SEVENTH FRAMEWORK PROGRAMME
THEME 3
Information and communication Technologies

PANACEA Project

Grant Agreement no.: 248064

Platform for **A**utomatic, **N**ormalized **A**nnotation and **C**ost-**E**ffective **A**cquisition
of Language Resources for Human Language Technologies

D4.1
Technologies and tools for corpus creation, normalization and annotation

Dissemination Level:	Public
Delivery Date:	July 16, 2010
Status – Version:	Final
Author(s) and Affiliation:	Prokopis Prokopidis, Vassilis Papavassiliou (ILSP), Pavel Pecina (DCU), Laura Rimel, Thierry Poibeau (UCAM), Roberto Bartolini, Tommaso Caselli, Francesca Frontini (ILC-CNR), Vera Aleksic, Gregor Thurmair (Linguatec), Marc Poch Riera, Núria Bel (UPF), Olivier Hamon (ELDA)

This document is part of technical documentation generated in the PANACEA Project, **P**latform for **A**utomatic, **N**ormalized **A**nnotation and **C**ost-**E**ffective **A**cquisition (Grant Agreement no. 248064).

Please send feedback and questions on this document to: iulatrl@upf.edu

TRL Group (Tecnologies dels Recursos Lingüístics), Institut Universitari de Lingüística Aplicada, Universitat Pompeu Fabra (IULA-UPF)

Table of contents

1 Introduction

The objectives of the Corpus Acquisition and Annotation (CAA) subsystem are the acquisition and processing of monolingual and bilingual language resources (LRs) required in the PANACEA context. Therefore, the CAA subsystem includes: i) a Corpus Acquisition Component (CAC) for extracting monolingual and bilingual data from the web, ii) a component for cleanup and normalization (CNC) of these data and iii) a text processing component (TPC) which consists of NLP tools including modules for sentence splitting, POS tagging, lemmatization, parsing and named entity recognition.

We present the terminology used in this document in Section 2. We discuss state-of-the-art and existing tools for corpus acquisition, corpus normalization, and text processing in Sections 3, 4 and 5 respectively. The resources to be produced in the context of WP4 are discussed in Section 6. In Section 7 we present the solution path we aim to explore for generating these resources.

2 Terminology

Corpus (or text corpus): a (large) set of texts. In PANACEA, we assume the texts are stored electronically, in a given file format and character encoding, without any formatting information, eventually provided with metadata and/or linguistic annotation. Often, texts are referred to as documents, in which case the texts are assumed to be topic-coherent.

Monolingual corpus: a corpus of texts in one language.

Bilingual corpus: a corpus of texts in two languages.

Parallel corpus: a bilingual corpus consisting of texts organized in pairs which are translations of each other, i.e. they include the same information (parallel texts). Usually, the pairs are identified at least for documents (parallel documents) and the corpus described as document-aligned parallel corpus. If the translation pairs are identified also for sentences (parallel sentences) we talk about sentence-aligned parallel corpus.

Comparable corpus: a bilingual corpus consisting of texts organized in pairs (comparable documents) which are only approximate translations of each other, i.e. they include similar information.

Web Crawler: a computer program that browses the World Wide Web in a methodical and automated manner in order to copy/store web documents (html pages, pdf documents, etc.) for later processing (e.g. indexing, creating corpora, etc.)

Focused web crawler: is a web crawler that downloads html pages that are relevant to a predefined topic in order to build topic-specific web collections.

Seed pages: Web pages known to be relevant to a specific domain. The crawler will be initialized with these pages.

3 Corpus Acquisition Component

The WP4.1 task involves the development of a Corpus Acquisition Component (CAC) for extracting monolingual and bilingual data from the web. The CAC is the first stage in the

PANACEA pipeline for building LRs by crawling web documents[1] with rich textual content. To implement the CAC, we will use and adapt an efficient and distributed web crawling methodology that will collect web pages with content belonging to specific languages and predefined domains. The CAC will also include modules that examine if the relevant pages come from sites with content available in more than one language. These pages will be used as the resources for creating parallel corpora collections in later processing stages.

We aim to contribute to the creation of monolingual corpora (1M words) for EN, EL, ES, IT, and FR, focusing on the environment and labor legislation domains. We also aim to contribute to the creation of parallel corpora (250-500K words) for the environment and legislation (EN-EL, EN-FR) domains needed in WP5. To this end, we aim to download enough *comparable corpora* from which the necessary parallel sentences will be extracted. Moreover, resources for another domain/language combination (i.e. automotive (EN-DE)) will be collected in the framework of the platform's final evaluation in WP8.

3.1 Task description

Efficient focused web crawlers try to acquire web documents relevant to a specific topic. The task of focused crawling consists of three major subtasks:

- Construction of a topic definition. The topic definition in the context of focused crawling is usually based on lists of positive and/or negative terms, optionally with weights.
- Construction of a list of seed pages. Focused crawlers are typically initialized with a list of seed pages. These lists are manually constructed either from existing collections such as the ones available from the Open Directory Project[2], or from responses by popular search engines to queries including the terms in the topic definition.
- Development of an efficient crawler that interacts with a text to topic classifier so that the crawler stores web documents relevant to specific domains. Typically a topic classifier is adapted so that it can provide relevance scores to web documents. The evolution of the crawl is affected by these scores.

For focused bilingual data acquisition, an additional subtask is required:

- Development of a module that, given a pair of languages SL and TL and a specific topic, examines each relevant SL page for links to approximate TL translations.

3.2 State of the art

This sub-section includes short descriptions of the most common algorithms exploited for crawling, followed by a discussion of web page classification techniques applied to focused crawling.

A. Crawling

1. Breadth-First (Pinkerton, 1994) is the simplest algorithm for crawling. It uses a list of URLs scheduled to be fetched called the frontier. In Breadth-First, the frontier is implemented as a First-In First-Out (FIFO) queue. Thus the pages are crawled in the order in which the links to other pages appear in the page under examination.

[1] In the initial version of both the monolingual and bilingual crawlers to be developed by T12 of the project, crawling will target HTML pages only.

[2] http://www.dmoz.org/

2. Best-First (Cho et al., 1998) is probably the most appropriate algorithm for a focused crawling task. Its basic idea is to select for crawling the best link from the frontier according to an estimation criterion. In its simplest form, a text to topic classifier (like Naive Bayes, Cosine Similarity, SVM, string matching, etc) is exploited to provide a score of relevance to each crawled page. This score is also assigned to each link within the page.

3. PageRank (Brin and Page, 1998) is based on the same idea but exploits the “popularity” of a web page instead of its relevance. The term “popularity” refers to the probability that a random crawler will visit that page at any given time. In other words, a page’s popularity score is estimated on the basis of the popularity scores of the pages that point to it. Consequently, the PageRank algorithm is more suitable for indexing web pages instead of collecting pages relevant to a specific domain.

4. Fish-search (De Bra and Post, 1994) could be considered as a combination of the Breadth-First and Best-First algorithms. It exploits a binary classifier in order to keep only the links within relevant pages. Then, the pages are crawled in the order in which the links to them appear in the page under examination.

5. Shark-search (Hersovici et al., 1998) is an improvement of the fish-search algorithm. The potential score of each link is influenced by the estimated relevance of its anchor text (i.e. the visible, clickable text in a link) and the source web page. Regression is adopted instead of binary classification.

7. InfoSpiders (Menczer and Belew, 2000) uses an adaptive population of agents searching for pages relevant to the topic using evolving query vectors and neural nets to decide which links to follow. InfoSpiders displays a disadvantage at the early stage of the crawling process, as the neural networks are not trained yet.

8. The Context Focused Crawler (Diligenti et al., 2000) builds a set of context graphs from seed pages. Associated classifiers are built and used to guide the search and update the context graphs.

9. Path algorithm (Passerini et al., 2001) adopts a similar idea with the Context Focused algorithm. It ranks each page according to its relevance to the topic and the distance from a relevant page (i.e. the number of links the crawler must be follow to visit this page starting from a relevant one).

10. Tunneling (Bergmark et al., 2002) is based on the assumption that relevant web pages are organized in clusters. The idea is that the crawler will not give up probing a direction immediately after it encounters an irrelevant page but will continue searching in that direction for a pre-defined number of steps. This allows the focused crawler to travel from one relevant web cluster to another when the gap (number of irrelevant pages) between them is within a limit.

B. Web page classification and focused crawling

Text classification addresses the task of labeling a text document with one or more labels from a set of predefined content-based categories. In the context of focused crawling, the task is developing efficient algorithms that classify web page content as relevant to a predefined domain or not. Qi and Davison (2009) review features and algorithms used in web page classification. In most of the algorithms reviewed, the plain text of the web page is represented as a high-dimensional feature vector, which encodes the presence of words or word n-grams in the document. Then, several machine-learning approaches such as density estimation using a naive Bayes classier (Joachims, 1997), a distance weighted k-nearest neighbor classifier (Yang, 1997), the C4.5 decision tree/rule learner (Quinlan, 1993) and SVMs with polynomial or RBF kernel (Joachims, 1998) are applied. In one of the first attempts to apply those text classification algorithms in collections of online documents, Joachims (1998) claims that SVMs significantly outperform other methods in this context.

Recently, many algorithms exploit additional information contained in HTML pages, such as HTML tags, hyperlinks and anchor text. Kwon and Lee (2003) classify Web pages using a modified k-Nearest Neighbor algorithm, in which terms contained within different HTML tags are given different weights. A similar approach (Golub and Ardo, 2005) assign different weight to terms depending on whether they are extracted from the title, the headings, the metadata, and/or the main text of the HTML page.

Other methods adopt the assumption that neighbouring pages are likely to be in the same topic (Davison, 2000, Chakrabarti et al., 2002, Menczer, 2005). In these approaches, it is it claimed that i) neighbouring pages involve common terms, ii) pages tend to link to pages on the same topic and iii) there is a strong correlation between the anchor text of the links and the content of the web pages links point to. Thus the topic of a target page can be guessed by examining a portion of the content (Utard and Furnkranz, 2005) or only the anchor text of the source page (Sun et al., 2002).

