Initiative museumsvokabular.de
Web Services für kontrolliertes Vokabular

museumvok-ws: Schnittstellen-Definition

Version 0.4

Stand 14.08.2007

Autoren: Regine Stein & Carlos Saro
Konrad-Zuse-Zentrum für Informationstechnik (ZIB)
Takustr. 7
D 14195 Berlin
stein@zib.de | saro@zib.de

Einleitung

In dieser Referenz wird die Schnittstelle des Vokabular Web Service der Initiative museumsvokabular.de definiert. Die Beschreibung richtet sich einerseits an die Entwickler von Museumsinformationssystemen, welche auf den Web Service zugreifen sollen. Andererseits können Anbieter von kontrolliertem Vokabular ihrerseits eigene Web Services gemäß dieser Schnittstellendefinition bereit stellen und damit das Angebot von museumsvokabular.de unter ihrer eigenen Ägide erweitern.

Die Schnittstellendefinition[1] basiert auf der unter http://museum.zib.de/museumsvokabular/documents/museumvok0.2.pdf verfügbaren, SKOS-kompatiblen[2] Formatbeschreibung museumvok für die strukturierte Bereitstellung von kontrolliertem Vokabular und enthält in ihrem Hauptteil die genaue Beschreibung der bereit gestellten SOAP-RPC Funktionen.

Ein vom Zuse-Institut Berlin implementierter Protoyp des Services ist zugänglich unter http://museum.zib.de/museumsvokabular/webservice/museumvok-ws0.4.wsdl
Sie können den Service mit einem von Digicult Schleswig-Holstein und ZIB zur Verfügung gestellten Client testen: http://museum.zib.de/museumsvokabular/webservice/museumvok-client0.4.php

Die Initiative museumsvokabular.de wird den Web Service kontinuierlich weiter entwickeln, bitte konsultieren Sie regelmäßig die Seite http://www.museumsvokabular.de > Tech-Dok.

[1] Als wesentliche Grundlage für die Schnittstellendefinition diente dabei ein von der US-amerikanischen National Agricultural Library bereit gestellter Thesaurus Web Service, s. das „NAL Thesaurus Web Services Reference Manual“, download unter http://agclass.nal.usda.gov/agt/progmanual.pdf
[2] S. http://www.w3.org/2004/02/skos/

museumvok-ws: Funktionen und Parameter

Beschreibung der Parameter

Die Übergabe- und Rückgabeparameter der weiter hinten beschriebenen Funktionen werden hier allgemein definiert, sie erschließen sich ggf. jedoch erst im Zusammenhang mit den Funktionsdefinitionen.

Übergabeparameter:

direction

direction ist ein **String** mit drei zulässigen Werten, der die Suchrichtung in der Hierarchie angibt bzw. nur die Top Terms anfordert:

ASC = suche aufsteigend, Oberbegriffe
DESC = suche absteigend, Unterbegriffe
TOP = Top Terms, suche nur die Begriffe der ersten Hierarchieebene

Id

Id ist ein **String** mit einem Identifikator.

Beispiel: $Id = '00000001';
$Id = 'Erntegerät';

IdList

IdList ist ein **Array** von Identifikatoren von Konzepten – sie können im Vokabular enthalten sein oder nicht.

Beispiel: $IdList = array('00000001', 'XYZ00099');

level

level ist ein **Integer,** der die Anzahl der Hierarchiestufen, die durchsucht werden sollen, angibt.
Wenn kein *level* angegeben ist, wird er standardmäßig mit 1 belegt, d.h. es wird nur eine Ebene nach oben oder unten durchsucht.
Wenn *level* gleich 0 ist, werden alle Hierarchiestufen bis zum Ende durchsucht.
Beachte, dass große Werte oder *level*=0 ggf. sehr große Ergebnis-Arrays zurückliefern können!

password

password ist ein **String**, der zum Login benötigt wird.

