SOAP-Faults und deren Behandlung		Konvention xml-sf 1.0.6 Empfehlung
Kurzbeschreibung	Im Rahmen der Kommunikation mittels SOAP-WebServices tritt die Problematik über den Umgang mit SOAP-Faults zutage. Dieses Dokument zielt nun darauf ab, eine einheitliche Vorgehensweise zu erstellen, wie SOAP-Faults organisatorisch und praktisch behandelt werden. Weiters soll eine einheitliche Übersicht über Fehlergruppen und über applikationsspezifische Erweiterungen derselben gegeben werden.	
Autor(en)	**Michael Liehmann** **Franz-Josef Herpers**	**Projektteam / Arbeitsgruppe** **Kommarch/SoapFaults**

1

Stelle	Vorgelegt am	Angenommen am	Abgelehnt am
IKT-Board **Länder** **Gemeindebund** **Städtebund**	**18.04.2005** **18.04.2005** **18.04.2005** **18.04.2005**	**04.05.2005** **04.05.2005** **04.05.2005** **04.05.2005**	

2

SOAP-Faults

Inhaltsverzeichnis

(1) Einleitung

Problemstellung und Ziel

WebServices sind Applikationen die einen programmatischen Zugang zu einer Fachapplikation über das Internet anbieten. SOAP(Simple Object Access Protocol [SOAP11], [SOAP12]) bezeichnet ein Protokoll, mit dem WebServices angesprochen werden können. Eine Vielzahl an Applikationen im Bereiche E-Government verwenden nun jene SOAP-WebServices zur Kommunikation zwischen Applikation und Client bzw. zwischen einzelnen Applikationen. Im Rahmen dieser Verbindungen treten bisweilen Probleme auf. Etwaige Fehlermeldungen können mittels SOAP-Faults an die aufrufende Applikation weitergegeben werden. In den üblichen Entwicklungsumgebungen, wie JAVA AXIS und MS .NET, führen diese SOAP-Faults zu applikationsspezifischen Exceptions. Manche Fehlermeldungen sollen zu einer solchen Exception führen, bei anderen Fehlern ist es anzuraten, diese auf andere Weise zu übermitteln.

In diesem Papier wird eine technische und applikationsübergreifende Einteilung getroffen, welche Fehler SOAP-Faults darstellen und somit Exceptions erzeugen und welche Meldungen besser als ordentliche SOAP-Meldung an die anfragende Applikation zurückgegeben werden sollen.

(2) SOAP-Faults Syntax

Laut der SOAP-Spezifikation des W3C soll ein SOAP-Fault Fehlerinformationen innerhalb einer normalen SOAP-Nachricht in dem Element *env:Fault* übertragen. Da die simple Existenz dieses Elements in einer SOAP-Nachricht nur auf einen Fehler hinweist, aber nicht auf welchen bestimmten, ist diesem Element eine Struktur zugeordnet, die genauere Aufschlüsse zulässt. Die SOAP-Fault Syntax veränderte sich jedoch von der Version in SOAP 1.1 auf die Version in SOAP 1.2. Zum bessern Verständnis werden hier beide Strukturen kurz vorgestellt.

(2.1) SOAP-Faults in SOAP 1.1

Abbildung 1 beschreibt das XML-Element SOAP-Faults laut SOAP-Spezifikation 1.2.

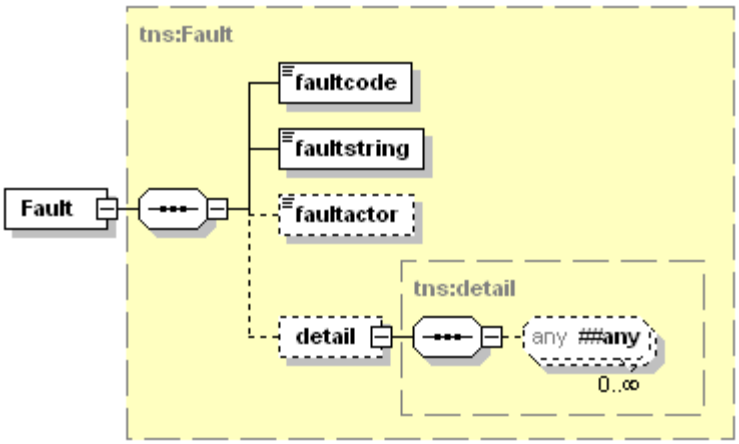

Abbildung 1

Im Folgenden eine Beschreibung der einzelnen Elemente:

