LIRICS

Deliverable D.5.1.B

API for lexica

Project reference number	e-Content-22236-LIRICS
Project acronym	LIRICS
Project full title	Linguistic Infrastructure for Interoperable Resource and Systems
Project contact point	Laurent Romary, INRIA-Loria 615, rue du jardin botanique BP101. 54602 Villers lès Nancy (France) romary@loria.fr
Project web site	http://lirics.loria.fr
EC project officer	Erwin Valentini
Document title	API for lexica
Deliverable ID	D.5.1.B
Document type	Report
Dissemination level	Public
Contractual date of delivery	M24
Actual date of delivery	16/01/2007
Status & version	V2.0
Work package, task & deliverable responsible	WP5 - UFSD
Author(s) & affiliation(s)	Marc Kemps-Snijders(MPI), Adam Funk (UFSD)
Additional contributor(s)	Ian Roberts, Julien Nioche (UFSD)
Keywords	API LMF

Document evolution

version	date	version	date
0.1	12-10-05		
0.2	12-12-05		
1.0	13-12-05		
2.0	16-01-07		

Content

1 Introduction

This document specifies the Application Programming Interface (API) for the **Lexical Markup Framework** (**LMF**). It is based on the ISO TC 37/SC 4 N130 Rev. 7 document, which describes the LMF model.

This document follows the guidelines of the LIRICS deliverable D1.1 ("Guidelines and tools for producing standards, test-suites and API's").

2 References:

- **Lexical Markup Framework** (**LMF**) ISO TC 37/SC 4 SC4 N130 Rev. 7 document
- LIRICS deliverable D1.1 **Guidelines and tools for producing standards, test-suites and API's**
- **Web Service Definition Language** (**WSDL**) *http://www.w3.org/TR/wsdl*
- **Google Web Service API** *http://www.google.com/apis/*

3 Overview of the API

The LIRICS deliverable D1.1 ("Guidelines and tools for producing standards, test-suites and API's") distinguishes between two levels of API:

- **Service API** for interactions with a processor (i.e. web services, local program)
- **Linguistic Content API** for manipulating the linguistic content returned by a processor

1.1 Service API

The **service API** specifies the syntax of a LMF Web Service. It is based on Web Service Definition Language (WSDL), as required by the deliverable D1.1. This API specifies remote operations which input / output are based on XML.

1.2 Linguistic content API

The **linguistic content API** describes a set of objects in a given implementation language. As specified in the LIRICS deliverable D1.1, this API will be defined using the Java language. However, it should be straightforward to use this API definition for an implementation in another language. The linguistic content API provides a direct access to the linguistic information represented as XML by the service API.

1.3 API use cases

In this chapter we describe how the LMF API can be used in different architectures.

1.3.1 Local access to a native LMF system

In this case the LMF system is directly accessible by the user application. Both programs are running on the same machine and are using the same implementation language. The LMF system runs as a library within the user application. The user application interacts with the LMF system by using instances of the LMF API objects in the implementation language of the user application. This approach is shown on Figure 1.

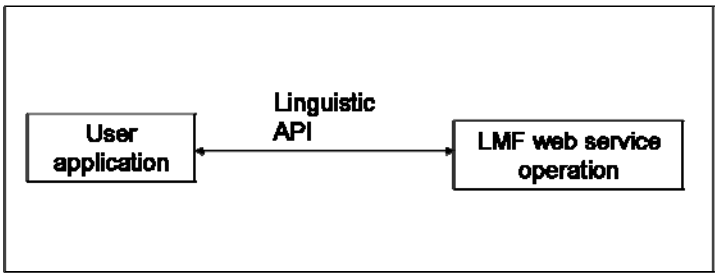

Figure 1 Direct access

1.3.2 Remote access

The LMF system runs on a remote server and is accessible through a Web Service. This approach is the one chosen in the context of the LIRICS project. The reference implementation will be based on a remote access.

A WSDL file describes the syntax and location of the annotation service. The user application sends a representation of a document to this service. The information returned can be of two types:

1. **Serialised objects:** the LMF service returns a set of serialized objects. These objects implement the LMF API. The user application de-serialises the objects and uses them directly as in 1.3.1. This approach is not chosen in the LIRICS project, since it requires that the user application and service use the same implementation language, which is not a portable solution.

2. **XML serialisation:** the LMF service returns a XML representation of the LMF API objects. This approach is illustrated by the Figure 2 **.**

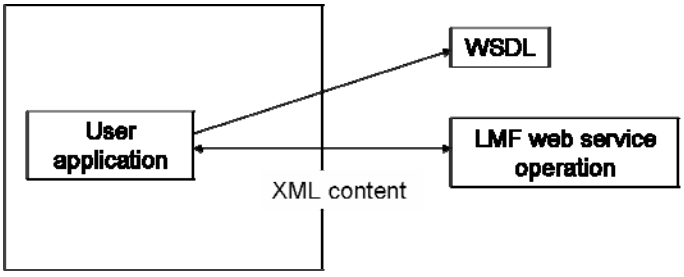

Figure 2 Remote access based on XML

The user application developer can choose whether to use this XML representation directly or to convert it into a LMF API implementation. The latter requires a conversion library. After the conversion the LMF information can be accessed directly as in 1.3.1. This approach is illustrated by the Figure 3.

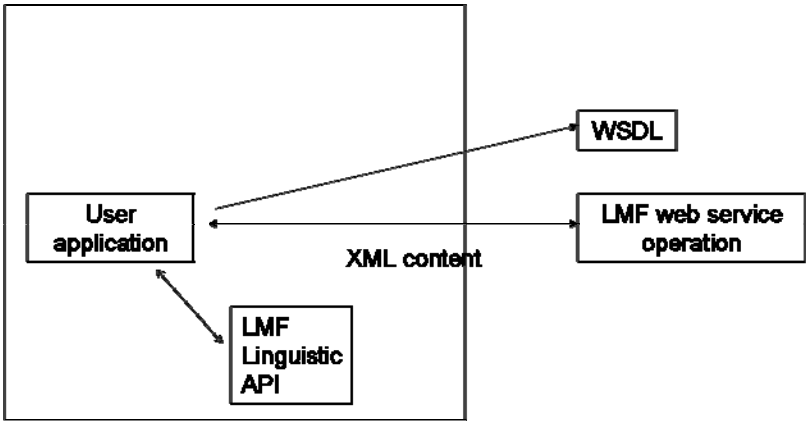

Figure 3 Remote access with conversion to the Linguistic API

1.3.3 Embedded access

In this case the user application uses a library that embeds the connection to the remote Web Service and the conversion of the returned information into an implementation of the LMF API. This architecture requires a specific library in order to convert from the XML content to the linguistic API, and also to connect to the web service. The user application would specify the required parameters through the API of this library, which subsumes the LMF linguistic content API. The parameters can be the URL of the WSDL file, identification data, etc…

1.4 API in LIRICS

As specified in the LIRICS deliverable D1.1, the service API is described with a WSDL file. The complete WSDL definition for LMF Services is provided in .