returnMode

returnMode ist ein **Integer** im Wertebereich [0-2], der das Rückgabeformat für das Ergebnis bestimmt:

0 = komplettes museumvok-Element
1 = SKOS-Element: about/labels/semanticRelations/notes (Bezeichnungen, Beziehungen und Semantik des Konzepts, keine Verwaltungsinformation)
2 = about/labels/semanticRelations (nur Bezeichnungen und Beziehungen, keine Definitionen/Beschreibungen/Anmerkungen, keine Verwaltungsinformation)

returnNumberOfConcepts

returnNumberOfConcepts ist ein **Integer**, der die Anzahl der zurückzugebenden Elemente, unabhängig von der Trefferzahl, beschränkt.

schema

schema ist ein **String,** der das anzusprechende Vokabular identifiziert.

Beispiel: $schema = ‘Gefaess’;

schemeList

schemeList ist ein **Array**, das angibt, für welche Schemata die Funktion getSchemeMetadata Informationen zurückliefern soll. schemeList = array('ALL') bedeutet, dass die Metadaten für alle bereitgestellten Schemata zurückgeliefert werden. Voreinstellung ist 'ALL'.

searchFields

searchFields ist ein **Array**, das angibt, welche Bezeichnungen zu durchsuchen sind, zulässige Werte sind

'prefLabel' => Suche nach Vorzugsbezeichnung
'altLabel' => Suche nach alternativen Bezeichnungen
'notation' => Suche nach Notation
'hiddenLabel' => Suche nach versteckten Bezeichnungen

und alle Kombinationen daraus, z.B.
$searchField = array('prefLabel', 'altLabel') => Suche in Vorzugs- und alternativen Bezeichnungen.

searchLang

searchLang ist ein **Array**, das angibt, in welchen Sprachen die Bezeichnungen zu durchsuchen sind, zulässige Werte sind Sprachattribute gemäß RFC 3066: Tags for the Identification of Languages und alle Kombination daraus. searchLang = array('ALL') bedeutet, dass alle Sprachen durchsucht werden. Voreinstellung ist 'de'.

searchTermMode

searchTermMode ist ein **Integer**-Wert (Flag Byte), der den Suchmodus bestimmt.

0 = exakte Suche
1 = Groß-/Kleinschreibung ignorieren
2 = Sonderzeichen ignorieren
4 = Linkstrunkierung: Suche nach *Begriff
8 = Rechtstrunkierung: Suche nach Begriff*

Kombination der Modi durch Addition, z.B.:
13 = Teiltextsuche, case-insensitive

server

server ist ein **String,** der den anzusprechenden Server identifiziert.

Beispiel: $server = 'museum.zib.de';

session_id

session_id ist ein **String**, der beim Login vom Server für die Session an den Client vergeben wird und bei jedem Funktionsaufruf zur Authentifizierung zwischen Client und Server übertragen wird.

sortField

sortField ist ein **String**, der das Sortierkriterium für die Ergebnismenge angibt, zulässige Werte sind

'prefLabel' => Sortiere nach Vorzugsbezeichnung.
'about' => Sortiere nach ID.
'notation' => Sortiere nach Notation.

termList

termList ist ein **Array** von Bezeichnungen, die im Vokabular enthalten sein können oder nicht.

Beispiel: $termList = array('Königskuchenform', 'Lammform', 'Napfkuchenform');

username

username ist ein **String**, der zum Login benötigt wird.

