP.....	12
4.1	Was ist SOAP?	12
4.2	Aufbau einer Nachricht	12
4.3	Syntax.....	13
4.4	Beispiel einer SOAP Nachricht.....	13
4.5	Sicherheit eines Nachrichtenaustauschs.....	14

1 Web Services

Mit Webservices werden Systeme für die automatische, interne Kommunikation zwischen Web Anwendungen bezeichnet. Heute benutzt man diese Technologie für verschiedenste Arten von Anwendungen wie etwa Web 2.0 Anwendungen und Cloud Computing. Die Integration von mehreren Web Anwendungen, Koordinierung der Standards für die Datenübertragung, Protokolle, Plattformen und mehr, wird durch Webservices erleichtert. Diese Aspekte sind interessant für Organisationen um ihre Anwendungen miteinander zu koordinieren, egal welche spezifische Anwendung, Design oder Laufzeit Umgebung verwendet wird (ASP.NET, J2EE, COM, PHP,...). Die Verwendung von Web Services ist weit verbreitet, daher ist der Blick auf die Sicherheit solcher Systeme besonders wichtig.

Diese Arbeit befasst sich zuerst mit den Grundlagen von Webservices. Anschließend werden die relevanten Sicherheitskriterien wie etwa Authentifizierung, Echtheit und Vertraulichkeit von den Webservice Technologien SOAP, UDDI und WSDL behandelt.

1.1 Einführung

Ein Webservice ist eine Software Anwendung die spezielle Funktionen ausführt und anderen Komponenten über ein Netzwerk eine Auskunft über seine Fähigkeiten anbietet. Die serviceorientierte Architektur (SOA) von Webservices definiert, dass Webservices beschrieben (description), veröffentlicht (published), gefunden (discovered) und dynamisch (invoked) werden können. Web Services müssen eindeutig durch einen Uniform Resource Identifier (URI) identifizierbar sein, über den die Anwendungsschnittstelle definiert und lokalisiert werden kann. Ein wesentlicher Unterschied zwischen normalen Webseiten und Webservices ist, dass Webservices für sogenannte „unintelligent agents“ ausgelegt sind anstatt für Endbenutzer.

Webservices basieren auf einer Menge von Internet Standards, wobei die elementaren Standards WSDL (Web Service Definition Language), UDDI (Universal Description Discovery and Integration) und SOAP (Simple Object Access Protocol) sind. Jede dieser Webservice Technologien wird über die XML Schemasprache definiert und meist in Verbindung mit HTTP genutzt. Diese drei Technologien können miteinander verbunden werden um Anwendungen und Prozesse zu kreieren.

1.2 Wichtige Merkmale

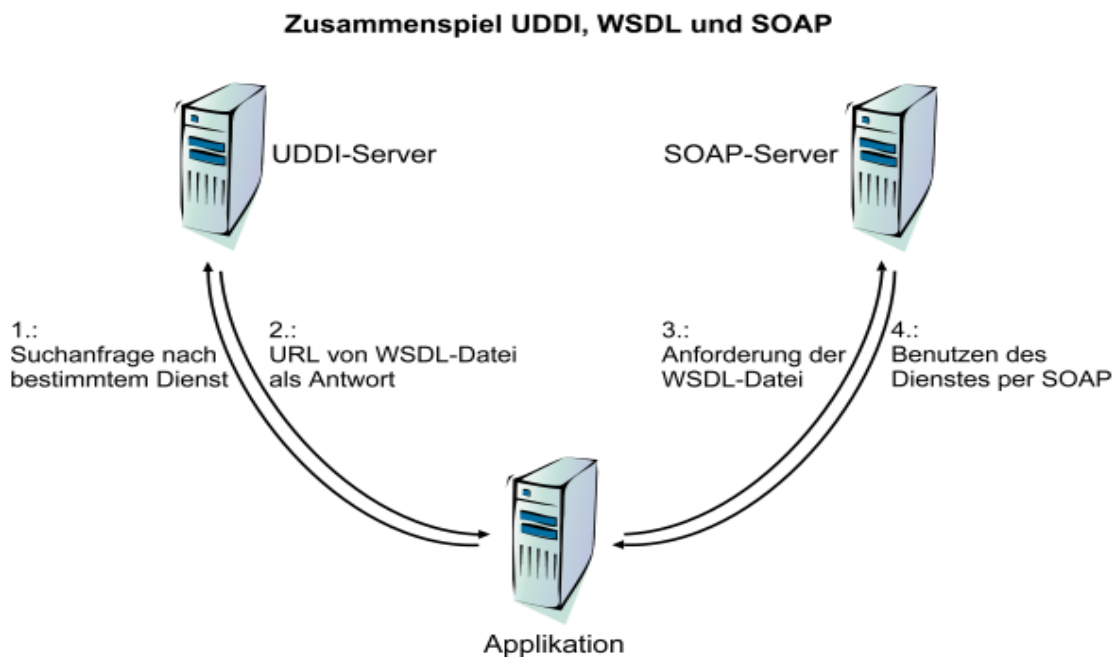
Die Technologie der Webservices ist eine zukunftsfähige Technologie, welche folgende entscheidende Merkmale besitzt.

