



LarKC

The Large Knowledge Collider

a platform for large scale integrated reasoning and Web-search

FP7 – 215535

D7b.2.1 Semantic annotation for carcinogenesis research

Coordinator: Angus Roberts

With contributions from: Danica Damljanovic, Mark Greenwood, Hamish Cunningham, Mattias Johansson, James McKay, Kurt Straif

Quality Assessor: Yi Zeng

Quality Controller: Angus Roberts

Document Identifier:	LarKC/2008/D7b.2.1/V0.1
Class Deliverable:	LarKC EU-IST-2008-215535
Version:	version 1.0
Date:	October 8, 2009
State:	final
Distribution:	public



EXECUTIVE SUMMARY

Semantic annotation is the extraction of structured information from textual documents. Importantly, the information extracted can be related to knowledge based resources, such as ontologies. This provides connectivity to these resources, and thus helps to define the document semantics.

In D7b1.1a (Requirements summary and data repository for the WP7b Carcinogenesis use case), two scenarios are described for which semantic annotation is required. First, improved literature search is required to assist with carcinogenesis reference production (Monographs). Second, literature knowledge mining is required to assist with gene-disease involvement in Genome Wide Association Studies (GWAS). In both cases, it is expected that the scenarios can be enhanced by provision of a semantically annotated corpus of MEDLINE abstracts, and potentially by annotation of other biomedical literature.

In the same deliverable, Chapter 4 describes *Semantic annotation of the biomedical literature* in detail. It defines the subject area, and the technologies. This deliverable is the practical and experimental outcome of the requirement for semantic annotation. The deliverable first describes the semantic annotation schema that has been created within LarKC, and the data annotated against that schema. It then describes the software used to create this data.

Finally, the deliverable introduces future work, including the creation of a human annotated gold standard. This may be used both for evaluation of existing annotation, and to improve performance by machine learning of statistical annotation models.



DOCUMENT INFORMATION

IST Project Number	FP7 – 215535	Acronym	LarKC
Full Title	The Large Knowledge Collider: a platform for large scale integrated reasoning and Web-search		
Project URL	http://www.larkc.eu/		
Document URL			
EU Project Officer	Stefano Bertolo		

Deliverable	Number	7b.2.1	Title	Semantic annotation for carcinogenesis research
Work Package	Number	7b	Title	Carcinogenesis reference production

Date of Delivery	Contractual	M18	Actual	19-Oct-09
Status	version 1.0		final	<input checked="" type="checkbox"/>
Nature	prototype <input type="checkbox"/> report <input type="checkbox"/> dissemination <input type="checkbox"/>			
Dissemination Level	public <input checked="" type="checkbox"/> consortium <input type="checkbox"/>			

Authors (Partner)	Angus Roberts (University of Sheffield), Danica Damljanovic (University of Sheffield), Mark Greenwood (University of Sheffield), Hamish Cunningham (University of Sheffield), Mattias Johannson (International Agency for Research on Cancer), James McKay (International Agency for Research on Cancer), Kurt Straif (International Agency for Research on Cancer)			
Resp. Author	Angus Roberts		E-mail	a.roberts@dcs.shef.ac.uk
	Partner	University of Sheffield	Phone	+44 (114) 222 1917

Abstract (for dissemination)	Semantic annotation is the extraction of structured information from textual documents. Importantly, the information extracted can be related to knowledge based resources, such as ontologies. This provides connectivity to these resources, and thus helps to define the document semantics. The LarKC carcinogenesis use case has a requirement for the semantic annotation of biomedical literature, in order to (a) improve literature search and (b) mine knowledge for modelling of gene-disease associations. This deliverable is the practical and experimental outcome of this requirement. The deliverable is in the form of semantic annotation software, an annotation schema, and the body of semantic annotation itself.
Keywords	Semantic annotation, Information Extraction, Text Mining, Biomedical Literature, Biomedical Informatics, Bioinformatics, Biomedical Data Sources, Biomedical Knowledge Sources, MEDLINE, Pubmed

Version Log			
Issue Date	Rev No.	Author	Change
15/09/2008	1	Angus Roberts	Created document, front matter
15/07/2009	2	Angus Roberts	Main document structure and outline
24/09/2009	3	Angus Roberts	Draft for review
06/10/2009	4	Angus Roberts	Corrections from review



PROJECT CONSORTIUM INFORMATION
















Participant's name	Partner	Contact
Semantic Technology Institute Innsbruck, Universitaet Innsbruck	 	Prof. Dr. Dieter Fensel Semantic Technology Institute (STI), Universitaet Innsbruck, Innsbruck, Austria Email: dieter.fensel@sti-innsbruck.at
AstraZeneca AB		Bosse Andersson AstraZeneca Lund, Sweden Email: bo.h.andersson@astrazeneca.com
CEFRIEL - SOCIETA CONSORTILE A RESPONSABILITA LIMITATA		Emanuele Della Valle CEFRIEL - SOCIETA CONSORTILE A RESPONSABILITA LIMITATA Milano, Italy Email: emanuele.dellavalle@cefriel.it
CYCORP, RAZISKOVANJE IN EKSPERIMENTALNI RAZVOJ D.O.O.		Michael Witbrock CYCORP, RAZISKOVANJE IN EKSPERIMENTALNI RAZVOJ D.O.O., Ljubljana, Slovenia Email: witbrock@cyc.com
Höchstleistungsrechenzentrum, Universitaet Stuttgart		Georgina Gallizo Höchstleistungsrechenzentrum, Universitaet Stuttgart Stuttgart, Germany Email : gallizo@hlrs.de
MAX-PLANCK GESELLSCHAFT ZUR FOERDERUNG DER WISSENSCHAFTEN E.V.		Dr. Lael Schooler, Max-Planck-Institut für Bildungsforschung Berlin, Germany Email: schooler@mpib-berlin.mpg.de
Ontotext AD		Atanas Kiryakov, Ontotext Lab, Sofia, Bulgaria Email: naso@ontotext.com
SALTLUX INC.		Kono Kim SALTLUX INC Seoul, Korea Email: kono@saltlux.com
SIEMENS AKTIENGESELLSCHAFT		Dr. Volker Tresp SIEMENS AKTIENGESELLSCHAFT Muenchen, Germany Email: volker.tresp@siemens.com
THE UNIVERSITY OF SHEFFIELD		Prof. Dr. Hamish Cunningham, THE UNIVERSITY OF SHEFFIELD Sheffield, UK Email: h.cunningham@dcs.shef.ac.uk
VRIJE UNIVERSITEIT AMSTERDAM		Prof. Dr. Frank van Harmelen, VRIJE UNIVERSITEIT AMSTERDAM Amsterdam, Netherlands Email: Frank.van.Harmelen@cs.vu.nl
THE INTERNATIONAL WIC INSTITUTE, BEIJING UNIVERSITY OF TECHNOLOGY		Prof. Dr. Ning Zhong, THE INTERNATIONAL WIC INSTITUTE Mabeshi, Japan Email: zhong@maebashi-it.ac.jp
INTERNATIONAL AGENCY FOR RESEARCH ON CANCER		Dr. Paul Brennan, INTERNATIONAL AGENCY FOR RESEARCH ON CANCER Lyon, France Email: brennan@iarc.fr
INFORMATION RETRIEVAL FACILITY		Dr. John Tait, Dr. Paul Brennan, INFORMATION RETRIEVAL FACILITY Vienna, Austria Email: john.tait@ir-facility.org