Rückgabeparameter:

Alle Funktionen außer Login/Logout geben dieselbe Struktur aus vier Elementen zurück, bestehend aus einem Ergebnis-String, der Trefferanzahl, einem numerischen Statuscode und der Session-ID. Login/Logout geben nur Statuscode und Session-ID zurück.

result

result ist ein **String**, i.d.R. museumvok-strukturiertes XML– je nach angefordertem Rückgabemodus enthält er die vollständigen Konzepte oder nur Teile davon.

resultNumber

resultNumber ist ein **Integer**, der die Anzahl der zurück gelieferten Konzepte oder Schemata angibt. Beachte, dass er auch bei erfolgreichem Funktionsaufruf null sein kann.

session_id

session_id ist ein **String**, der beim Login vom Server für die Session an den Client vergeben wird und bei jedem Funktionsaufruf zur Authentifizierung zwischen Client und Server übertragen wird.

statusCode

statusCode ist ein **Integer**, der gleich null ist, wenn der Funktionsaufruf erfolgreich abgearbeitet werden konnte, oder einen der folgenden Fehlercodes enthält.
Beachte, dass ein erfolgreicher Funktionsaufruf nicht notwendig Konzepte zurückliefert!

0 = Ok.
101 = Authentifikation fehlgeschlagen.
102 = Funktion nicht realisiert.
103 = Suchparameter fehlerhaft:
Keine *IdList* fuer Funktion searchConceptsById / Kein arrayOfStrings uebergeben.
Keine *termList* fuer Funktion searchConceptsByTerm/ Kein arrayOfStrings uebergeben.
Keine *Id* fuer Funktion fetchHierarchy / Kein string uebergeben.
104 = Server nicht erreichbar / Uebergebener Parameter *server* fehlerhaft.
105 = Schema existiert nicht / Uebergebener Parameter *schema* fehlerhaft.
106 = Unzulaessiger *searchMode*-Parameter fuer Funktion searchConceptsById oder searchConceptsByTerm

107 = Unzulaessiger *returnMode*-Parameter fuer Funktion searchConceptsById oder searchConceptsByTerm
108 = Unzulaessiger *level*-Parameter fuer Funktion fetchHierarchy / Kein integer-Wert uebergeben.
109 = Unzulaessiger *direction*-Parameter fuer Funktion fetchHierarchy

Beschreibung der Funktionen

getSchemeMetadata

FUNKTION NOCH NICHT IMPLEMENTIERT!

SYNOPSIS

getSchemeMetadata (schemeList, server, returnMode, returnNumberOfConcepts, session_id)

BESCHREIBUNG

Die Funktion liefert Metadaten zu einem oder mehreren Schemata wie Versionsnummer, Beschreibung und Autor zurück. Für schemeList=array('ALL') werden die angeforderten Informationen, in Abhängigkeit des returnMode, für alle verfügbaren Schemata zurückgeliefert.

RÜCKGABE
getSchemeMetadata gibt ein Array aus vier Elementen zurück (*result*, *resultNumber*, *statusCode*, *session_id*), *resultNumber* enthält hier die Anzahl der gefundenen Schemata.

searchConceptsById

SYNOPSIS

searchConceptsById (IdList, server, schema, sortField, returnMode, returnNumberOfConcepts, session_id)

BESCHREIBUNG

Für jeden Identifikator aus der Liste *IdList* sucht die Funktion die zugehörigen Konzepte im angegebenen Vokabular (*server + schema*). *sortField* gibt das Sortierkriterium für die Ergebnismenge an. Der Rückgabemodus *returnMode* bestimmt, in welchem Format die gefundenen Konzepte zurück geliefert werden, entsprechend der obigen Parameter-Definitionen. *returnNumberOfConcepts* beschränkt die Anzahl der zurückzugebenden Elemente. Die *session_id* dient der Authentifizierung am Server.
Beachte: Falls aus einer Liste von Identifikatoren auch nur ein einzelner Identifikator nicht gefunden wird, ist die Treffermenge leer!