- **Automatisierung:** Automatisieren von Diensten über Netzwerke (Internet bzw. Intranet)
- **Remote Control:** Funktionsaufruf auf fremden Rechnern wird ermöglicht
- **Standardisierung:** hoher Grad an Standardisierung ermöglicht auch Kommunikation über Plattformgrenzen hinweg (Microsoft / .NET ↔ Sun / Java)

- **Kombinierbarkeit:** leichte Kombinierbarkeit einzelner Softwarebausteine über SOAP bzw. WebServices
- **Skalierbarkeit:** Problemlose Erweiterung zu anspruchsvollen Systemen mit:
 - HTTP-Load-Balancing zur Performancesteigerung und Skalierung
 - leistungsfähigem Web Server (z.B. Apache oder IIS) für bewährte Sicherheitsmechanismen
 - einem Directory Service zur Authentifizierung und Autorisierung, zum Beispiel OpenLDAP, Microsoft NT Domains bzw. Active Directory Service oder Novell eDirectory

1.3 Beispielszenario

Die folgende Abbildung veranschaulicht die Interaktionen zwischen den Webservice Technologien. Es stellt die Suche eines Web Services per UDDI, die Untersuchung der Schnittstelle per WSDL und die Benutzung per SOAP dar.



2 UDDI

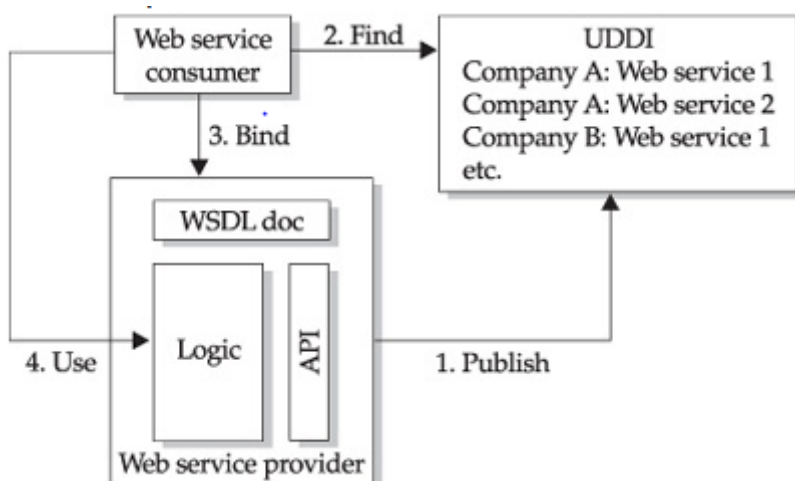
UDDI (Universal Description, Discovery and Integration) beschreibt ein Protokoll zum Veröffentlichen und Nachschlagen von Webservices. UDDI ist ein sogenannter Verzeichnisdienst mit einer SOAP Schnittstelle, der eine Übersicht über die gesamten Netzwerk-Ressourcen bietet. Webservices können dadurch dynamisch gesucht, gefunden und verwendet werden. Es gibt zahlreiche UDDI-Server und UDDI-Browser wo nach registrierten Web Services gesucht werden kann. Es wird auch ermöglicht selbst einen Eintrag erzeugen, sprich sein eigenes Webservice eintragen zu lassen (meist ist eine kostenlose Registrierung erforderlich).

- **Description:** Abrufen der Informationen über Unternehmen
- **Discovery:** Geschäftspartner ermitteln
- **Integration:** Anwendungen universell einsetzbar gestalten

2.1 Funktionsweise

Der Anbieter des Webservices veröffentlicht Informationen über den Dienst den er zur Verfügung stellen will, dazu werden die jeweiligen APIs benutzt. Anschließend können die Benutzer das Webservice im UDDI Verzeichnis nachschlagen. Es gibt viele UDDI Clients um Verzeichnisse nach Webservices zu durchsuchen, ein Beispiel wäre der Generic Soap Client. Der Eintrag in dem UDDI Verzeichnis zeigt auf das entsprechende WSDL Dokument des Webservice Anbieters.

Die folgende Grafik veranschaulicht das sogenannte "publish, find, bind" Konzept eines Webservices.



2.2 Verzeichnis Kategorien

Die Verwendung von UDDI ermöglicht die Benutzung von zwei verschiedenen Verzeichnistypen.

Public UDDI Directories

Öffentlich UDDI Verzeichnisse werden benutzt um Dienste der Öffentlichkeit zur Verfügung zu stellen. Zum Beispiel, die Internet Plattform xmethods.net stellt ein Verzeichnis mit verschiedensten Webservices zu Verfügung. Bevor ein Webservice mittels UDDI veröffentlicht wird, sollte man sich gut überlegen ob wirklich die „ganze Welt“ Zugriff darauf haben soll. Wenn ein Verzeichnis der jeweiligen Webservices benötigt wird, ist es generell besser eine private UDDI mit Authentifizierung zu erstellen.