TABLE OF CONTENTS

LIST OF FIGURES	6
LIST OF TABLES	7
ABBREVIATIONS	8
1 INTRODUCTION	9
1.1 Location of the software and data	10
1.2 Overview of this report	10
2 SEMANTIC ANNOTATION	11
2.1 Introduction	11
2.2 Schema	11
2.2.1 High level domain schema	11
2.2.2 Detailed description of annotations	13
2.3 Data sources	15
2.3.1 Corpus	15
2.3.2 Ontologies and vocabularies	15
2.4 Format and delivery	16
2.5 Viewing the data	17
3 SOFTWARE FOR SEMANTIC ANNOTATION	19
3.1 Introduction	19
3.2 Delivery	19
3.3 Pipeline description	20
3.3.1 Termino	20
3.3.2 Pipeline description	21
3.3.3 Non-standard components	21
3.4 Preparing the data sources	23
3.4.1 Term lookup from UMLS	23
3.4.2 MEDLINE preparation	24
3.5 Running the pipeline	24
4 FUTURE PLANS AND CONCLUSIONS	26
4.1 Future plans	26
4.1.1 Future data development and delivery	26
4.1.2 Future software delivery	26
4.1.3 Evaluation	26
4.2 Conclusions	27
REFERENCES	28
A EXAMPLE ANNOTATED DOCUMENT	29



LIST OF FIGURES

2.1	Example view of annotations	18
-----	---------------------------------------	----



LIST OF TABLES

2.1	Original markup	13
2.2	Original markup	14
2.3	Semantic annotation	14
3.1	Annotation pipeline	22



LIST OF ABBREVIATIONS

CLEF	Clinical e-Science Framework
CUI	Concept unique identifier (UMLS)
FSM	Finite State Machine
GATE	General Architecture for Text Engineering
GO	Gene Ontology
GWAS	Genome Wide Association Study
IARC	International Agency for Research on Cancer
IE	Information Extraction
LarKC	The Large Knowledge Collider project
MRC	UK Medical Research Council
PR	Processing Resource (GATE)
TUI	Type unique identifier (UMLS)
UMLS	Unified Medical Language System
WHO	World Health Organisation



1. Introduction

This report accompanies deliverable data and software delivered as **LarKC deliverable D7b.2.1, Semantic annotation for carcinogenesis research**.

The data and software have been produced for the LarKC carcinogenesis use case. This use case has a requirement for semantic annotation. Semantic annotation and the use case requirement are described in detail in **LarKC deliverable D7b.1.1a Requirements summary and data repository** [11], which gives a full bibliography on semantic annotation. We summarise the approach below, paraphrasing from that deliverable.

Semantic Annotation is about assigning to entities and relations in text links to their semantic descriptions in an ontology. This sort of semantic metadata provides both class and instance information about the entities.

Semantic annotation is a specific metadata generation and usage schema aiming to enable new information access methods and to enhance existing ones. The semantic annotation provided by the accompanying deliverable is based on linguistic pre-processing of textual data, and on information extraction (IE) of the entities of interest. Information discovered in the documents by an IE system constitute an important part of their semantics. Moreover, by using text redundancy and external or background knowledge, this information can be connected to formal descriptions, i.e., ontologies, and thus provide semantics and connectivity to other resources using the same ontology — in the case of biomedicine, connectivity to the wealth of biomedical knowledge bases.

Automatic semantic annotation enables many new applications: highlighting, semantic search, categorisation, generation of more advanced metadata, smooth traversal between unstructured text and formal knowledge. The aim of semantic annotation in the LarKC cancer research use case, is to support sophisticated search and knowledge discovery for carcinogenesis researchers:

- **Literature search** Rich search to support production of carcinogen reference works
- **Gene disease association** Prior knowledge discovery to enhance the statistical power of gene disease association studies

These use cases are fully described in **LarKC deliverable D7b.1.1a Requirements summary and data repository**. This report describes the semantic annotation data and software that will support these scenarios.

The accompanying deliverable provides semantic annotation of the complete MEDLINE corpus, approximately 18 million citations from the biomedical literature [3]. The citations are annotated against a collection of important biomedical ontologies and terminologies, taken from UMLS [4]. Annotations include diseases, anatomy, processes, and measurements.



1.1 Location of the software and data

The data referred to in this report is available from the author on request. Software to replicate the data is available from the LarKC version control repository ¹.