3.3 Existing tools

This subsection includes short descriptions of available open source tools for crawling. An overview of their functionalities is presented in Table 1. A comparison of these tools in terms of licensing, languages supported and availability as web services is illustrated in Table 2. Furthermore, we compare tools according to the usability requirements described in *D8.1. User Requirements* Specifically, we focus on performance, processing speed, feedback provided about the crawl progress, error handling, and quality of the documentation (Table 3).

The BootCat toolkit (Baroni et al., 2004) is a well-known suite of Perl scripts for bootstrapping specialized language corpora from the web. Bootcat initially creates random tuples from a seed term list and runs a query for each tuple (on the Yahoo! search engine). After keeping the first 10 results from each query, it constructs a URL list, downloads corresponding web pages and removes boilerplate. Two software applications that integrate the BootCat tools for constructing simple web corpora are the WebBootCat (Baroni et al., 2006) and the BootCat front-end, a web service front-end and a graphical user interface to the core tool, respectively. Although these applications are primarily designed for end users, they can be employed for certain initial (off-line) tasks (e.g. construction and testing of a seed URL list in a specific domain). In addition, a modified version of the BootCat toolkit can be used as an alternative tool for acquisition of monolingual corpora in specific domains.

Heritrix (Mohr et al., 2004) is an open-source and extensible web crawler. It is implemented in Java and the main interface is accessible using a web browser. Heritrix is one of the most configurable tools for crawling. To the best of our knowledge, it does not include functions about focused crawling based on a predefined list of terms of a specific topic.

Combine (Ardo, 2005) is an open system, implemented in Perl, for crawling Internet resources. It is based on a combination of a general Web crawler and an automated topic classifier. The classification is provided by a focus filter using a topic definition implemented as a list of terms describing this topic. One critical issue is the fact that, Combine, in its current implementation, does not sort and follow the most promising URLs in the frontier (i.e. links within pages with high relevance to the topic). In other words, it is a breadth-first crawler followed by a topic classifier. We believe that a modification to the Combine's strategy, which will include sorting URLs in the frontier, would be beneficial for Panacea purposes. It is worth mentioning that Combine: a) is an "active project", b) includes modules for language identification and topic classification, c) is modular and open-source and d) allows monitoring of the crawl progress by logging its actions in a relational database.

HTTrack (Roche, 2007) is actually a web site copier. In general, it downloads Web sites from the Internet to a local directory, building recursively all directories and storing HTML pages, images, and all other files from the server. It is fully configurable and supports filters and parameters that guide the harvesting. For example, a combination of the filters "-* +*/el/*.pdf +*/en/*.pdf downloads only pdf files from URLs which contain the string "/el/" or "/en/". HTTrack is a component of Bitextor (Esplà-Gomis, 2010), which to the best of our knowledge, is the only open-source application for building bilingual comparable corpora from multilingual websites. It uses HTTrack as mentioned above and makes two assumptions: i) candidate parallel pages should be under the same web domain and ii) should have similar html structure. Bitextor extracts four features (file size, length of plain text, tag structure, and list of numbers in the web page) for each downloaded page. Then it computes the relative differences of the first two features and the edit distances of the others (for every candidate pair of pages in different languages). Then pairs are classified as bitexts[3] or not based on a comparison of computed values against predefined thresholds. After comparing the positions of text blocks (i.e. text between html tags) in each bitext, the tool stores those pairs of text segments that are strong candidates for being translations of each other. The main shortcoming might be that the term segment does not correspond to any linguistic unit. Our experiments showed that segments are actually text blocks between some predefined html tags. Thus a segment could be a sentence, a paragraph, a part of a sentence, etc.

[3] In the Bitextor terminology, bitexts are pairs of files which contain approximately the same text in two different languages.

	Input	Output	Functionality
BootCat toolkit	List of terms in a specific topic	XML file that contains a monolingual corpus and metadata (url, date, size in words, domain)	Toolkit including scripts to bootstrap specialized corpora and terms from the web
Heritrix	Seed url list	Html files, multiple log files	Multithreaded, breadth-first web crawler. All parameters can be configured via web based user interface
Combine	1) Seed url list 2) List of terms in a specific topic	1) XML file including metadata (date, url, topic, language, etc.), 2) HTML files in UTF-8.	Multithreaded, best-first web crawler. Configurations can be set via an XML file.
HTTrack	Seed URL list	A mirror directory of each downloaded web site.	Multithreaded web copier. Filters and parameters can be set via a GUI.
Bitextor	Seed URL list	1) log file in which generated bitexts are recorded 2) HTML files in UTF-8 3) TMX file, which contains pairs of text segments (i.e. parts of text between successive html tags) that are strong candidates for being translations of each other	An automatic bitext generator integrating HTTrack. All parameters for filtering and comparing can be configured via an XML file.

Table 1 Overview of the functionalities of the available tools.

	Licence	Languages supported	WS
BootCat toolkit	GPL	Several (including all Panacea languages)	WebBootCat
Heritrix	LGPL	Not applicable	Web GUI
Combine	GPL	Uses Lingua::Identify Perl module for language identification. 33 languages supported (not Greek).	No
HTTrack	GPL	Not applicable	No
Bitextor	GPL	The integrated LibTextCat library contains fingerprints for 69 languages.	No

Table 2 Licensing, languages supported and availability as WS.

	Performance[4]	Feedback	Error handling	Documentation
BootCat toolkit	Given 4 terms in topic "Machine Translation", 41 pages were retrieved, resulting in a corpus of 144K words in 2.5 minutes	Progress bar (for WebBootCat)	Fully integrated	In progress
Heritrix	Speed depends on configuration	Multiple log files (e.g. crawl path, filtering results), full report at the end	Fully integrated	Comprehensive[5]. Large developer and user community.
Combine	Handles up to 200 URLs per minute. In an experiment for the topic "carnivorous plants", about 35% of all visited pages were judged relevant	Log table in an SQL DB	Fully integrated	Comprehensive [6]
HTTrack	Not mentioned	Progress bar	Fully integrated	Comprehensive [7]
Bitextor	Results depend on the structure of each website. On the well structured website of the Parliament of Canada, a 99% precision and a 85.33% recall were reported. Respective values on a heterogeneous web site were 86% and 61%.	Log messages for each major step (e.g. downloading, comparing, generating bitexts) ; needs improvement	Needs improvement	Needs improvement

Table 3 Comparison of available tools according to the usability requirements

[4] All performance reports are provided by developers or members of the developer groups for each toolkit

[5] http://crawler.archive.org/articles/user_manual/index.html

[6] http://combine.it.lth.se/documentation/

[7] http://www.httrack.com/html/index.html

4 Clean-up and normalization component

Corpus clean-up and normalization involve removing irrelevant parts of downloaded web pages in order to produce clean monolingual and bilingual data in uniform format applicable for training an MT system.

4.1 Task description

In this subsection we describe required modules for the development of a corpus clean-up and normalization component (CNC).

A. Text normalization

Web is highly heterogeneous not only in terms of content but also in terms of form. Documents and pages available on-line can have different file formats (html, pdf, doc, txt, etc.) and text encodings (UTF-8, ISO-8859-x, etc.). Text normalization involves detection of the formats and text encodings of the downloaded web pages and converting them into unified format (plain text) and text encoding (UTF-8).

B. Language identification

During language identification, each downloaded web page (or its part) is analysed and its language is identified. Then, documents (or their parts) which are not in the target language are discarded.

C. Web-page cleaning

Apart from a main textual content, a typical web page also contains certain “noise” including navigation links, advertisements, disclaimers, etc. (often called boilerplate) of only limited or no use for the purposes of training an MT system. Such irrelevant parts should be removed and only the main content should be kept in order to produce good-quality language resources. This is the most challenging task of the CNC and special attention will be paid to it in WP4.

D. Duplicate detection

The Web contains many duplicate pages, texts and their parts. Ignoring this phenomenon and including duplicate documents (or their parts) in the corpus could have negative effect on training the MT system. Duplicate detection involves identification of documents (or their parts) already appearing in the corpus and their elimination. In the area of web page crawling, the attention is focused on detection of near duplicate pages. Two pages with the same main content can differ in other parts (boilerplate) and therefore duplicate detection algorithms would fail in identifying them as full duplicates.

4.2 State of the art

A. Text normalization

Text normalization is rather a technical problem. File format detection is done simply by checking file endings (html, txt, doc, pdf, etc.) or by the Linux/Unix *file(1)* command which identifies format specific character sequences in the files. Text encoding is identified e.g. by the Linux/Unix command *enca(1)* which identifies encoding specific sequences in the files. The same tool can be used for text encoding conversion.

B. Language identification

The problem of automatic language identification has been extensively studied since the 1960's. An overview of applied approaches and methods is presented e.g. in Hughes et al. (2006): Cavnar and Trenkle (1994) used statistical models of character n-gram cooccurrence (fingerprints). Dunning (1994) employed Bayesian models for character sequence prediction. Grefenstette (1995) used concurrence of word and part of speech as the basis for determining if two given text samples were from the same, or different languages. Aslam and Frost (2003) applied an information-theoretic measure of document similarity. Some authors used more advanced statistical methods, e.g Poutsma (2001) who applied Monte Carlo based sampling to generate large random feature models and used them to identify a language based on the occurrence of the features, or Teytaud and Jalam (2001) who used kernel methods based on n-grams derived from inverse document frequency indices.