Private UDDI Directories

Private UDDI Verzeichnisse werden gewöhnlich intern bei großen Konzernen verwendet. Diese Verzeichnisse sind meist nur zugänglich für Beschäftigte oder Partner des Konzerns. Ein internes Register liegt im allgemeinen hinter einer Firewall, und ist somit isoliert vom öffentlichen Netzwerk. Bei privaten Verzeichnissen wird oftmals angenommen, dass Sie ohnehin gegen Angriffe von außen geschützt sind und daher wird eher wenig Augenmerk auf die Sicherheit gelegt. Da aber auch solche Systeme durchaus geknackt werden können sollte auch hier auf die Implementierung der Sicherheit großer Wert gelegt werden.

2.3 Informationen

Die bereitgestellten Informationen von UDDI werden in die folgenden drei Hauptkategorien eingeteilt.

- **White Pages:** (Basisinformationen): geben Auskünfte über die Identität des Serviceanbieters, beispielsweise eine weltweit eindeutige Unternehmenskennzahl (Data Universal Numbering System - DUNS), Kontaktdaten, Namensregister und mehr.
- **Yellow Pages:** (Servicekategorisierung): Klassifizierung hinsichtlich des Zwecks des Dienstes, beispielsweise die Einteilung der Unternehmen hinsichtlich Branchen; Dienstleistung.
- **Green Pages:** liefern Schnittstellenbeschreibung eines Webservices. Speichert technische Informationen über Nachrichtenformate, Protokolle (WSDL).

2.4 Architektur

Die Architektur besteht im Wesentlichen aus einem Client und ein oder mehreren UDDI Anwendungen (Knoten) die eine Registratur bilden. Die folgenden Punkte beschreiben die Hauptklassen (Entities) aus denen UDDI zusammengesetzt ist.

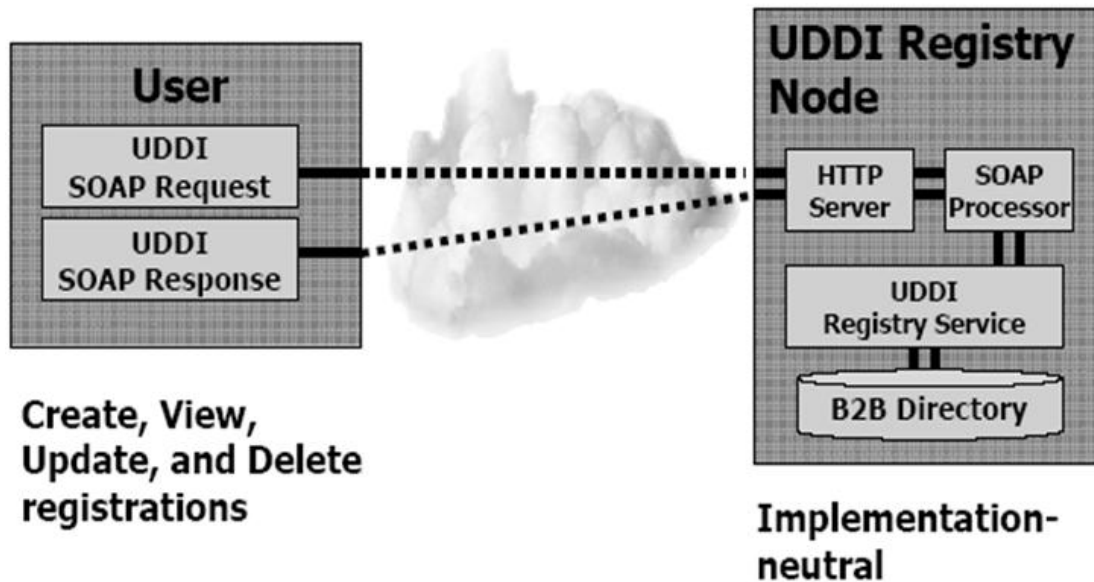
- **businessEntity:** Informationen über den Anbieter des Dienstes (Unternehmen, Person,...) veröffentlicht im UDDI Register
- **businessService:** Beschreibung der Serviceklassen, Klassifizierungs-Informationen, Beschreibung des Zwecks des Services
- **bindingTemplate:** Beschreibung der technischen Eigenschaften des Services, Service Zugriff und Metadaten
- **tModel (technical Model):** Verweis auf technische Anforderungen, Speicherung erfolgt mittels generischer Daten, Grundlage für technische Fingerabdrücke, beinhaltet Information über das verwendete Protokoll (HTTP, SMTP).
- **Publisher Assertion:** Beschreibung der Beziehung zwischen den Anbietern
- **Subscription:** Registrierung, Empfang von Änderungsmitteilungen (Änderungen im Registry)

2.5 UDDI Register

UDDI Register bestehen aus UDDI Konten, folgendes sind die verfügbaren APIs wobei laut Spezifikation mindestens eines davon implementiert werden muss.