Ein PHP-Funktionsaufruf könnte z.B. folgendermaßen aussehen:

```
$IdList = array('51','2','17');
$server = 'museum.zib.de';
$schema = ‘Gefaess’;
$sortField = 'prefLabel';
$result = $soap->__soapCall("searchConceptsById",
                                array('IdList'=>$IdList, 'server'=>$server, 'schema'=>$schema,
                                'sortField'=>$sortField, 'returnMode'=>0,
                                'returnNumberOfConcepts'=>10, 'session_id'=$session_id));
```

RÜCKGABE
searchConceptsById gibt ein Array aus vier Elementen zurück (*result*, *resultNumber*, *statusCode*, *session_id*), wobei *result* die Ergebniselemente im museumvok-Format enthält.

searchConceptsByTerm

SYNOPSIS

searchConceptsByTerm (termList, server, schema, searchTermMode, searchFields, searchLang, sortField, returnMode, returnNumberOfConcepts, session_id)

BESCHREIBUNG

Für jede Bezeichnung aus der Liste *termList* sucht die Funktion die zugehörigen Konzepte im angegebenen Vokabular (*server + schema*). Der Suchmodus *searchTermMode* bestimmt, ob und wie die Suche normalisiert und/oder trunkiert werden soll. *searchFields* enthält die zu durchsuchenden Bezeichnungen, *searchLang* die zu durchsuchenden Sprachen. *sortField* gibt das Sortierkriterium für die Ergebnismenge an. Der Rückgabemodus *returnMode* bestimmt, in welchem Format die gefundenen Konzepte zurück geliefert werden, entsprechend der obigen Parameter-Definitionen. *returnNumberOfConcepts* beschränkt die Anzahl der zurückzugebenden Elemente. Die *session_id* dient der Authentifizierung am Server.

Ein PHP-Funktionsaufruf könnte z.B. folgendermaßen aussehen:

```
$termList = array('Königskuchenform', 'Lammform', 'Napfkuchenform');
$server = 'museum.zib.de';
$schema = ‘Gefaess’;
$searchFields = array('prefLabel', 'altLabel');
$sortField = 'prefLabel';
$result = $soap->__soapCall("searchConceptsByTerm",
                                array('termList'=>$termList, 'server'=>$server, 'schema'=>$schema,
                                      'searchTermMode'=>13, 'searchFields'=>$searchFields,
                                      'searchLang'=>array('ALL'), 'sortField'=>$sortField,
                                      'returnMode'=>0, 'returnNumberOfConcepts'=>10,
                                      'session_id'=$session_id));
```

RÜCKGABE
searchConceptsByTerm gibt ein Array aus vier Elementen zurück (*result*, *resultNumber*, *statusCode*, *session_id*), wobei *result* die Ergebniselemente im museumvok-Format enthält.

fetchHierarchy

SYNOPSIS

fetchHierarchy (Id, server, schema, level, direction, returnMode, returnNumberOfConcepts, session_id)

BESCHREIBUNG

Die Funktion gibt zu einem mit dem Parameter *Id* eindeutig adressierbaren Konzept im angegebenen Vokabular (*server* + *schema*) den angeforderten Hierarchiezweig zurück: Je nach Suchtiefe *level* und Suchrichtung *direction* werden die in Hierarchie über- oder untergeordneten Konzepte als XML im museumvok-Format zurückgeliefert. Das Ergebnis erlaubt die Rekonstruktion der Hierarchie durch die broader-/narrower-Relationen. Für *direction*='TOP' werden nur die Top Terms zurückgeliefert.

Ein PHP-Funktionsaufruf könnte z.B. folgendermaßen aussehen:

```
$Id = '51';
$server = 'museum.zib.de';
$schema = ‘Gefaess’;
$result = $soap->__soapCall("fetchHierarchy",
                                    array('Id'=>$Id, 'server'=>$server,'schema'=>$schema,
                                          'level'=>0, 'direction'=>'DESC',
                                          'returnMode'=>0, 'returnNumberOfConcepts'=>10,
                                          'session_id'=$session_id));
```

RÜCKGABE
fetchHierarchy gibt ein Array aus vier Elementen zurück (*result*, *resultNumber*, *statusCode*, *session_id*), wobei *result* die Ergebniselemente im museumvok-Format enthält.

Beachte, dass große Werte für den Parameter *level* oder *level*=0 ggf. sehr große Ergebnis-Arrays zurückliefern können!