C. Web-page cleaning

Current state of the art in the area of corpus cleaning is briefly described in Spousta et al. (2008): Interest in web page cleaning originated in the area of web mining and search engines (see e.g. Cooley at al. (1999) or Lee et al. (2000)). Bar-Yossef and Rajagopalan (2002) introduced a notion of 'pagelet' determined by the number of hyperlinks in the HTML element used to segment a web page; pagelets with high frequency of hyperlinks were removed. Lin and Ho (2002) extracted keywords from each block of text to compute entropy, low-entropy blocks were removed. In Yi et al. (2003) and Yi and Liu (2003), a tree structure was introduced to capture the common presentation style of web pages and similarly as in the previous work entropy of its elements was computed to determine which element should be removed. Chen et al., (2006) proposed another two-stage cleaning method: in the first phase, web pages are segmented into blocks and blocks are clustered according to their layout features, in the second phase, the blocks with similar layout style and content are deleted.

Several new methods and approaches were introduced during the CLEANEVAL 2007 contest (http://cleaneval.sigwac.org.uk/) organized by the ACL Web as Corpus interest group. Competitive systems used both heuristics (often based on the observations that HTML tag density within boilerplate text is higher than within the main content and that main content is usually longer than boilerplate text) as well as sound machine learning methods (often requiring annotated data to optimize parameters of the systems), such as Support Vector Machines (Bauer et al., 2007), decision trees, genetic algorithms, and language models (Hofmann and Weerkamp, 2007). The best performing system was described in Marek et al. (2007) and later improved by Spousta et al. (2008). It employes multi-feature sequence labelling of textual blocks based on Conditional Random Fields.

D. Duplicate detection

(Near) duplicate detection is a difficult task because, generally, it is a quadratic problem: each new candidate document before being added to the corpus it must be checked against all other documents appearing in the corpus (e.g. by document similarity measures). Although such methods are quite accurate, the speed becomes a serious problem in large document collections. Therefore, several authors proposed methods that reduce the time complexity to sub-quadratic:

Shingling (Broder, 1997), I-Match (Chowdhury et al., 2002), Locality Sensitive Hashing (Gionis et al., 1999) and SpotSigs (Theobald et. al., 2008). SpotSigs, which specifically targets duplicate detection for web crawling, represents each web page as a set of spot signatures. A spot signature is a chain of words that follow frequent words as these are attested in a corpus. These signatures are rarely present in advertisements and navigational components of web pages. Thus, the signatures are built from portions of pages with "real" content. Then, SpotSigs adopts an efficient and self-tuning matching algorithm based on Jaccard similarity of sets of spot signatures, in order to derive an optimal partitioning of the web page collection into buckets of potentially matching documents, and thus to reduce the problem of identifying duplicates into a sub-quadratic one. Theobald et al. (2008) report that SpotSigs outperformed Shingling and I-Match algorithms in terms of recall and precision, and Locality Sensitive Hashing in efficiency over the TREC WT10g Web collection.

4.3 Existing Tools

A. Corpus normalization

file(1) is a Linux/Unix command used to determine type of a given file. It employs a sequence of tests and the first test that succeeds causes the file type to be returned. It recognized wide range of file types and for text file types target by the project works quite well.

enca(1) is a Linux/Unix command used to detect and convert encoding of text files. It supports all known text encodings and languages.

B. Language identification

A long list with language identification (LI) tools is presented in http://www.let.rug.nl/~vannoord/TextCat/competitors.html. Since, we are interested in using efficient and free/open source tools, we compare relevant tools in terms of performance, number of languages supported and licensing.

LibtextCat[8] is a free Perl library for efficient LI that is based on the text categorization algorithm presented by Cavnar and Trenkle (1994). LibtextCat supports 69 languages and is released under the BSD License. LibtextCat ports exist in C, Python and Java.

Lingua:Identify[9] is an open-source and flexible LI implemented in Perl. The user can define critical parameters such as languages activated during search, maximum size of input to analyze and part(s) of input to analyze. The tool supports 33 languages.

Lextek Language Identifier[10] is a fast and accurate LI tool, which supports about 260 language/charset combinations. An end user application is available but the corresponding SDK is a commercial product.

C. Corpus cleaning

Based on our best knowledge – only three web page cleaning tools are available as open source (Table 4). All of them expect HTML file on input and produce clean plain text on output.

Victor is (http://ufal.mff.cuni.cz/victor/) a tool for cleaning arbitrary web pages developed at Charles University in Prague (Spousta et al., 2008). It employs a sequence-labelling approach based on Conditional Random Fields. Every block of text (separated by a sequence of one or

[8] http://software.wise-guys.nl/libtextcat/
[9] http://search.cpan.org/~ambs/Lingua-Identify-0.26/README
[10] http://www.lextek.com/langid/

more HTML tags) in an analysed document is assigned a set of feature scores extracted from the textual content and HTML structure of the page. The blocks are automatically labelled either as content segments containing main web page content, which are preserved, or as boilerplate segments not suitable for further linguistic processing, which are eliminated. CRFs belong to the supervised group of machine learning algorithms and as such Victor requires manually annotated data to be trained on. The Victor's authors provide an AJAX application for easy annotation of such data (http://ufal.mff.cuni.cz/victoria/).

Boilerpipe (http://code.google.com/p/boilerpipe/) is a library implementing algorithms for detecting and removing boilerplate around the main textual content of a web page (Kohlschütter et al., 2010). It employs only a small set of shallow text features for classifying the individual text elements in a Web page -- number of words and link density. It provides specific strategies for some common task settings (for example: news article extraction) and can be easily adapted for other tasks.

NCleaner (http://webascorpus.sourceforge.net/) is a simple boilerplate removal tool employing character-level n-gram models as classifiers (Evert, 2008). It depends on Lynx, a text-mode Web browser, for converting HTML pages to plain text. It does not use HTML structure of the analyzed web pages. The tool is distributed with two pre-trained models for English pages and new models can easily be trained to other languages from manually cleaned training data.

	License	**Code**	**Languages**	**Requirements**	**Performance**	**Performance**
Victor	GPL	Perl	English +	Perl, CRF++	3.5 pages/s	84.1
Boilerpipe	Apache	Java	Independent	Java RE	~	~
NCleaner	GPL	Perl/C	English +	Perl, Lynx	20M words/s	82.9

Table 4 Web page cleaning tools

D. Duplicate detection

To the best of our knowledge, the only open source tool is SpotSigs[11] developed by Martin Theobald (2008). The package contains implementations of three of the four effective algorithms mentioned in section 4.2 of this report (SpotSigs, Locality Sensitive Hashing and I-Match).

5 Text Processing Component

5.1 Task description

The WP4.3 task involves developing a Text Processing Component (TPC) that, following operations the CAC and the CNC, will deal with the processing of the automatically acquired and normalized corpora. Partners involved in this task will adapt existing NLP tools for the languages addressed by the project. At the first development cycle of PANACEA, available lingware in the consortium and other open source tools for sentence splitting, POS tagging, lemmatization, and parsing/chunking will be integrated. Integration of other tools like named entity recognizers and term extractors for certain languages will also be examined based on the

[11] http://www.mpi-inf.mpg.de/~mtb/

project needs. We aim to use scalable tools that will be able to efficiently process the large amounts of data expected from CAC. PANACEA partners will take care of developing web services for the tools they will support for each language, as well as input and output converters to and from the common encoding format proposal documented in *D3.1 Architecture and Design of the Platform.*

In more detail, according to a) the [PANACEA Annex I] sections concerning WP5 (Parallel corpus and derivatives) and WP6 (Lexical Acquisition) and b) the user requirements documented in *D8.1 User Requirements*, the minimum set of NLP tools required for the project needs can be grouped as follows

1. Sentence splitters and tokenizers, POS taggers and lemmatizers for all languages. These basic tools perform basic preprocessing and will guide other NLP tools deployed in the factory.
2. Chunkers and/or parsers for all languages. These tools will generate annotations required for WP5 and WP6 technologies.
3. Named entity recognizers and term extractors for EN and DE at least. These tools will generate annotations required for the MT adaptation tasks to be performed in WP8.

5.2 State of the art

In this section we provide brief summaries of the state of the art of NLP tools for each language addressed by the project.

5.2.1 English

For sentence splitting, the RASP system (Briscoe et al., 2006) uses a set of deterministic finite-state rules, as do the FreeLing tools (Atserias et al., 2006). The LT-TTT tools (Grover et al., 2000) use a maximum entropy sentence boundary disambiguator following tokenization. Evaluation data is not available for these systems. Punkt (Kiss and Strunk, 2006), a language-independent, unsupervised approach to sentence boundary detection which focuses on accurate identification of abbreviations, achieved an accuracy of 98.4% on sections 3-6 of the Penn Treebank. Clark et al. (2009) reported 95.5% accuracy of Punkt on sections 2-21, and improved on Punkt's performance by using left- and rightward searching rather than regular expressions, as well as some additional rules, to achieve accuracy of 98.5%.

For tokenization, rule- and regular expression-based systems are the norm, including the tokenizers in the RASP system (Briscoe et al., 2006), the LT-TTT tools (Grover et al., 2000), the FreeLing tools (Atserias et al., 2006), and the Stanford tokenizer, which is based on Penn Treebank tokenization (included as part of the Stanford parser, Klein and Manning, 2003). Evaluation data is not readily available, but the tokenizers are considered to be highly accurate on newspaper data.

There is a wide variety of POS tagger, all achieving high accuracy. The C&C POS tagger (Curran and Clark, 2003) is a Maximum Entropy tagger trained on the Penn Treebank and using PTB POS tags, and achieves over 97% accuracy on Section 23 of the Penn Treebank. The Stanford POS tagger (Toutanova et al., 2003) uses log-linear models with dependency networks, with an English model trained on the Penn Treebank using PTB POS tags, and achieving 97.2% accuracy on Sections 22-24 of the Penn Treebank. The RASP parser includes a first order HMM POS tagger based on Elworthy (1994) and augmented with an unknown word model and an extended lexicon. It is trained on 3 million words of text from the Susanne, LOB and BNC corpora and uses a subset of the CLAWS tagset (Jurafsky and Martin, 2000). It achieves 97% accuracy on DepBank.