- **Inquiry:** Suchanfragen
- **Publication:** Eintragen von Service Metadaten in ein Register
- **Security:** Sicherheit (Signaturen)
- **Custody Transfer:** Festlegung / Änderung der Besitzrechte an den verwalteten Daten
- **Subscription:** Anmelden von Nutzern, Update über Änderungen im Register benötigt
- **Replication:** Replizieren von Registern

- **SubscriptionListener:** Callback an Klienten, Information über bestimmte Änderungen an der Registry informieren.
- **Value Set:** externe Validierung von UDDI Daten (Neueinträge, Änderungen)



Diese obige Grafik veranschaulicht die UDDI SOAP Anfragen und Antworten zwischen dem Klienten und dem UDDI Register Node.

2.6 Sicherheit

Das Sicherheitsmodell des UDDI Registers ist geprägt durch eine Sammlung von Richtlinien betreffend die Implementierung der Register und Knoten. Es ist beispielsweise festgelegt, dass das binding-Template die Modellierung des Sicherheitsprotokolls übernimmt. Generell wird das gesamt Sicherheitskonzept über eine spezielle Sicherheit API und das tModel abgewickelt.

2.6.1 Allgemein

Da diese Web Technologie auf XML basiert empfiehlt es sich darauf zu achten, dass keine sensiblen Daten direkt in einer XML Datei gespeichert werden. Solche Daten können ansonsten relativ einfach in die falschen Hände geraten. Weiter ist es empfehlenswert eine Authentifizierung des Zugriffs zum Verzeichnis, welches die entsprechenden Daten enthält umzusetzen. Standard HTTP Authentifikation Techniken wie Basic, Digest, Windows Integrated und SSL Client-Side Zertifikate können verwendet.

2.6.2 tModel

Das tModel (technische Model) ist ein entscheidender Teil der digitalen Signatur, da hier die Zuweisung der Signatur implementiert ist. Nachfolgend werden die Elemente des tModels kurz beschrieben.

- **tModelKey**: Identifizierung des tModel
- **delete**: logisch gelöscht?
- **name**: Name des tModel mittels URL
- **description**: beschreibt das tModel
- **overviewDoc**: zusätzliche Merkmale
- **categoryBag**: technische Beschreibung
- **dsig:Signature**: digitale Signatur wird an businessEntity zugewiesen – dsig:Signature[0...*]

Beispiel eines tModel:

```
<tModel tModelKey="uddi:uddi.org:v3_publication">
  <name>uddi-org:publication_v3</name>
  <description>UDDI Publication API V3.0</description>
  <overviewDoc>
    <overviewURL useType="wsdlInterface">
      http://uddi.org/wsdl/uddi_api_v3_binding.wsdl#UDDI_Publication_SoapBinding
    </overviewURL>
  </overviewDoc>
  <overviewDoc>
    <overviewURL useType="text">
      http://uddi.org/pubs/uddi_v3.htm#PubV3
    </overviewURL>
  </overviewDoc>
  <categoryBag>
    <keyedReference keyName="uddi-org:types:wsdl"
      keyValue="wsdlSpec"
      tModelKey="uddi:uddi.org:categorization:types"/>
    <keyedReference keyName="uddi-org:types:soap"
      keyValue="soapSpec"
      tModelKey="uddi:uddi.org:categorization:types"/>
    <keyedReference keyName="uddi-org:types:xml"
      keyValue="xmlSpec"
      tModelKey="uddi:uddi.org:categorization:types"/>
    <keyedReference keyName="uddi-org:types:specification"
      keyValue="specification"
      tModelKey="uddi:uddi.org:categorization:types"/>
  </categoryBag>
</tModel>
```

2.6.3 Identifikation

Die Verwendungen eines Identifikationssystems um die Zugriffskontrolle zu unterstützen, kann mittels den entsprechenden Authentifikation und Autorisation APIs umgesetzt werden. UDDI selbst bietet zur Identifizierung ein System genannt DUNS (Data Universal Numbering

System) an. Dieses System bietet eine weltweit eindeutige Identifikation von einem Webservice an. Es hat sich bei den Standardisierungsgesellschaften (ANSI, ISO,...) etabliert und wird von Millionen Unternehmen benutzt.

2.6.4 Authentifikation

Die Sicherheit APIs von UDDI beinhaltet folgende Methoden um die Authentifikation zu implementieren.

- **discard_authToken:** Informiert einen UDDI Knoten, dass der vorher erhaltene Authentifizierung Token nicht mehr benötigt wird und als nicht mehr gültig angesehen wird.
- **get_authToken:** Stellt eine Anfrage nach einem Authentifizierung Token und gibt ein authInfo Element von einem UDDI Knoten zurück. Ein authInfo Element wird möglicherweise von dem Inquiry API Set, Publication API Set, Section Custody and Ownership Transfer API Set oder Subscription API Set benötigt.

