



Project no. 027391

Project acronym: LT4eL

Project title: Language Technology for eLearning

Instrument Specific Targeted Research Project

Thematic Priority Information Society Technology

D2.2 Documented keyword extraction tool and integration report (1st cycle)

Due date of deliverable: 30-11-2006

Actual submission date: 21-12-2006

Start date of project: 1-12-2005

Duration: 30 Months

Organisation name of lead contractor for this deliverable: University of Tübingen (UTU)

Revision [1]

Project co-funded by the European Commission within the Sixth Framework Programme (2002-2006)		
Dissemination Level		
PU	Public	x
PP	Restricted to other programme participants (including the Commission Services)	
RE	Restricted to a group specified by the consortium (including the Commission Services)	
CO	Confidential, only for members of the consortium (including the Commission Services)	

D2.2 - Documented keyword extraction tool and integration report

Contents

- 1 Introduction
- 2 Architecture of the tool
- 3 Integration into the Learning Management System
- 4 Evaluation strategy
- 5 Problems and solutions
- 6 References
- 7 Appendices
 - 7.1 A. Use cases
 - 7.1.1 Author annotates learning object with keywords (File resource)
 - 7.1.2 User searches for learning object
 - 7.2 B. User manual
 - 7.2.1 Send mode
 - 7.2.2 Analyse mode
 - 7.3 C. Related deliverables

Introduction

In the *Language Technology for eLearning* (LT4eL) project, we address one of the major problems users of LMSs will be confronted with: how to retrieve learning content from an LMS. In eLearning, we deal with learning objects of varying granularity. A learning object should be accompanied by metadata which describes it, among others, in terms of its content. The "Learning Object Metadata" (LOM, cf. [1]) has become the most widespread and well-known standard for the encoding of the metadata. The aim of the *LT4eL* project is to improve the retrieval and accessibility of content through the identification of the learning material by means of descriptive metadata. To this end, we employ available Language Technology resources to develop functionalities which facilitate the semi-automatic generation of metadata from content. Metadata generation should be considered as part of the life cycle of learning materials. Language Technology can provide significant support for this task. In particular, within LOM, the section *General* allows for the representation of a document's content by topical keywords.

Keywords which describe the topics and contents of a document are essential for the retrieval and accessibility of this document and are therefore a key feature of the metadata. A software assistant will support authors and content managers in selecting appropriate keywords. The keyword extractor will be used for semi-automatic selection of topical keywords for each learning object. Semi-automatic means that the keywords proposed by the program will be reviewed and selected by humans (typically – authors or submitters of the learning objects) before being added to the metadata of the learning object. Storing such information will improve the retrieval and accessibility of content.

The keyword extractor will draw on quantitative and qualitative characteristics of keyword candidates. Keywords are supposed to characterize the topic(s) of a learning object. They therefore tend to appear more often in a document than can be expected if all words would be distributed randomly. This behaviour is modelled by weighting the frequency of a term in the current document, against the share of documents in a collection in which this document

occurs at all (the measure is known as $TF*IDF$).

Keywords tend to cluster in certain documents and will not appear at all in other documents. A statistics which is frequently used the model the distribution of words in texts is Poisson or, alternatively, a mixture of Poissons (cf. [2]). While the distribution of function words is close to the expected distribution under these models, good keyword candidates deviate significantly from it. The score of this deviation can be used as a statistics by which the lexical units are ranked (this measure is known as RIDF - Residual Inverse Document Frequency, cf. [3]). RIDF, however, does not take into account the term frequency in the current document. We therefore adjusted RIDF by term frequency (we call this modified measure ADRIDF). Currently, these three measures are implemented in the keyword extractor.

A further distributional characteristics of keywords is their burstiness. Good keywords tend to appear, within documents, in clusters. Once a keyword appeared in a text, it tends to appear in shorter intervals. After a while, the word might disappear and appear - and burst - again, if the topic is resumed. Reference to a certain topic is a local phenomenon. The concept of term burstiness can be formalized by measuring the gaps between the individual occurrences of each term in the text. Sarkar et al. use a combination of two exponential functions the distribution within bursts and outside bursts. Through an iterative process the variables values in these functions are modified to optimize the fit between observed and expected distribution values. Optimal values for these three variables are derived once the fitting reaches a stable maximum. These values turned out to be a good indicator for the keywordiness of words in texts (cf. [4]). Due to problems with adapting third-party software in needed to measure burstiness, we have delayed the implementation of this measure to the second year of the project.

It still remains to be determined how the different measures should be combined and weighted to get the best results. The output of the various statistics will be compared to manual keyword annotation, performed by experienced annotators. A comparison of high ranking keyword candidates with keywords selected by humans will approximate recall and precision values. The (mix of) statistics which we will use in the final system will be selected on the basis of these evaluation results.