The Web Services Description Language (WSDL) is an XML format for describing network services as a set of operations on messages containing either document-oriented or procedure-oriented information. The operations and messages are described abstractly, and then bound to a concrete network protocol and message format to define an endpoint. Related concrete operations are combined into services.

The content of the messages is the lexical information returned by a LMF server and will be in XML format. The XML format of this information will be described in the LMF norm. In the WSDL file, references to the LMF XML elements are done through their namespace only. This way, the independence between the API description and the XML representation of LMF is preserved.

The following chapters describe the use cases and how they are related to the service definition.

3.1 Use case scenarios

In using the API five use cases can be distinguished, browsing a catalogue, searching a catalogue and adding, removing or updating elements in a catalogue. For easy reference the LMF core model is provided in Figure 4.

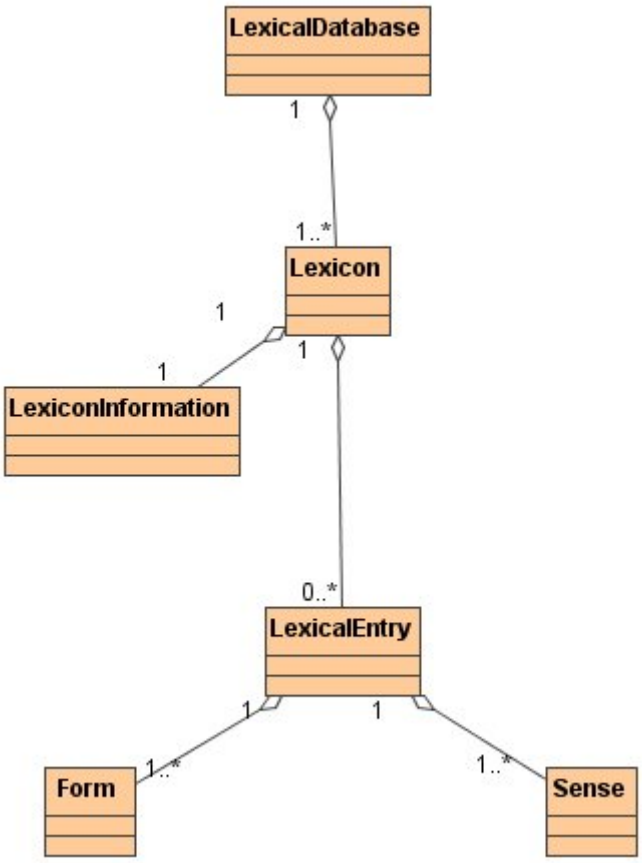

Figure 4: LMF Core model

Applications accessing the LMF API will wish to retrieve information regarding Components stored by the LMF system. For most application the process of retrieving information is user driven were the user will either browse or search the LMF system. The figure below shows the use cases envisaged for traversing the LMF system.

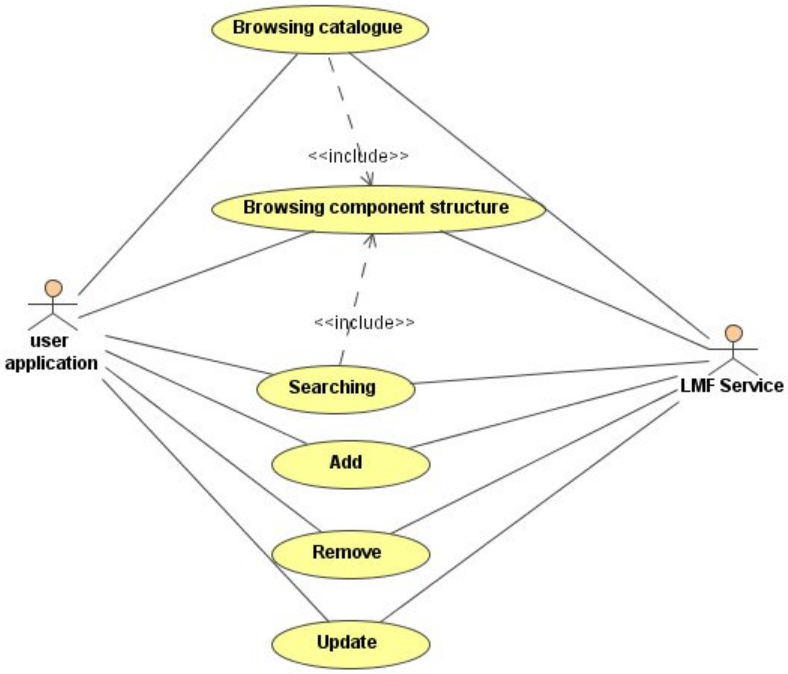

Figure 5: Use case diagram LMF Service

3.1.1.1 Browsing a catalogue.

When accessing a catalogue for the first time a user will first retrieve a list of all LexicalDatabase instances present in the LMF system. The list of LexicalDatabases should contain sufficient information to allow users to quickly select the LexicalDatabase they are interested in. Since most users are directly interested in the information of the lexica contained in a LexicalDatabase, the LexicalDatabase must contain the LexiconInformation of each contained Lexicon. The user may select the LexicalDatabase of interest from the list and proceed to view the LexicalDatabase details. He/she will commence to traverse through the component structure.

3.1.1.2 Searching

The user wants to search one or multiple lexica for a feature/value pairs of interest. He/she might for example be interested in all lexical entries whose partOfSpeech is ‘noun’ and gerund contains ‘walk’. This search may not be limited to one lexicon since for comparison it is important that multiple lexica can be searched at the same time. These lexica may be part of different LexicalDatabase instances.

3.1.1.3 Browsing a component structure

Once a LexicalDatabase, for example, is retrieved the user wishes to browse the structure of the LexicalDatabase to view details of the underlying components. These components include Lexicon, LexiconInformation, Form, Sense and any user defined Component structures. It should be possible to traverse a component structure both top down, as well as bottom-up allowing a two-way navigation through the structure.

Additionally, for a Lexicon component the user may also want to view the schema of the Lexicon, revealing structural information of the lexicon as a whole.

3.1.2 Sequence diagrams

The various use cases may be broken down to display detailed interaction between the application and LMF API. The next paragraphs describe the interaction for the various use cases.

3.1.2.1 Browse catalogue

Browsing the catalogue consists of two steps. In the first step all LexicalDatabase elements are retrieved. In the second step all information which is available for an underlying component is accessed.