The standard tool for EN lemmatization is Morpha (Minnen et al., 2000). It is based on finite-state techniques and built on data from several large corpora. It comprises a set of morphological generalizations together with a list of exceptions for specific (irregular) word forms, for a total of about 1,650 rules. A 99.97% type accuracy has been reported with respect to the CELEX lexical database and a 99.98% token accuracy with respect to relevant tokens in the BNC.

For parsing English we focus on the unlexicalized parsers since these are most relevant for lexical acquisition. These include the RASP parser (Briscoe et al., 2006), a wide-coverage unification-based tag-sequence parser with a statistical parse selection component; and the Berkeley (Petrov et al., 2006) and Stanford parsers (Klein and Manning, 2003), both constituent PCFG parsers. RASP achieves 76.3 F-score on grammatical relations in the re-annotated DepBank corpus. The Stanford and Berkeley parsers both report over 90% accuracy on Section 23 of the Penn Treebank. Also relevant are dependency parsers such as MSTParser (McDonald et al., 2005), a graph-based dependency parser; and MaltParser (Nivre et al., 2006), a transition-based dependency parser. In a comparison on Section 23 of the Penn Treebank converted to Stanford dependencies (de Marneffe, 2006), the Stanford parser achieved a labeled F-score of 84.2, the Berkeley parser of 87.9, MaltParser of 81.1, and MSTParser of 78.8 (Cer et al., 2010).

For EN NER, the named entity recognizer in the LT-TTT tools (Grover et al., 2000) uses a staged combination of rule-based processing with probabilistic partial matching. On the MUC-7 competition it achieved an F-score of 93.39. FreeLing (Atserias et al., 2006) includes a simple NER system that matches on capitalized words and simple function words, as well as an HMM model which can be trained. Evaluation data is not available.

5.2.2 French

Unitex (Paumier, 2010) is a corpus processor implementing finite state technology that can be used for sentence splitting and tokenization for French. It can be freely downloaded and contains resources for various languages. TreeTagger (Schmid, 1997) is a language independent, probabilistic part-of-speech tagger, that uses decision trees to disambiguate word forms (POS). The tool has been adapted to French and a parameter file for FR is available from the author’s site. The error-driven transformation-based Brill POS tagger (Brill, 1995) has also been adapted to French and can be downloaded from the website of the ATILF-CNRS NLP group[12]. F. Béchet at the Laboratoire Informatique d’Avignon has developed a freely downloadable[13] dictionary-based POS tagger that also integrates an unknown word guesser and a lemmatizer. A named entity tagger using Conditional Random Fields has also been developed by the same author. The multi-language CST Lemmatiser (Jongejan and Dallianis, 2009) has been adapted to French as well.

As far as we know, no formal public evaluation is available for the tools above.

The GRACE evaluation campaign (Adda et al., 1998) was the first evaluation campaign on POS taggers for French. Competing systems were trained on a 10M tokens collection and tested on 650k tokens. Paroubek et al. (2006) discuss EASY, an evaluation of syntactic analysers for the

[12] http://www.atilf.fr/
[13] http://lia.univ-avignon.fr

French language, tested on a manually annotated 83K manually annotated collection. The PASSAGE project (Villemont de la Clergerie et al., 2008) is a follow-up of the Easy evaluation campaign (2007-2010) that aims at building semi-automatically a French Treebank of large size (more than 100 million words) by combining the output of several parsers.

Three standard implementations of well known algorithms for POS tagging (Treetagger, Brill, HMM) were trained and evaluated against a derivative corpus of the GRACE campaign in Allauzen and Bonneau-Maynard (2008). Treetagger was found to be the best system with 95.7% accuracy.

5.2.3 German

There are some free-available sentence segmentation tools for German. Sentrick is a tool based on the Punkt-System for sentence boundaries detection. There are no evaluation data for Sentrick available, whereby the Punkt-System reports an accuracy of 98.74% in sentence boundary detection on newspaper corpora in eleven languages (Kiss and Strunk, 2006). DKPro (Gurevych et al., 2007) is a framework that consists of a number of UIMA-components for various kinds of NLP tasks like tokenization, sentence splitting, POS-tagging, lemmatization, parsing and the like. The sentence segmentation component is based on the Java BreakIterator class. SPre (Hermes and Benden, 2005), is a user-configurable pre-processing tool, initially implemented as a plug-in for the GATE system and later within the Tesla framework. No evaluation data for these systems is available. The problem of sentence segmentation for German is discussed in (Palmer and Hearst, 1997).

JTok is a configurable tokeniser for German, developed at DFKI by Joerg Steffen. It is part of ‘Heart of Gold’, a XML-based middleware for integrating shallow and deep NLP components (Schaefer, 2007). jTokeniser is a Java library for tokenising strings into a list of tokens. It is a package comprising of 4 tokenisers that range from basic whitespace tokenisers to more complex ones: WhiteSpace-, Regex-, BreakIterator- and SentenceTokeniser. Evaluation data is not available.

Lemmatisation for morphologically complex languages like German is not a simple task. Some problems cannot be solved solely through a rule-based algorithm. Therefore, performing an accurate lemmatization for German requires a lexicon. The Durm German Lemmatisation System, developed at the University of Karslruhe (Perera and Witte, 2005), consists of a number of GATE components and resources that perform morphological analysis and lemmatisation for German nouns only. The lemmatizer shows accuracies of around 97% on a collection with 350 newspaper articles, the electronic version of one book, and a selection of 6 Wikipedia articles. The CST Lemmatiser (Jongejan and Haltrup, 2005) is written in C++ and is available for Danish, English, German, Icelandic, Dutch, Russian, Polish and French. The accuracy of the lemmatiser is estimated to be at least 87% for words that are not in the dictionary. However, the CST report contains no evaluation data for German.

There is a variety of German POS taggers, including freely available ones. The TnT (Brants, 2000) tagger is a statistical part-of-speech tagger that is trainable on different languages and tagsets. The tagger is an implementation of the Viterbi algorithm for 2nd order Markov models. An accuracy of 96.7% is reported on the NEGRA German Corpus. TreeTagger (Schmidt, 1994, 1995) is a language independent, probabilistic part-of-speech tagger that uses decision trees to disambiguate word forms. TreeTagger can also be used as a chunker for English, German, and French. An accuracy of 97.53% is reported on a small hand tagged German newspaper corpus.

Brill-Tagger is an error-driven transformation-based, rule-based tagger. There are many implementations of the Brill-Tagger for German, e.g. at the University of Zurich, (Schneider and Volk, 1998) where the STTS tagset (Stuttgart/Tübing Tagset) is used. Evaluation data is not available for this system.

The Proceedings of the Workshop on Parsing German (Kübler and Penn, 2008) provide a collection of references to various methods for parsing German. LoPar (Schmid, 2000) is an implementation of a parser for head-lexicalised probabilistic context-free grammars. A comparative study at the University of Tübingen (Kübler et al., 2006) had shown two different results of parsing German using two different treebanks: Negra and TüBa-D/Z. When trained on the Negra Corpus LoPar achieved a labelled F-score of 71.67, whereby when trained on the TüBa-D/Z it reached an F-score of 85.44. Kübler and Prokić (2006) made a comparison between LoPar (a constituency parser) and the MaltParser (a dependency parser). The dependency parser performed better with an accuracy of 83.4%, whereby the F-scores for constituents plus grammatical functions parses ranged between 51.4 and 75.3, depending on the treebank, NEGRA or TüBa-D/Z. BitPar (Schmid, 2004) is a parser for highly ambiguous probabilistic context-free grammars (such as treebank grammars). It uses bit-vector operations to speed up the basic parsing operations by parallelization (bitext parsing: automatically finding the syntactic structures of the parallel sentences in a text and its translation). In experiments at the University of Stuttgart with the BitPar trained on a portion (3718 sentences) of the Penn Treebank together with their translations into German, Fraser, Wang, and Schütze (2009) reported an accuracy of 87.89%. Berkeley Parser is based on probabilistic context-free grammars (Petrov et al., 2006). The research on parsing at the University of California at Berkeley focuses on unlexicalised parsers, automatically learned grammars and training without additional human input. In experiments for parsing German (Petrov and Klein, 2007) report a labelled precision and recall of 80.1%.

For Named Entity Recognition, Florian et al. (2003) presented an experimental framework combining four diverse classifiers (robust linear classifier, maximum entropy, transformation-based learning, and Hidden Markov Models). They achieved an overall F_1-score of 72.41% on identifying *person*, *locations* and *organizations* in German texts (and an F_1-score of 88.76% for English).

5.2.4 Greek

For sentence boundary detection in Greek texts, Stamatatos et al. (1999) employ transformation-based learning to extract sentence boundary disambiguation rules and report an accuracy of 99.4% in a test corpus with 8.7K sentences. Papageorgiou et al. (2000) discuss a regex-based tokenizer and sentence splitter that contains gazetteers of abbreviations.

Several data-driven approaches for POS tagging of Greek have been reported. Dermatas and Kokkinakis (1995) implemented an HMM tagger and reported an accuracy of around 94% when using a tagset of 11 tags. Orphanos and Christodoulakis (1999) combined a high-coverage lexicon of 870K wordforms and an induced decision tree disambiguator/guesser into a POS tagger that yielded 94.5% accuracy in 10-fold cross-validation tests on a 138K corpus. Using Brill's tagger and a tagset of 58 tags, Petasis et al. (1999) conducted experiments with two small corpora of 125K tokens and 65K tokens and reported accuracies of around 95%. Papageorgiou et al. (2000) used a variation of transformation-based learning and a Parole-compatible tagset of 584 labels on a well balanced training corpus of 447K tokens. They reported accuracies of 96.28% on POS and POS subtype. Maragoudakis et al. (2003) present a Bayesian Belief

Network tagger that shows accuracies of 96% and 97%, on disambiguating POS and case respectively.

For parsing Greek text, Boutsis et al. (2000) discuss a system based on a manually developed grammar consisting of non-recursive regular expressions. The system has been evaluated against a manually annotated corpus of 33K tokens and shown precision and recall results well above 94% for most types of non-recursive phrasal constituents. Maragoudakis et al. (2004) report on a Bayesian shallow parser that detects subject-object pairs with an accuracy of 92% on gold input from preprocessing stages.

