Project no. 027391
Project acronym: LT4eL
Project title: Language Technology for eLearning

Instrument Specific Targeted Research Project
Thematic Priority Information Society Technology

**D4.1b ILIAS LMS with integrated functionalities and documentation – first cycle**

Due date of deliverable: 30-11-2007
Actual submission date: 21-12-2007

Start date of project: 1-12-2005
Duration: 30 Months

Organisation name of lead contractor for this deliverable: Swiss Federal Institute of Technology (ETHZ)

Revision [1]

---

<table>
<thead>
<tr>
<th>Dissemination Level</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>PU</td>
<td>Public</td>
</tr>
<tr>
<td>PP</td>
<td>Restricted to other programme participants (including the Commission Services)</td>
</tr>
<tr>
<td>RE</td>
<td>Restricted to a group specified by the consortium (including the Commission Services)</td>
</tr>
<tr>
<td>CO</td>
<td>Confidential, only for members of the consortium (including the Commission Services)</td>
</tr>
</tbody>
</table>

---

**Project co-funded by the European Commission within the Sixth Framework Programme (2002-2006)**
Overview

LT4eL is a project that offers a set of language technology based services for e-Learning platforms. The acronym stands for Language Technology for e-Learning. Basic services are keyword generation, definition detection and enhanced semantic and multilingual search capabilities. The tools support a more effective metadata annotation of learning objects, authoring of glossaries and better search functionalities in learning management systems.

The LT4eL tools are implemented as web services that are deployed in a Java web server architecture and can be used via a SOAP interface by other applications. This makes the LT4eL tools independent from the technology that is used in the e-Learning platform, since SOAP libraries are available for all major programming languages like Java, C++, C#, PHP, Python or Perl.

This document is part one of the deliverable 4.1 as described in the technical annex of the project. It is targeted at developers of other learning managements systems and provides a step-by-step procedure that explains how to download, install, configure the LT4eL tools. It also documents the web services offered by the tools and explains how to integrate them into another LMS. To see how these functionalities can be used and affect a user interface within a learning management system, refer to the User Documentation document of the LT4eL project. For possible future improvements please refer to the Changes and Future Plans Annex provided with the 4.1 deliverables.

All source code of the project is available at https://sourceforge.net/projects/lt4el.

Change Log

Version 0.1

- Basic implementation of sendNewLO(), getStatus(), deleteLO(), findKeywordCandidates(), getDefinitionCandidates(), search() and getConceptNeighbourhood()

Version 0.2

- Changes in properties file
- Search returns concepts related to search results
- Revised search implementation
- Increased stability of tools (especially for concurrent requests)
- getDomain() and domain parameters abandoned
- sendNewLo() input parameters corrected
- removed restriction of deleteLO() (language model is updated, when deleteLO is called now)
- removed restriction of getDefinitionCandidates() (local grammars are included within the package now)
- removed restriction for search
- search result includes related concepts for each LO now
• sendApprovedDefinitions() abandoned
• For more information see project’s M24 Changes Report
# Table of Contents

OVERVIEW ................................................................................................................................. 2
CHANGE LOG ............................................................................................................................... 2
TABLE OF CONTENTS .................................................................................................................. 4
ARCHITECTURE ........................................................................................................................... 5
INSTALLATION ............................................................................................................................. 5
  PREREQUISITES ....................................................................................................................... 5
  DOWNLOAD .............................................................................................................................. 6
  CONFIGURATION ..................................................................................................................... 6
  BUILDING AND DEPLOYING .................................................................................................... 7
WEB SERVICES ........................................................................................................................... 8
  SENDNEWLO .......................................................................................................................... 8
  GETSTATUS ........................................................................................................................... 8
  DELETELO .............................................................................................................................. 9
  FINDKEYWORDCANDIDATES .............................................................................................. 9
  SENDAPPROVEDKEYWORDS .............................................................................................. 10
  GETDEFINITIONCANDIDATES ............................................................................................ 10
  SEARCH ............................................................................................................................... 11
  GETCONCEPTNEIGHBOURHOOD ...................................................................................... 11
CLIENT EXAMPLE CODE ......................................................................................................... 12
  PHP ................................................................................................................................. 12
  JAVA ............................................................................................................................... 16
Architecture

The below picture shows the major components of the integration setup. The language technology server on the left provides the keyword extractor, definitory context finder and ontology management system functionalities.

The tools themselves are developed using the Java programming language and are hosted on a Java web server. The functionalities can be accessed by a learning management system through the web service interface. The web service interface separates the technologies used on both sides.

Installation

This section explains which prerequisites are required to use the LT4eL tools and how to build and deploy them. Please note that these are requirements for running the LT4eL tools server. Your learning management system may run on any other technology.

Prerequisites

To use the LT4eL tools you need:

- Apache Tomcat 5.5 or later
- Apache Ant 1.6.x or later

Apache Tomcat is a web container developed at the Apache Software Foundation (ASF). Apache Tomcat is the servlet container that is used in the official Reference Implementation for the Java Servlet and JavaServer Pages technologies. You can get Apache Tomcat and instructions about installation at [http://tomcat.apache.org/](http://tomcat.apache.org/).

Apache Ant is a Java-based build tool, like Make, but without Make's wrinkles and with the full portability of pure Java code. The LT4eL tools use an Ant script to build and deploy the tools in the Apache Tomcat web container. You can get Apache Ant and instructions about the installation at [http://ant.apache.org/](http://ant.apache.org/).
**Download**

You can find the latest release of the LT4eL tools at the SourceForge.net project page at [http://sourceforge.net/projects/lt4el/](http://sourceforge.net/projects/lt4el/). The package will be named something like lt4el.0.m24.beta.tar.gz. Please use the `tar` command to extract the package to a local folder.

```
> tar -xzvf lt4el.0.m24.beta.tar.gz
```

If you want to use the latest development snapshot of the tools, you can checkout them from the subversion server on Sourceforge.net.

```
> svn co https://lt4el.svn.sourceforge.net/svnroot/lt4el/trunk/ lt4el
```

**Configuration**

Before building the tools please make sure, that the CATALINA_HOME environment variable is set to your Tomcat directory. For example if you are using the bash shell, you could add the following line to your `~/.bash_profile` file.

```
export CATALINA_HOME=/<your_path>/apache-tomcat-5.5.20
```

To configure the LT4eL tools make a copy of the file `lt4el.props.template` int the LT4eL main directory and name this copy `lt4el.props`. This properties file contains a number of paths to directories that store various data of the LT4eL tool set.

```
#
# LT4eL properties file
#

# 1. Save a copy of this file as 'lt4el.props':
#    lt4el.props.template -> lt4el.props
# 2. Change the values according to your environment. Save the file.

# LO files directory, must include the necessary .dtd files
Docs=/path/to/lo/directory

# internal DB directory
Db=/path/to/database/directory

# lxtransduce tool path
Lxtransduce=/path/to/lxtransduce/binary

# grammars path
Grammars=/path/to/grammars/directory

# lexicons path
Lexicons=/path/to/lexicons/directory

# Search Index directory
```
Building and Deploying

After configuration you just need to call the ant command in the main directory to build and deploy the LT4eL tools. The default target of the ant script builds and deploys everything into two directories in the Tomcat webapps/ folder. The directory „lt4eL“ contains all tools except the web services, whereas the web services are deployed into a directory called „lt4elservices“.