- **<faultcode>** (mandatory)
  SOAP-Fault Codes helfen bei einer Klassifizierung der SOAP Fehlermeldungen. Mögliche Fault Codes sind:

- VersionMismatch: Der SOAP-*Envelope* verwendet einen ungültigen Namensraum für das SOAP-Element *Envelope*. Mit diesem Fehler wird in SOAP auf Protokollversionierungprobleme hingewiesen.
- MustUnderstand: Ein Headerblock, der das Flag *mustUnderstand=true* enthält, wurde vom Empfänger der Nachricht nicht verstanden.
- Client: Der Empfänger hat ein Problem mit der SOAP-Nachricht und kann diese nicht abarbeiten. Üblicherweise liegt bei dieser Art von Fehler das Problem beim Client.
- Server: Es ist ein Fehler aufgetreten, der nicht unmittelbar mit der Verarbeitung der Nachricht in Verbindung gebracht werden kann.

Detaillierter Beschreibungen sind der Spezifikation zu entnehmen [SOAP11].

- **<faultstring>** (mandatory)
  In diesem Element befindet sich eine lesbare Erläuterung des Fehlers.
- **<faultactor>** (optional)
  Dieses Element zeigt, mittels eines xs:anyURI Elements, auf jenen Knoten, welcher den Fehler verursachte. Es ist durchaus möglich, dass ein Webservice die Verarbeitung der Nachricht an ein anderes Webservice auslagert bzw. abgibt; somit wird eine Kaskadierung von WebServices erstellt. Tritt nun an einer Stelle dieser Kaskade ein Fehler auf, insofern ein WebService nicht antwortet, so wird jene Adresse des Knotens in diesem Element angegeben.
- **<detail>** (optional)
  Applikationsspezifische Fehlerinformationen, deren Syntax keinem Schema obliegt. Dieses Element muss inkludiert werden, falls der Fehler von einem Element innerhalb des *Body*-Elements verurscaht wurde. Er darf nicht inkludiert werden, wenn der Fehler bei der Abarbeitung des Headers auftrat.

Ein HTTP-Response mit einer SOAP-Fehlermeldung laut SOAP 1.1 hat folgendes Aussehen:

```
HTTP/1.1 500 Internal Server Error
Content-Type: text/xml; charset="utf-8"
Content-Length: nnnn
<env:Envelope xmlns:env="http://schemas.xmlsoap.org/soap/envelope/">
  <env:Body>
        <env:Fault>
              <faultcode>env:Server</faultcode>
              <faultstring>Server Error</faultstring>
              <faultactor>http://www.foo.com</faultactor>
              <detail>
                    <e:myfaultdetails xmlns:e="Some-URI">
                          <message>My application didn't work</message>
                          <errorcode>1001</errorcode>
                    </e:myfaultdetails>
              </detail>
        </env:Fault>
  </env:Body>
</env:Envelope>
```

(2.2) SOAP-Faults in SOAP 1.2

Abbildung 2 beschreibt das XML-Element SOAP-Faults laut SOAP-Spezifikation 1.2.:

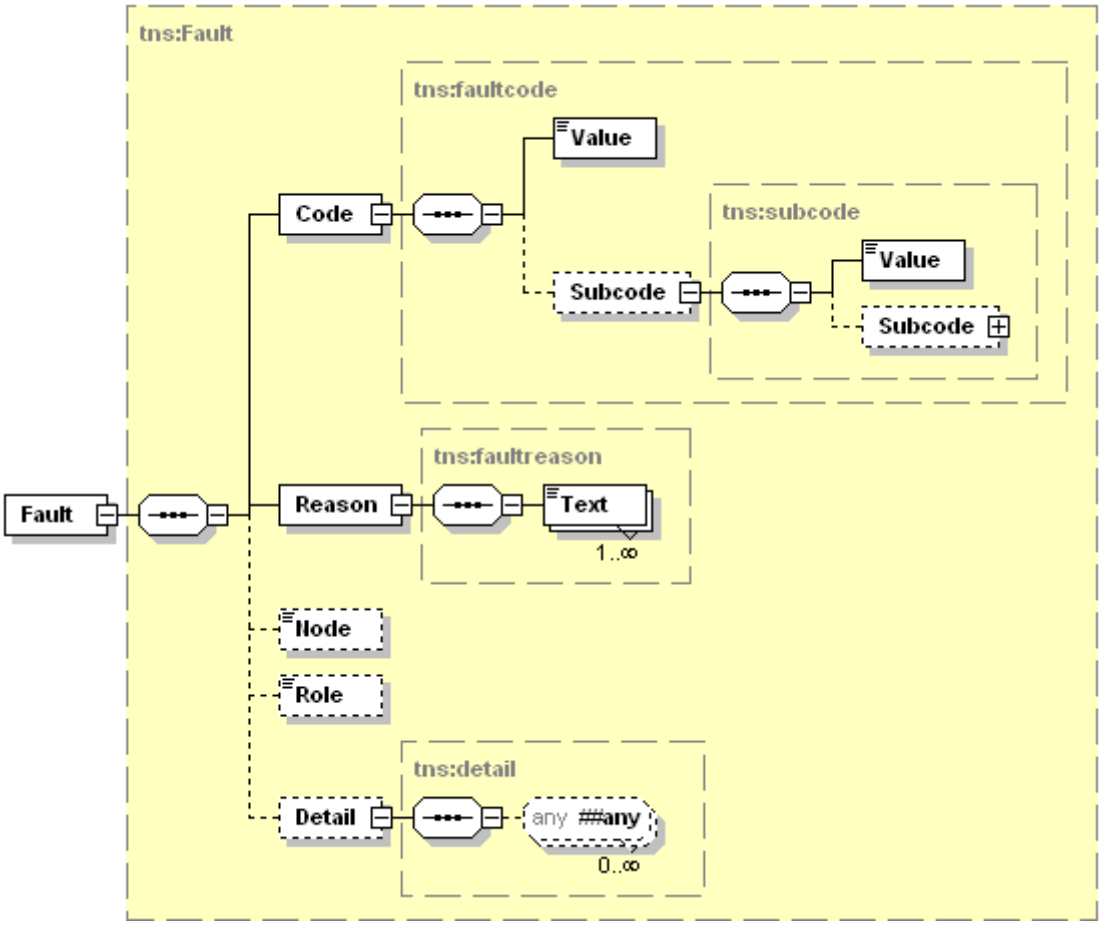

Abbildung 2

Im Folgenden eine Beschreibung der einzelnen Elemente:

- **<Code>** (mandatory)
  SOAP-Fault Codes helfen bei einer hierarchischen Klassifizierung der SOAP-Fehlermeldungen. Die Codierung kann beliebig granular mittels Subcodes definiert werden. Mögliche Fault Codes sind:
  - VersionMismatch: siehe SOAP 1.1
  - MustUnderstand: siehe SOAP 1.1
  - DataEncodingUnknown: Dieses Element zeigt einen Fehler des Empfängers auf, der durch einen nicht erkennbaren Wert des *encodingStyle* Attributs in der SOAP-Message verursacht wurde.
  - Sender: siehe SOAP 1.1
  - Receiver: siehe SOAP 1.1

  Eine detailliertere Beschreibung ist wiederum der Spezifikation zu entnehmen [SOAP12].

- **<Reason>** (mandatory)
  In diesem Element befindet sich eine oder mehrere lesbare Erklärungen des Fehlers.
- **<Node>** (optional)
  Siehe SOAP 1.1 **<faultactor>**.
- **<Role>** (optional)
  Hier wird die Rolle des in **<Node>** bezeichnetet Knotens beschrieben, in dem der Knoten zum Zeitpunkt des Fehlers agierte.
- **<Detail>** (optional)
  Applikationsspezifische Fehler Informationen, deren Syntax keinem Schema obliegt.