Login

FUNKTION NOCH NICHT IMPLEMENTIERT!

SYNOPSIS

Login (username, password)

BESCHREIBUNG

Login-Funktion zum Beginn einer Session.

RÜCKGABE
Login gibt ein Array aus zwei Elementen zurück (*statusCode*, *session_id*), die *session_id* wird dabei vom Server erzeugt.

Logout

FUNKTION NOCH NICHT IMPLEMENTIERT!

SYNOPSIS

Logout (session_id)

BESCHREIBUNG

Logout-Funktion zur Beendigung einer Session.

RÜCKGABE
Logout gibt ein Array aus zwei Elementen zurück (*statusCode, session_id)*.

museumvok-ws: WSDL Web Service Definitionsdatei

URL: http://museum.zib.de/museumsvokabular/webservice/museumvok-ws0.4.wsdl

```
<?xml version="1.0" encoding="UTF-8"?>
<wsdl:definitions xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/"
xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/" xmlns:tns="urn:museumvok"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:soapenc="http://schemas.xmlsoap.org/soap/encoding/"
xmlns="http://schemas.xmlsoap.org/wsdl/" targetNamespace="urn:museumvok"
name="museumvok">
  <wsdl:types>
    <xsd:schema targetNamespace="urn:museumvok">
      <xsd:complexType name="arrayOfStrings">
        <xsd:complexContent>
          <xsd:restriction base="soapenc:Array">
            <xsd:sequence>
              <xsd:element maxOccurs="unbounded" minOccurs="0" name="item"
type="xsd:string"/>
            </xsd:sequence>
            <xsd:attribute ref="soapenc:arrayType" wsdl:arrayType="xsd:string[]"/>
          </xsd:restriction>
        </xsd:complexContent>
      </xsd:complexType>
      <xsd:complexType name="arrayOfSearchFields">
        <xsd:complexContent>
          <xsd:restriction base="soapenc:Array">
            <xsd:sequence>
              <xsd:element maxOccurs="unbounded" minOccurs="0" name="oneField"
type="tns:searchFields"/>
            </xsd:sequence>
            <xsd:attribute ref="soapenc:arrayType" wsdl:arrayType="xsd:string[]"/>
          </xsd:restriction>
        </xsd:complexContent>
      </xsd:complexType>
      <xsd:simpleType name="searchTermMode">
        <xsd:annotation>
          <xsd:documentation xml:lang="de">Suchmodus<br/>
          0 = exakte Suche
          1 = Gro?/Kleinschreibung ignorieren
```

```
        2 = Sonderzeichen ignorieren
        4 = Linkstrunkierung
        8 = Rechtstrunkierung
        Kombination der Modi durch Addition
        </xsd:documentation>
      </xsd:annotation>
      <xsd:restriction base="xsd:integer">
        <xsd:pattern value="[0-15]"/>
      </xsd:restriction>
    </xsd:simpleType>
    <xsd:simpleType name="returnMode">
      <xsd:annotation>
        <xsd:documentation xml:lang="de">Rueckgabemodus:
         0 = komplettes museumvok-Element
         1 = SKOS-Element: about/labels/semanticRelations/notes (Bezeichnungen, Beziehungen
und Semantik des Konzepts, keine Verwaltungsinformation)
         2 = about/labels/semanticRelations (nur Bezeichnungen und Beziehungen, keine
Definitionen/Beschreibungen/Anmerkungen, keine Verwaltungsinformation)
         3 = about/labels (nur Bezeichnungen)
        </xsd:documentation>
      </xsd:annotation>
      <xsd:restriction base="xsd:integer">
        <xsd:pattern value="[0-2]"/>
      </xsd:restriction>
    </xsd:simpleType>
    <xsd:simpleType name="direction">
      <xsd:annotation>
        <xsd:documentation xml:lang="de">Suchrichtung in der Hierarchie:
         ASC = aufsteigend, Oberbegriffe
         DESC = absteigend, Unterbegriffe
         TOP = TopTerms, alle Begriffe der ersten Hierarchieebene
        </xsd:documentation>
      </xsd:annotation>
      <xsd:restriction base="xsd:string">
        <xsd:enumeration value="ASC"/>
        <xsd:enumeration value="DESC"/>
        <xsd:enumeration value="TOP"/>
      </xsd:restriction>
    </xsd:simpleType>