1.2 Overview of this report

This report starts with a description of the data part of the deliverable, in Chapter 2. Details are given of the annotation schema, the corpus used, how semantics are applied to annotations by relating them to external ontologies, and the delivery format.

This is followed by Chapter 3, which describes the software used to annotate the data. The components are listed and described, pointers to further documentation are provided, and the different ways in which the software may be run are discussed.

Finally, Chapter 4 discusses future work and concludes the report.

¹<https://larkc.svn.sourceforge.net/svnroot/larkc/branches/wp7b/medline-preprocess/pipeline>



2. Semantic annotation

2.1 Introduction

This chapter describes the semantic annotation data for the cancer use case. The semantic annotation consists of annotations over the whole of MEDLINE [3], against a set of biomedical terminologies and ontologies. All of the terminologies and ontologies used are included within UMLS [8], and so the annotation has been carried out against UMLS, restricted to just those sources.

This chapter starts with a high level description of each of the main annotation types, followed by detail of all annotations. This is followed by descriptions of the UMLS datasets used for annotation, and the MEDLINE corpus. Finally, the delivery format is described.

2.2 Schema

This section describes the annotation schema, first at a high level, and then in greater detail.

2.2.1 High level domain schema

This section describes each of the annotation types in general language. In addition, the way in which semantics are defined is described.

Anatomy

An anatomical location may be: an anatomical structure or location; a whole organ; an organ component; a tissue. Occasionally, a non-normative body structure will be mentioned. These are also considered anatomical locations.

Anatomy excludes pathological tissues and proliferations, cells and sub-cellular components, and body substances.

Disease

Disease is a very overloaded word, and has many different meanings to different people. A disease in these semantic annotations may refer to any of following:

- Diseases
- Syndromes
- Signs
- Symptoms
- Injuries
- Conditions
- Diagnosis
- Pathological tissues e.g. melanomas, tumours, adenocarcinomas



Investigation

The general definition of an investigation used is to observe or study by close examination and systematic inquiry. An investigation may be:

- a laboratory procedure
- a diagnostic procedure
- a preventative procedure
- a laboratory test
- a research activity
- a research technique
- a research study type

Measurement

Measurements include scalar values, discrete values, units, statistics, intervals of the preceding measures, and compounds of the preceding measures.

Process

A molecular or cellular process includes all normal and pathological functions at the cellular and molecular level, i.e. below the level of tissues. It excludes anything covered by the entity type, which describe processes above the cellular level.

Species

Species is the most basic taxonomic unit used in biological classification. A species will be defined as a group of organisms that differ from all other groups of organisms and that are capable of breeding and producing fertile offspring.

Defining semantics

In addition to the semantic annotation types described above, each span of text annotated with one of the above will also be annotated with one or more UMLSTerm semantic annotations. These relate the span to terminologies and vocabularies in the UMLS, via unique identifiers from the UMLS:

- **TUI** Type unique identifier
- **CUI** Concept unique identifier



Annotation	Features	Description
MedlineCitation	pmid	Equivalent to the MEDLINE delivery XML element of the same name.
Article	<i>none</i>	Equivalent to the MEDLINE delivery XML element of the same name.
ArticleTitle	<i>none</i>	Equivalent to the MEDLINE delivery XML element of the same name.
Abstract	<i>none</i>	Equivalent to the MEDLINE delivery XML element of the same name.
AbstractText	<i>none</i>	Equivalent to the MEDLINE delivery XML element of the same name.

Table 2.1: Original markup

2.2.2 Detailed description of annotations

This section describes all of the annotations and their features in detail. Three types of annotations are provided, and each described in a separate table. The three types are:

- **Original markup** MEDLINE citation XML elements, expressed as annotations
- **Linguistic annotation** Basic lexico-syntactic annotation
- **Semantic annotation** The annotation types described above

Original markup

The original markup comprises those elements from the original MEDLINE XML that directly wrap the MEDLINE abstract text and title, and all ancestors of those elements, to the root element. It is provided in order that semantic annotations may be placed back in the context of the original MEDLINE citation. These are shown in Table 2.1.

Linguistic annotation

The linguistic annotations are a by-product of the semantic annotation process. They codify the lexico-syntactics of the text, such as tokens, parts-of-speech, sentences. In addition, annotations for raw term lookup against UMLS are given. These are shown in Table 2.2.

Semantic annotation

The semantic annotations are the final product of the annotation task, and comprise the high level biomedical annotation types described above, together with a UMLS-Term annotation, which relates these high level types to entries in the UMLS source vocabularies. These are shown in Table 2.3.



Annotation	Features	Description
DBInstance	id	A raw term found by the term lookup component.
Number	value	Any numbers
RomanNumeral	value	Roman numerals
Sentence	<i>none</i>	A sentence.
SpaceToken	kind, length	All spaces between tokens.
Split	kind	The split between two sentences
Token	category, affix, kind, length, orth, root, string	Generally a word.
terminoTerm	type	A term typed against UMLS into one of a small number of coarse categories.
terminoExtraTerm	majorType, minorType	A term found in an additional lookup list of stop words.

Table 2.2: Linguistic annotation

Annotation	Features	Description
UMLSTerm	type, CUIs, TUIs, description	The basic semantic annotation, against UMLS type systems. A term associated with type information from UMLS.
Anatomy	Type	A coarsely-typed semantic annotation. Type takes a value of tissue , organ , structure , organ component , system
Disease	<i>none</i>	A coarsely-typed semantic annotation.
Investigation	<i>none</i>	A coarsely-typed semantic annotation.
Measurement	unit, value, type	A coarsely-typed semantic annotation.
Process	<i>none</i>	A coarsely-typed semantic annotation.
Species	<i>none</i>	A coarsely-typed semantic annotation.

Table 2.3: Semantic annotation



2.3 Data sources

This section gives a high level description of the annotated corpus, and of the terminologies and ontologies against which the corpus has been semantically annotated.