Ein HTTP-Response mit einer SOAP-Fehlermeldung laut SOAP 1.2 hat folgendes Aussehen:

```
HTTP/1.1 500 Internal Server Error
Content-Type: text/xml; charset="utf-8"
Content-Length: nnnn
<env:Envelope xmlns:env="http://www.w3.org/2003/05/soap-envelope"
xmlns:m="http://www.example.org/timeouts"
xmlns:xml="http://www.w3.org/XML/1998/namespace">
  <env:Body>
          <env:Fault>
                  <env:Code>
                          <env:Value>env:Sender</env:Value>
                          <env:Subcode>
                                  <env:Value>m:MessageTimeout</env:Value>
                          </env:Subcode>
                  </env:Code>
                  <env:Reason>
                          <env:Text xml:lang="en">Sender Timeout</env:Text>
                  </env:Reason>
                  <env:Detail>
                          <m:MaxTime>P5M</m:MaxTime>
                  </env:Detail>
          </env:Fault>
  </env:Body>
</env:Envelope>
```

(3) Fehlerunterscheidung

Sowohl laut der SOAP Spezifikation 1.1, wie auch laut Version 1.2 wird ein SOAP-Fehler über HTTP mit dem HTTP-Statuscode 500 (InternalServerError) [HTTP] zurückgesandt. Dies geschieht auch falls der Fehler auf der Clientseite liegt.

Da nicht immer ein interner Server Error den Fehler verursacht hat, ist es für eine detaillierte Beschreibung des Umgangs mit SOAP-Faults wichtig zu verstehen, in welchen Ebenen der WebService-Architektur Fehler auftreten können und somit woher diese Fehler kommen. Die einzelnen Fehlercodierungen in SOAP 1.1 und SOAP 1.2 weisen schon auf die Fehlerquellen hin.

(3.1) Fehler in der Applikation

Während der Verarbeitung der Daten, die über das WebService an die Fachapplikation gesandt wurden, kann ein Fehler auftreten. Üblicherweise tritt ein Fehler auf, wenn der Server den eingegangenen Request nicht weiter bearbeiten kann.

(3.2) Fehler auf der Clientseite

Falls auf der Clientseite Fehler auftreten, so ergeben sich dadurch normalerweise keine SOAP-Faults. Es kann aber durchaus vorkommen, dass der Client eine Anfrage an ein WebService stellt und diese nicht der vorgegebenen Syntax entspricht. Die Ursache des daraus entstehenden SOAP-Faults ist somit auf dem Client zu suchen.

(3.3) Fehler in der Transportschicht

Eine weitere Möglichkeit, wo sich Fehler in der Kommunikation ereignen können, besteht in der Transportschicht zwischen Client und WebService bzw. bei kaskadierenden WebServices zwischen diesen. Hier kann es zu physischen Problemen bei der Übermittlung der Daten kommen oder bei der Erreichbarkeit von anderen Quellen

(4) SOAP-Faults vs. XML-Fehlermeldung

Folgendes Kapitel behandelt die Unterschiede zwischen einer XML-Fehlermeldung und einem SOAP-Fault.

(4.1) SOAP-Fault

Beim Bearbeiten des WebServices trifft der Server auf eine für ihn unlösbare Situation. Er erzeugt einen SOAP-Fault, der mit der HTTP Statusmeldung 500 – Internal Server Error an die aufrufende Applikation retour gesandt wird. Wie schon eingangs erwähnt führt ein SOAP-Fault in den allgemein bekannten WebService Frameworks eine Exception. Diese muss programmatisch abgefangen werden.

Um zu erfahren, wo der jeweilige Fehler gelegen hat, muss die Exception-Message, die übermittelt wird, vom Client geparst werden. Danach kann der WebService-Client dementsprechende Schritte setzen, um eine etwaige Fehlermeldung an den Benutzer weiterzugeben. Gegebenenfalls kann das Client-Programm auch eine andere Fehlerbehandlung erwägen.

Somit liegt bei einem SOAP-Fault die Intelligenz vor allem beim Client, der auf die Fehlermeldungen selektiv reagieren muss.