    <xsd:simpleType name="statusCode">
      <xsd:annotation>
        <xsd:documentation xml:lang="de">Rueckgabestatus / Fehlercodes:<br/>
```

```
            0 = Ok.<br/>
            101 = Authentifikation fehlgeschlagen.<br/>
            102 = Funktion nicht realisiert.<br/>
            103 = Suchparameter fehlerhaft:
                Keine IdList fuer Funktion searchConceptsById / Kein arrayOfStrings
uebergeben.<br/>
                Keine termList fuer Funktion searchConceptsByTerm/ Kein arrayOfStrings
uebergeben. <br/>
                Keine Id fuer Funktion fetchHierarchy / Kein string uebergeben.<br/>
            104 = Server nicht erreichbar / Uebergebener Parameter Server fehlerhaft.<br/>
            105 = Schema existiert nicht / Uebergebener Parameter Schema fehlerhaft.<br/>
            106 = Unzulaessiger searchMode-Parameter fuer Funktion searchConceptsById oder
searchConceptsByTerm<br/>
            107 = Unzulaessiger returnMode-Parameter fuer Funktion searchConceptsById oder
searchConceptsByTerm<br/>
            108 = Unzulaessiger level-Parameter fuer Funktion fetchHierarchy / Kein integer-Wert
uebergeben.<br/>
            109 = Unzulaessiger direction-Parameter fuer Funktion fetchHierarchy<br/>
          </xsd:documentation>
        </xsd:annotation>
        <xsd:restriction base="xsd:integer">
          <xsd:pattern value="[0,100-199]"/>
        </xsd:restriction>
      </xsd:simpleType>
      <xsd:simpleType name="searchFields">
        <xsd:annotation>
          <xsd:documentation xml:lang="de">Angabe der zu durchsuchenden Bezeichnungen
           prefLabel = Vorzugsbezeichnung
           altLabel = Alternative Bezeichnungen
           notation = Notation
           hiddenLabel = Versteckte Bezeichnungen
          </xsd:documentation>
        </xsd:annotation>
        <xsd:restriction base="xsd:string">
          <xsd:enumeration value="altLabel"/>
          <xsd:enumeration value="hiddenLabel"/>
          <xsd:enumeration value="notation"/>
          <xsd:enumeration value="prefLabel"/>
        </xsd:restriction>
      </xsd:simpleType>
      <xsd:simpleType name="sortField">
        <xsd:annotation>
```

```
            <xsd:documentation xml:lang="de">Sortierung nach
             about = ID
             prefLabel = Vorzugsbezeichnung
             notation = Notation
            </xsd:documentation>
          </xsd:annotation>
          <xsd:restriction base="xsd:string">
            <xsd:enumeration value="about"/>
            <xsd:enumeration value="notation"/>
            <xsd:enumeration value="prefLabel"/>
          </xsd:restriction>
        </xsd:simpleType>
        <xsd:complexType name="baseConcept">
          <xsd:annotation>
            <xsd:documentation xml:lang="de">
             Einfacher String, der entweder komplettes museumvok-Element oder Teile davon in XML-
Syntax enthaelt. <br/>
             Kann ggf. zu strukturiertem Element ausgebaut werden.
            </xsd:documentation>
          </xsd:annotation>
          <xsd:all>
            <xsd:element maxOccurs="1" minOccurs="1" name="concept" type="xsd:string"/>
          </xsd:all>
        </xsd:complexType>
        <!-- Nachfolgender Type wird derzeit nicht gebraucht -->
        <xsd:complexType name="arrayOfConcepts">
          <xsd:complexContent>
            <xsd:restriction base="soapenc:Array">
              <xsd:sequence>
                <xsd:element maxOccurs="unbounded" minOccurs="0" name="oneConcept"
type="tns:baseConcept"/>
              </xsd:sequence>
              <xsd:attribute ref="soapenc:arrayType" wsdl:arrayType="xsd:string[]"/>
            </xsd:restriction>
          </xsd:complexContent>
        </xsd:complexType>
      </xsd:schema>
    </wsdl:types>