NER systems for Greek include Karkaletsis et al. (1999) who compare manually compiled NER grammars to induced decision trees that distinguish between classes of person, organization and non-NE. Giouli et al. (2006) discuss a maximum entropy system that yields an overall F1-score of 94.87% on loc, org and person NEs of a news corpus. Lucarelli et al. (2007) present a NER system that identifies person and organization names by the use of SVMs that scan the text in two passes, where the second pass identifies re-occurrences of NEs in different contexts. The authors report F1-scores of 94.05% and 79.39% for person and orgs, respectively.

5.2.5 Italian

There is a wide variety of POS Taggers for Italian with high accuracy. In the SemaWiki Project (Attardi et al., 2009) two well-known taggers are used: TreeTagger (Schmid, 1994) and Hunpos (Halácsy et al., 2007) which have been trained on Italian. The results of the two taggers have been combined in order to improve their accuracy, by achieving 96.75% accuracy in the Evalita 2009 evaluation campaign of Natural Language Processing tools for Italian[14]. TagPro (Pianta and Zanoli, 2007a) is a system based on Support Vector Machines (SVM). TagPro exploits a rich set of features, including morphological analysis. It scored as the best system in the Italian Pos Tagging task at Evalita 2007 (98.04% of accuracy). The ILC-UniPi MaxEnt PoS Tagger. (Dell'orletta et al., 2007) is a Maximum Entropy PoS tagger operating on the output of MAGIC, an Italian rule-based morphological parser. It has been evaluated on the Evalita 2007 data set reporting 97.65% accuracy. C4 (Romagnoli, 2007) is a portable statistical part of speech tagger based on a second order Markov model technique. No evaluation measure is reported. CORIStagger (Tamburini, 2007) is an evolution of the tool developed inside the CORIS project and is based on a HMM system. It has reported 97.59% accuracy at Evalita 2007. Enriched TreeTagger (Baroni et al., 2007) is a probabilistic system based on decision trees. It has achieved 97.89% accuracy at Evalita 2007. UniPISyntema (Aliprandi et al., 2007) is a system for handling inflected languages. The system combines statistical and rule based methods and is an on-the-fly POS tagger for Open-Domain texts, combining second-order Markov Models, Lexical Resources and deep morpho-syntactic annotations. It achieved 88.71% of accuracy at Evalita 2007. The Evalita 2007 data set is available through the campaign website.

For parsing in Italian different systems are available. We report here state of the art systems, without taking into account the difference between lexicalised and unlexicalised ones. Lavelli et al. (2009) have compared four different parsing algorithms using an open-source transition-based dependency parser in the Dependency Parsing Task of Evalita 2009. They achieved the following results: Labeled Attachment Score (LAS): 83.43; Unlabeled Attachment Score (UAS): 87.91; and Label Accuracy (LA): 89.57. Among dependency parsers we have also Lesmo_PAR (Lesmo and Lombardo, 2002), DeSR (Attardi, 2006) and Shen’s Bidirectional

[14] http://evalita.fbk.eu/

Parser (Shen, 2006). Lesmo_PAR is a rule-based system that includes chunking followed by attachment of verb dependents driven by both rules manually developed and data about verbal subcategorization. The results reported at the Evalita 2007 are LAS: 86.94; UAS: 90.90; LA: 91.59. DeSR is a statistical dependency parser. DeSR implements a deterministic shift-reduce methodology to parsing that handles non-projective dependencies incrementally. The results at Evalita 2007 are LAS: 77.88; UAS: 88.43; LA: 83.00. Finally, Shen's Bidirectional Parser is a statistical bidirectional dependency parser which does a greedy search over the sentence and picks the relation between two words with the best score each time and builds the partial tree. It achieved LAS: 74.85; UAS: 85.88; LA: 81.59 at Evalita 2007. The Berkeley parser for Italian (Lavelli and Corazza, 2009) is a constituency parser obtained by adapting the Berkeley Parser to Italian. It achieved the best F1 score (78.73, recall: 80.02; precision: 77.48) at Evalita 2009. Another constituency based parser is the Data-Oriented Parsing based system by Sangati (2009). The main idea behind the implementation of this parser is to extract as many as possible fragments from the training corpus, and recombine them via a probabilistic generative model, in order to parse novel sentences. The system achieves 75.76% in labeled F-score. Evalita 2007 and 2009 data sets are available through the campaign website.

Named Entity Recognizers for Italian include the system by Mehdad et al. (2009), which is based on the YAMCHA classifier machine. It achieved 81.09 F-measure at Evalita 2009. EntityPro (Pianta and Zanoli, 2007b) is a system based on SVM. For each running word, EntityPro extracts a rich set of linguistic features in a one-word window. It achieved an F-measure of 82.14% at Evalita 2007. Finally, Yahoo NER System (Ciaramita and Atserias, 2007) implements a Hidden Markov Model, based on a regularized perceptron classifier. It achieved an F-measure of 68.99% at Evalita 2007. Evalita data set on NER is available through the campaign website.

5.2.6 Spanish

Martínez et al. (2010) have developed the IULA Processing Tool, a system for sentence splitting, tokenization and named-entity recognition of Spanish. The tool is based on rules which depend on a series of resources to improve obtained results: a grammatical phrase list, a foreign expression list, a follow-up abbreviation list, a word-form lexical database and a stop-list to increase lexical-lookup efficiency. The tool has been evaluated against a hand-tagged corpus divided in two domain specific topics (Press and Genomics). Accuracies of 99.39% and 91.55% are reported for sentence splitting in the two collections. Respective results for NER are 95.43% and 99.76%. Accuracies of 76.85% and 85.00% are reported for recognizing Named Entities at the beginning of a sentence due to the ambiguous capitalization problem. Europarl tools for sentence splitting and tokenization (http://www.statmt.org/europarl/) include specific scripts for Spanish.

Giménez and Màrquez (2004) have developed the SVMTool, a POS tagger generator based on Support Vector Machines, coded in Perl/C++. The tagger, which is available from http://www.lsi.upc.edu/~nlp/SVMTool/, achieves a state of the art accuracy of 97.2%. Ferrández and Peral (2005) trained an HMM PoS tagger on the 50K words CLIP-TALP corpus for Spanish. An accuracy of 95.36% was reported on a test corpus of 10K words. GRAMPAL (Moreno and Guirao, 2006) is a tagger and lemmatizer focusing on spoken Spanish. GRAMPAL relies on a large lexicon and a disambiguation process based on statistical training. An accuracy of over 95% has been reported on spontaneous speech. The IULA PoS Tagger (Vivaldi, 2009) is an adaptation of the TreeTagger (Schmidt, 1994) that integrates a lemmatizer and uses the IULA tagset for Spanish. The accuracy for both tagging and lemmatization is 98% tested against a 100K words test set. MOSTAS (Iglesias et al., 2008) is an NLP tool focusing on semi-structured information from clinical reports. This tool tags clinical texts with morpho-semantic information and anonymizes the reports by covering sensitive patient information. It also detects biomedical concepts using specialized biomedical lexica and thesauri.

SUPAR (Ferrández et al., 1998) is an NLP tool aiming at full or partial syntactic analysis and anaphora resolution. SUPAR works for Spanish and English and outputs morphosyntactically tagged sentences with pronominal anaphora resolution. An accuracy of 81% was reported for partial parsing on a 9600 words test set.

For named-entity recognition of Spanish, Kozareva et al. (2007) developed NERUA, a tool able to classify entities in four groups: person, location, organization and other. NERUA is based on 3 different learning algorithms: HMM, Maximum Entropy and Memory-based learning and employs a simple voting strategy to obtain the combined result of the three methods. F_1-scores of 92.96% for detection and 78.59% for classification are reported on the CoNLL-2002 data sets.

5.3 Existing tools

In this section we summarize the results of a survey on several aspects of TPC tools *already available* to PANACEA partners. The tools surveyed include both tools developed by consortium partners, and open source tools developed by third parties and adapted/extended by partners. The survey was conducted via a template that is included in Appendix A. Overall, descriptions for 33 tools or pipelines of tools were submitted by PANACEA partners for all languages addressed by the project.

As examples of the descriptions provided by partners, we also include in the Appendix descriptions for two tools: a) the Decomposer for German Compounds developed by Linguatec, and b) the ILSP FBT POS Tagger. An overview of all documented tools classified according to their functionalities is presented in Table 5 below.

	English	**French**	**German**	**Spanish**	**Italian**	**Greek**
Sentence splitting	EuroParl tools FreeLing LT-SentenceSegmentiser	EuroParl tools	EuroParl tools LT-SentenceSegmentiser	EuroParl tools IULA Processing Tool FreeLing	EuroParl tools FreeLing Syn SG	EuroParl tools ILSP SST
Tokenization	EuroParl tools, Charniak FreeLing, LT-Tokeniser RASP, Stanford	EuroParl tools	EuroParl tools LT-Tokeniser	Morfette EuroParl tools FreeLing	FreeLing Syn SG	ILSP SST
POS tagging	Charniak, Bikel, Berkeley, FreeLing, LT-Tagger, RASP C&C, Stanford	Bikel BitPar LT-Tagger	Berkeley	Morfette IULA POS Tagger FreeLing	FreeLing Syn SG	ILSP FBT
Parsing	Charniak, Bikel, Berkeley, FreeLing, LT-Parser, RASP, C&C, Stanford	Bikel Berkeley BitPar	Berkeley LT-Parser	Bikel FreeLing		
Function labeling	FunTag		Labeller	FunTag		
Lemmatization	XLE grammar lexicon, Freeling, LT-Lemmatiser, RASP, C&C	LT-Lemmatiser	TreeTagger LT-Lemmatiser	Morfette FreeLing LT-Lemmatiser	Syn SG FreeLing LT-Lemmatiser	ILSP Lemmatizer
LFG parsing	LFG AA	LFG AA	LFG AA	LFG AA		
Dependency parsing	Maltparser, MSTparser, FreeLing			FreeLing	Syn SG	
Chunking					Syn SG	ILSP Chunker
Named Entity Recognition	FreeLing, LT-Namer, ILSP MENER, C&C	LT-Namer	LT-Namer	FreeLing LT-Namer	FreeLing Syn SG LT-Namer	ILSP MENER
Multiword Expression Extraction	FreeLing			FreeLing	FreeLing	
Word Sense Disambiguation	FreeLing			FreeLing	FreeLing	
Other	LT-TopicIdentifier		LT-Decomposer LM-MonoTermExtract			

Table 5 Text Processing Component Tool Overview

5.3.1 Availability and licensing

As an initial comment to the overview of tool availability provided in the overview table, we can see that partners have already developed and/or adapted tools corresponding to the sets of required tools for all PANACEA languages (DE, EN, EL, ES, FR, IT). EN and DE term extractors and named entity recognizers are available for the WP8 tasks. Noticeable tools missing include constituency parsers for Italian and Greek, although a dependency parser and a chunker are available for these two languages respectively. DE, EL, ES and IT are mostly covered by tools developed by PANACEA partners, while for EN and FR available tools are open source solutions.