Beside the default target, the ant script offers a number of other targets for the different build and deployment steps. If you type „ant help“ you will get a list of all targets.

ant deploy.all: Compiles and deploys everything into Tomcat webapp folder. (default target)
ant deploy.restart: Same as deploy.all, tries to restart Tomcat (Unix only).
ant help: This help message.
ant clean: Cleans build and deploy directory.
ant deploy.kwext: Compiles and deploys keyword extractor component.
ant compile.kwext: Compiles keyword extractor component.
ant deploy.defcon: Compiles and deploys definition finder component.
ant compile.defcon: Compiles defcontext component.
ant deploy.gui: Compiles and deploys GUI component.
ant compile.gui: Compiles GUI component.
ant build.war: build war-file for tomcat deployment
ant compile.webservices: Compiles webservices component.
ant deploy.webservices: Deploy webservises component.
Web Services

All functionalities of the LT4eL tools are implemented as web services. You can use them with standard technologies like WSDL and SOAP. After successfully installing the tools you can access the web service description at:

http://<your_tomcat_url>/lt4elservice/services/Lt4elService?wsdl

The current specification lists eight implemented methods.

sendNewLo

Sending a new learning object to the language technology server. This is the first method that should be invoked right after a new learning object is created in or uploaded to a learning management system.

Current Restrictions

• The (text) content of the learning object is not attached to this function as it is intended to be in the final version. Instead of this, a local path to the learning object is the third input parameter (filename) of the function.

• The input file must be an annotated learning object, i.e. an XML file that respects the LT4eLAna DTD.

• The parameter attach is not used yet.

Input Parameters

• loid (xsd:string): Learning Object ID. This ID is used to identify the learning object. It is assumed, that this ID is generated in the learning management system, when new learning objects are created. This ID is used as an input/output parameter in most if the other functions.

• language (xsd:string): The (main) language of the learning object. The language must be represented by a two-letter code as defined in ISO 639-1. See http://www.oasis-open.org/cover/iso639a.html for details.

• filename (xsd:string): Local path to the learning object file. (see Current Restrictions)

• attach (xsd:boolean): Specifies, whether keywords are automatically attached to the learning object or not. Usually keywords are not attached at this point. The user has to confirm appropriate keywords in the learning management, before they are attached to the learning object.

• filename2 (xsd:string): Filename of ontological annotated file. If semantic search should be available, this file must be

Output Parameters

• accepted (xsd:boolean): True, if the language technology server successfully received the learning object, false otherwise.

getStatus

Get processing status of a learning object, that has been sent to the language server. This function can be used after sendNewLO has been invoked for a learning object. Since the processing and conversion of a new learning object may take several minutes, this function tells the learning management system the status of the processing. It can be used to be
displayed for the user and to deactivate certain functions, that cannot be used until the processing status is **finished** (2).

**Current Restrictions**

- A processing chain starting from raw learning objects to representations used internally by the language technology server (linguistic annotation) has not been implemented yet. The only status values that are used at this time are 0 and 2.

**Input Parameters**

- **loid (xsd:string):** Learning Object ID. (see `sendNewLo`).

**Output Parameters**

- **status (xsd:int):** Current processing status.
  - 1  Failed.
  - 0  Unknown.
  - 1  In process.
  - 2  Finished.

**deleteLO**

Delete a learning objects representation on the language technology server. This function should be called when a learning object is deleted in the learning management system. After successfully invoking `deleteLO`, subsequent calls to `getStatus` will return Unknown (0) again.

**Input Parameters**

- **loid (xsd:string):** Learning Object ID. (see `sendNewLo`).

**Output Parameters**

- **success (xsd:boolean):** True, if the language technology server successfully removed the learning object, false otherwise.

**findKeywordCandidates**

Find candidate terms for keyword annotation of a learning object. This method should be used by a learning management system, when a learning object is annotated with keywords. A lot of learning management systems come with support for LOM or Dublin Core meta data. Both of these standards allow the annotation with keywords. However, simple tagging systems work the same way and could use this function to propose keywords to an annotator.

**Current Restrictions**

- In general this function works better, when the internal language model gets larger. This means if only a small number of learning objects have been sent to the language technology server by using the function `sendNewLo`, the quality of the results is suboptimal. Good quality can be expected after 30-50 mid-size learning objects have been sent to the language technology server.

**Input Parameters**

- **loid (xsd:string):** Learning Object ID. (see `sendNewLo`).
- **maxnum (xsd:int):** Maximum number of keywords that should be returned by the function.
- **method (xsd:string):** Method of keyword detection.
tfidf  TF-IDF
ridf  R-IDF
adridf  ADR-IDF (currently best performing)

Output Parameters

- *keywords (ArrayOfString)*: Ranked keywords.

Types

- ArrayOfString
  - minOccurs 0
  - maxOccurs unbounded
  - type xsd:string

`sendApprovedKeywords`

Send all keywords related to a learning object, that have been approved by a human annotator back to the learning technology server. The language technology server could later use this information, e.g. during the search.

Input Parameters

- *loid (xsd:string)*: Learning Object ID
- *keywords (tns:ArrayOfString)*: Keywords approved by an author.

Output Parameters

- *success (xsd:boolean)*: Success true/false

`getDefinitionCandidates`

Get a set of terms and candidate definitions for a learning object. This method can be used by learning managements systems to support semi-automatic generation of glossaries with terms and definitions found in a learning object.

Input Parameters

- *loid (xsd:string)*: Learning Object ID. (see `sendNewLo`).

Output Parameters

- *definitions (ArrayOfDefinition)*: Array of terms and related defining texts.

Types

- ArrayOfDefinition
  - minOccurs 0
  - maxOccurs unbounded
  - type Definition

- Definition
  - type sequence of
    - definedTerm (xsd:string)
    - definingText (xsd:string)
**search**

Search for learning objects. This function supports extended search capabilities based on fulltext search, keyword based search and semantic search. Semantic search supports multilingual retrieval of learning objects by using lexicons and an ontology.