    <message name="searchConceptsByIdRequest">
      <part name="IdList" type="tns:arrayOfStrings"/>
      <part name="server" type="xsd:string"/>
      <part name="schema" type="xsd:string"/>
```

```
    <part name="sortField" type="tns:sortField"/>
    <part name="returnMode" type="tns:returnMode"/>
    <part name="returnNumberOfConcepts" type="xsd:integer"/>
    <part name="session_id" type="xsd:string"/>
  </message>
  <message name="searchConceptsByIdResponse">
    <part name="result" type="xsd:string"/>
    <part name="resultNumber" type="xsd:integer"/>
    <part name="statusCode" type="xsd:integer"/>
    <part name="session_id" type="xsd:string"/>
  </message>
  <message name="searchConceptsByTermRequest">
    <part name="termList" type="tns:arrayOfStrings"/>
    <part name="server" type="xsd:string"/>
    <part name="schema" type="xsd:string"/>
    <part name="searchTermMode" type="tns:searchTermMode"/>
    <part name="searchFields" type="tns:arrayOfSearchFields"/>
    <part name="searchLang" type="tns:arrayOfStrings"/>
    <part name="sortField" type="tns:sortField"/>
    <part name="returnMode" type="tns:returnMode"/>
    <part name="returnNumberOfConcepts" type="xsd:integer"/>
    <part name="session_id" type="xsd:string"/>
  </message>
  <message name="searchConceptsByTermResponse">
    <part name="result" type="xsd:string"/>
    <part name="resultNumber" type="xsd:integer"/>
    <part name="statusCode" type="tns:statusCode"/>
    <part name="session_id" type="xsd:string"/>
  </message>
  <message name="fetchHierarchyRequest">
    <part name="Id" type="xsd:string"/>
    <part name="server" type="xsd:string"/>
    <part name="schema" type="xsd:string"/>
    <part name="level" type="xsd:integer"/>
    <part name="direction" type="tns:direction"/>
    <part name="returnMode" type="tns:returnMode"/>
    <part name="returnNumberOfConcepts" type="xsd:integer"/>
    <part name="session_id" type="xsd:string"/>
  </message>
  <message name="fetchHierarchyResponse">
    <part name="result" type="xsd:string"/>
    <part name="resultNumber" type="xsd:integer"/>
```

```
    <part name="statusCode" type="tns:statusCode"/>
    <part name="session_id" type="xsd:string"/>
  </message>
  <message name="getSchemeMetadataRequest">
    <wsdl:documentation>
      * schemeList
        An array of schemata of which requested information is returned.
        Default is "ALL": returns information about all available schemata.
   </wsdl:documentation>
    <part name="schemeList" type="tns:arrayOfStrings"/>
    <part name="server" type="xsd:string"/>
    <part name="returnMode" type="tns:returnMode"/>
    <part name="returnNumberOfConcepts" type="xsd:integer"/>
    <part name="session_id" type="xsd:string"/>
  </message>
  <message name="getSchemeMetadataResponse">
    <part name="result" type="xsd:string"/>
    <part name="resultNumber" type="xsd:integer"/>
    <part name="statusCode" type="xsd:integer"/>
    <part name="session_id" type="xsd:string"/>
  </message>
  <message name="LoginRequest">
    <part name="username" type="xsd:string"/>
    <part name="password" type="xsd:string"/>
  </message>
  <message name="LoginResponse">
    <part name="statusCode" type="tns:statusCode"/>
    <part name="session_id" type="xsd:string"/>
  </message>
  <message name="LogoutRequest">
    <part name="session_id" type="xsd:string"/>
  </message>
  <message name="LogoutResponse">
    <part name="statusCode" type="tns:statusCode"/>
    <part name="session_id" type="xsd:string"/>
  </message>
  <portType name="museumvokPortType">
    <operation name="searchConceptsById">
      <input message="tns:searchConceptsByIdRequest"/>
      <output message="tns:searchConceptsByIdResponse"/>
    </operation>