2.6.5 Vertraulichkeit

Mit UDDIv3 wurde neue Vertrauensebene eingeführt, indem sämtliche UDDI - Basisdatentypen von dem Veröffentlichter signiert werden können. Ein Benutzer kann somit die Authentizität der Suchergebnisse selbst überprüfen. Die Anfrage Sprache für ein UDDIv3 Register kann bei Bedarf die Ergebnismenge so einschränkt, dass nur Einträge geliefert werden, die digital signiert sind. Auch innerhalb von Multi-Registry Umgebungen ist dieses Konzept wichtig, da somit sichergestellt werden kann, dass Daten zwischen Root-Register und Affiliate-Register nicht durch Dritte unbemerkt abgeändert worden sind.

3 WSDL

3.1 Was ist WSDL?

WSDL bedeutet **Web Services Description Language** und ist eine XML Beschreibung eines Web Service. In dieser Beschreibung werden die Zugriffsmöglichkeiten sowie die veröffentlichten Funktionen beschrieben.

3.2 WSDL Dokumentenstruktur

Die Struktur eines WSDL Dokuments basiert auf 4 verschiedenen Hauptelementen:

Element	Defines
<types>	The data types used by the web service
<message>	The messages used by the web service
<portType>	The operations performed by the web service
<binding>	The communication protocols used by the web service

Typen

Dieser Bereich beschreibt die Datentypen des Web Services. Um plattformunabhängig in Bezug auf Datentypen zu sein, werden diese Datentypen in XML beschrieben.

Nachrichten

Im message-Bereich werden die Datentypen einer Operation beschrieben, vergleichbar mit der Parameterliste bei einem Funktionsaufruf.

Ports

Ein Port beschreibt ein Web Service und dessen Operationen und kann mit einer Funktionsbibliothek verglichen werden.

Bindungen

Definiert das Nachrichtenformat und Protokoll Details für einen Port.

3.3 Arten von Funktionen

Der meistgenutzte Funktionstyp von WSDL ist „request-response“, allerdings sind 3 weitere Typen definiert:

- One-way

Die Funktion empfängt eine Nachricht, gibt aber keinen Rückgabewert aus.

```
<message name="newTermValues">
  <part name="term" type="xs:string"/>
  <part name="value" type="xs:string"/>
</message>

<portType name="glossaryTerms">
  <operation name="setTerm">
    <input name="newTerm" message="newTermValues"/>
  </operation>
</portType >
```

- Request-response

Die Funktion empfängt eine Nachricht und gibt eine Antwort zurück.

```
<message name="getTermRequest">
  <part name="term" type="xs:string"/>
</message>

<message name="getTermResponse">
  <part name="value" type="xs:string"/>
</message>

<portType name="glossaryTerms">
  <operation name="getTerm">
    <input message="getTermRequest"/>
    <output message="getTermResponse"/>
  </operation>
</portType>
```

- Solicit-response

Die Operation sendet eine Anfrage und wartet auf eine Antwort

- Notification

Die Operation sendet eine Nachricht, erwartet aber keine Antwort.

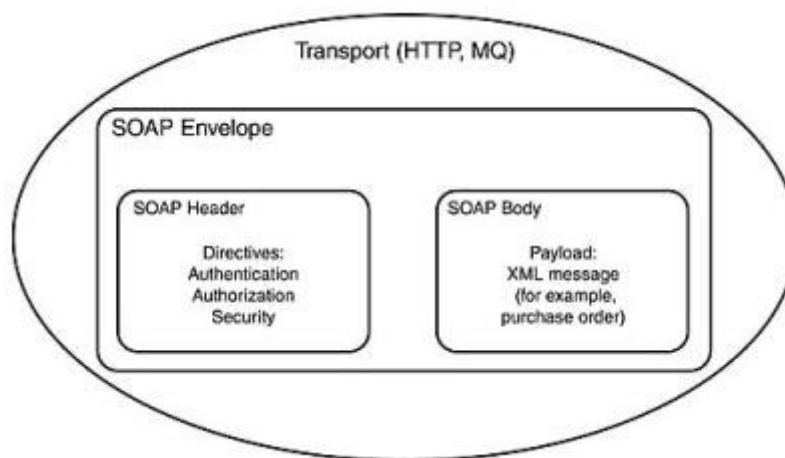
4 SOAP

4.1 Was ist SOAP?

SOAP bedeutet **S**imple **O**bject**A**ccess **P**rotocol und ist ein XML-basiertes Protokoll zum Austausch von Nachrichten über HTTP. Es bietet Applikationen eine Möglichkeit zum Austausch von Nachrichten übers Internet und kann durch den Transport via HTTP durch Firewalls geschleust werden. Es wird für Anwendungen mittels SOAP möglich, über das Internet plattformunabhängig Nachrichten und Services auszutauschen.