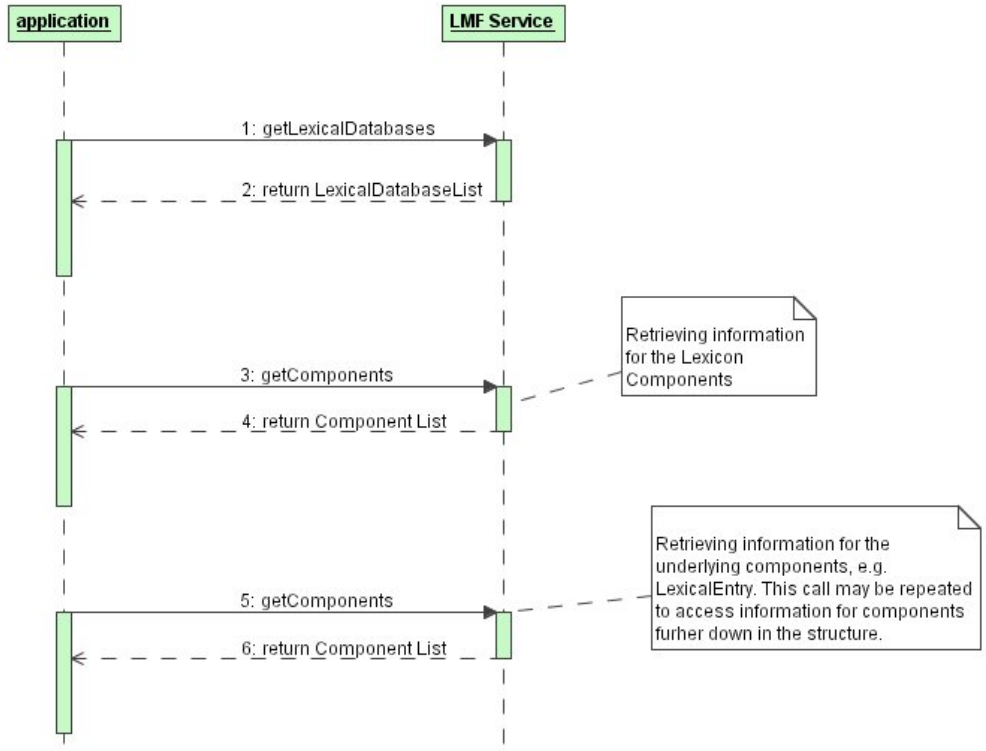

Figure 6: Browsing a component structure

3.1.2.2 GetSchema

A user is interested in the schema information of a Lexicon. Retrieving the schema is done in the following 2 steps:

- Lexicon retrieval. The user retrieves the Lexicon containing the schema of interest.
- View the schema

The following interactions may de identified. The difference between these is the manner in which the lexicon is retrieved. Retrieval of the schema is in all cases identical.

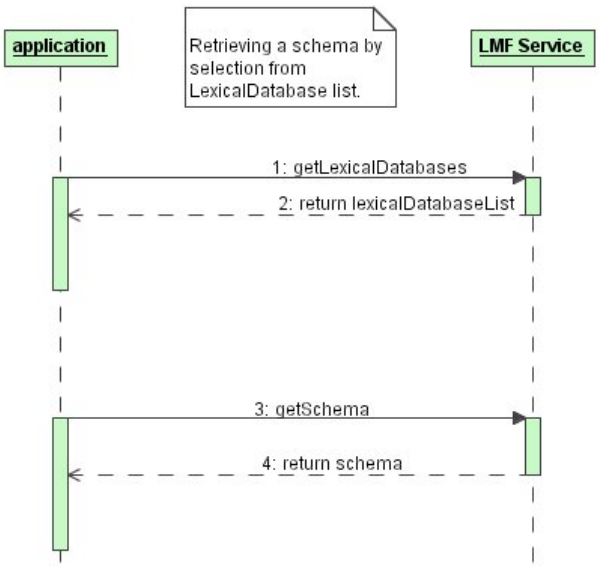

Figure 7: Retrieving a schema by selection from LexicalDatabase list.

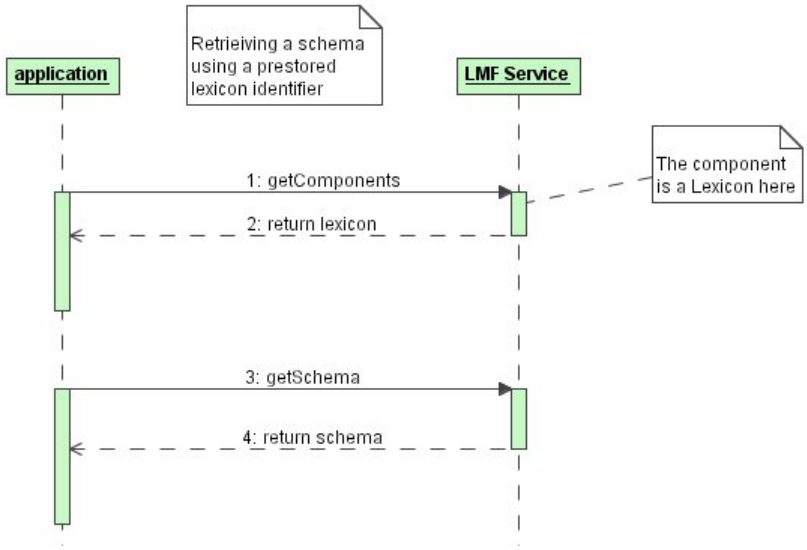

Figure 8: Retrieving a schema using a pre-stored Lexicon identifier.

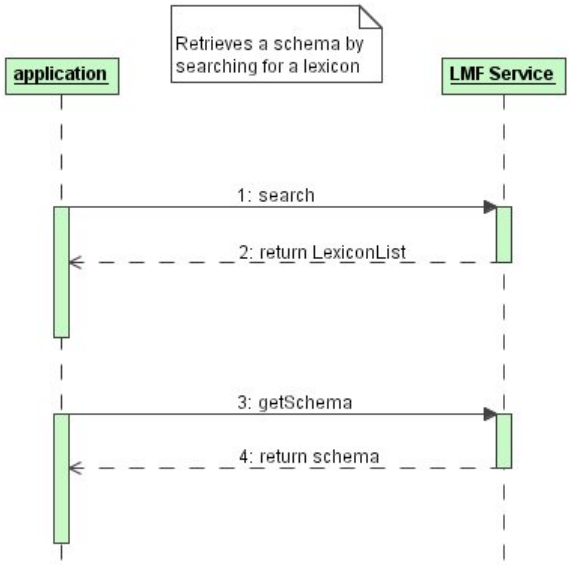

Figure 9: Retrieving a lexicon using search.

3.1.2.3 Search

When a user is interested in particular feature/value pair combinations he/she may decide to search the system for any occurrences of the two. Once a combination has been found the user may traverse the structure of the retrieved components to reveal more details on the contents of he found components.

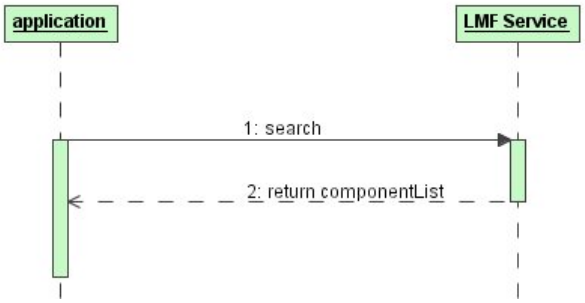

Figure 10: searching the system

3.1.2.4 Add

When a user want to add component or data category information to a LexicalDatabase structure, the information to be added is passed to the system. The system will add the information to the specified node. When the user has insufficient privileges to perform this operation he/she must be notified. Also, when the node to which the specified information is to be added cannot be located the user is to be informed.