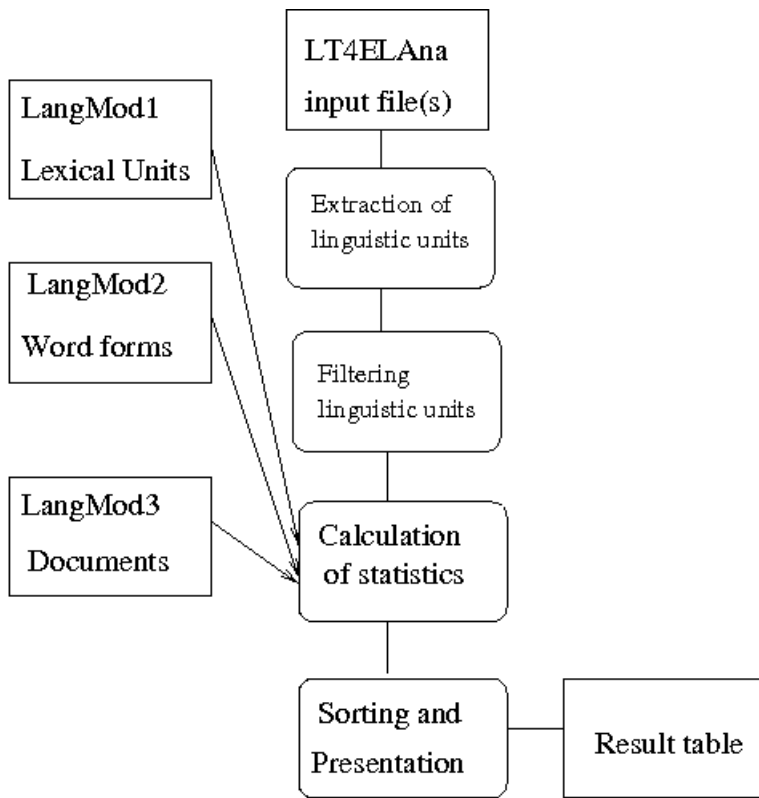
In the context of a Learning Management System, the extractor analyses a set of documents and returns the best keyword candidates for each learning object. These keywords will be presented in a canonical citation form (e.g. the base forms for nouns and verbs). Additionally, the most frequently occurring inflected forms of these lexical units will be presented to the user as attested forms. The user decides on the inclusion of these candidates into the metadata.

It has been shown that the results of the keyword selection, measured against the performance of human readers, improve significantly if the text is annotated linguistically (cf. [5]). Therefore, linguistically annotated learning objects are used as input. Furthermore, using part-of-speech information allows us to eliminate some units (using their part of speech tag) that are unlikely to be good keywords, even though the statistical measures might state so.

The assessment of manually annotated keywords which have been provided within this project revealed that up to 70% of the keywords are multi-word units (this is the case for Polish). Therefore it is essential that a keyword extractor also finds and weights multi-word units. This is one of the functions of the tool.

Architecture of the tool

The software consists of a set of Java classes.



The data flow is the following: The user of the tool submits a file or a set of files. Java methods are provided to handle the files (incl. validation of the XML file format). Note that in the current stage of the project it is assumed that the files which are sent to the tool are linguistically annotated according to the LT4EL annotation format specification.

If the file is in correct format, the information extracted from it is added to the language model for specified language. The language model consists of 3 parts (that can be thought of as database tables):

- **Lexical Units:** those are the basic units for which statistics are calculated; lexical units are also returned as keyword candidates; a lexical unit is defined by a lemma and POS tag (so that *book-Verb* and *book-Noun* are separate units). Lexical units can consist of a single word or of multiple words;
- **Word Forms Types:** those are the actual representations of lexical units in the input files; word form type is defined by an orthographic representation and morphological description (therefore, two syncretic forms, identical orthographically but having different morphological annotation, are separate units). Word Form Types also keep the information about the position of their occurrences (= tokens) in the corpus. Only those word form types that qualify as "potential keyword candidates" are held in the database. The set of potential keyword candidates is constrained, for each language individually, by the part of speech and morpho-syntactic features.
- **Documents:** represent the documents (learning objects) stored in the database; a simple structure used mainly as a foreign key, holding information about names and domains of the learning objects.

In the first step - *Extraction of linguistic units* - all potentially interesting sequences of words are extracted. This is done using the suffix array data structure (cf. [6]) that allows for recognising potential sequences of any length. A indispensable condition for their inclusion, is, however, that they must appear at least twice in the text.

In the second step – *Filtering linguistic units* – the sequences that begin/end with functional lexical units, are trimmed, so that the initial and final words are within a certain set of word classes which are defined separately for each individual language. Then, sequences longer than a specified threshold (specified as a program parameter) are discarded. Optimal value of

the threshold will most probably be different for different languages, and after it is determined experimentally for each project language it can be added to the language specific information in the KWE code.