4.2 Aufbau einer Nachricht

SOAP ist für die Übermittlung von Nachrichten zwischen Applikationen übers Internet zuständig. Im XML Format werden diese Nachrichten folgendermaßen strukturiert:



- **SOAP Envelope**
Diese "Umhüllung" kennzeichnet die XML Syntax als SOAP Nachricht und erleichtert einer Anwendung damit den Datenaustausch in einem festgelegten Format, unabhängig von der Art des Transports der Nachricht.
- **SOAP Header**
Der Header enthält Metainformationen über die SOAP Nachricht und beschreibt die enthaltenen Anfragen oder Antworten. Über den SOAP Header werden die beinhalteten Daten verwaltet und beschrieben. Erweiterungen und Sicherheitsmaßnahmen finden hier ihren Platz.
- **SOAP Body**
Im Body einer Nachricht findet sich der Inhalt dieser Nachricht, also den Gegenstand oder das Objekt, das eine Applikation einer anderen sendet. Dieser Inhalt kann dabei ein XML Dokument sein, das eine Bestellung oder einen Vertrag – also ein konkretes Objekt – beinhaltet, oder ein Remote Procedure Call spezifiziert durch die aufzurufenden Funktionen und die dazugehörigen Parameter.
Im Body gibt es außerdem noch ein sogenanntes Fault Element, das Fehler und ihre Stati, sowie Beschreibung zum Fehler und Fehlercode enthält.

4.3 Syntax

Für den Aufbau einer SOAP Nachricht gibt es einige syntaktische Regeln:

- Eine SOAP Nachricht MUSS im XML Format geschrieben sein
- Eine SOAP Nachricht MUSS im SOAP Envelope Namespace sein
- Eine SOAP Nachricht MUSS den SOAP Encoding Namespace verwenden
- Eine SOAP Nachricht darf keine DTD Definitionen enthalten
- Eine SOAP Nachricht darf keine XML Verarbeitungsinstruktionen enthalten

```
<?xml version="1.0"?>
<soap:Envelope
xmlns:soap="http://www.w3.org/2001/12/soap-envelope"
soap:encodingStyle="http://www.w3.org/2001/12/soap-encoding">

<soap:Header>
...
</soap:Header>

<soap:Body>
...
  <soap:Fault>
    ...
  </soap:Fault>
</soap:Body>

</soap:Envelope>
```

4.4 Beispiel einer SOAP Nachricht

Ein Beispiel für eine SOAP Anfrage ist der Aufruf einer Funktion „GetStockPrice“ im Namespace www.example.org/stock mit dem Parameter „StockName“. Es wird hier also einer Funktion der Name einer Aktie übergeben und als Antwort bekommt man den aktuellen Wert zu diesem Wertpapier.

Anfrage über HTTP und XML (=SOAP):

```
POST /InStock HTTP/1.1
Host: www.example.org
Content-Type: application/soap+xml; charset=utf-8
Content-Length: nnn

<?xml version="1.0"?>
<soap:Envelope
xmlns:soap="http://www.w3.org/2001/12/soap-envelope"
soap:encodingStyle="http://www.w3.org/2001/12/soap-encoding">

<soap:Body xmlns:m="http://www.example.org/stock">
  <m:GetStockPrice>
    <m:StockName>IBM</m:StockName>
  </m:GetStockPrice>
</soap:Body>

</soap:Envelope>
```

Antwort auf die Anfrage:

```
HTTP/1.1 200 OK
Content-Type: application/soap+xml; charset=utf-8
Content-Length: nnn

<?xml version="1.0"?>
<soap:Envelope
xmlns:soap="http://www.w3.org/2001/12/soap-envelope"
soap:encodingStyle="http://www.w3.org/2001/12/soap-encoding">

  <soap:Body xmlns:m="http://www.example.org/stock">
    <m:GetStockPriceResponse>
      <m:Price>34.5</m:Price>
    </m:GetStockPriceResponse>
  </soap:Body>

</soap:Envelope>
```

4.5 Sicherheit eines Nachrichtenaustauschs

Für einen sicheren Nachrichtenaustausch über Medien wie das Internet sind im Allgemeinen folgende Aspekte zu berücksichtigen:

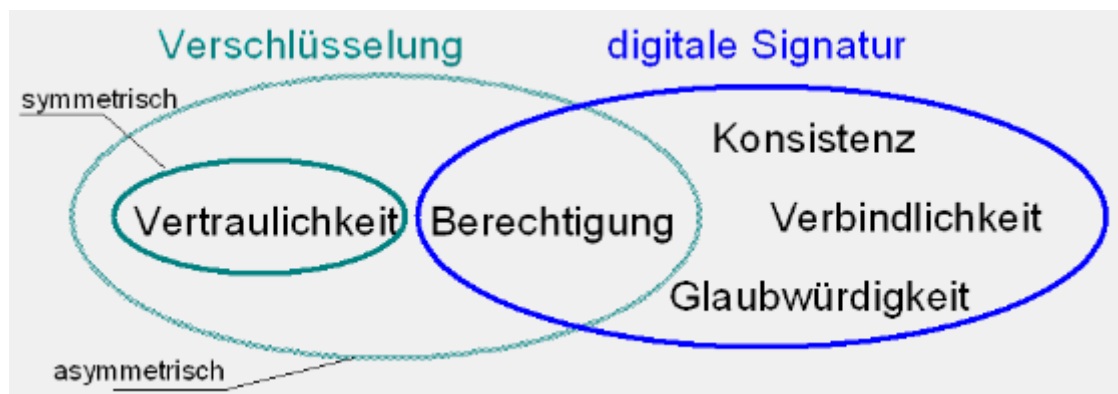
- **Vertraulichkeit:**
Eine Nachricht soll ausschließlich für den jeweils adressierten Empfänger lesbar sein.
- **Berechtigung:**
Ein Dienstanforderer muss auch zu dessen Nutzung berechtigt sein. Insbesondere soll es einem potentiellen Eindringling nicht möglich sein, eine andere Identität vorzutäuschen.
- **(Daten-)Konsistenz:**
Eine versandte Nachricht muss ohne Modifikationen beim Empfänger eintreffen.
- **Glaubwürdigkeit:**
Eine Nachricht muss nachprüfbar durch den vermeintlichen Sender erstellt worden sein.
- **Verbindlichkeit:**
Der Sender soll im Nachhinein die Urheberschaft einer durch ihn erstellten Nachricht nicht leugnen können.

Zur Realisierung der verschiedenen Sicherheitsaspekte haben sich eine Reihe technischer Lösungen herausgebildet, die sich grob in zwei Verfahrensklassen einordnen lassen:

- Kryptographie und
- Digitale Unterschrift.

Während durch kryptographische Bearbeitung der ursprüngliche Nachrichtentext so modifiziert wird, dass er für einen Angreifer nicht (oder nur mit großem Aufwand) lesbar ist, zielt der Einsatz digitaler Unterschriften auf die Bestätigung der Echtheit der inhaltlich unverändert übermittelten Information ab.

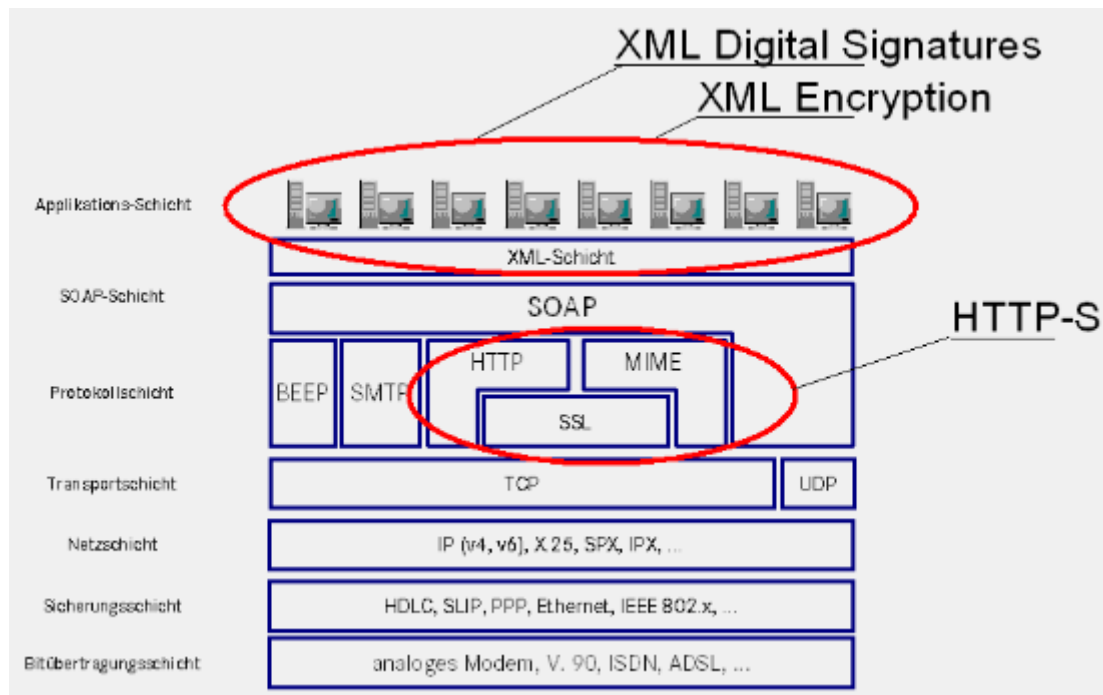
Für die konkrete Datenübertragung mittels SOAP wird also eine Kombination aus Sicherheitsmechanismen nötig, um die Anforderung an eine sichere Datenübertragung erfüllen zu können.