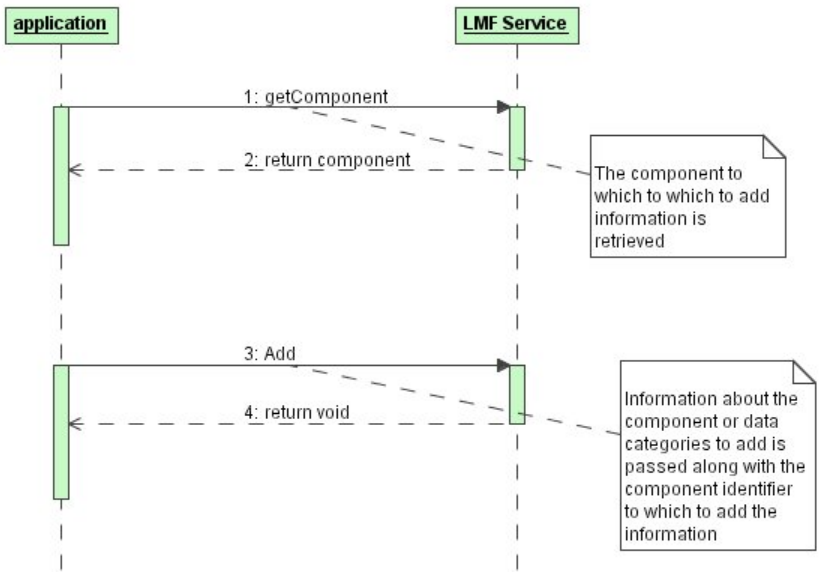

Figure 11: Adding information

3.1.2.5 Delete

A user may decide that information present in a LexicalDatabase structure is no longer needed and decides to remove it. The information to be removed is passed to the service after which it is removed from the LexicalDatabase structure. When the user has no privileges to perform this operation he/she is notified. Also, when the information to be removed cannot be located the user is to be notified.

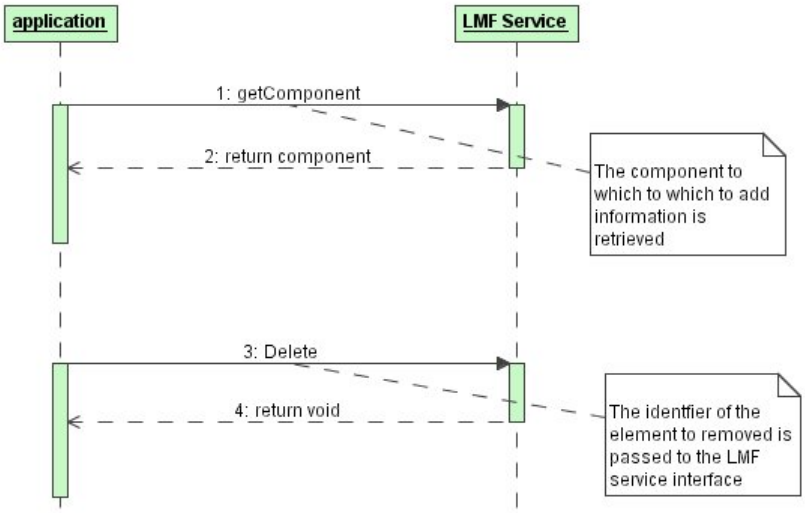

Figure 12: Removing information

3.1.2.6 Update

Information present in a LexicalDatabase may not always be accurate. When the user wants to make a modification to this information he/she passes the new information to the service. This includes information on what element is to be modified. When a user has insufficient privileges to perform this operation the user is to be notified. When the element to be modified cannot be located by the service the user is also to be notified.

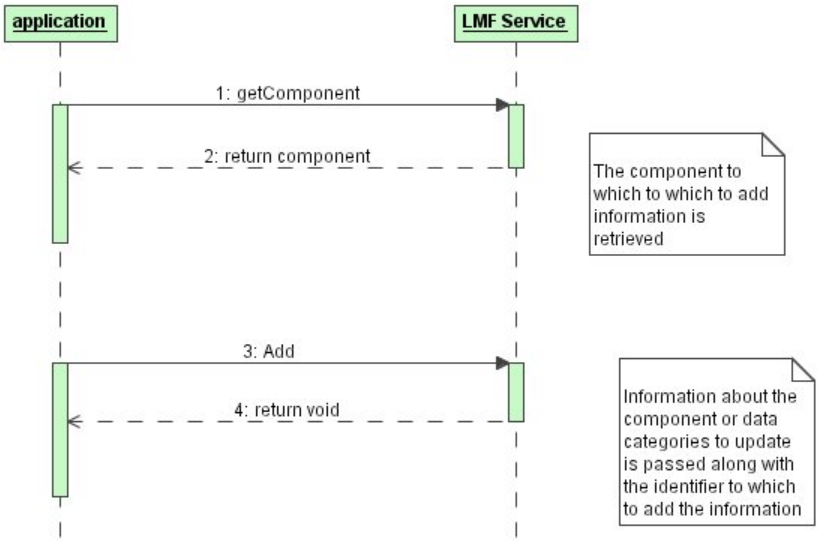

Figure 13:Updating information

4 Description of the LMF API

The service interface will be specified using WSDL. This supports a service interface definition that is distinct from the protocol bindings used for service invocation. The service interface is described in terms of message structures and sequences of simple message exchanges. The complete WSDL description is defined in Appendix A and includes a SOAP protocol binding as an example binding. Part of this description describing the interface is included below. Interface methods are highlighted in red.

The remainder of this chapter describes the individual methods of the interface. For each method the input and output parameters are specified and examples of the xml fragment are given. These xml fragments are intended to provide an illustrative to the xml structure which may be returned and is by no means intended to be regarded as normative in nature.

Each method of the interface is shortly described below in pseudo code for clarity purposes.

4.1 getLexicalDatabases

This method returns a list of Ids for all LexicalDatabase instances present in the LMF system. It presents the primary entry point to the LMF service. A SecurityException fault is thrown if access to the system is denied;

In	
WSDL type	-
Description	-

Out	

WSDL type	`<complexType name="ArrayOf_soapenc_string">`
	`<complexContent>`
	`<restriction base="soapenc:Array">`
	`<attribute ref="soapenc:arrayType"`
	`wsdl:arrayType="soapenc:string[]" />`
	`</restriction>`
	`</complexContent>`
	`</complexType>`

Description	Array of IDS for the LexicalDatabases

Example	`<struct type="LexicalDatabases_overview">`
	`<struct id="345" type="LexicalDatabase" />`
	`<struct id="345" type="LexicalDatabase" />`
	`<struct id="345" type="LexicalDatabase" />`
	`</struct>`

Fault type	**Description**
SecurityException	This fault occurs when a user has no access privileges to the LMF service

4.2 getSchema

This method returns the schema of the Lexicon identified by the specified ID. A SecurityException is thrown if access to the lexicon is denied. An ObjectNotFoundException is thrown if the object can not be found.

The returned schema is specified in relaxNG.

In	
WSDL type	`type="soapenc:string" />`
Description	The lexicon ID of the lexicon to retrieve the schema for.