2.3.1 Corpus

The corpus comprises all citations from the 2009 baseline MEDLINE, approximately 18 million in total [3]. MEDLINE is distributed as approximately 600 `.xml.gz` files. Each is a single xml file gzipped, containing 20 000 – 40 000 citations. Each citation is within a `MedlineCitation` element. Each citation comprises a title, an abstract, and a large amount of meta data. In a significant number of cases, only a title is provided, and no abstract.

2.3.2 Ontologies and vocabularies

The title and abstract was annotated for all annotations, or title only where no abstract is provided. Annotations have been added for terms from the following ontologies and vocabularies:

- Foundational Model of Anatomy 2.0
- Gene Ontology 2008_04_01
- ICD10 1998
- NCBI Taxonomy 2008_05_07
- NCI Thesaurus 2008_05D
- SNOMED Clinical Terms 2008_07_31
- University of Washington Digital Anatomist 1.7.3

In all cases, these were taken from UMLS version 2008 AB [4]. Full details of each of the source terminologies can be found in the UMLS documentation.

Individual terms are typed according to the UMLS semantic type assigned to concepts related to that term. Each semantic annotation type is defined by a set of UMLS semantic types, as described in the sections below. In these sections, TUI and STNRTN refer to IDs in the UMLS semantic type system.

Anatomical loci

- Body system (UMLS TUI T022)
- Body Location or Region (UMLS TUI T029)
- Body Space or Junction (UMLS TUI T030)
- Anatomical Abnormality (UMLS STNRTN A1.2.2%)
- Body Part, Organ or Organ Component (UMLS TUI T023)
- Tissue (UMLS TUI T024)



Diseases

- Pathological Function (UMLS STNRTN B2.2.1.2)
- Neoplastic Process (UMLS TUI T191)
- Injury or Poisoning (UMLS TUI T037)
- Finding (UMLS STNRTN A2.2%) excluding Laboratory or Test Result (UMLS TUI T034)

Processes

- Cell Function (UMLS TUI T043)
- Molecular Function (UMLS STNRTN B2.2.1.1.4%)
- Cell or Molecular Dysfunction (UMLS TUI T049)

Species

- Organism (UMLS STNRTN A1.1%)

2.4 Format and delivery

Annotated MEDLINE abstracts are delivered as a number of tar files, one for each of the MEDLINE distribution `.xml.gz` files, and with the same filename prefix as these files. Each tar file contains around 40 000 individual files, one for each MedlineCitation in the MEDLINE distribution `.xml.gz` file.

The individual files are gzipped GATE document xml, [1] with an extension `.gate.xml.gz`. The individual files contain a representation of the complete text and markup of the ArticleTitle and AbstractText elements of the original MedlineCitation, together with a representation of the linguistic and semantic annotations added by processing. The representation conforms to the following schema:

```
<!ELEMENT GateDocument (GateDocumentFeatures,  
                          TextWithNodes, (AnnotationSet+))>  
<!ELEMENT GateDocumentFeatures (Feature+)>  
<!ELEMENT Feature (Name, Value)>  
<!ELEMENT Name (\#PCDATA)>  
<!ELEMENT Value (\#PCDATA)>  
<!ELEMENT TextWithNodes (\#PCDATA | Node)*>  
<!ELEMENT AnnotationSet (Annotation*)>  
<!ATTLIST AnnotationSet Name CDATA \#IMPLIED>  
<!ELEMENT Annotation (Feature*)>  
<!ATTLIST Annotation Type CDATA \#REQUIRED  
                      StartNode CDATA \#REQUIRED  
                      EndNode CDATA \#REQUIRED>  
<!ELEMENT Node EMPTY>  
<!ATTLIST Node id CDATA \#REQUIRED>
```



Three AnnotationSet elements are given in each document, corresponding to the three annotation types described above:

- **Original markup AnnotationSet** contains annotations corresponding to MEDLINE markup;
- **Nameless, default AnnotationSet** contains annotations from linguistic preprocessing;
- **Filtered AnnotationSet** contains the semantic annotation.

Fragments of an example document are given in Appendix A.

2.5 Viewing the data

The data can be viewed in several ways. Until the data is provided in a public interface, one of the easiest is in the GATE Developer interface. View as follows:

1. Untar a data file (e.g. `tar xvf medline001.tar`)
2. Gunzip the individual files (e.g. `gunzip medline001/*.gz`)
3. Download, install and run GATE, following the instructions at <http://gate.ac.uk>
4. In GATE, create a new corpus with these menus: **File / New language resource / GATE corpus / OK**
5. Right click the new corpus in the left hand pane, and choose **Populate**
6. Fill in the following options:
 - Directory URL: navigate to the directory containing the unzipped files (`medline001` above)
 - Extensions: add `xml` to the list
 - Encoding: `UTF-8`
 - Mime type: `blank`
 - Recurse: leave as it is

This will open all documents. Double click one to open it, and navigate around the annotation sets and annotations using the buttons above the document, and the right hand pane.

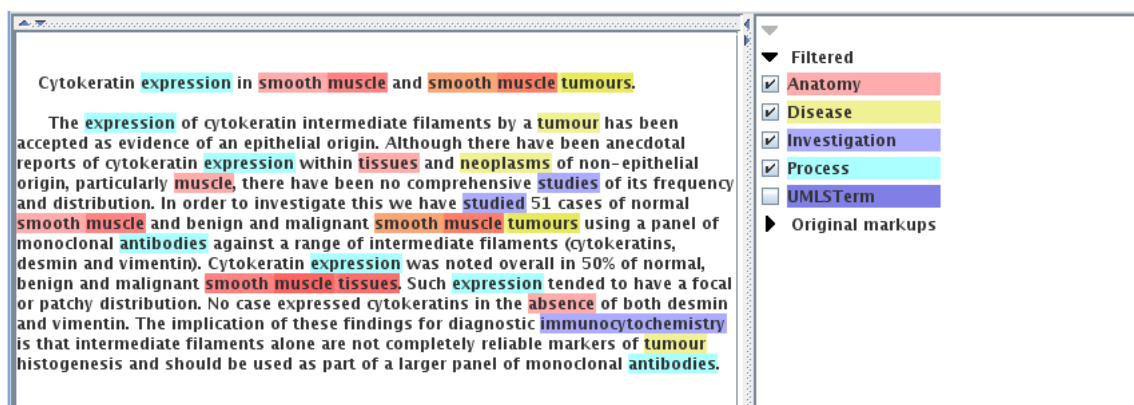


Figure 2.1: A view of annotations in a MEDLINE abstract. Note that the interface shown is for indication only, and the delivered annotations can be viewed in any viewer capable of understanding the delivery format. Viewing is not restricted to the tool shown here, which is a portion of the GATE interface.