(4.2) XML-Fehlermeldung bzw. XML-Statusmeldung

Der Ablauf der Fehlererzeugung bzw. die Behandlung des Fehlers am Client ist bei einer XML-Fehlermeldung anders, als bei einem SOAP-Fault.

Der Fehler wird ebenfalls am Server erkannt. Die Exception, auf die der Server bei der Abarbeitung stösst, wird jedoch schon am Server abgefangen.

Der Client erhält eine standardisierte Nachricht, die auf einen Fehler hindeutet und die genaue Fehlerbeschreibung und andere beschreibende Daten enthält. Es wird am Client keine Exception geworfen. Nach dem Parsen der übermittelten Daten, wird programmatisch darauf reagiert.

Die Intelligenz bei dieser Fehlerbehandlung liegt vordergründig am Server, jedoch muss auch der Client selektiv auf die ihm gesandten Mitteilungen reagieren.

Es können durch diesen Prozeß nicht nur standardisierte Fehlermeldungen an den Client geliefert werden, sondern auch reine Statusmitteilungen, die Informationen über die Transaktionen auf Applikationsseite beinhalten.

Die vorliegende Empfehlung stellt folgendes Schema zur Verfügung, das über ein *<xs:include>* in ein anwendungsspezifisches Schema importiert werden kann:

```
<xs:schema elementFormDefault="qualified" xmlns:xs="http://www.w3.org/2001/XMLSchema">

    <xs:element name="Message" type="CustomFaultType"/>
    <xs:complexType name="CustomFaultType" final="extension">
        <xs:annotation>
            <xs:documentation>CustomFault reporting structure</xs:documentation>
        </xs:annotation>
        <xs:sequence>
            <xs:element name="Code" type="xs:integer"/>
            <xs:element name="Reason" type="xs:string" maxOccurs="unbounded"/>
            <xs:element name="Help" type="xs:string" minOccurs="0"/>
            <xs:element name="Detail" type="xs:anyType" minOccurs="0"/>
        </xs:sequence>
    </xs:complexType>
</xs:schema>
```

Applikationsspezifische Schemata können sowohl ein eigenes Element für ihre XML-Fehlermeldungen definieren und ihm den global deklarierten komplexen Typ `CustomFaultType` zuweisen oder direkt das global deklarierte `Message`-Element referenzieren.

(5) Vorgehensweise bei Fehlern

Wir empfehlen Fehlermeldungen in WebService generell als SOAP-Faults zu erzeugen. In Ausnahmenfällen, insbesondere, wenn trotz eines Fehlers noch sinnvolle Ergebnisse zurückgeliefert werden können, kann als Alternative die Antwort mit einer standardisierten XML-Fehlermeldung ausgeliefert werden. Als genaues Unterscheidungsmerkmal sind zwei Sichtweisen anzuführen:

(5.1) Fehler aus Sicht der Transportschicht (System Exception)

Eine System Exception tritt immer dann auf, wenn aufgrund eines Fehlers in der Transportschicht oder eines Fehlers in der Serverinfrastruktur,. der Request nicht bearbeitet werden kann.

Beispiele:

- Ausfall von sekundären Systemen (Datenbank, ...)
- Fehlerhaftes Routing bei kaskadierenden WebServices
- HTTP Timeout

Durch Behandlung dieser Fehler mit einem SOAP-Fault, kann der Client mit einer, von den Frameworks erstellten Exception, auf die Problematik hingewiesen werden, dass momentan keinerlei Ergebnisauslieferung stattfinden kann.
Das Auftreten einer System Exception sollte grundsätzlich einen SOAP-Fault erzeugen.

(5.2) Fehler aus Sicht der angesprochenen Systeme (Application Exception)

Eine Application-Exception bezeichnet einen Fehler, bei dem aus Sicht der Anwendung ein Request nicht korrekt bearbeitet werden kann, weil entweder der Request nicht korrekt ist oder die Anwendung einen Request nicht korrekt bearbeiten kann.

Beispiele:

- Datenbankabfrage mit leerer Ergebnismenge
- Anfrage mit ungültigem Wertebereich

Dieser Art von Fehler empfehlen wir ebenfalls mittels SOAP-Faults zu übertragen.