Out	
WSDL type	<complexType name="**xml_element**"> <sequence> <xsd:any namespace="**##any**" minOccurs="**1**" maxOccurs="**1**" processContents="**skip**" /> </sequence> </complexType>
Description	An XML element representing the schema structure
Example	<grammar xmlns="**http://relaxng.org/ns/structure/1.0**" datatypeLibrary="**http://www.w3.org/2001/XMLSchema-datatypes**"> <start> <ref name="**struct_lexicon**" /> </start> <define name="**struct_lexicon**"> <element name="**struct**"> <attribute name="**type**"> <value>**Lexicon**</value> </attribute> <ref name="**struct_lexiconInformation**" /> <zeroOrMore> <ref name="**struct_lexicalEntry**" /> </zeroOrMore> </element> </define> <define name="**struct_lexiconInformation**"> <element name="**struct**"> <attribute name="**type**"> <value>**LexiconInformation**</value> </attribute> <element name="**feat**"> <attribute name="**type**"> <value>**name**</value> </attribute> <text /> </element> <element name="**feat**"> <attribute name="**type**"> <value>**description**</value> </attribute> <text /> </element> <element name="**feat**"> <attribute name="**type**"> <value>**creationDate**</value> </attribute> <text /> </element> </element> </define> <define name="**struct_lexicalEntry**"> <element name="**struct**"> <attribute name="**type**"> <value>**LexicalEntry**</value> </attribute> <element name="**feat**"> <attribute name="**type**"> <value>**partOfSpeech**</value> </attribute> <text /> </element> <oneOrMore> <ref name="**struct_form**" /> </oneOrMore> <zeroOrMore> <ref name="**struct_sense**" /> </zeroOrMore> </element> </define> **…… ……** </grammar>

Fault type	Description
SecurityException	This fault occurs when a user has no access privileges to the LMF service or this method
ObjectNotFoundException	This fault occurs when the schema associated with the specified identifier can not be found

4.3 getComponents

This method returns an XML representation of the components identified by the specified Ids. A SecurityException is thrown if access to any of the components is denied. An ObjectNotFoundException is throws if any of the specified Ids cannot be associated with a component.

An example of the returned XML is shown below. The returned XML only represents the information at the component level of the specified ID. To retrieve information for the Form, for example, a subsequent call using getComponents("456") is needed.

In	

WSDL type	`<complexType name="ArrayOf_soapenc_string">`
	`<complexContent>`
	`<restriction base="soapenc:Array">`
	`<attribute`
	`ref="soapenc:arrayType"`
	`wsdl:arrayType="soapenc:`
	`string[]" />`
	`</restriction>`
	`</complexContent>`
	`</complexType>`

Description	The array of component IDs identifiying the components to be retrieved..

Out	

WSDL type	`<complexType name="xml_element">`
	`<sequence>`
	`<xsd:any namespace="##targetNamespace"`
	`minOccurs="1" maxOccurs="1"`
	`processContents="skip" />`
	`</sequence>`
	`</complexType>`

Description	XML element representing the component structures of the retrieved components

Example	`<struct id="123" type="LexicalEntry">`
	`<feat id="345" type="partOfSpeech">noun</feat>`
	`<feat id="345" type="lexeme">voorbeeld</feat>`
	`<struct id="456" type="Form" />`
	`<struct id="356" type="Sense" />`
	`</struct>`

Fault type	Description
SecurityException	This fault accurs when a user has no access privileges to the LMF service or this method
ObjectNotFoundException	This fault occurs when one or more of the components associated with the specified identfiers can not be found

4.4 getParents

This method returns the information of the parent components identified by the specified ID's. A SecurityException will be thrown if access is denied. An ObjectnotFoundException is thrown if any of the specified Ids refers to a non existent component. An example of the returned XML is shown below for the LexicalEntry shown above.

In	

WSDL type	`<complexType name="ArrayOf_soapenc_string">`
	`<complexContent>`
	`<restriction base="soapenc:Array">`
	`<attribute`
	`ref="soapenc:arrayType"`
	`wsdl:arrayType="soapenc:`
	`string[]" />`
	`</restriction>`
	`</complexContent>`
	`</complexType>`

Description	The array of component IDs identifiying the components for which the parent components are to be retrieved..

Out	

WSDL type	`<complexType name="xml_element">`
	`<sequence>`
	`<xsd:any namespace="##targetNamespace"`
	`minOccurs="1" maxOccurs="1"`
	`processContents="skip" />`
	`</sequence>`
	`</complexType>`

Description	XML element representing the component structures of the retrieved parent components

Example	`<struct id="678" type="Lexicon">`
	`<struct id="684" type="LexiconInfo" />`
	`<struct id="123" type="LexicalEntry" />`
	`<struct id="111" type="LexicalEntry" />`
	`<struct id="222" type="LexicalEntry" />`
	`...`
	`</struct>`

Fault type	Description
SecurityException	This fault accurs when a user has no access privileges to the LMF service or this method
ObjectNotFoundException	This fault occurs when one or more of the components associated with the specified identfiers can not be found

4.5 getValues

This method returns the value for the list of specified Ids and allows direct retrieval of attribute values given an ID. A SecurityException is thrown if access is denied; an ObjectNotFoundException is thrown if any of the ID's refers to a non existent attribute value. An example of the returned XML is shown below.

In	

WSDL type	`<complexType name="ArrayOf_soapenc_string">`
	`<complexContent>`
	`<restriction base="soapenc:Array">`
	`<attribute`
	`ref="soapenc:arrayType"`
	`wsdl:arrayType="soapenc:`
	`string[]" />`
	`</restriction>`
	`</complexContent>`
	`</complexType>`

Description	The array of value IDs identifiying the values to be retrieved..

Out	

WSDL type	`<complexType name="xml_element">`
	`<sequence>`
	`<xsd:any namespace="##targetNamespace"`
	`minOccurs="1" maxOccurs="1"`
	`processContents="skip" />`
	`</sequence>`
	`</complexType>`

Description	XML element representing the component structures of the retrieved parent components
Example	`<feat id="345" type="partOfSpeech">noun</feat>`

Fault type	**Description**
SecurityException	This fault accurs when a user has no access privileges to the LMF service or this method
ObjectNotFoundException	This fault occurs when one or more of the values associated with the specified identfiers can not be found

4.6 searchName

Searches the LMF system using the specified *searchString*. More details on the LMF query language can be found in Section 5.