3. Software for semantic annotation

3.1 Introduction

This chapter describes software for creating the semantic annotation described in the previous chapter. The software is provided as a GATE application pipeline. GATE is a widely used toolkit and architecture for text engineering. The application pipeline makes use of standard GATE components, together with term lookup and semantic annotation components that have been adapted for LarKC from previous projects. Specifically, the *Termino* term annotation software [7] developed for the UK MRC CLEF project [9] has been adapted to the LarKC cancer use case.

The first section of this Chapter describes the delivery of the software, pointing to the source code and configuration files, and to the GATE documentation needed to for a deeper understanding. This is followed by a component-by-component description of the pipeline. A brief description is then given of loading the Termino term annotator with the required UMLS data sources, and with preparing MEDLINE for annotation. The final section describes how the pipeline can be run.

Note on use of the word *pipeline*. The LarKC platform is considered to run workflows, and not linear pipelines. The semantic annotation software described here is one part of such a workflow. The software itself, however, is considered internally to be a linear, serial pipeline - this is the word used throughout the documentation of the GATE software used to implement semantic annotation. When we use the work pipeline in this document, therefore, we are referring to the internal structure of a single workflow step in LarKC, not to an entire LarKC application.

3.2 Delivery

The annotation software is delivered as a GATE application pipeline. GATE is fully documented at [1]. The pipeline can be found in the LarKC version control repository ¹. The software as delivered contains all of the configuration files and software required in addition to the GATE software itself. The delivered software configures GATE, either as an embedded application (e.g. within LarKC), or with the GATE Developer, an IDE for developing language engineering applications. The delivered also software provides components not found within the standard GATE distribution.

In addition, a small number of components are provided that are not used in the semantic annotation software, but which may be found useful. These include alternative components and components for extra levels of annotation. This is described below.

The delivered software consists of the following directory structure:

- **medline.xgapp** GATE configuration file
- **plugins** directory of GATE plugins

¹<https://larkc.svn.sourceforge.net/svnroot/larkc/wp7b/medline-preprocess/pipeline>



- **ANNIE** Standard Named Entity recogniser, including basic lexico-syntactic annotation components.
- **Chemistry_Tagger** Chemical compound and formulae. Not used - provide as additional annotation software if required.
- **NEFilter** Grammar for filtering stop words and terms
- **Numbers** Number grammars
- **TermMatcher** Termino term matchers
- **TerminoAnnotators** Semantic annotation of terms against UMLS
- **Tools** More basic lexico-syntactic tools and ancillary tools
- **measurements** Measurement grammars
- **termino-utils** Shared libraries for TermMatcher and TerminoAnnotators
- **resources** Configuration data for some of above components
 - **gazetteer** A species lookup list (not used - the application uses Termino for species annotation)
 - **jape** Number grammars and other grammars.

3.3 Pipeline description

NOTE *In order to replicate this section, a user will require a good knowledge of GATE [1]. The application uses non-standard GATE components.*

This section describes the GATE [1] pipeline for semantic annotation in the carcinogenesis use case. The pipeline configuration file and all components can be found in the LarKC version control repository ².

The software can be seen as a pipeline, down which each document is passed. (Of course, this is parallelised in reality). Each component adds annotation to the document. The early components add basic linguistic annotations to the document. Later components consume these early annotations, themselves producing annotations describing richer levels of detail.

The pipeline consists of a sequence of *processing resources*, or PR. These are listed in Table 3.1, in the order they are applied. For each PR, notes are given on annotations created and the effect of the component. In the PR type column, a link is given to further information about this component.

Most of the PRs used are standard GATE PRs, and are part of the GATE distribution. These are documented in the GATE user guide [6]. Links are given to the appropriate sections in the manual. Those PRs that are not standard GATE PRs, are described by further notes in the sections below.

3.3.1 Termino

Several components make use of a large-scale term recogniser developed in previous projects, Termino [7]. Termino consists of:

²<https://larkc.svn.sourceforge.net/svnroot/larkc/branches/wp7b/medline-preprocess/pipeline>



- A database schema for terms and their relationship to their source (such as a source vocabulary or ontology)
- A compile with which to compile out and serialise a FSM matcher.
- A GATE PR with which to run the FSM over a document, annotating terms found in the Termino database.
- A GATE PR with which to annotate terms with identifiers from source ontologies and vocabularies (i.e. to add semantics to the annotations)

The loading of Termino with the relevant portions of UMLS for the semantic annotation for the carcinogenesis use case, is describe in Section 3.4.1.

3.3.2 Pipeline description

3.3.3 Non-standard components

This section gives brief descriptions of those components that are specific to the application. The above pipelines also use many standard GATE components. These will not be described here, see the GATE manual for details.