The survey also showed that the majority of the tools are not currently available as Web Services, i.e. software systems "designed to support interoperable **machine-to-machine** interaction over a network"[15]. Two of the tools were available for testing via web demos (ILC SynSG, DCU LFG AA), one tool (C & C parser) provides code from its site for an optional installation of a SOAP web service, while others (ILSP SST, ILSP Tagger) have been deployed using the UIMA AS Client - Service Architecture[16] but are not publicly accessible yet.

Tools are made available under a variety of licenses (see Figure 1). Tools developed by partners are mostly available for research purposes or on a case-to-case basis. As expected, the majority of tools developed by third parties and adapted/extended by partners are available as free or open software.

5.3.2 Implementation and performance

For certain languages targeted by PANACEA, standalone tools for different processing stages are not available. Instead, pipelines or comprehensive one-does-it-all tools cover all processing stages up to and including syntactic analysis (e.g. the Italian SynSg System, or the Charniak and RASP parsers). Almost half of the tools are implemented in Java, with C/C++ coming second together with some hybrid systems (Figure 2) involving Perl, Python, etc. More than half of the tools claim to be OS independent or, at least, operating on both Windows and Linux (Figure 3).

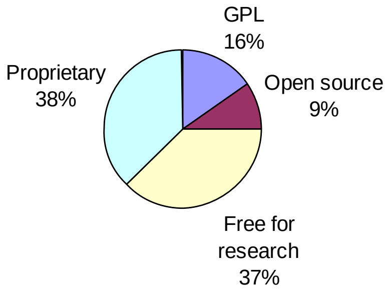
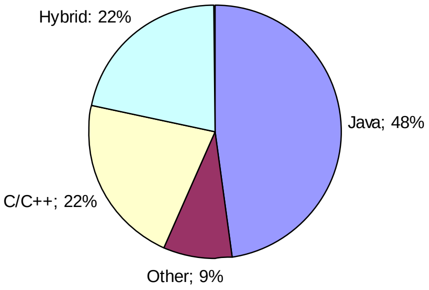


Figure 1 Licenses of NLP tools

[15] http://www.w3.org/TR/ws-gloss/
[16] http://uima.apache.org/doc-uimaas-what.html



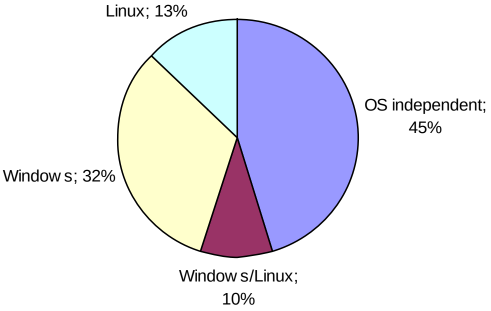


Figure 2 Programming Languages

Figure 3 Operating Systems

It is not easy to compare the processing speed of different categories of NLP tools, especially if pipelines of tools instead of standalone versions are documented. Depending on the perspective, the majority of most basic tools (i.e. up to lemmatization) seem to perform relatively fast (Figure 5). Nevertheless, higher level tools, like for example constituency parsers for EN, are much slower in comparison (Figure 5).

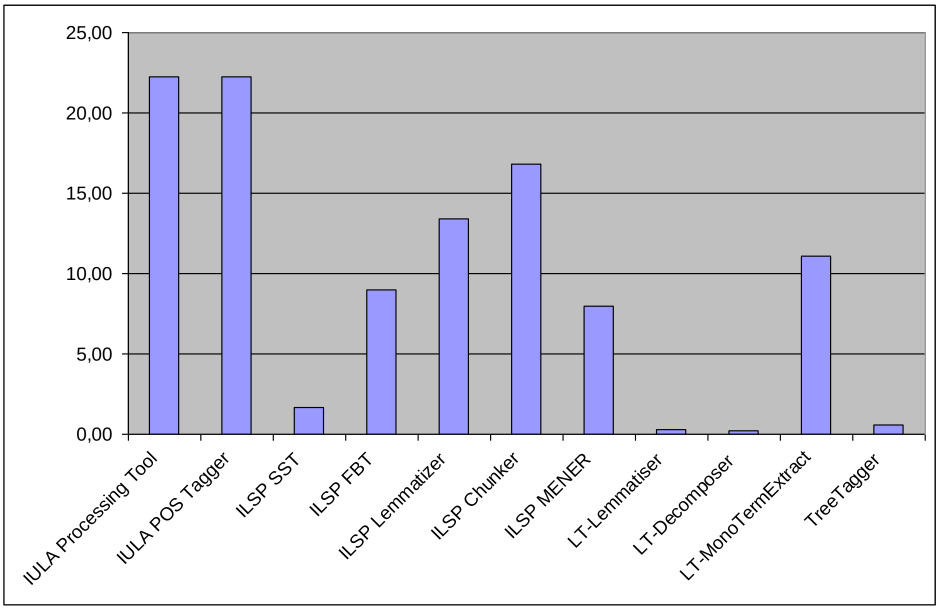


Figure 4 Performance (hours for 20M tokens) for a subset of basic NLP tools

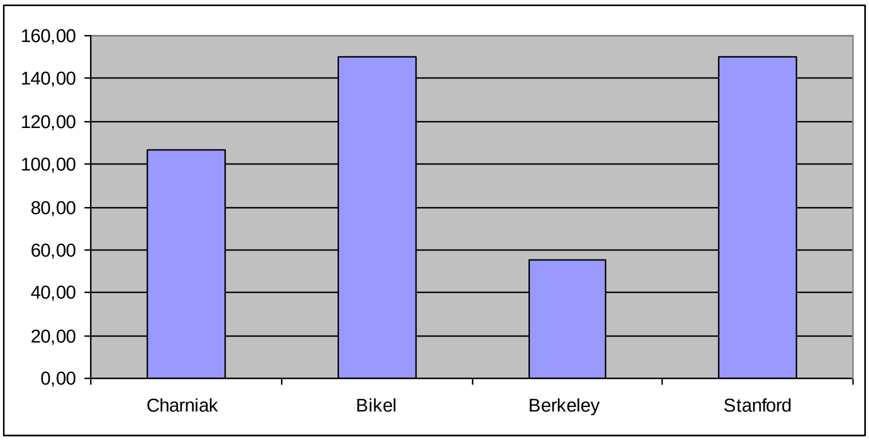


Figure 5 Performance (hours for 20M tokens) for EN parsers (projected from their performance on Sec. 22 of Penn TB)

5.3.3 Character Encodings, Formats and Linguistic Representations

Over 85% of the tools surveyed are compatible with the UTF-8 character encoding for Unicode. As far as encoding formats are concerned, around 70% of the tools use some kind of vertical (one token per line – tab separated annotations) or inline format, while 13% export their results in both standoff and inline annotation files. The rest of the formats include lists of annotations like terms and topics. Only a small percentage of tools (28%) use some well defined encoding format like XCES[17] or UIMA CAS[18]. The rest tend to use proprietary vertical formats or Penn Treebank[19] style bracketed trees.

As far as linguistic data categories are concerned, UPF, ILC and ILSP use EAGLES/Parole compatible tagsets for POS tagging. Linguatec uses a tagset compatible to the STTS standard for DE[20]. For EN constituency parsers (and integrating taggers) the Penn Treebank categories are clear winners.

5.3.4 Evaluation

A summary of evaluation results is hard to produce since quantitive evaluation figures for all tools are not available. POS taggers are among the tools for which most comparative figures have been provided during the survey. As an indication, TreeTagger, IULA POS and ILSP FBT show relatively high accuracies of 96-98%. Of course, differences in tagset size and evaluation test sets make such results difficult to compare. Other parameters may affect results as well. For example, when below-POS features (like case) are taken into account, ILSP FBT's performance drops to a 93%.

On the other hand, comparison of other categories of tools like EN constituency parsers seems more straightforward, since 4 out of 5 surveyed have been tested on the Penn Treebank. The Berkeley, Charniak, Bikel, Bitpar and Stanford parsers all have F_1-scores above 86%, with Berkeley leading with a 91% score.

[17] http://www.xces.org/
[18] http://docs.oasis-open.org/uima/v1.0/uima-v1.0.html
[19] http://www.cis.upenn.edu/~treebank/
[20] http://www.sfb441.uni-tuebingen.de/a5/codii/info-stts-en.xhtml

6 Resource description

In this section we describe the resources to be produced in WP4.

A. Domain-specific monolingual corpora

We aim to create the domain-specific monolingual corpora presented in the table below. The minimum amount of monolingual data has been decided to be 1M words for each language/domain combination. The "news" domain is the fall-back option, in case we do not achieve to acquire the proper number of words for the other domains.