In	
WSDL type	<complexType name="**xml_element**"> <sequence> <xsd:any namespace="**##targetNamespace**" minOccurs="**1**" maxOccurs="**1**" processContents="**skip**" /> </sequence> </complexType>
Description	The XML element describing the search terms

Out	
WSDL type	<complexType name="**xml_element**"> <sequence> <xsd:any namespace="**##targetNamespace**" minOccurs="**1**" maxOccurs="**1**" processContents="**skip**" /> </sequence> </complexType>
Description	XML element representing the structures of the found components or datacategories
Example	<struct id="**123**" type="**LexicalEntry**"> <feat id="**345**" type="**partOfSpeech**">**noun**</feat> <feat id="**345**" type="**lexeme**">**voorbeeld**</feat> <struct id="**456**" type="**Form**" /> <struct id="**356**" type="**Sense**" /> </struct>

Fault type	Description
SecurityException	This fault accurs when a user has no access privileges to the LMF service or this method

4.7 add

This method adds information to a LMF structure

In	
WSDL type	<complexType name="**xml_element**"> <sequence> <xsd:any namespace="**##targetNamespace**" minOccurs="**1**" maxOccurs="**1**" processContents="**skip**" /> </sequence> </complexType>
Description	XML element representing the structures of the found components or datacategories
Example	<struct id="**123**" type="**LexicalEntry**"> <feat type="**partOfSpeech**">**noun**</feat> </struct> The datacategory partOfSpeech with value noun is added to lexicalEntry '123'

Out	
WSDL type	-
Description	-

Fault type	Description
SecurityException	This fault accurs when a user has no access privileges to the LMF service or this method
ObjectNotFoundException	This fault occurs when one or more of the components associated with the specified identfiers can not be found

4.8 delete

This method removes information from an LMF structure

In	
WSDL type	<complexType name="**xml_element**"> <sequence> <xsd:any namespace="**##any**" minOccurs="**1**" maxOccurs="**1**" processContents="**skip**" /> </sequence> </complexType>
Description	XML element representing the structures of the found components or datacategories
Example	<struct id="**123**" type="**LexicalEntry**"> </struct> The lexicalentry with ID '123' is removed from the lexicon

Out	
WSDL type	-
Description	-

Fault type	Description
SecurityException	This fault accurs when a user has no access privileges to the LMF service or this method
ObjectNotFoundException	This fault occurs when one or more of the components associated with the specified identfiers can not be found

4.9 update

This method updates existing information in a LMF structure.

In	

WSDL type	`<complexType name="xml_element">`
	`<sequence>`
	`<xsd:any namespace="##any"`
	`minOccurs="1" maxOccurs="1"`
	`processContents="skip" />`
	`</sequence>`
	`</complexType>`

Description	XML element representing the structures of the found components or datacategories
Example	`<feat id="321" type="partOfSpeech">verb</feat>` Data category partofSpeech value with id '321'is updated to value verb

Out	
WSDL type	-
Description	-

Fault type	**Description**
SecurityException	This fault accurs when a user has no access privileges to the LMF service or this method
ObjectNotFoundException	This fault occurs when one or more of the components associated with the specified identfiers can not be found

5 LMF Query language

The purpose of the *searchName* method described earlier is to search the LMF service for elements matching user defined search criteria.

The proposed query language will be Xpath or at least Xpath orientated. This seems to be most logical approach since the lexicon information is specified in terms of XML. However, other query specification mechanisms may be adopted by different implementations of the LMF service interface.

Query is XPath, e.g.

```
LexicalEntry[../@LexiconID = 'lex1' and LemmatisedForm[@value = 'dog']]

Lexicon[@ID = 'lex1']/LexicalEntry[LemmatisedForm/@value = 'dog']/Lexeme/FrisianWord

Lexicon[@ID = 'lex2']/LexiconInformation
```

Or XPath-like:

```
<Query>
  <Component type='FrisianWord'>
    <Constraint>
      <Parent>
        <Child type='LemmatisedForm'>
          <DatCat domain='iso-12620' id='POS' />
          <Value>NOUN</Value>
        </Child>
      </Parent>
    </Constraint>
  </Component>
</Query>
```