Term matchers and annotators

These components can be found in the LarKC version control repository ³. They are used in a number of pipeline steps as follows:

1. A number of Term Matchers are used to match terms.
 - Each of these term matchers is compiled from a different part of the termino database (or a different termino database). There are three matchers (one from UMLS, one from an ad hoc list, and one that looks for invalid terms that are candidates for deletion).
 - These matchers add DBInstance annotations with id features that hold termino IDs.
2. Termino Mapping Annotator
 - This retrieves additional info from the termino database
 - This is run with the config file TerminoAnnotators/resources/termino/umls.xml
 - It adds UMLSTerm annotations with cui and lui features.
3. UMLS TUI Adder
 - This retrieves additional info from the UMLS database
 - It is run without a config file
 - It adds a feature tui, with a list of TUIs, to each UMLSTerm annotation

³They are found in the plugins/TermMatcher and plugins/TerminoAnnotators sub-directories in <https://larkc.svn.sourceforge.net/svnroot/larkc/branches/wp7b/medline-preprocess/pipeline>



Name	PR type	Notes
Reset all annotations	Document reset [1]	Removes all annotations, so we can run this application repeatedly on the same documents
Tokenise	ANNIE english tokeniser [1]	Standard GATE tokeniser.
Sentence split	Regex sentence splitter [1]	Standard GATE regular expression sentence splitter.
POS Tag	ANNIE pos tagger [1]	Standard GATE pos tagger
Morphology	GATE morphological analyser [1]	Standard morphological analyser.
Roman Numerals	RomanNumeralsFinder	Specialised roman numeral matcher
Numbers in Words	NumbersInWordsFinder	Specialised number word matcher
Numbers	JAPE grammar	JAPE grammar
Measurement Tagger	Measurements Tagger	JAPE grammar
UMLS term matcher	Term matcher (see Section 3.3.3)	A compiled FSM, built from UMLS terms. Creates DBInstance annotations with termino id features.
Introspection filter matcher	Term matcher (see Section 3.3.3)	A compiled FSM, built from stop words.
Map DBInstance to UMLS terms	Termino mapping annotator (see Section 3.3.3)	For each !DBInstance with UMLS information in termino, adds UMLSTerm annotations with CUI and LUI
Map DBInstance to terminoExtraTerms	Lists annotator (see Section 3.3.3)	Annotates stop words
Add TUIs to UMLSTerms	UMLS TUI adder (see Section 3.3.3)	For each UMLSTerm annotation, looks up the TUI in the umls database, and adds it as a feature
Add termino types and annotations	UMLS tags adder (see Section 3.3.3)	Creates final terminoTYPE annotations.
Filter out invalid terms	JAPE grammar: see Section	Removes all terminoTerm annotations and terminoTYPE annotations (eg terminoCondition) created above, where they are coextensive with filter terms.

Table 3.1: Annotation pipeline



4. UMLS Tags Adder

- This adds entity type information
- This is taken from the tags table in the UMLS database. This maps TUIs to entity type
- Each UMLSTerm annotation has a feature `terminoTypes` added, with its value a list of entity types (such as Disease)
- In addition, for each entity type on a UMLSTerm, an annotation with the name of that type, and no features, is added. e.g. a Disease annotation might be added. In addition, a `terminoTerm` annotation is added with an appropriately set type feature.

Invalid terms filter

These components can be found in the LarKC version control repository ⁴. This contains a single grammar file, `InvalidTermsFilter.jape`, which removes any `terminoTerm` coextensive with a `terminoExtraTerm` with `minorType` of *invalid*. Essentially, this is a stop term list, built from introspection on development documents.

3.4 Preparing the data sources

3.4.1 Term lookup from UMLS

Documents are annotated against vocabularies from UMLS, using the term matcher, `Termino` [7]. `Termino` and its use in the pipeline are described in Section . `Termino` was originally developed for other projects, and has been adapted for the LarKC use case.

Full details of loading `Termino` from UMLS, and of creating term matchers from a `Termino` database, are provided with the `Termino` loading software. The software can be found in the LarKC version control repository ⁵. This section outlines the steps, for reference and background.

Create UMLS subset The tool `metamorphosys` (provided with UMLS) is used to build subsets of UMLS. Selective subsets can be created, based on various criteria, such as language and licensing restrictions. For LarKC, a subset based on a small number of required terminologies and ontologies was built.

Load UMLS Metathesaurus subset into mysql The `metamorphosys` subsetting tool creates `mysql` load scripts. These are used to load UMLS into a MySQL database.

Load UMLS Semantic Net into mysql Again, The `metamorphosys` subsetting tool creates semantic net load scripts.

⁴They are found in the `plugins/NEFilter` sub-directory in <https://larkc.svn.sourceforge.net/svnroot/larkc/branches/wp7b/medline-preprocess/pipeline>

⁵<https://larkc.svn.sourceforge.net/svnroot/larkc/branches/wp7b/medline-preprocess/termino-loading>



Make terminoTYPE tables and tags table This adds a table for each entity type into the UMLS database. The tag table maps the terminoTYPE table names to the entity types that will be used in GATE annotations.

Filter terminoTYPE tables to create termoid lists This takes the terminoTYPE tables and filters them through a set of rules to create lists of "termoids": the things that live in termino. Filtering removes terms that can lead to poor performance, or that are non-linguistic in origin.

Create a termino database Create the base Termino tables.

Load termoids into Termino This takes the termoid list files created in the step before last, and places them in the termino database.

Compile a Termino matcher This uses the GATE TermMatcher Plugin provided in the source code, to create a finite state matcher from the database.

Loading with filter terms In addition to matchers built from UMLS terms, termino can use many other matchers. A typical one is a matcher for terms to exclude, or filter. e.g. stop words. The termoid list for such a filter is provided with the source code.

3.4.2 MEDLINE preparation

NOTE *In order to replicate this section, a user will require a knowledge of MEDLINE and its delivery format.*

The pipeline was run over all MEDLINE citations [3]. MEDLINE is distributed as a set of gzipped xml files. Each file contains up to 40 000 individual citations, comprising titles, abstracts and meta data. For annotation, files were unzipped, and the text and title of each citation stripped out into individual xml files, one for each citation. The script for doing this can be found in the LarKC version control repository⁶. The script can be run with the Groovy interpreter, available from [2].