Language/domain	ENVIRONMENT	LEGAL	NEWS
EN	√	√	*
ES	√	√	*
IT	√	√	*
EL	√	√	*
FR	√	√	*

Table 6 Monolingual Corpora to be produced in WP4

The corpus acquisition, clean-up and normalization phase (WP4.1 and WP4.2) aims to fetch and store web pages (relevant to these domains and in the selected languages) that contain enough plain text to cover the amount of data needed. During this phase, boilerplate will be removed from the crawled documents, and near duplicates will be detected and rejected.

The output of this phase will include the original HTML files and corresponding XML files with basic metadata as described in Panacea Deliverable *D3.1 Architecture and Design of the Platform*, section 6.1.2. The XML files will also contain the extracted text converted in UTF-8. Paragraph indicators from the HTML pages will be transferred will guide paragraph segmentation of the text in the XML files.

The text from these XML files will be fed to the Text Processing Component (WP4.3). NLP tools for sentence splitting, tokenization, POS tagging, lemmatization and parsing will output new XML files including annotations from all the above processing stages in the format described in Panacea Deliverable *D3.1 Architecture and Design of the Platform*, section 6.1.3. An example of such an XML file (the final output of WP4) is provided and discussed in Appendix D.

B. Domain-specific bilingual comparable corpora

We aim to produce bilingual, document-aligned comparable corpora in the domain/language combinations shown in Table 7. The "news" domain is again the fall-back option.

From these comparable corpora, WP5 will generate sentence-aligned corpora of 250-500K words for each language/domain combination. At this stage of the project it is hard to estimate the appropriate size of comparable data we should acquire in the context of WP4.

Comparable corpora	AUTOMOTIVE[21]	ENVIRONMENT	LEGAL	NEWS
EN-DE	√			
EN-EL		√	√	*
EN-FR		√	√	*

Table 7 Bilingual comparable corpora to be produced in WP4

The data flow will be the same as in the case of monolingual data with some additional information. The crawled html pages will be stored and a log file which will denote pairs of documents (document alignment) will also be generated. Furthermore, we will examine in combination with WP5, if storing segment alignments, like the ones generated by the bilingual crawlers described in 3.3, would be useful for assisting the task of parallel sentences extraction.

We plan to take into account feedback from WP5 and WP6 regarding content and size and improve the functionality and the output of the corpus acquisition and cleaning tools during the second development cycle of the project.

7 Possible workflow

Based on the task descriptions for the Corpus Acquisition, Clean-up and Normalization and Text Processing Components we show in Figure 6 the overall proposed workflow for the creation of the resources described above. Descriptions of the workflow for each component are discussed in the following subsections. A list of the tools to be deployed as web services in the PANACEA factory is presented in 7.4 below.

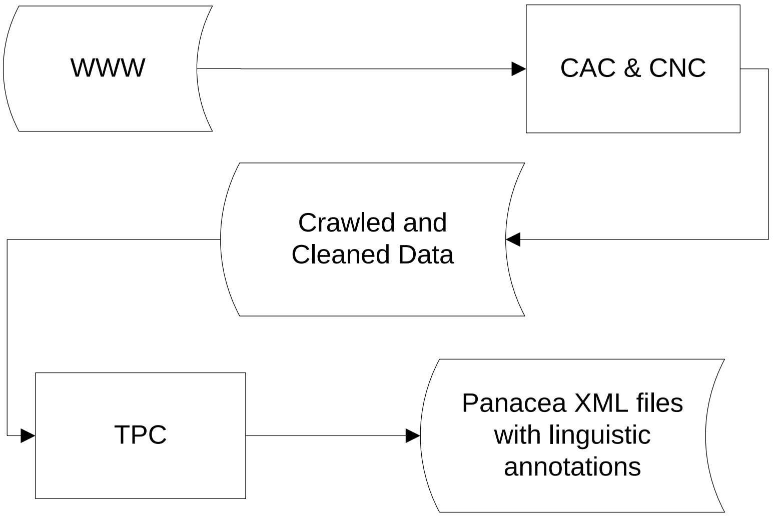


Figure 6 Overall workflow for CAA subsystem

[21] As stated above, the resources for the automotive (EN-DE) language/domain combination will be collected in the framework of the platform's final evaluation in WP8.

7.1 Monolingual data acquisition and clean-up

A possible workflow for monolingual data acquisition and clean-up in a specific domain is presented in Figure 7. To implement this workflow, we initially plan to create a modified version of the Combine crawler. This version will use the Best-First algorithm instead of the default Breadth-First. It is worth mentioning that the proposed workflow includes modules for language identification, boilerplate removal, text normalization and duplicate detection. This way we will avoid crawling for data in non-target languages or data that have already been downloaded.

Input for the PANACEA monolingual focused crawler will include the domain definition and a list of seed URLs for each domain-specific crawl (labour legislation and environment). In order to define the domain, we will adopt a strategy followed by many researchers (Menczer et al., 2004, Ardo and Golub, 2007, Dorado, 2008), that is manually constructing a definition by exploiting already available lists or thesauri. We will examine, among others, the Eurovoc[22] multilingual thesaurus in order to create appropriate term lists for specific domains in many European languages. The definition will also include a short prose-like description of the topic and metadata like the title of the topic, comments, the author's name, the date, etc. The initial list of URLs will be manually constructed either from existing collections like as the ones available from the Open Directory Project[23].

We present the data flow in more detail below, following the steps in Figure 7:

1. At the beginning of crawling, the frontier is empty. Therefore, the frontier is filled with the URLs from the seed URL list.
2. A URL is selected and removed from the frontier. A multithreaded crawling implementation will ensure concurrent visiting of more than one page.
3. A page is fetched and the list of visited URLs is updated.
4. Normalization involves format identification and character set conversion. If the recognized format is not the targeted one (i.e. html for the first version of the Panacea crawler), the page is discarded. Libraries like LibEnca and LibIconv will be used for character encoding guessing and conversion to UTF-8. Conversion of the content to an appropriate format is necessary for comparison with the language fingerprint and the terms in the domain definition.
5. During cleaning (by a library like Boilerpipe) html tags, java script sections and boilerplate are marked for removal.
6. Language identification (by a library like LibTextCat) involves identifying and discarding pages in non-target languages.
7. Domain filtering. In this step, the relevance of the page to the domain is estimated. We plan to compare the text of the page with the term list and provide a relevance score. For calculating this score, we will initially exploit Combine's string-to-string matching and/or linear SVM classifier.
8. Duplicate detection. We plan to use the SpotsSig module mentioned above for near duplicate detection.
9. The page is stored if its relevance score exceeds a threshold that will be defined after initial experimentation. The cleaned version of the HTML page is stored in an XML file which also contains a) automatically extracted essential metadata for each page (e.g. url,

[22] http://europa.eu/eurovoc/

[23] http://www.dmoz.org/

download date, language, topic, etc) and b) the paragraph-segmented textual content of the HTML pages.

10. The links are extracted from both relevant and non-relevant pages. The idea of keeping links from irrelevant pages is called tunnelling. A focused crawler using tunneling will not give up probing a direction immediately after it encounters an irrelevant page. Instead, it continues searching in that direction for a pre-set number of steps. This allows the focused crawler to travel from one relevant web cluster to another when the gap between the two clusters (the irrelevant pages) does not exceed a predefined limit. Links found in irrelevant pages get a lower score.
11. Extracted links are added to the list of new URLs, which are sorted according to their relevance scores.
12. When the frontier has no more URLs (after many repetitions of step 2) it is filled with the top-scored items of the list of new URLs.
13. Steps 2-12 are repeated until a criterion is met (e.g. the time of crawling has expired or the number of stored html files has reached a certain threshold).

As described in the steps above, the output of the above process will include a pool of stored HTML pages and a corpus of corresponding XML files with content as in step 9 above.