Annex A LMF WSDL

The WSDL file describing the LMF service is given below. The file was created by first defining a Java interface and subsequently using Apache Axis 1.2.1 to generate the WSDL file. Modifications were made by hand to to refer to the XML namespace for LMF XML entities. Thus the schema of these entities will not be defined in this document. Examples have been given in the document for the xml fragments, but are merely to be considered as indicative to the structure and are by no means intended to be normative in nature.

```
<?xml version="1.0" encoding="UTF-8" ?>

<wsdl:definitions targetNamespace="urn:LIRICS-LMF"
     xmlns:impl="urn:LIRICS-LMF"
     xmlns:intf="urn:LIRICS-LMF"
     xmlns:apachesoap="http://xml.apache.org/xml-soap"
     xmlns:wsdlsoap="http://schemas.xmlsoap.org/wsdl/soap/"
      xmlns:soapenc="http://schemas.xmlsoap.org/soap/encoding/"
     xmlns:xsd="http://www.w3.org/2001/XMLSchema"
     xmlns:tns1="http://DefaultNamespace"
     xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/"
     xmlns:lmf="urn:LIRICS-LMF">

   <wsdl:types>

      <schema xmlns="http://www.w3.org/2001/XMLSchema"
            targetNamespace="http://DefaultNamespace">
         <import namespace="urn:LIRICS-LMF" />
         <import namespace="http://schemas.xmlsoap.org/soap/encoding/"
               />
         <complexType name="SecurityException">
            <sequence />
         </complexType>
         <complexType name="ObjectNotFoundException">
            <sequence />
         </complexType>
         <complexType name="SchemaViolationException">
            <sequence />
         </complexType>
      </schema>

      <schema xmlns="http://www.w3.org/2001/XMLSchema"
            targetNamespace="urn:LIRICS-LMF">
         <import namespace="http://DefaultNamespace" />
         <import namespace="http://schemas.xmlsoap.org/soap/encoding/"
               />
         <complexType name="ArrayOf_soapenc_string">
            <complexContent>
               <restriction base="soapenc:Array">
                  <attribute ref="soapenc:arrayType"
                        wsdl:arrayType="soapenc:string[]" />
               </restriction>
            </complexContent>
         </complexType>
      </schema>

      <complexType name="xml_element">
         <sequence>
            <xsd:any namespace="##targetNamespace" minOccurs="1"
                  maxOccurs="1" processContents="skip" />
         </sequence>
      </complexType>

      <complexType name="relaxng_schema">
         <sequence>
            <xsd:any namespace="http://relaxng.org/ns/structure/1.0"
                  minOccurs="1" maxOccurs="1" processContents="skip" />
         </sequence>
      </complexType>
   </wsdl:types>

   <wsdl:message name="SecurityException">
      <wsdl:part name="fault" type="tns1:SecurityException" />
   </wsdl:message>
   <wsdl:message name="createRequest">
      <wsdl:part name="in0" type="soapenc:string" />
   </wsdl:message>
   <wsdl:message name="getParentsRequest">
      <wsdl:part name="in0" type="impl:ArrayOf_soapenc_string" />
   </wsdl:message>
```

```
    <wsdl:message name="searchNameRequest">
       <wsdl:part name="in0" type="soapenc:string" />
    </wsdl:message>
    <wsdl:message name="deleteResponse" />
    <wsdl:message name="getComponentsRequest">
       <wsdl:part name="in0" type="impl:ArrayOf_soapenc_string" />
    </wsdl:message>
    <wsdl:message name="getSchemaRequest">
       <wsdl:part name="in0" type="soapenc:string" />
    </wsdl:message>
    <wsdl:message name="getValuesRequest">
       <wsdl:part name="in0" type="impl:ArrayOf_soapenc_string" />
    </wsdl:message>
    <wsdl:message name="deleteRequest">
       <wsdl:part name="in0" type="soapenc:string" />
    </wsdl:message>
    <wsdl:message name="getValuesResponse">
       <wsdl:part name="getValuesReturn" type="soapenc:string" />
    </wsdl:message>
    <wsdl:message name="updateResponse" />
    <wsdl:message name="updateRequest">
       <wsdl:part name="in0" type="soapenc:string" />
    </wsdl:message>
    <wsdl:message name="createResponse">
       <wsdl:part name="createReturn" type="soapenc:string" />
    </wsdl:message>
    <wsdl:message name="ObjectNotFoundException">
       <wsdl:part name="fault" type="tns1:ObjectNotFoundException" />
    </wsdl:message>
    <wsdl:message name="getParentsResponse">
       <wsdl:part name="getParentsReturn" type="impl:xmlstring" />
    </wsdl:message>
    <wsdl:message name="getSchemaResponse">
       <wsdl:part name="getSchemaReturn" type="impl:relaxng_schema" />
    </wsdl:message>
    <wsdl:message name="SchemaViolationException">
       <wsdl:part name="fault" type="tns1:SchemaViolationException" />
    </wsdl:message>
    <wsdl:message name="getComponentsResponse">
       <wsdl:part name="getComponentsReturn" type="impl:xmlstring" />
    </wsdl:message>
    <wsdl:message name="getLexicalDatabasesResponse">
       <wsdl:part name="getLexicalDatabasesReturn"
            type="impl:ArrayOf_soapenc_string" />
    </wsdl:message>
    <wsdl:message name="searchNameResponse">
       <wsdl:part name="searchNameReturn" type="impl:xmlstring" />
    </wsdl:message>
    <wsdl:message name="getLexicalDatabasesRequest" />
    <wsdl:portType name="ILMFService">

       <wsdl:operation name="delete" parameterOrder="in0">
          <wsdl:input name="deleteRequest" message="impl:deleteRequest" />
          <wsdl:output name="deleteResponse" message="impl:deleteResponse"
                />
          <wsdl:fault name="ObjectNotFoundException"
                message="impl:ObjectNotFoundException" />
          <wsdl:fault name="SchemaViolationException"
                message="impl:SchemaViolationException" />
          <wsdl:fault name="SecurityException"
                message="impl:SecurityException" />
       </wsdl:operation>

       <wsdl:operation name="create" parameterOrder="in0">
          <wsdl:input name="createRequest" message="impl:createRequest" />
          <wsdl:output name="createResponse" message="impl:createResponse"
                />
          <wsdl:fault name="SchemaViolationException"
                message="impl:SchemaViolationException" />
          <wsdl:fault name="SecurityException"
                message="impl:SecurityException" />
       </wsdl:operation>

       <wsdl:operation name="update" parameterOrder="in0">
          <wsdl:input name="updateRequest" message="impl:updateRequest" />
          <wsdl:output name="updateResponse"
                message="impl:updateResponse" />
          <wsdl:fault name="SchemaViolationException"
                message="impl:SchemaViolationException" />
          <wsdl:fault name="SecurityException"
                message="impl:SecurityException" />
       </wsdl:operation>

       <wsdl:operation name="searchName" parameterOrder="in0">
          <wsdl:input name="searchNameRequest"
                message="impl:searchNameRequest" />
          <wsdl:output name="searchNameResponse"
```

```
            message="impl:searchNameResponse" />
        <wsdl:fault name="SecurityException"
            message="impl:SecurityException" />
    </wsdl:operation>

    <wsdl:operation name="getLexicalDatabases">
        <wsdl:input name="getLexicalDatabasesRequest"
            message="impl:getLexicalDatabasesRequest" />
        <wsdl:output name="getLexicalDatabasesResponse"
            message="impl:getLexicalDatabasesResponse" />
        <wsdl:fault name="SecurityException"
            message="impl:SecurityException" />
    </wsdl:operation>

    <wsdl:operation name="getSchema" parameterOrder="in0">
        <wsdl:input name="getSchemaRequest"
            message="impl:getSchemaRequest" />
        <wsdl:output name="getSchemaResponse"
            message="impl:getSchemaResponse" />
        <wsdl:fault name="ObjectNotFoundException"
            message="impl:ObjectNotFoundException" />
        <wsdl:fault name="SecurityException"
            message="impl:SecurityException" />
    </wsdl:operation>

    <wsdl:operation name="getComponents" parameterOrder="in0">
        <wsdl:input name="getComponentsRequest"
            message="impl:getComponentsRequest" />
        <wsdl:output name="getComponentsResponse"
            message="impl:getComponentsResponse" />
        <wsdl:fault name="ObjectNotFoundException"
            message="impl:ObjectNotFoundException" />
        <wsdl:fault name="SecurityException"
            message="impl:SecurityException" />
    </wsdl:operation>

    <wsdl:operation name="getValues" parameterOrder="in0">
        <wsdl:input name="getValuesRequest"
            message="impl:getValuesRequest" />
        <wsdl:output name="getValuesResponse"
            message="impl:getValuesResponse" />
        <wsdl:fault name="ObjectNotFoundException"
            message="impl:ObjectNotFoundException" />
        <wsdl:fault name="SecurityException"
            message="impl:SecurityException" />
    </wsdl:operation>

    <wsdl:operation name="getParents" parameterOrder="in0">
        <wsdl:input name="getParentsRequest"
            message="impl:getParentsRequest" />
        <wsdl:output name="getParentsResponse"
            message="impl:getParentsResponse" />
        <wsdl:fault name="ObjectNotFoundException"
            message="impl:ObjectNotFoundException" />
        <wsdl:fault name="SecurityException"
            message="impl:SecurityException" />
    </wsdl:operation>

</wsdl:portType>

<wsdl:binding name="LMFAPISoapBinding" type="impl:ILMFService">
    <wsdlsoap:binding style="rpc"
        transport="http://schemas.xmlsoap.org/soap/http" />
    <wsdl:operation name="delete">
        <wsdlsoap:operation soapAction="" />