Lediglich im Falle, in dem trotz des Auftretens eines Fehlers noch eine sinnvolle Ergebnismenge zurück geliefert werden kann, kann eine standardisierte XML-Fehlermeldung erzeugt werden.

Dem Client kann mitgeteilt werden, dass es bei der Abarbeitung der Anfrage zu Problemen gekommen ist, diese jedoch nicht kritisch sind bzw. zu einem möglicherweise ungewollten Ergebniss führen. Ein typisches Beispiel wäre die fehlende Unterstützung für ein optionales Feature wie z.B. Sortierung der Ergebnismenge einer Registerabfrage. Ein Client, der die Sortierung angefordert hat, kann durch eine XML-Meldung darauf hingewiesen werden, dass dieses Feature nicht unterstützt wird, wobei ihm trotzdem die Ergebnisse (allerdings natürlich unsortiert) ausgeliefert werden können.
Es ist danach dem Client überlassen, seine Anfrage mit anderen Parametern neu abzusenden, oder die Daten so zu übernehmen.

(6) Fehler- und Statuscodes

Zu einer standardisierten Kommunikationsarchitektur zählen auch vorgegebene Statusmeldungen und Fehlercodes. Eine Einteilung der Fehler und der Status in Gruppen und die durchgehende logische Nummerierung derselben, hilft nicht nur den Entwicklern der Applikationen, sondern bietet auch bei der Erstellung der Clientapplikationen Unterstützung.
Weiters soll eine Standardisierung der Fehlercodes, durch die Verwendung eines eigens dafür zu definierenden Namespaces, vorangetrieben werden.

(6.1) Einteilung der Fehler- bzw. Statuscodes

Die auftretenden Status bzw. Fehler können in Anlehnung an die HTTP Statuscodierung in folgende Gruppen unterteilt werden:

- Informationen über den momentanen Status
  Diese Gruppe beinhaltet keine Fehlermeldungen, sondern ordentliche Mitteilungen über den momentanen Status einer Transaktion.
- Erfolgsmeldungen
  In dieser Gruppen werden alle Erfolgsmeldungen, die im Laufe einer Kommunikation übermittelt werden sollen, gesammelt.
- Zusätzliche Informationen nötig
  Die Meldungen in dieser Klasse zeigen auf, dass zusätzliche Informationen nötig sind, um die Transaktion zu beenden. Dies bedeutet jedoch nicht, dass die Anfrage fehlgeschlagen ist, sondern nur, dass die Applikation nun mit der vorhandenen Information versucht die Transaktion zu beenden.
- Fehler des Clients
  In dieser Klasse werden Meldungen gesammelt, die anzeigen, dass dem Client ein Fehler unterlaufen ist.
- Fehler des Servers
  Fehlermeldungen in dieser Klasse lassen auf interne Fehler des Servers schliessen. Der Server zeigt auf, dass er nicht in der Lage ist, die Transaktion zu beenden, oder die Anfrage zu verarbeiten.
- Andere Fehler
  Diese Klasse bietet Fehlern oder Status, die logisch nicht in den vorangegangenen Klassen Einzug finden, Platz.

Durch die obige Standardeinteilung ergibt sich eine 5-stufige Fehlereinteilung. Die HTTP Codierung nutzt eine dreistellige zusätzliche Interpretation dieser 5 Stufen (zB.: 4001 für

Unauthorisierten Zugriff). Um den vielfältigen Applikation im Bund eine Möglichkeit zu bieten, alle Fehler und Status in dieses Schema überzuführen, empfehlen wir eine vierstellige Interpretation.