    <operation name="searchConceptsByTerm">
```

```
      <input message="tns:searchConceptsByTermRequest"/>
      <output message="tns:searchConceptsByTermResponse"/>
    </operation>
    <operation name="fetchHierarchy">
      <input message="tns:fetchHierarchyRequest"/>
      <output message="tns:fetchHierarchyResponse"/>
    </operation>
    <operation name="getSchemeMetadata">
      <input message="tns:getSchemeMetadataRequest"/>
      <output message="tns:getSchemeMetadataResponse"/>
    </operation>
    <operation name="Login">
      <input message="tns:LoginRequest"/>
      <output message="tns:LoginResponse"/>
    </operation>
    <operation name="Logout">
      <input message="tns:LogoutRequest"/>
      <output message="tns:LogoutResponse"/>
    </operation>
  </portType>
  <binding name="museumvokBinding" type="tns:museumvokPortType">
    <soap:binding style="rpc" transport="http://schemas.xmlsoap.org/soap/http"/>
    <operation name="searchConceptsById">
      <soap:operation soapAction="urn:soapservice#searchConceptsById"/>
      <input>
        <soap:body use="encoded" encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
namespace="urn:soapservice"/>
      </input>
      <output>
        <soap:body use="encoded" encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
namespace="urn:soapservice"/>
      </output>
    </operation>
    <operation name="searchConceptsByTerm">
      <soap:operation soapAction="urn:soapservice#searchConceptsByTerm"/>
      <input>
        <soap:body use="encoded" encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
namespace="urn:soapservice"/>
      </input>
      <output>
        <soap:body use="encoded" encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
namespace="urn:soapservice"/>
```

```
      </output>
    </operation>
    <operation name="fetchHierarchy">
      <soap:operation soapAction="urn:soapservice#fetchHierarchy"/>
      <input>
        <soap:body use="encoded" encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
namespace="urn:soapservice"/>
      </input>
      <output>
        <soap:body use="encoded" encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
namespace="urn:soapservice"/>
      </output>
    </operation>
    <operation name="getSchemeMetadata">
      <soap:operation soapAction="urn:soapservice#getSchemeMetadata"/>
      <input>
        <soap:body use="encoded" encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
namespace="urn:soapservice"/>
      </input>
      <output>
        <soap:body use="encoded" encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
namespace="urn:soapservice"/>
      </output>
    </operation>
    <operation name="Login">
      <soap:operation soapAction="urn:soapservice#Login"/>
      <input>
        <soap:body use="encoded" encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
namespace="urn:soapservice"/>
      </input>
      <output>
        <soap:body use="encoded" encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
namespace="urn:soapservice"/>
      </output>
    </operation>
    <operation name="Logout">
      <soap:operation soapAction="urn:soapservice#Logout"/>
      <input>
        <soap:body use="encoded" encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
namespace="urn:soapservice"/>
      </input>
      <output>
```

```
        <soap:body use="encoded" encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
namespace="urn:soapservice"/>
      </output>
    </operation>
  </binding>
  <service name="museumvok">
    <port name="museumvokPort" binding="tns:museumvokBinding">
      <soap:address location="http://museum.zib.de/museumsvokabular/webservice/museumvok-
ws0.4.php"/>
    </port>
  </service>
</wsdl:definitions>
```