Statistics are calculated only for sequences that got through the filters. The keyword candidates are sorted by the value of chosen statistic. Only the top part of the list is presented to the user. In the current version, the number of the top candidates presented can be defined using a program parameter; if it is not defined, all candidates will be returned (including the worst ones at the end of the list). In the final version, the threshold needs to be set according to user preferences.

Integration into the Learning Management System

The full picture is given in the integration report. We therefore give a rough outline of the integration of the tool with the ILIAS Learning Management System (and other potential software clients)

- The tool will be placed on the language technology server of the project, provided with a webservice interface and servlet/JSP web interface.
- The Learning Management System (and possible other clients) will communicate with the tool using the webservice interface; test users will use the servlet/JSP interface.
- The tool will receive a document, a language code which specifies the language of the document, and the domain code
- the tool will return a list of proposed keywords for this document (to be reviewed by the author)

Evaluation strategy

The evaluation of the tool will proceed in three steps.

- The first method is an automatic procedure:
 - first, the manually extracted keywords are matched against the list of first k automatically extracted keywords; k is the number of manually extracted keywords.
 - for each file and method, the number of full matches and partial matches is recorded, and final score (1 point per full match, half point per partial match) counted.
 - to see how far from the top of the automatically extracted list other manually extracted keywords are, for each of them the position on the list is checked. This lets us see a difference between a situation when the manually annotated keyword is just after the threshold on the automatic list (an "almost-match") and when it has been assigned a really low value.
- The second method is based on human evaluation of the quality of automatically extracted keywords. Test users will read a text and, after this, will be confronted with those words which have been ranked highest by the keyword extractor for this text. Based on their knowledge of the text, the test persons rate the quality of each word as a keyword, on a scale from 1 to 5. This evaluation helps us to assess the quality of the extracted keywords - which can be acceptable even though they do not match the manually annotated ones, because the annotator had different opinion on what should and what should not be qualified as a keyword in the given document.
- The third method is a step towards a gold standard for keyword evaluation. It is based on the assumption that individuals will widely diverge in their choice of keywords for a certain document, but that there might be a core of keywords on which most annotators of a text would agree. We will therefore let at least 12 persons annotate the same text

with a limited number of keywords. The keywords which are collected from all annotators will be ranked by the frequency with which they have been selected. The keywords which are chosen by 'the majority of annotators' (the exact number is to be determined) will be considered to be a gold standard for that document, against which the performance of the automatic keyword extraction can be evaluated. Secondly, the output of the experiment allows us to measure the average inter-annotator agreement and to check where the agreement between the automatic extractor and any annotator ranks relatively to the average agreement.

The automatic evaluation (step 1) is currently being performed by the language resource providers. The evaluation tool is a part of this deliverable. Pilots for the third evaluation method have recently been launched for German and English. Based on the results of the evaluation, improvements to the extractor will be implemented. One crucial question which will be investigated is for the best statistics and for the best combination of quantitative and qualitative features of the keyword candidates. This issue will be further investigated in the second project year. The evaluation methods described above are suited for a tool-centered evaluation. Additionally, a task-based, user-centered evaluation will be launched once the extractor is integrated into the Learning Management System. This is the task of WP5.

Problems and solutions

The first phase of development and testing of the keyword extractor revealed the following problems:

- For different languages different maximum length of multi-word keywords is optimal. The first version of software extracted only one-word keywords, but that turned out to give poor results for Polish; on the other hand, after allowing longer keywords the results for other languages like Dutch seemed to be worse. The current solution is that the maximum keyword length can be specified by a program parameter, thus allowing to use different values for different languages.
- At the beginning all words were allowed in multi-word keywords. The undesirable result was that sequences like (an artificial example) "of the operating system that" could have been proposed. The solution was to trim the initial and final functional words from the sequences, to get, in this case, "operating system" alone.
- For Slavic languages multi-word lexical units presented to the user as sequences of lemmas of separate words (instead of intuitively understood lemma of the whole sequence) were sometimes unreadable. This has been solved by presenting all attested forms of the lexical units along with the lemma.

References

- [1] *Final Draft, Standard for Learning Object Metadata*, IEEE, 2002 -- P1484.12.1
- [2] K. Church and W. Gale. 1995. *Poisson mixtures*. In: *Natural Language Engineering*. Vol. 1, No 2, pp. 163-190.
- [3] K. Church, W. Kenneth and W. Gale. 1995. *Inverse Document Frequency (IDF): A Measure of Deviations from Poisson*. In: *Proc. of Third Workshop on Very Large Corpora*.
- [4] Avik Sarkar, Paul H. Garthwaite, and Anne De Roeck. 2005. *A Bayesian Mixture Model for Term Re-occurrence and Burstiness*. In: *Proc. of the Ninth Conference on Computational Natural Language Learning (CoNLL-2005)*. Ann Arbor, Michigan. ACL. pp. 48-55.
- [5] Anette Hulth. 2003. *Improved Automatic Keyword Extraction Given More Linguistic Knowledge*. In: *Proc. of the 2003 Conference on Empirical Methods in Natural Language Processing*. pp 216--223.