3.5 Running the pipeline

This section gives high level details of how the semantic annotation software can be run, and refers to other documentation for the detail. The software can be run in three ways:

- Via the LarKC semantic annotation plugin [5]
- In the GATE Developer GUI [1]
- From other software, via the GATE embedded library [1]

⁶<https://larkc.svn.sourceforge.net/svnroot/larkc/branches/wp7b/medline-preprocess/ParseMedline.groovy>



At the time of annotation, the LarKC platform did not allow for multi-threading of the semantic annotation plugin. The data described in this report was therefore created using the third option, using a simple piece of code to multi-thread annotation of the documents via the GATE embedded library.



4. Future plans and conclusions

4.1 Future plans

The semantic annotation deliverable accompanying this report is a complete annotation of the major literature resource in biomedicine, and is usable as is. As the both the use cases and LarKC platform develop, however, other requirements and technical solutions will present themselves for the semantic annotation task. This section looks at likely directions

4.1.1 Future data development and delivery

As end users start to make use of semantic annotation in LarKC use case prototypes, it is likely that they will develop further requirements that were not apparent in the original requirements gathering exercise. In addition, it is likely that on using the current semantic annotation, end users will point out flaws and problems. The semantic annotation software developed is flexible enough to be adapted to new use cases, and to be improved over existing cases, in two respects:

- **Addition of new terminologies.** The term recogniser is not static, and can readily be adapted to new terminologies, for recognition of new semantic types.
- **Adaptable annotation grammars** The annotation technology used supports modular and customisable grammar writing, and so can be adapted to new annotation tasks.

4.1.2 Future software delivery

As the LarKC platform develops, it will be possible to use new developments to assist with semantic annotation. Two cases are obvious:

- **LarKC data layer** Term recognition and semantic addition are currently provided by the Termino recogniser, described in previous Chapters. Once the LarKC data layer can be used for this, the annotation software should use term recognition against the data layer.
- **Parallelisation** The semantic annotation software delivered, is a configuration of GATE. This can be run in a variety of ways. A semantic annotation plugin has been written for the LarKC platform [5]. However, at the time of producing the data associated with this deliverable, the LarKC platform could not parallelise plugins. Annotation was therefore performed outside of the platform. When parallelisation of plugins is available on the platform, annotation should be carried out with the plugin.

4.1.3 Evaluation

In addition to semantically annotating text for the use cases, we should provide some means of evaluating that annotation. The general approach taken is to manually annotate a small portion, and to compare the manual annotation with the machine



annotation, to generate measures of precision and recall. Such an approach on a similar task to that within LarKC is described in [10].

Such a gold standard corpus may also be used to train supervised machine learning methods of semantic annotation **roberts08a**, **roberts08b**.

In brief, the process of creating the gold standard involves:

1. Write a set of annotation guidelines to describe the annotation required.
2. Iteratively test these guidelines against a small corpus, and rewrite them as appropriate.
3. Train human annotators.
4. Annotate a representative sample of the whole corpus.

We have begun to develop such a gold standard, having completed step 1 and part of step 2, and have carried out a pilot annotation of a small number of MEDLINE citations.

4.2 Conclusions

This report accompanies and describes software for semantic annotation of biomedical literature, and semantic annotation data produced by running this software over all MEDLINE citations. The software and data have been produced for the LarKC carcinogenesis research use case.

The software provided is a configuration of the popular GATE text engineering platform, extending it with term recognisers and annotators adapted for the LarKC carcinogenesis use case. The software annotates text against a number of significant biomedical ontologies, drawn from the UMLS. Semantics are provided via linkage back to the UMLS.

The data provided includes annotations not commonly found in similar annotated MEDLINE corpora, such as annotations for diseases, anatomy, processes, and measurements.

Finally, we have introduced ongoing work on the evaluation of the annotation, by provision of a human annotated gold standard.



REFERENCES

- [1] Gate — general architecture for text engineering, 2009. <http://gate.ac.uk>.
- [2] Groovy, 2009. <http://groovy.codehaus.org/>.
- [3] Medline/pubmed baseline repository, 2009. <http://mbr.nlm.nih.gov/>.
- [4] Unified medical language system (umls), 2009. <http://www.nlm.nih.gov/research/umls/>.
- [5] H. Cunningham, Y. Li, M. Greenwood, A. Kiryakov, V. Momtchev, I. Peikov, A. Roberts, N. Aswani, and D. Damljanovic. D2.2.1, 2.5.1 month 12 selection components. Technical report, LarKC project deliverable, 2009.
- [6] H. Cunningham, D. Maynard, K. Bontcheva, V. Tablan, and C. Ursu. *The GATE User Guide*. <http://gate.ac.uk/>, 2002.
- [7] Henk Harkema, Robert Gaizauskas, Mark Hepple, Angus Roberts, Ian Roberts, Neil Davis, and Yikun Guo. A Large Scale Terminology Resource for Biomedical Text Processing. In Lynette Hirschman and James Pustejovsky, editors, *HLT-NAACL 2004 Workshop: BioLINK 2004, Linking Biological Literature, Ontologies and Databases*, pages 53–60, Boston, MA, USA, May 2004. ACL.
- [8] D. Lindberg, B. Humphreys, and A. McCray. The Unified Medical Language System. *Methods Inf Med*, 32(4):281–291, 1993.
- [9] A Rector, J Rogers, A Taweel, D Ingram, D Kalra, J Milan, P Singleton, R Gaizauskas, M Hepple, D Scott, and R Power. CLEF — joining up health-care with clinical and post-genomic research. In *Proceedings of UK e-Science All Hands Meeting 2003*, pages 264–267, Nottingham, UK, 2003.
- [10] A. Roberts, R. Gaizauskas, M. Hepple, G. Demetriou, Y. Guo, A. Setzer, and I. Roberts. Building a semantically annotated corpus of clinical texts. *J Biomed Inform*, 2009. Epub ahead of print.
- [11] Angus Roberts, Kurt Straif, James McKay, Martin Stetter, and Hamish Cunningham. D7b.1.1a requirements summary and data repository. Technical report, LarKC project deliverable, 2009.