**Input Parameters**

- **terms (ins:ArrayOfString):** The search terms
- **languages (ins:ArrayOfString):** Search term languages
- **retrieval_languages (ins:ArrayOfString):** Languages of target learning objects
- **method (xsd:string):** Search Method ("SEMANTIC", "KEYWORD", "FULLTEXT")
- **search_concepts (ins:ArrayOfArrayOfString):** Concepts, if learning objects related to concepts are searched

**Output Parameters**

- **result (xsd:ArrayOfAnyType):** Array of list of found learning object IDs and list of found ontology fragments. result[0] contains the learning objects IDs in a key, value array list. Keys are the learning objects IDs, value is a list of concepts related to each learning object. result[1] holds the ontology fragments.

**Types**

- **ArrayOfAnyType**
  
  - minOccurs: 0
  - maxOccurs: unbounded
  - type: xsd:anyType

**getConceptNeighbourhood**

Get relations and related concepts of a ontology concept. This function can be used to support browsing through the ontology in the learning management system’s interface.

**Input Parameters**

- **concepts (ins:ArrayOfString):** Concepts
- **languages (ins:ArrayOfString):** Languages (the entries in the lexicons for these languages and the concepts included in the fragments will be returned)

**Output Parameters**

- **fragments (ins:ArrayOfString):** Ontology fragments for concepts.
Client Example Code

SOAP libraries are available for all major programming languages. In this section we want to provide some example implementations that show how to use the LT4eL web services. If you write a similar code in a programming language that is not listed here, please do not hesitate to send us your code. We would like to extend this chapter with additional examples.

To write an adapter to the LT4eL webservices is the major step to use the functionalities within your application. You can then call the adapter’s functions to extend your application with the functionalities of the LT4eL tools.

**PHP**

For PHP a popular SOAP library is nuSOAP (see [http://sourceforge.net/projects/nusoap/](http://sourceforge.net/projects/nusoap/)). The following example code uses this library. It implements a wrapper class for all functions of the language technology server.

```php
<?php

// LT4eL Example Class

// You must install nuSoap to use this example.
// See http://sourceforge.net/projects/nusoap/
include_once ("./nusoap.php");

/**
 * LT Server Adapter
 */
class LTServerAdapter
{

    /**
     * Constructor
     */
    function __construct()
    {
        // replace the following URL with your LT server address
        $this->setServer("http://localhost:8080/lt4elservice/services/Lt4elService");

        // initialize soap client
        $this->client = new soap_client($this->getServer());
        $this->client->soap_defencoding = 'UTF-8';
        $this->client->decode_utf8 = false;
        $this->client->response_timeout = 240;
        if ($err = $this->client->getError())
        {
            die('SOAP Client constructor error: ' . $err);
        }
    }

    /**
     * Set Server.
     * @param string $a_server Server
     */
    function setServer($a_server)
    {
        $this->server = $a_server;
    }
}
```
function getServer()
{
    return $this->server;
}

function sendNewLO($a_lo_id, $a_language, $a_filename, $a_attach, $a_ont_filename)
{
    // sendNewLO call
    $accepted = $this->client->call('sendNewLO',
        array('loid' => (string) $a_lo_id,
            'language' => $a_language,
            'filename' => $a_filename,
            'attach' => $a_attach,
            'filename2' => $a_ont_filename));

    return $accepted;
}

function getStatus($a_lo_id)
{
    // getStatus call
    $status = $this->client->call('getStatus',
        array('loid' => (string) $a_lo_id));

    switch($status)
    {
    case 0:
        $str_status = "UNKNOWN";
        break;
    case 1:
        $str_status = "IN_PROCESS";
        break;
    case -1:
        $str_status = "FAILED";
        break;
    case 2:
        $str_status = "FINISHED";
        break;
    }

    return $str_status;
}

function getProcessingStatus($a_lo_id)
{
    // getProcessingStatus call
    $status = $this->client->call('getProcessingStatus',
        array('loid' => (string) $a_lo_id));

    switch($status)
    {
    case 0:
        $str_status = "UNKNOWN";
        break;
    case 1:
        $str_status = "IN_PROCESS";
        break;
    case -1:
        $str_status = "FAILED";
        break;
    case 2:
        $str_status = "FINISHED";
        break;
    }

    return $str_status;
}

function deleteLearningObject($a_lo_id)
{
    // deleteLearningObject call
    $status = $this->client->call('deleteLearningObject',
        array('loid' => (string) $a_lo_id));

    switch($status)
    {
    case 0:
        $str_status = "UNKNOWN";
        break;
    case 1:
        $str_status = "IN_PROCESS";
        break;
    case -1:
        $str_status = "FAILED";
        break;
    case 2:
        $str_status = "FINISHED";
        break;
    }

    return $str_status;
}
* @param string $a_lo_id learning object id
* @return boolean success
*/
function deleteLO($a_lo_id)
{
    // deleteLo call
    $success = $this->client->call('deleteLO',
        array('loid' => (string) $a_lo_id));

    return $success;
}

/**
* Find Keyword Candidates
* @param string $a_lo_id learning object id
* @param string $a_number number of keywords
* @param string $a_method method
* @return string keyword candidates
*/
function findKeywordCandidates($a_lo_id, $a_number, $a_method = "adridf")
{
    // findKeywordsCandidates call
    $keywords = $this->client->call('findKeywordCandidates',
        array('loid' => (string) $a_lo_id,
            'maxnum' => (int) $a_number,
            'method' => $a_method));

    return $keywords;
}

/**
* Send approved keywords
* @param string $a_lo_id learning object id
* @param string $a_keywords approved keywords
* @return boolean success
*/
function sendApprovedKeywords($a_lo_id, $a_keywords)
{
    if (!is_array($a_keywords))
    {
        $a_keywords = array($a_keywords);
    }

    // sendApprovedKeywords call
    $success = $this->client->call('sendApprovedKeywords',
        array('loid' => (string) $a_lo_id,
            'keywords' => $a_keywords));

    return $success;
}

/**
* Get Definition Candidates
* @param string $a_lo_id learning object id
* @return array definition candidates array of array
*
function getDefinitionCandidates($a_lo_id)
{
    // findKeywordsCandidates call
    $definitions = $this->client->call('getDefinitionCandidates',
        array('loid' => (string) $a_lo_id));

    return $ret;
}

/**
* Get concepts and relations of a concept
* @param string $a_concept concept (in english)
* @param string $a_preferred_lang preferred languages
* @return string ontology fragments (for XML see WP3)
*/
function getConceptNeighbourhood($a_concepts, $a_preferred_langs = "en")
{
    if (!is_array($a_concepts))
    {
        $a_concepts = array($a_concepts);
    }

    if (!is_array($a_preferred_langs))
    {
        $a_preferred_langs = array($a_preferred_langs);
    }