[6] M. Yamamoto, and K. Church. 2001. *Using Suffix Arrays to Compute Term Frequency and Document Frequency for All Substrings in a Corpus*. In: *Computational Linguistics* 27(2001),1. pp. 1-30.

Appendices

A. Use cases

The term LTools refers to the keyword extractor which is the subject of this report, as well as additional classes and methods which couple the extractor with the web service interface.

Author annotates learning object with keywords (File resource)

Brief Description of Context and Goal: An author wants to provide a new learning object within the learning management system. He uploads an input file (PDF, HTML, DOC, ODT etc.) file as a learning object. ILIAS+LTools will assist him to assign useful keywords to the learning object.

Related LT-Functionality and WP: Semi-automated keyword annotation

Primary Actor: Author

Preconditions: Author is within ILIAS repository and has permission to create file objects.

Postconditions: A new file object has been created, the author has chosen appropriate keywords from a list of candidate keywords, provided by ILIAS+LTools.

Main Success Scenario

1. Author selects File in the new resource selection list and hits Add
2. ILIAS+LTools display a form including input fields for title, language and filename
3. Author enters title and language, selects a local file and hits Upload File
4. ILIAS+LTools display the (LOM) metadata input form, including a list of suggested keywords
5. Author selects some of the suggested keywords, enters some new keywords and hits Save
6. ILIAS+LTools save the metadata

There will also be another simpler interface (independent from ILIAS), doing the same basic task: Keyword Extractor will get a file (already converted to XML and linguistically annotated by other software), analyse it and return keyword candidates.

User searches for learning object

Brief Description of Context and Goal: A user wants to search for a learning object by keywords.

Primary Actor: User

Preconditions: User is logged into ILIAS.

Postconditions: ILIAS+LTools list a set of learning objects that match the keywords entered by the user.

Main Success Scenario

1. User hits Search in the main menu
 2. ILIAS+LTools display a search form
 3. User enters search terms and hits Search
 4. ILIAS+LTools display a list of learning objects that match the search terms
- The ranking of the documents is the following: If the search term is part of the ontology, all documents which are linked to that ontology are ranked highest. If the search term matches a keyword, all document which have this keyword in their meta-data are ranked second highest. If both of the above are not the case, but the search term can be found in the full text index, all the documents which contain the search term are listed. The quality of the result -

ontological mapping, keyword mapping, full text mapping - will be indicated by the system.

B. User manual

The user manual presented below describes the command line interface available now. In the next step, even the test users will use a web-based interface. Command-line interface can be used on any operating system with Java Runtime Environment 1.5.

Send mode

In the Send mode, a file is added to the language model. Required parameters:

- mode (-s)
- file path (-f <file>)
- language (-L <language code>)

Optional parameters:

- maximum length in words of a keyword (-k <length>); the default is 1
- database environment directory (-p <directory>); the default is current directory
- log file name (-l <filename>)

Example: in the directory where the jar file (kwe0.4.0.jar) resides, run

```
java -jar kwe0.4.0.jar -s -f <path to LT4eLAna XML file> -L <language_code>
```

Analyse mode

In the Analyse mode, a file already present in the language model is analysed. Those lexical units that

- appear in the input document at least twice

and

- qualify as keywords according to the statistics based on the whole language model

are written to the output file.

Required parameters:

- mode (-a)
- method (-m <method>); available methods are TFIDF, RIDF, ADRIDF
- document id (-i <id>); id is a consecutive number assigned when document is added to the language model
- language (-L <language code>)

Optional parameters:

- database environment directory (-p <directory>); the default is current directory
- number of keywords to be extracted (-n <number>); the default is all (no limit)
- log file name (-l <filename>)

Example: in the directory where the jar file (kwe0.4.0.jar) resides, run

```
java -jar kwe0.4.0.jar -a -m RIDF -i 0
```

Now you can check the output file and assess the results. The name of the file will be constructed according to the pattern

```
<original document name>_<method>.txt
```

In the web-based version the list of keywords will be presented to the user through a web form.

C. Related deliverables

- Guidelines for the annotation of keywords (on the portal, additional documents section)
- Lists of keywords for each language (on the portal, keywords section)
- Keyword extractor executable (on the portal, tools section)
- Tools for the automatic evaluation of the keyword extractor (on the portal, tools section)