Daraus ergibt sich folgende Zahlencodierung der Fehler- und Statusgruppen:

- XML-Statusmeldungen
  - Statusmeldungen – **1xxx**
    1000 – Continue
    1001 – Idle
  - Erfolgsmeldungen – **2xxx**
    2000 – OK
    2001 – Created
    2002 – Accepted
- XML-Fehlermeldungen
  - Zusatzinformation – **3xxx**
    3000 – Multiple Choice
    3001 – Moved Permanently
  - Clientfehler – **4xxx**
    4000 – BadRequest
    4001 – Unauthorized
    4003 – Forbidden
- SOAP-Faults
  - Serverfehler – **5xxx**
    5000 – Internal Server Error
    5001 – Not implemented
  - Sonstiger Fehler – **6xxx**

Die oben vorgeschlagenen Fehlercodes sind nummerisch und sollen in einem `<faultcode>`-Element eines SOAP-Faults zur Anwendung kommen. Laut SOAP-Schema [SOAP12] sind dort aber nur Qualified Names zulässig, d.h. Zeichenketten, die nicht mit einer Ziffer beginnen dürfen. Aus diesem Grund müssen die Fehlercodes in SOAP-Faults mit einem führenden „F“ versehen werden (zb.: „F4010“, „F5018“, …).

(6.2) Fehler- bzw. Statuscodes in SOAP-Faults

Gemäß WS-I-Empfehlung (s. [WSI], 3.3.6) sollten die Fehlercodes dem Namensraum des Applikationsschemas zugewiesen werden, indem er das für diesen definierte Präfix erhält. Ein Beispiel zeigt die korrekte Verwendung der Fehlercodes im `<faultcode>`-Element eines SOAP-Faults:

```
<soap:Fault xmlns:soap="http://schemas.xml.soap.org/soap/envelope/" xmlns:
        sw=" http://reference.e-government.gv.at/namespace/xml-sw/1#">
       <faultcode>sw: F4010</faultcode>
     <faultstring> Required search criteria missing</faultstring>
</soap>
```

(7) Referenzen

[HTML]

Raggett, Dave; Le Hors, Arnaud; Jacobs, Ian: HTML 4.01 Specification W3C Recommendation 24 December 1999, Abgerufen aus dem World Wide Web am 29.06.2004 unter http://www.w3.org/TR/html4/

[HTTP]

Fielding, R. and et al.: Hypertext Transfer Protocol, Juni 1999, Abgerufen aus dem World Wide Web am 21.12.2004 unter http://www.ietf.org/rfc/rfc2616.txt?number=2616

[SOAP11]

Box, D. and et al: Simple Object Access Protocol (SOAP) 1.1 , Abgerufen aus dem World Wide Web am 21. Dez

ember 2004 unter http://www.w3.org/TR/2000/NOTE-SOAP-20000508/

[SOAP12]

Gudgin, M. and et al: SOAP Version 1.2 Part 1: Messaging Framework, Abgerufen aus dem World Wide Web am 20. Dezember 2004 unter http://www.w3.org/TR/soap12-part1/

[WSI]

WebServices Interoperability Organization: WS-I Basic Profile Version 1.1., Abgerufen aus dem World Wide Web am 25. Jänner 2005 unter http://www.ws-i.org/Profiles/BasicProfile-1.1-2004-08-24.html#SOAPFAULT

[VALIDATE]

Liehmann, M: Erklärung des Validate Ansatzes im Rahmen der Kommunikationsarchitektur, Abgerufen aus dem World Wide Web am 01. Jänner 2005 unter http://reference.e-government.gv.at/Zwischenergebnisse.571.0.html

[XML-SW]

Herpers, F.J.: XML-Search:XML-basiertes Protokoll für Suchanfragen via Webservices, vorgelegt in der ARGE Kommarch am 9.11.2004

(8) Literaturvorschläge

Using SOAP Faults:
http://msdn.microsoft.com/library/default.asp?url=/library/en-us/dnservice/html/service09172002.asp

Einführung in SOAP:
http://www.oreilly.de/catalog/progwebsoapger/chapter/

Using SOAP Faults in WebLogic:
http://dev2dev.bea.com/products/wlworkshop81/articles/soap_wlw_ws.jsp

Using SOAP with J2EE: (SOAP 1.1)
http://www.informit.com/articles/article.asp?p=169106&seqNum=6

Using the SOAP Protocol with Java: (SOAP 1.2)
http://www.quepublishing.com/articles/article.asp?p=328640&seqNum=11

What's new in SOAP 1.2:
http://www.hadleynet.org/marc/whatsnew.html