    $frags = $this->client->call('getConceptNeighbourhood',
        array('concept' => $a_concepts,
            'preferred_languages' => $a_preferred_langs));

    return $frags;
}

/**
* Search for learning objects
* @param string $a_search_terms array of search string
* @param array $a_languages search terms languages (only needed for semantic)
* @param array $a_r_languages retrieval languages (LO languages)
* @param string $a_method "SEMANTIC", "KEYWORD", or "FULLTEXT"
* @param array $a_concepts look for LOs that are related to concepts
* @return array $result[0] contains found LOs $result[1] contains found fragments
*/
function search($a_search_terms, $a_languages, $a_r_languages, $a_method, $a_concepts = "")
{
    if ($a_search_concepts == "")
    {
        $a_search_concepts = array(array());
    
    
    
    
}
Java

For Java a popular SOAP library is Axis (see http://ws.apache.org/axis/). This example uses Axis Version 1.4. The first thing you have to do is to generate the Mapping-Classes, WebService-Stubs and the Definition-Classes. To do this download the axis-distribution, extract it, change to the axis-directory/lib and then run the following command:

code

```
java -cp commons-logging-1.0.4.jar:axis.jar:commons-discovery-0.2.jar:saaj.jar:jaxrpc.jar:log4j-1.2.8.jar:wsdl4j-1.5.1.jar org.apache.axis.wsdl.WSDL2Java
```

http://localhost:8080//lt4elservice/services/Lt4elService?wsdl

The classes are generaed in the eu-subfolder The following example code uses axis and the autogenerated classes. It implements a wrapper class for all functions of the language technology server.

```java
package eu.lt4el.webservices;

import java.rmi.RemoteException;
import org.apache.axis.AxisFault;
import eu.lt4el.defcontext.Definition;
import eu.lt4el.webservices.service.Lt4ElServiceHttpBindingStub;
import eu.lt4el.webservices.service.Lt4ElServiceLocator;
import eu.lt4el.webservices.service.Lt4ElServicePortType;

public class LTServerAdapter {
    private Lt4ElServiceLocator serviceLocator;
    private Lt4ElServicePortType serviceBinding;

    public ClientImpl() throws WebserviceException {
        serviceLocator = new Lt4ElServiceLocator();
        try {
            serviceBinding = new Lt4ElServiceHttpBindingStub(serviceLocator);
        } catch (AxisFault e) {
            throw new Exception("Can't instaniate ClientImpl!", e);
        }
    }

    /**
     * Constructor
     **/
    public LTServerAdapter(String address) throws AxisFault {
        serviceLocator = new Lt4ElServiceLocator();
        serviceLocator.setLt4elServiceHttpPortEndpointAddress(address);
        serviceBinding = new Lt4ElServiceHttpBindingStub(serviceLocator);
    }
}
```
public boolean deleteLO(String loid) throws Exception {
    try {
        return serviceBinding.deleteLO(loid);
    } catch (RemoteException e) {
        throw new Exception("Can't delete LO. with id = " + loid, e);
    }
}

public String[] findKeywordCandidates(String loid, int maxnum, String method) throws Exception {
    try {
        return serviceBinding.findKeywordCandidates(loid, maxnum, method);
    } catch (RemoteException e) {
        throw new Exception("Error! Write your message here!", e);
    }
}

public String[] getConceptNeighbourhood(String[] concepts, String[] languages) throws Exception {
    try {
        return serviceBinding.getConceptNeighbourhood(concepts, languages);
    } catch (RemoteException e) {
        throw new Exception("Error! Write your message here!", e);
    }
}

public Definition[] getDefinitionCandidates(String loid) throws Exception {
    try {
        return serviceBinding.getDefinitionCandidates(loid);
    } catch (RemoteException e) {
        throw new Exception("Error! Write your message here!", e);
    }
}

public int getStatus(String loid) throws Exception {
    try {
        return serviceBinding.getStatus(loid);
    } catch (RemoteException e) {
        throw new Exception("Error! Write your message here!", e);
    }
}
public Object[] search(String[] terms, String[] languages,
    String[] retrieval_languages, String method, String[] search_concepts,
    String sys_lang) throws Exception {
    try {
        return serviceBinding.search(terms, languages, retrieval_languages,
                                         method, search_concepts, sys_lang);
    } catch (RemoteException e) {
        throw new Exception("Error! Write your message here!", e);
    }
}

public boolean sendApprovedKeywords(String loid, String[] keywords) throws Exception {
    try {
        return serviceBinding.sendApprovedKeywords(loid, keywords);
    } catch (RemoteException e) {
        throw new Exception("Error! Write your message here!", e);
    }
}

public boolean sendNewLO(String loid, String language, String filename, boolean attach, String filename2) throws Exception {
    try {
        return serviceBinding.sendNewLO(loid, language, filename, attach, filename2);
    } catch (RemoteException e) {
        throw new Exception("Error! Write your message here!", e);
    }
}
LT4eL – Work Package 4
D4.1 Annex. M24 Changes and Future Plans
Change Log

Version 0.1

• Changes done due to testing and validation since initial integration up to M24.
Table of Contents

CHANGE LOG ........................................................................................................................................................................... 2
TABLE OF CONTENTS ................................................................................................................................................................. 3
OVERVIEW .................................................................................................................................................................................. 4
MAJOR CHANGES UP TO M24 ...................................................................................................................................................... 4
PLANS FOR M25 TO M30 ............................................................................................................................................................ 4
BUG TRACKER REPORT .............................................................................................................................................................. 6
Overview
This document gives a broad overview about the changes that have been made to the LT4eL tools and the extended ILIAS prototype due to testing and validation activities after the initial integration of the tools into ILIAS up to M24. To facilitate the management of issues a bug tracker has been setup at the sourceforge.net’s project page. The tracker includes detailed information on many issues including detailed description and correspondence between reporters and developers. The tracker is openly accessible at: https://sourceforge.net/tracker/?atid=881658&group_id=177544&func=browse

Major Changes up to M24
During September we have split the ongoing development of the tools into two development branches. One dedicated for new development, a second for bug fixing. At the end of October we decided to backport the new development version of the semantic search to the bugfix branch used on the validation server. This has been done due to stability problems of the first version, that occured when multiple users accessed the system.

