LIRICS

Deliverable 5.2.B

LIRICS LMF Implementation

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	LIRICS LMF Implementation
Deliverable ID	**D5.2.B**
Document type	Report
Dissemination level	Public
Contractual date of delivery	M30
Actual date of delivery	14 September 2007
Status & version	Final
Work package, task & deliverable responsible	WP5, USFD
Author(s) & affiliation(s)	Adam Funk (USFD)
Additional contributor(s)	Marc Kemps-Snijders (MPI)
Keywords	LIRICS LMF Implementation

Document evolution

version	date	version	date
1.0	20/08/06		
1.9	24/08/07		
2.0	14/09/07		

1   Introduction

This brief document accompanies and explains the demonstrator/prototype software deliverable D5.2.B, "LMF: reference implementation".

2   Requirements

The LMF service is provided by the LEXUS server operated by MPI.

The LMF client is provided as a language resource plug-in for the GATE (General Architecture for Text Engineering) software, which is available from the http://gate.ac.uk/ GATE website.

3   Installation and use

The client software is delivered as a file **lmf-plugin.tar.gz**, which can be downloaded from the LIRICS website and unpacked using any standard unzip tool into a directory of the user's choice.   The plugin itself is contained in the **lmf-plugin** directory (which contains the **creole.xml** file).

The user follows the standard instructions (in the GATE User Guide) for downloading, installing and launching GATE, then clicks on the "Manage CREOLE plugins" icon and, in the plugin manager, adds a new plugin from the **lmf-plugin** directory.

This will add to the list of CREOLE plugins a line for the "lmf-plugin", which when selected will indicate that it provides the following resources: Lexicon Repository Viewer, Lexicon Viewer, Lexicon Repository and Lexicon.

The user then ticks the "Load now" box in this line and clicks "OK" at the bottom of the CREOLE plugin manager.

The user then right-clicks on "Language Resources" in the GATE GUI and selects "New" then "Lexicon Repository". Filling in a "Name" is optional. The repositoryURL field is completed by default with the URL for MPI's LEXUS service, but can be changed to use another service. The user should then fill in the userName and userPassword fields with his account details.

(The LEXUS service requires authentication; different accounts have access to a different range of lexica. It is possible to obtain a guest account by following the "register as guest" link from the http://lux07.mpi.nl/mpi/lexus/ web page.)

This will create a GATE language resource (LR) for the lexicon repository in the list of LRs in the GATE GUI. Double-clicking on it will open in the main pane, which will contain a list of the availabe lexica. Double-clicking on a lexicon from that list will create a corresponding lexicon LR in the list, which can be double-clicked to open and browse it.

4   Lexus web service

The lexus web service currently offers functionality for accessing and browsing the online environment. This enables web service clients to extract complete lexica and their accompanying schema definitions and to search for lexical entry matching supplied search criteria. The following methods are supported:

GetResources

Returns the list of resources accessible to the current user.

GetResource

Returns all data for a specific resource. Each resource is associated with ist own schema which may be obtained using the *GetSchema* method.

GetSchema

Returns the schema for a specific resource. Each schema is associated with ist own name spece. The name space identifier points to a location on the server which generates the available human readable documentation for the schema.

Search

Searches for lexical entries from a resource matching the specified search criteria. The method takes a query and a Map containing the name space declaration. The query interface allows searches across any part of the lexicoal entry content.

Example query

```
"/lex:lexicon/lex:lexicalEntry/lex:sense/lex:transcription[.
contains(\"ylap\")]"
       This example will return all lexical entries for which the
transcription attribute contains 'ylap'.
```

This example will contains a 'lex' name space identifier that is mapped to the actual name space :
```
nsMap.put( "lex",
"http://localhost:8080/lexus/schema/bGV4aWNvbi8yYzkwOTQ0YzEwYTYxNzgzM
DExMGE2MmMzZjQzMDAwMw==/");
```