A. Example annotated document

Below is an example annotated MEDLINE abstract, in raw xml form. Some parts of the document have been removed for clarity:

- The document starts with a set of document-level features, given in their entirety.
- This is followed by the document content. In this case, it is very short: “PAPYRUS EBERS”: the title of a 1500 BC Egyptian papyrus indexed by MEDLINE. The content also includes spaces and line returns. Node elements are inserted into the content. This demarcates it, with nodes being referenced by annotation elements later in the document. Some spaces and returns, and their associated nodes, have been removed for clarity.
- This is followed by the default annotation set. This is large, and so only a couple of annotations are given.
- Next comes the un-named default annotation set. This contains basic linguistic annotations e.g. for tokens and spaces. It has been truncated for clarity.
- Next, the **Filtered** annotation set is given, containing the final annotations. In this case, we have a **Species** annotation for “PAPYRUS”, and a **UMLSTerm** annotation tying it back to the UMLS.
- This is followed by the **Original markup** annotation set, containing the markup in the original MEDLINE XML document.

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

<!-- The document's features-->

<GateDocumentFeatures>
<Feature>
  <Name className="java.lang.String">gate.SourceURL</Name>
  <Value className="java.lang.String">
    file:/data/medline/baseline/split/medline09n0001/16015741.xml
  </Value>
</Feature>
<Feature>
  <Name className="java.lang.String">
    gate.sam.a3.PooledAlexandriaProcessor.fileSize
  </Name>
  <Value className="java.lang.Long">192</Value>
</Feature>
<Feature>
  <Name className="java.lang.String">MimeType</Name>
  <Value className="java.lang.String">text/xml</Value>
</Feature>
<Feature>
```



```
<Name className="java.lang.String">
  gate.sam.a3.PooledAlexandriaProcessor.processingTime
</Name>
<Value className="java.lang.Long">37</Value>
</Feature>
</GateDocumentFeatures>

<!-- The document content area with serialized nodes -->

<TextWithNodes><Node id="0"/>

[... other annotations removed for brevity and clarity...]

<Node id="8"/>PAPYRUS<Node id="15"/> <Node id="16"/>EBERS<Node id="21"/>

[... other annotations removed for brevity and clarity...]

<Node id="23"/> <Node id="24"/> <Node id="25"/> <Node id="26"/>

[... other annotations removed for brevity and clarity...]

<Node id="43"/></TextWithNodes>

<!-- The default annotation set -->

<AnnotationSet>

<Annotation Id="6" Type="SpaceToken" StartNode="1" EndNode="2">
<Feature>
  <Name className="java.lang.String">length</Name>
  <Value className="java.lang.String">1</Value>
</Feature>
<Feature>
  <Name className="java.lang.String">kind</Name>
  <Value className="java.lang.String">space</Value>
</Feature>
<Feature>
  <Name className="java.lang.String">string</Name>
  <Value className="java.lang.String"> </Value>
</Feature>
</Annotation>

[... other annotations removed for brevity and clarity...]

<Annotation Id="13" Type="Token" StartNode="8" EndNode="15">
<Feature>
  <Name className="java.lang.String">length</Name>
```



```
<Value className="java.lang.String">7</Value>
</Feature>
<Feature>
  <Name className="java.lang.String">category</Name>
  <Value className="java.lang.String">NNP</Value>
</Feature>
<Feature>
  <Name className="java.lang.String">orth</Name>
  <Value className="java.lang.String">allCaps</Value>
</Feature>
<Feature>
  <Name className="java.lang.String">root</Name>
  <Value className="java.lang.String">papyrus</Value>
</Feature>
<Feature>
  <Name className="java.lang.String">kind</Name>
  <Value className="java.lang.String">word</Value>
</Feature>
<Feature>
  <Name className="java.lang.String">string</Name>
  <Value className="java.lang.String">PAPYRUS</Value>
</Feature>
</Annotation>
</AnnotationSet>

<!-- Named annotation set -->

<AnnotationSet Name="Filtered">
<Annotation Id="54" Type="UMLSTerm" StartNode="8" EndNode="15">
<Feature>
  <Name className="java.lang.String">cui</Name>
  <Value className="java.lang.String">C1085733</Value>
</Feature>
<Feature>
  <Name className="java.lang.String">tuiDesc</Name>
  <Value className="java.util.ArrayList"
    itemClassName="java.lang.String">Plant</Value>
</Feature>
<Feature>
  <Name className="java.lang.String">clefTypes</Name>
  <Value className="java.util.HashSet"
    itemClassName="java.lang.String">Species</Value>
</Feature>
<Feature>
  <Name className="java.lang.String">lui</Name>
  <Value className="java.lang.String">L6605108</Value>
</Feature>
```



```
<Feature>
  <Name className="java.lang.String">tuis</Name>
  <Value className="java.util.ArrayList"
    itemClassName="java.lang.String">T002</Value>
</Feature>
</Annotation>
<Annotation Id="53" Type="Species" StartNode="8" EndNode="15">
</Annotation>
</AnnotationSet>

<!-- Named annotation set -->

<AnnotationSet Name="Original markups">
<Annotation Id="0" Type="MedlineCitation" StartNode="0" EndNode="43">
<Feature>
  <Name className="java.lang.String">pmid</Name>
  <Value className="java.lang.String">16015741</Value>
</Feature>
</Annotation>
<Annotation Id="1" Type="Article" StartNode="3" EndNode="42">
</Annotation>
<Annotation Id="2" Type="ArticleTitle" StartNode="8" EndNode="22">
</Annotation>
<Annotation Id="3" Type="Abstract" StartNode="27" EndNode="39">
</Annotation>
<Annotation Id="4" Type="AbstractText" StartNode="34" EndNode="34">
<Feature>
  <Name className="java.lang.String">isEmptyAndSpan</Name>
  <Value className="java.lang.String">>true</Value>
</Feature>
</Annotation>
</AnnotationSet>

</GateDocument>
```