Da im Fall SOAP Nachrichten notwendigerweise im XML Format ausgetauscht werden, ergeben sich für die Sicherheit gewissen Besonderheiten. Am augenfälligsten ist sicherlich zunächst die durchgängige Formatierung der übertragenen Daten als XML-Dokument. Zwar handelt es sich dabei lediglich um einen Unicode-Textstrom; jedoch ergibt sich aus der Anlage der XML selbst bereits ein Gesichtspunkt, dem bei der Absicherung der Inhalte Aufmerksamkeit zuteilwerden muss: die mögliche fein-granulare Steuerung der Sicherheit. Durch die in XML realisierte Separierung von Inhalten und beschreibender Information, die in XML-Elemente und -Attribute zerfällt, lässt sich bereits intuitiv eine Trennlinie ausmachen. So ist es beispielsweise denkbar, lediglich die Nutzinformation eines Dokuments zu verschlüsseln und dabei die Tags unberührt zu belassen. Genauso können wahlfrei auch ganze Elemente unter Einbeziehung ihrer Kind Elemente, mithin abgegrenzte Dokumentteile, kryptographisch bearbeitet werden.

Derselbe Freiheitsgrad steht auch für die Anbringung von digitalen Signaturen zur Verfügung. So können sie in variabler Granularität von Einzelementen bis hin zum gesamten Dokument angebracht werden.

In Anlehnung an die Schichten des bekannten ISO-OSI Referenzmodells lässt sich SOAP primär als Protokoll der Darstellungsschicht einordnen. Allerdings kann - je nach gebundenem Netzprotokoll, etwa bei den Transportprotokollen TCP oder UDP - auch die Einordnung von SOAP als Protokoll der Sitzungsschicht erfolgen.



Generell lässt sich die Sicherheit der übermittelten Nachrichten „unterhalb“ bzw. „oberhalb“ der SOAP-Schicht einbringen. Zeitlich der SOAP-Nachrichtenerzeugung nachgelagert und damit im Referenzmodell unterhalb der SOAP-Schicht angeordnet sind Sicherheitsmechanismen wie der Secure Sockets Layer (SSL) zu sehen. Diese operieren auf beliebigen Datenströmen, welche durch die darüber liegenden Protokollschichten erzeugt und über definierte Schnittstellen bereitgestellt werden. Die Benutzung dieser Klasse von Kommunikationsabsicherung ist für die SOAP verwendenden Applikationen im Betrieb vollkommen transparent und erfordert nur minimale Eingriffe in den Kommunikationsablauf.

Wird hingegen die Absicherung der im SOAP-Umschlag enthaltenen Daten mit Mitteln der XML-Technologie gewünscht, so sind Eingriffe in die Applikation oder zumindest die Kommunikationskomponente notwendig. Hierfür existieren die beiden eingangs skizzierten Ansätze der XML-Signaturen und der XML-Verschlüsselung, die sich vergleichsweise einfach in das SOAP-Umfeld integrieren lassen.

Beispiel einer verschlüsselten SOAP Nachricht

```
(1) <SOAP-ENV:Envelope xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/"
(2) xmlns:xsd="http://www.w3.org/2001/XMLSchema"
(3) xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
(4) <SOAP-ENV:Body>
(5)   <EncryptedData xmlns="http://www.w3.org/2001/04/xmlenc#"
(6)     Type="http://www.w3.org/2001/04/xmlenc#Element">
(7)     <EncryptionMethod Algorithm="http://www.w3.org/2001/04/xmlenc#tripledes-cbc">
(8)       <IV>ZG9uJ3RTcHk=</IV>
(9)     </EncryptionMethod>
(10)    <CipherData>
(11)    <CipherValue>sVjhJIW5QnIeY3brbpamNOcz/Ja+RmnG0pLo7vWnmTp+vpVs53c0Y
(12)    VDeb4gmYEcoBTAe00S8l0cySjPkgkqbVksD9zo6U2LpS466KXIp5NDVRgJcHZnp8tr
(13)    o5mb90g2gB56bw+IyskKh7QDMbvM7ACdT6SGauu0dSGIT3Q5kTT1qQWQ4easDZ1Shh
(14)    pVYrBXPRI//3QGyjrnoOclh8T5eqRhRRAuwUc3A4RqaH7M6QTIb36vOTBRuFbfFESB
(15)    w8648Xi8G1Spu39cVoMjEUTrlDnJo1gpWdU5JWFF9RqzhmBQ=</CipherValue>
(16)    </CipherData>
(17)  </EncryptedData>
(18) </SOAP-ENV:Body>
(19) </SOAP-ENV:Envelope>
```

Literaturverzeichnis

- [1] Hacking Exposed Web Applications– Chapter 7Attacking XML Web Services
- [2]<http://www.torsten-horn.de/techdocs/soap.htm>
- [3]<http://www.bitpipe.com/tlist/UDDI-Specification.html>
- [4]http://www.uddi.org/pubs/uddi_v3.htm
- [5]<http://msdn.microsoft.com/de-de/library/bb979447.aspx>
- [6]http://www.tecchannel.de/webtechnik/soa/472223/verzeichnisdienste_fuer_web_services_ws_inspection_und_uddi/
- [7]<http://han.ubl.jku.at/han/ieeexploreiel/ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=4024048>
- [8]<http://www.w3schools.com/wsd/>
- [9] <http://www.w3schools.com/soap/>
- [10] <http://www.jeckle.de/secureSOAP.html> (Stand 21.11.2011)