The result list of the semantic search displays now all concepts that are related to the query per result item. If multiple topics are found within a document, the document is ranked higher.

The ontology browser now features the term definitions from the lexicon as a tooltip box. Since we do not have translations of definitions in all languages (only English and Bulgarian), the system falls back to English, if the definition is not available in the user’s language.

The list of keyword extractor candidates listed per block in the ILIAS user interface has been increased from 5 to 10 due to pre-tests feedback and WP2 findings. The overall list of keywords has been decreased from 100 to 50. The keyword generation method has been switched from RIDF to ADRIDF since WP2 results have shown that ADRIDF performs better.

Various changes have been done related to usability of the glossary generator in ILIAS. In the first version the glossaries have not been attached closely enough to the learning objects. They are now listed together with the learning objects within the ILIAS repository and can be accessed directly from the learning objects. Tutors generating a glossary, can delete them from within the learning object’s properties screen now. (See D4.1 User Documentation for screenshots)

The translations of the ILIAS user interface into the languages of all partners have been constantly improved. Misleading messages and terms in interface elements have been replaced by more suitable ones.

Plans for M25 to M30
Beside the integration of the automatic annotation processing chain several remaining issues form the validation feedback will be tackled. The focus will be more on search functionalities and the ontology presentation, since the feedback related to the integration of the keyword generator and the definition finder was quite positive.
There are also some open issues related to the ILIAS learning management system itself (e.g. usability of the user registration). However these will be tackled with a low priority since they are not directly related to the tools developed within the project.

The user interface of the search screen will need a revision. We need to make clearer, how the ordering of the search results reflects the relevance of the learning objects that have been found. We also plan to display Google-like snapshots of the documents which highlight search terms within their surrounding context to give users a better picture of the learning object’s content.

It may be necessary to introduce an additional search mode that enables experienced users to formulate advanced search queries (conjunction / disjunction / not). However, other users would appreciate a more simple search user interface. This is especially the case for the combination of search interface and ontology browser. We will work on alternatives and discuss the interface with the beginning of 2008.

All improvements should be finally integrated into a new version of the extended prototype by February 2008 before the final validation phase will start.
Bug Tracker Report

The table lists all closed issues of the project’s bug tracker. There are currently (6. Dec 2007) 155 issues listed in the tracker, 34 of them are open issues.