        <wsdl:input name="deleteRequest">
            <wsdlsoap:body use="encoded"
                encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
                namespace="urn:LIRICS-LMF" />
        </wsdl:input>
        <wsdl:output name="deleteResponse">
            <wsdlsoap:body use="encoded"
                encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
                namespace="urn:LIRICS-LMF" />
        </wsdl:output>
        <wsdl:fault name="ObjectNotFoundException">
            <wsdlsoap:fault name="ObjectNotFoundException" use="encoded"
                encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
                namespace="urn:LIRICS-LMF" />
        </wsdl:fault>
        <wsdl:fault name="SchemaViolationException">
            <wsdlsoap:fault name="SchemaViolationException" use="encoded"
                encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
                namespace="urn:LIRICS-LMF" />
        </wsdl:fault>
        <wsdl:fault name="SecurityException">
            <wsdlsoap:fault name="SecurityException" use="encoded"
```

```
                              ="http://schemas.xmlsoap.org/soap/encoding/"
                    namespace="urn:LIRICS-LMF" />
            </wsdl:fault>
        </wsdl:operation>
        <wsdl:operation name="create">
            <wsdlsoap:operation soapAction="" />
            <wsdl:input name="createRequest">
                <wsdlsoap:body use="encoded"
                    encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
                    namespace="urn:LIRICS-LMF" />
            </wsdl:input>
            <wsdl:output name="createResponse">
                <wsdlsoap:body use="encoded"
                    encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
                    namespace="urn:LIRICS-LMF" />
            </wsdl:output>
            <wsdl:fault name="SchemaViolationException">
                <wsdlsoap:fault name="SchemaViolationException" use="encoded"
                    encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
                    namespace="urn:LIRICS-LMF" />
            </wsdl:fault>
            <wsdl:fault name="SecurityException">
                <wsdlsoap:fault name="SecurityException" use="encoded"
                    encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
                    namespace="urn:LIRICS-LMF" />
            </wsdl:fault>
        </wsdl:operation>

        <wsdl:operation name="update">
            <wsdlsoap:operation soapAction="" />
            <wsdl:input name="updateRequest">
                <wsdlsoap:body use="encoded"
                    encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
                    namespace="urn:LIRICS-LMF" />
            </wsdl:input>
            <wsdl:output name="updateResponse">
                <wsdlsoap:body use="encoded"
                    encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
                    namespace="urn:LIRICS-LMF" />
            </wsdl:output>
            <wsdl:fault name="SchemaViolationException">
                <wsdlsoap:fault name="SchemaViolationException" use="encoded"
                    encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
                    namespace="urn:LIRICS-LMF" />
            </wsdl:fault>
            <wsdl:fault name="SecurityException">
                <wsdlsoap:fault name="SecurityException" use="encoded"
                    encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
                    namespace="urn:LIRICS-LMF" />
            </wsdl:fault>
        </wsdl:operation>

        <wsdl:operation name="searchName">
            <wsdlsoap:operation soapAction="" />
            <wsdl:input name="searchNameRequest">
                <wsdlsoap:body use="encoded"
                    encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
                    namespace="urn:LIRICS-LMF" />
            </wsdl:input>
            <wsdl:output name="searchNameResponse">
                <wsdlsoap:body use="encoded"
                    encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
                    namespace="urn:LIRICS-LMF" />
            </wsdl:output>
            <wsdl:fault name="SecurityException">
                <wsdlsoap:fault name="SecurityException" use="encoded"
                    encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
                    namespace="urn:LIRICS-LMF" />
            </wsdl:fault>
        </wsdl:operation>

        <wsdl:operation name="getLexicalDatabases">
            <wsdlsoap:operation soapAction="" />
            <wsdl:input name="getLexicalDatabasesRequest">
                <wsdlsoap:body use="encoded"
                    encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
                    namespace="urn:LIRICS-LMF" />
            </wsdl:input>
            <wsdl:output name="getLexicalDatabasesResponse">
                <wsdlsoap:body use="encoded"
                    encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
                    namespace="urn:LIRICS-LMF" />
            </wsdl:output>
            <wsdl:fault name="SecurityException">
                <wsdlsoap:fault name="SecurityException" use="encoded"
                    encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
                    namespace="urn:LIRICS-LMF" />
```

```
      </wsdl:fault>
    </wsdl:operation>

    <wsdl:operation name="getSchema">
      <wsdlsoap:operation soapAction="" />
      <wsdl:input name="getSchemaRequest">
        <wsdlsoap:body use="encoded"
            encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
            namespace="urn:LIRICS-LMF" />
      </wsdl:input>
      <wsdl:output name="getSchemaResponse">
        <wsdlsoap:body use="encoded"
            encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
            namespace="urn:LIRICS-LMF" />
      </wsdl:output>
      <wsdl:fault name="ObjectNotFoundException">
        <wsdlsoap:fault name="ObjectNotFoundException" use="encoded"
            encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
            namespace="urn:LIRICS-LMF" />
      </wsdl:fault>
      <wsdl:fault name="SecurityException">
        <wsdlsoap:fault name="SecurityException" use="encoded"
            encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
            namespace="urn:LIRICS-LMF" />
      </wsdl:fault>
    </wsdl:operation>

    <wsdl:operation name="getComponents">
      <wsdlsoap:operation soapAction="" />
      <wsdl:input name="getComponentsRequest">
        <wsdlsoap:body use="encoded"
            encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
            namespace="urn:LIRICS-LMF" />
      </wsdl:input>
      <wsdl:output name="getComponentsResponse">
        <wsdlsoap:body use="encoded"
            encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
            namespace="urn:LIRICS-LMF" />
      </wsdl:output>
      <wsdl:fault name="ObjectNotFoundException">
        <wsdlsoap:fault name="ObjectNotFoundException" use="encoded"
            encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
            namespace="urn:LIRICS-LMF" />
      </wsdl:fault>
      <wsdl:fault name="SecurityException">
        <wsdlsoap:fault name="SecurityException" use="encoded"
            encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
            namespace="urn:LIRICS-LMF" />
      </wsdl:fault>
    </wsdl:operation>

    <wsdl:operation name="getValues">
      <wsdlsoap:operation soapAction="" />
      <wsdl:input name="getValuesRequest">
        <wsdlsoap:body use="encoded"
            encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
            namespace="urn:LIRICS-LMF" />
      </wsdl:input>
      <wsdl:output name="getValuesResponse">
        <wsdlsoap:body use="encoded"
            encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
            namespace="urn:LIRICS-LMF" />
      </wsdl:output>
      <wsdl:fault name="ObjectNotFoundException">
        <wsdlsoap:fault name="ObjectNotFoundException" use="encoded"
            encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
            namespace="urn:LIRICS-LMF" />
      </wsdl:fault>
      <wsdl:fault name="SecurityException">
        <wsdlsoap:fault name="SecurityException" use="encoded"
            encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
            namespace="urn:LIRICS-LMF" />
      </wsdl:fault>
    </wsdl:operation>

    <wsdl:operation name="getParents">
      <wsdlsoap:operation soapAction="" />
      <wsdl:input name="getParentsRequest">
        <wsdlsoap:body use="encoded"
            encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
            namespace="urn:LIRICS-LMF" />
      </wsdl:input>
      <wsdl:output name="getParentsResponse">
        <wsdlsoap:body use="encoded"
            encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
            namespace="urn:LIRICS-LMF" />
      </wsdl:output>
```

```
            <wsdl:fault name="ObjectNotFoundException">
               <wsdlsoap:fault name="ObjectNotFoundException" use="encoded"
                     encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
                     namespace="urn:LIRICS-LMF" />
            </wsdl:fault>
            <wsdl:fault name="SecurityException">
               <wsdlsoap:fault name="SecurityException" use="encoded"
                     encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
                     namespace="urn:LIRICS-LMF" />
            </wsdl:fault>
         </wsdl:operation>
      </wsdl:binding>

</wsdl:definitions>
```