<table>
<thead>
<tr>
<th>Summary</th>
<th>Date</th>
<th>Pr</th>
<th>Status</th>
<th>Assigned</th>
<th>Submitted</th>
</tr>
</thead>
<tbody>
<tr>
<td>Keyword Extractor on ILIAS is generating funny results</td>
<td>15-Nov-07 2:05</td>
<td>9</td>
<td>Closed</td>
<td>qiq</td>
<td>claudiaborg</td>
</tr>
<tr>
<td>Account</td>
<td>13-Nov-07 2:35</td>
<td>5</td>
<td>Closed</td>
<td>alexkill</td>
<td>elinewest</td>
</tr>
<tr>
<td>Dutch translation of concepts</td>
<td>13-Nov-07 0:44</td>
<td>5</td>
<td>Closed</td>
<td>qiq</td>
<td>elinewest</td>
</tr>
<tr>
<td>no language restriction with definition finder</td>
<td>30-Oct-07 4:42</td>
<td>5</td>
<td>Closed</td>
<td>alexkill</td>
<td>jantinetrap</td>
</tr>
<tr>
<td>Duplicated LOs on l5as portal</td>
<td>5-Oct-07 5:49</td>
<td>5</td>
<td>Closed</td>
<td>picispic</td>
<td>qiq</td>
</tr>
<tr>
<td>glossary link to LOs</td>
<td>3-Oct-07 4:14</td>
<td>5</td>
<td>Closed</td>
<td>alexkill</td>
<td>de_ou_lt4el</td>
</tr>
<tr>
<td>glossary offline</td>
<td>3-Oct-07 4:05</td>
<td>9</td>
<td>Closed</td>
<td>nobody</td>
<td>de_ou_lt4el</td>
</tr>
<tr>
<td>search forgets language settings</td>
<td>2-Oct-07 4:29</td>
<td>5</td>
<td>Closed</td>
<td>alexkill</td>
<td>rybencjusz</td>
</tr>
<tr>
<td>Minimum size of search token</td>
<td>1-Oct-07 8:19</td>
<td>5</td>
<td>Closed</td>
<td>qiq</td>
<td></td>
</tr>
<tr>
<td>Semantic Search Performance (Development Server)</td>
<td>1-Oct-07 7:11</td>
<td>5</td>
<td>Closed</td>
<td>qiq</td>
<td>mfa1974</td>
</tr>
<tr>
<td>ILIAS LO status</td>
<td>1-Oct-07 1:43</td>
<td>5</td>
<td>Closed</td>
<td>alexkill</td>
<td>mfa1974</td>
</tr>
<tr>
<td>ILIAS unavailable</td>
<td>28-Sep-07 3:36</td>
<td>5</td>
<td>Closed</td>
<td>alexkill</td>
<td>bwojtowicz</td>
</tr>
<tr>
<td>validation server for tutors only in English or Czech</td>
<td>26-Sep-07 3:12</td>
<td>5</td>
<td>Closed</td>
<td>qiq</td>
<td>jantinetrap</td>
</tr>
<tr>
<td>Semantic Search Performance (Validation Server)</td>
<td>26-Sep-07 1:11</td>
<td>4</td>
<td>Closed</td>
<td>alexkill</td>
<td></td>
</tr>
<tr>
<td>Searching for a definition</td>
<td>24-Sep-07 3:19</td>
<td>9</td>
<td>Closed</td>
<td>de_ou_lt4el</td>
<td>eelcomosse</td>
</tr>
<tr>
<td>language consistency in interface</td>
<td>19-Sep-07 8:31</td>
<td>5</td>
<td>Closed</td>
<td>alexkill</td>
<td>eelcomosse</td>
</tr>
<tr>
<td>LOs &quot;Textverarbeitung Word &amp; Homepage erstellen&quot;</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Webdesign</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>LO &quot;Was ist eigentlich eLearning&quot; still not available</td>
<td>14-Sep-07 5:31</td>
<td>6</td>
<td>Closed</td>
<td>eelcomosse</td>
<td>fabiennel</td>
</tr>
<tr>
<td>Dutch: only html files available</td>
<td>14-Sep-07 5:28</td>
<td>6</td>
<td>Closed</td>
<td>picispic</td>
<td>fabiennel</td>
</tr>
<tr>
<td>search also when pressing enter</td>
<td>14-Sep-07 3:18</td>
<td>7</td>
<td>Closed</td>
<td>alexkill</td>
<td>elinewester</td>
</tr>
<tr>
<td>LO &quot;ACL course&quot;: view is empty</td>
<td>12-Sep-07 4:38</td>
<td>5</td>
<td>Closed</td>
<td>alexkill</td>
<td>fabiennel</td>
</tr>
<tr>
<td>English LO in German repository</td>
<td>12-Sep-07 3:33</td>
<td>3</td>
<td>Closed</td>
<td>alexkill</td>
<td>fabiennel</td>
</tr>
<tr>
<td>Semantic vs fulltext search</td>
<td>10-Sep-07 3:12</td>
<td>5</td>
<td>Closed</td>
<td>rybencjusz</td>
<td>bwojtowicz</td>
</tr>
<tr>
<td>fwp2ont files disappearing along with fwp2</td>
<td>7-Sep-07 9:19</td>
<td>5</td>
<td>Closed</td>
<td>picispic</td>
<td>rybencjusz</td>
</tr>
<tr>
<td>definition finder: not active</td>
<td>7-Sep-07 1:43</td>
<td>5</td>
<td>Closed</td>
<td>qiq</td>
<td>osenova</td>
</tr>
<tr>
<td>No search results at all</td>
<td>6-Sep-07 6:39</td>
<td>5</td>
<td>Closed</td>
<td>nobody</td>
<td>jantinetrap</td>
</tr>
<tr>
<td>No search results at all</td>
<td>6-Sep-07 5:52</td>
<td>5</td>
<td>Closed</td>
<td>nobody</td>
<td>jantinetrap</td>
</tr>
<tr>
<td>Removing LO's from a list not possible</td>
<td>6-Sep-07 5:37</td>
<td>5</td>
<td>Closed</td>
<td>nobody</td>
<td>jantinetrap</td>
</tr>
<tr>
<td>semantic search gives no results</td>
<td>6-Sep-07 3:55</td>
<td>8</td>
<td>Closed</td>
<td>qiq</td>
<td>elinewester</td>
</tr>
<tr>
<td>error in files</td>
<td>3-Sep-07 6:58</td>
<td>6</td>
<td>Closed</td>
<td>rosgaudio</td>
<td>eelcomosse</td>
</tr>
<tr>
<td>navigate through ontology in Internet Explorer</td>
<td>3-Sep-07 2:50</td>
<td>5</td>
<td>Closed</td>
<td>alexkill</td>
<td>eelcomosse</td>
</tr>
<tr>
<td>No Polish docs in semantic search</td>
<td>31-Aug-07 7:44</td>
<td>5</td>
<td>Closed</td>
<td>qiq</td>
<td>bwojtowicz</td>
</tr>
<tr>
<td>Wrong links on portal</td>
<td>30-Aug-07 12:45</td>
<td>6</td>
<td>Closed</td>
<td>picispic</td>
<td>eelcomosse</td>
</tr>
<tr>
<td>KW Search EN: not all LOs listed in results</td>
<td>30-Aug-07 9:56</td>
<td>5</td>
<td>Closed</td>
<td>nobody</td>
<td>claudiaborg</td>
</tr>
<tr>
<td>EN LO: Definition Finder producing strange results</td>
<td>30-Aug-07 9:16</td>
<td>5</td>
<td>Closed</td>
<td>rybencjusz</td>
<td>claudiaborg</td>
</tr>
<tr>
<td>LuceneRepository &amp; IndexFiles must be configurable</td>
<td>30-Aug-07 0:18</td>
<td>4</td>
<td>Closed</td>
<td>neli_tinchev</td>
<td>qiq</td>
</tr>
<tr>
<td>Repository does not reflect search database!</td>
<td>29-Aug-07 5:23</td>
<td>8</td>
<td>Closed</td>
<td>qiq</td>
<td>eelcomosse</td>
</tr>
<tr>
<td>deleteLO procedure stopped suddenly by tomcat/ilias?</td>
<td>28-Aug-07 8:04</td>
<td>5</td>
<td>Closed</td>
<td>holtmann</td>
<td>rybencjusz</td>
</tr>
<tr>
<td>RO: status FINISHED &amp; error parsing</td>
<td>27-Aug-07 16:39</td>
<td>5</td>
<td>Closed</td>
<td>corinfor</td>
<td>corinfor</td>
</tr>
<tr>
<td>LO EN: not viewable</td>
<td>27-Aug-07 9:28</td>
<td>5</td>
<td>Closed</td>
<td>claudiaborg</td>
<td>claudiaborg</td>
</tr>
<tr>
<td>En/RO LO: Status Unknown</td>
<td>27-Aug-07 3:32</td>
<td>8</td>
<td>Closed</td>
<td>rybencjusz</td>
<td>claudiaborg</td>
</tr>
<tr>
<td>LT server database corruption</td>
<td>24-Aug-07 8:18</td>
<td>9</td>
<td>Closed</td>
<td>rybencjusz</td>
<td>qiq</td>
</tr>
<tr>
<td>BG LOs: Semantic search</td>
<td>24-Aug-07 5:57</td>
<td>5</td>
<td>Closed</td>
<td>alexkill</td>
<td>osenova</td>
</tr>
</tbody>
</table>
Definitions created by grammar???
Definitions without Defined Terms
Semantic Search floods Tomcat Log
Fulltext search not working anymore
Keyword based search is not working.
variable may not have been initialized
KeyWord Search: problems with multi-words keywords
files disappeared from portal
Multiple grammars for some languages on the portal
LO can be added twice
Search language
Definitions of concepts
Search using 'related concepts'
Definitions concepts
Search language
Links in files
Automatic log out
GCD and KW not working for previously uploaded docs
need to kill database and create it again
Portuguese LOs in working version
Tool recommendation for browsing Berkeley DB files
variable may not have been initialized
The semantic search is a little bit slow
Keyword based search is not working.
Fulltext search not working anymore
Retrieval Languages are not respected
Semantic Search floods Tomcat Log
Definitions without Defined Terms
Problem in getConceptNeighbourhood
definitions created by grammar???
Added German LO empty
<table>
<thead>
<tr>
<th>Title</th>
<th>Date</th>
<th>Status</th>
<th>Assignee</th>
<th>REVIEWER</th>
</tr>
</thead>
<tbody>
<tr>
<td>UI inconsistency: table width</td>
<td>22-Jun-07 0:20</td>
<td>Closed</td>
<td>alexkill</td>
<td>juanromero</td>
</tr>
<tr>
<td>Support of space in document ID</td>
<td>21-Jun-07 10:02</td>
<td>Closed</td>
<td>neli_tinchev</td>
<td>eelcomosse</td>
</tr>
<tr>
<td>non-ASCII comments in common.WFTSearchKey.java:21</td>
<td>21-Jun-07 8:35</td>
<td>Closed</td>
<td>rybencjusz</td>
<td>eelcomosse</td>
</tr>
<tr>
<td>suggestion: change layout definitions</td>
<td>21-Jun-07 5:12</td>
<td>Closed</td>
<td>alexkill</td>
<td>eelinewester</td>
</tr>
<tr>
<td>Dutch UNKNOWN LOs</td>
<td>21-Jun-07 4:30</td>
<td>Closed</td>
<td>eelinewester</td>
<td>eelinewester</td>
</tr>
<tr>
<td>Portuguese «Unfinished» LOs</td>
<td>20-Jun-07 7:00</td>
<td>Closed</td>
<td>alexkill</td>
<td>mfa1974</td>
</tr>
<tr>
<td>non-English characters seen as ? in Ilias</td>
<td>15-Jun-07 1:47</td>
<td>Closed</td>
<td>alexkill</td>
<td>rybencjusz</td>
</tr>
<tr>
<td>Lexicons are ignored</td>
<td>13-Jun-07 6:36</td>
<td>Closed</td>
<td>qiq</td>
<td>qiq</td>
</tr>
<tr>
<td>Adapt list/get Scripts to retrieve LOs for search</td>
<td>13-Jun-07 4:39</td>
<td>Closed</td>
<td>qiq</td>
<td>alexkill</td>
</tr>
<tr>
<td>Management of ontological annotated LOs for Search</td>
<td>13-Jun-07 4:33</td>
<td>Closed</td>
<td>picispic</td>
<td>alexkill</td>
</tr>
<tr>
<td>English Keywords in Ilias - double entries</td>
<td>12-Jun-07 8:02</td>
<td>Closed</td>
<td>alexkill</td>
<td>claudiaborg</td>
</tr>
<tr>
<td>Build for ose, oms and lex</td>
<td>12-Jun-07 5:22</td>
<td>Closed</td>
<td>alex_simov</td>
<td>eelcomosse</td>
</tr>
<tr>
<td>Search build</td>
<td>12-Jun-07 4:49</td>
<td>Closed</td>
<td>eelcomossel</td>
<td>alexkill</td>
</tr>
<tr>
<td>NullPointerException in transformSimpleQuery</td>
<td>12-Jun-07 4:38</td>
<td>Closed</td>
<td>neli_tinchev</td>
<td>alexkill</td>
</tr>
<tr>
<td>no eu.lt4el.ose in build.xml</td>
<td>12-Jun-07 2:39</td>
<td>Closed</td>
<td>alexkill</td>
<td>rybencjusz</td>
</tr>
<tr>
<td>deleteLO() must update language model</td>
<td>12-Jun-07 2:26</td>
<td>Closed</td>
<td>rybencjusz</td>
<td>alexkill</td>
</tr>
<tr>
<td>English Grammar / Definition Finder</td>
<td>12-Jun-07 1:43</td>
<td>Closed</td>
<td>qiq</td>
<td>alexkill</td>
</tr>
<tr>
<td>output of search: ranking</td>
<td>11-Jun-07 9:53</td>
<td>Closed</td>
<td>eelcomossel</td>
<td>eelcomosse</td>
</tr>
<tr>
<td>output of search(): use super/subconcepts</td>
<td>11-Jun-07 9:51</td>
<td>Closed</td>
<td>eelcomossel</td>
<td>eelcomosse</td>
</tr>
<tr>
<td>output of search: shared concepts</td>
<td>11-Jun-07 9:50</td>
<td>Closed</td>
<td>eelcomossel</td>
<td>eelcomosse</td>
</tr>
<tr>
<td>adding LOs</td>
<td>11-Jun-07 7:16</td>
<td>Closed</td>
<td>qiq</td>
<td>abbranco</td>
</tr>
<tr>
<td>sendApprovedKeywords()</td>
<td>8-Jun-07 11:37</td>
<td>Closed</td>
<td>rybencjusz</td>
<td>alexkill</td>
</tr>
<tr>
<td>Adding LOs using cmd-line interface is slow.</td>
<td>8-Jun-07 1:38</td>
<td>Closed</td>
<td>qiq</td>
<td>qiq</td>
</tr>
</tbody>
</table>
Overview

LT4eL is a project that offers a set of language technology based services for e-Learning platforms. The acronym stands for Language Technology for e-Learning. Basic services are keyword generation, definition detection and enhanced semantic and multilingual search capabilities. The tools support a more effective metadata annotation of learning objects, authoring of glossaries and better search functionalities in learning management systems.

This document is part two of the deliverable 4.1 as described in the technical annex of the project. This document is an extension to the documentation of the ILIAS learning management system (Version 3.7.6). It addresses all functionalities that have been changed, extended or added due to the developments of the LT4eL project:

- Glossary Generation: Semi-automatic extraction of glossary terms and definitions from existing learning objects to facilitate the generation of glossaries.
- Extended Search Engine: The search engine makes use of additional ontology-based metadata-based (keywords) information. Users can browse the ontology and perform subsequent searches to find learning objects related to concepts in the ontology.
- Personal Desktop: Users can create references to learning objects on their personal desktop on the basis of search results. They can rearrange learning objects according to their individual needs.

All source code of the project is available at https://sourceforge.net/projects/lt4el.
Change Log

Version 0.1
• Basic implementation LT4eL functionalities.

Version 0.2
• Show 10 instead of 5 keywords per block. Show max. 50 in total.
• Added administration section.
• Added context of definitions in glossary generation screen.
• „Generate Glossary“ button renamed to „Finish and Save Glossary“.
• Glossaries are directly attached to learning object. They are listed together with the learning objects within the repository.
• Attached glossaries can be viewed and deleted from the info screen of the parent learning object.
• Moved chapter on learning object deletion before chapter on glossary generation.
• Search terms languages are selective immediately on the search screen.
• Topics related to learning objects are listed within semantic search results.
• Added lexicon definitions to ontology browser.
• Added learning path reordering on personal desktop section.
# Table of Contents

OVERVIEW ........................................................................................................................................... 2

CHANGE LOG .......................................................................................................................................... 3

TABLE OF CONTENTS .......................................................................................................................... 4

USING LT4EL FUNCTIONALITIES IN ILIAS .......................................................................................... 5

  ADDING A NEW LEARNING OBJECT ................................................................................................. 5
  DELETING A LEARNING OBJECT ........................................................................................................ 6
  GENERATE KEYWORDS FOR A LEARNING OBJECT ........................................................................... 7
  GENERATE A GLOSSARY ....................................................................................................................... 8
  SEARCH .............................................................................................................................................. 10
  USING THE ONTOLOGY BROWSER ..................................................................................................... 11
  MANAGING LEARNING PATH ON PERSONAL DESKTOP ..................................................................... 12
Using LT4eL Functionalities in ILIAS

The changes done in ILIAS within the LT4eL project are related to the handling of learning objects, their metadata and their retrieval. Other major functionalities of ILIAS like course management, communication tools, test and assessment are not affected.

Adding a New Learning Object

The creation process of learning objects has been adopted to enable the use of a repository of pre-annotated learning objects (linguistically and ontologically). This annotation process should be carried out automatically in the final product. However, at the time being the user cannot upload raw learning object documents. ILIAS will present a list of pre-annotated learning objects of the repository that is connected with the system.

After logging into ILIAS click "Repository" in the main menu. The repository contains all learning material and tools organised in categories. If you start with an empty installation, please add at least one category to the repository first. To enter a category, simply click on the name of the category. To add a learning object into the current category, select "Learning Module" in the selection list at the top right, then click on "Add".

A dialog opens, asking for the language and the learning object. To list learning objects of another language, select the language and click "Change Language". After that select the Learning Object you want to add and click "Add". The processing of the learning object may take several seconds (up to a minute, if multiple users work on the system).
After this step, the "Information Screen" of the learning object opens. It displays the processing status of the LT4eL server, which should be "FINISHED" if nothing went wrong. When you create a learning object, you become the owner of the object. The owner has all permissions (view, edit, delete) related to the learning object. You can click „View“ and the original learning object will be shown.

**Deleting a Learning Object**

To delete a learning object, navigate to the corresponding repository category first (e.g. "English Learning Objects"). Now click on "Switch Administration Commands On" at the bottom of the screen. This will add some additional functionalities for users with administrative rights. Click on the "Delete" button of the learning object that should be deleted.
A confirmation screen is displayed. Click on "Confirm" and the learning object will be deleted finally. Please note, that "Delete" buttons will not appear for learning objects, if you do not have the corresponding permission.

Generate Keywords for a Learning Object

After adding a learning object, keywords can be entered in the "Metadata" tab. To open it, simply click on "Metadata". The screen offers a form to enter title, language, description, keywords, authors, copyright and typical learning time information. It also offers a button called "Generate Keywords".
Clicking on "Generate Keywords" lists the first ten keywords delivered by the keyword extractor. The link "Show More" will display the next ten keywords. The link "Show Less" will hide the last added keywords from the list. To select keywords, simply check the corresponding checkboxes. You can add additional keywords in the text area next to the list. The list will show a maximum of 50 keywords.

If you are satisfied with your selected and entered keywords, click "Save" at the bottom of the form. All selected keywords will be put into the text area as a comma separated list. Switch back to the "Information Screen" by clicking "Info". The selected keywords will be displayed on the information screen, too. This screen and the view button is accessible by students.

**Generate a Glossary**

The LT4eL tools support the semi-automatic generation of glossaries. After adding a learning object, a glossary related to the learning object can be generated in the "Properties" tab of the learning object. To open it, simply click on "Properties".
You will find a button called "Generate Glossary". After clicking on this button ILIAS will list candidates for terms and their definition. You can choose which terms/definitions should be included in the glossary by checking the corresponding "Include in Glossary" checkbox. You may also make changes to the text. If you want to add new term/definition pairs, click on "Add Term/Definition" at the bottom of the page.

To generate the glossary, click the "Finish and Save Glossary" button at the bottom of the page. You will return to the initial "Properties" screen where two new buttons will appear, one for viewing the glossary another for deleting it.

Within the repository the glossary will be listed beneath the parent learning object. All glossaries are additionally listed at the bottom of each repository category.
Search

The search function of ILIAS has been extended to enable ontology based learning object retrieval. The extended ILIAS search also offers a keyword based search that uses the keywords as stored by the content provider during the learning object creation and metadata annotation process. Additionally a search for definitions within the generated glossaries and a standard fulltext search using a standard implementation of the Apache Lucene search engine are available.

To search for learning objects and definitions click on "Search" in the main menu. This will open the following form.

After entering the search term, you can select the languages of the learning objects you are interested in under „Retrieval Languages“. If you use semantic search, you also should ensure, that the list of „Search Terms Language(s)“ reflect the languages you used in the search terms input field.

After entering all necessary values, hit the „Search“ button. ILIAS will display the search result as a list of learning objects. Learning objects can be put on the personal desktop of ILIAS for further management of your personal learning path.
Using the Ontology Browser

If a semantic search is invoked and the LT4eL semantic search finds any concepts (called topics in the ILIAS user interface) that are related to the search terms entered, ILIAS will present fragments of the ontology as a result of the search. You may browse through the ontology by clicking on super- or subconcepts. To search for learning objects that are related to the topics, select a number of topics and hit „Search“. ILIAS will present a list of learning objects that are related to these topics.

Hovering the mouse over a topic displays a tooltip with the lexicon entry of the topic. Please note that term definitions of the lexicon are currently only available in English and Bulgarian. Other languages will fall back to the English definitions.
Managing Learning Path on Personal Desktop

Within the repository and on search results screens you can put single listed learning objects on your personal desktop. This function adds a reference to the learning object on your personal desktop and makes it easy to quickly access learning objects you are interested in next time without searching or browsing the repository again.

You can switch to your personal desktop anytime by clicking on „Personal Desktop“ in the main menu. You will get a list of all the learning objects, that you have selected before.

If you want to reconfigure the list, you can click on the „Reorder“ button in the upper right corner of the list.

You can change the positions by entering new numbers for each item. Lower numbers will be listed first. After entering the numbers, hit „Save Order“ to save your changes.

<table>
<thead>
<tr>
<th>Personal Items</th>
<th>Hide Details</th>
<th>Reorder</th>
</tr>
</thead>
<tbody>
<tr>
<td>Latex Tutorial</td>
<td>Edit</td>
<td>Remove from desktop</td>
</tr>
<tr>
<td>Owner: Claudia Borg</td>
<td></td>
<td></td>
</tr>
<tr>
<td>01 Introduction to Word</td>
<td>Edit</td>
<td>Remove from desktop</td>
</tr>
<tr>
<td>Owner: Claudia Borg</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

-12-