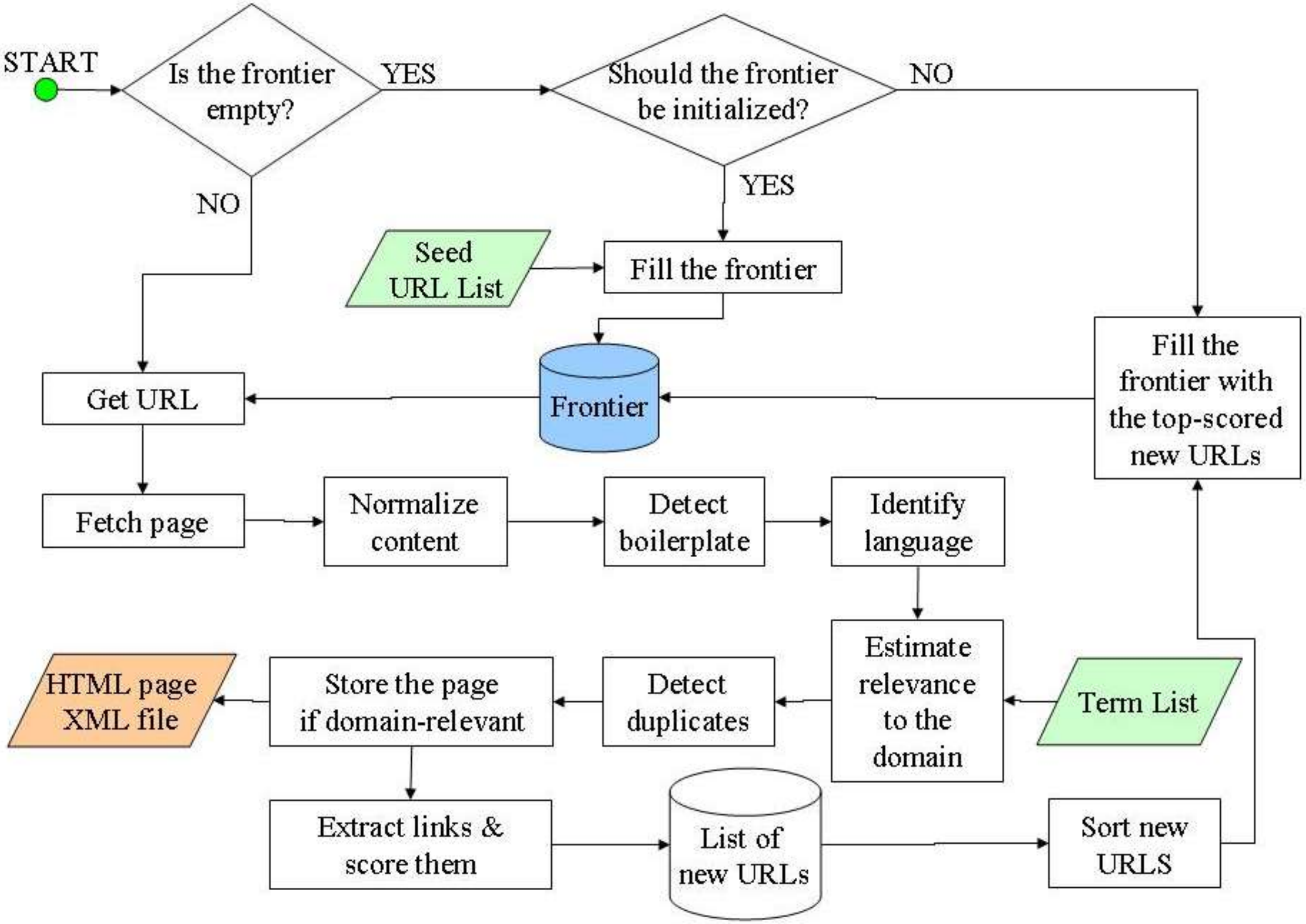


Figure 7 A possible workflow for focused monolingual data acquisition

7.2 Bilingual data acquisition and clean-up

Input for the PANACEA bilingual focused crawler will include domain definitions in both languages L1 and L2 targeted by the crawl and a list of seed URLs for each domain-specific crawl (labour legislation and environment). A possible workflow is presented in Figure 8. To implement this workflow we initially plan to integrate the modified version of Combine with Bitextor (cf. 3.3).

We present the data flow in more detail below, following the steps in Figure 8 and skipping steps 1-8 which are identical to the monolingual workflow:

9. URL filtering. In this step, the URL of the current page is compared with some predefined patterns for one of the targeted languages, L1. For example, suppose that the targeted languages are English and Greek. Possible patterns include “/en/”, “lang=en”, “/el/”, and “lang=el” which denote that the current page is part of a multilingual web site.
10. The page is stored if it is classified as domain-relevant and its URL matches the patterns. An XML file is generated as in step 9 of the monolingual workflow.
11. Links are extracted from both relevant and irrelevant pages.
12. In the next step an Inverse URL Filtering is performed. Each extracted link is compared to predefined templates from L2. For example, suppose that the URL of the current page contains “/en/”. An extracted link D will be assigned a score that on the basis of a) whether it matches the “/el/” pattern, and b) on the link’s similarity (e.g. its edit distance) to the current URL.

Steps 11-13 of the monolingual workflow are repeated in this workflow as well. The final step of the bilingual workflow includes examining the pool of stored HTML pages and deciding which pages can be considered as pairs from which parallel sentences can be extracted. The detection of these pairs will be based on a set of measures like: a) relative difference in file size, b) relative difference in length of plain text, c) edit distance of the HTML structures and d) edit distance of the lists of numbers contained in the stored pages. We will initially use these measures as defined by Bitextor, and compare values with corresponding predefined thresholds.

As described in the steps above, the output of the above process will include a pool of stored HTML pages and a corpus of corresponding XML files with content as defined above. Additionally, the output will include a log file in which detected pairs of documents are recorded.

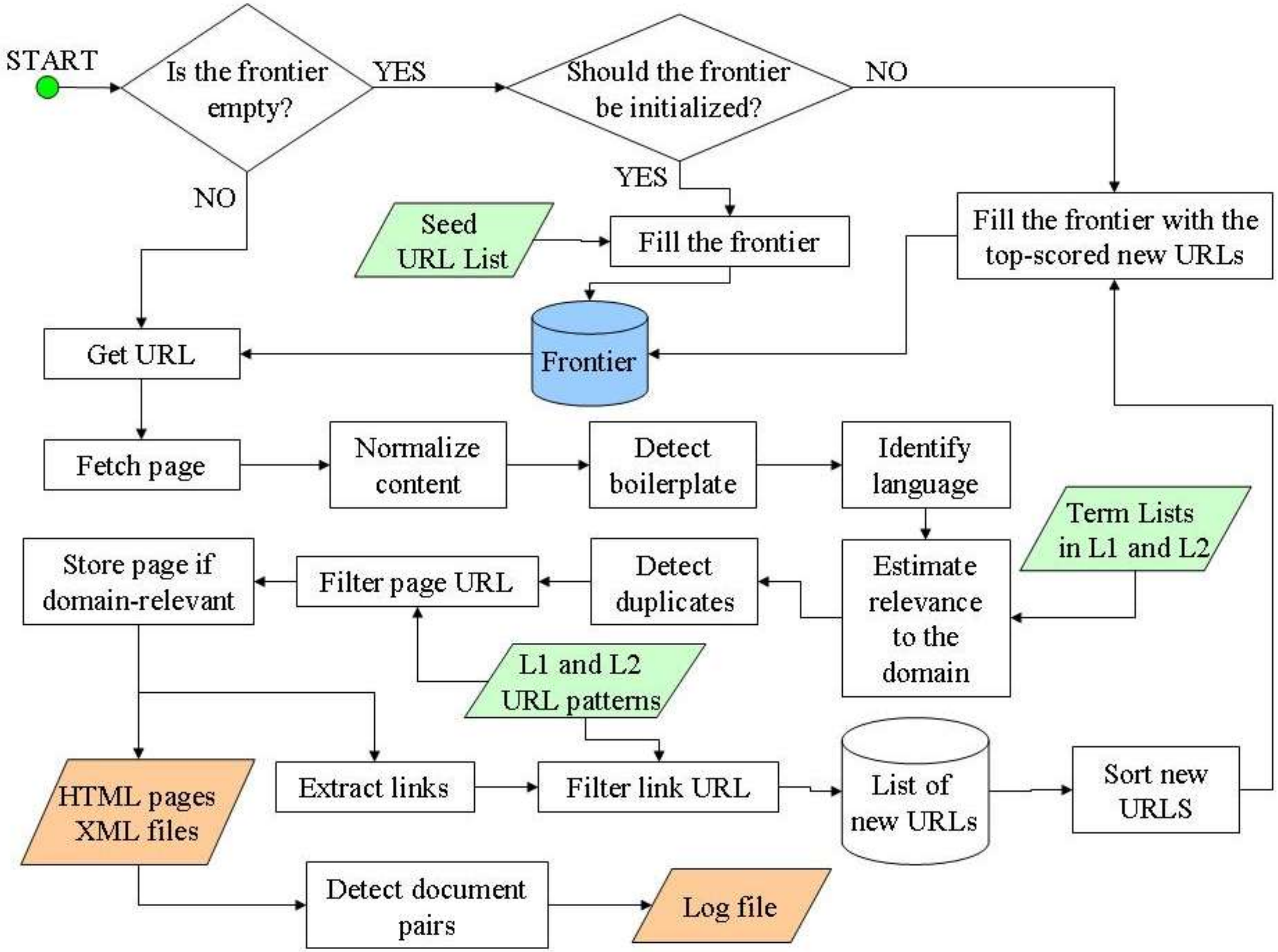


Figure 8 A possible workflow for bilingual data acquisition

7.3 Text processing

Once clean and normalized text from the crawled texts has been stored, data will be processed by the TPC tools (sentence splitters and tokenizers; POS taggers and lemmatizers; constituency and dependency parsers; named entity recognizers and term extractors if needed) as in Figure 9. Annotations generated during the text processing stages will be added to the PANACEA XML files described in *D3.1 Architecture and Design of the Platform*.

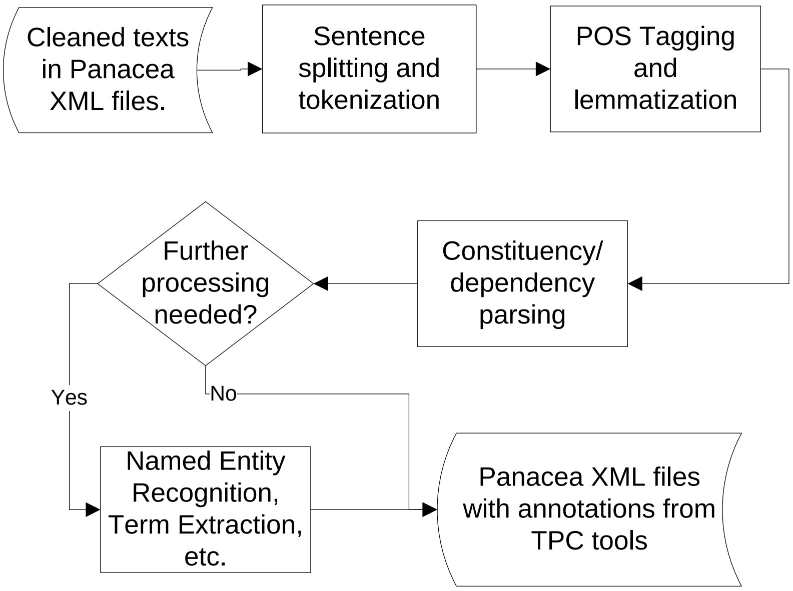

Cleaned texts in Panacea XML files.
Sentence splitting and tokenization
POS Tagging and lemmatization
Constituency/ dependency parsing
Further processing needed?
Yes
No
Named Entity Recognition, Term Extraction, etc.
Panacea XML files with annotations from TPC tools

Figure 9 A typical text processing workflow

7.4 Tools to be deployed as web services in the PANACEA factory

In subsections 7.1, 7.2 and 7.3 we proposed workflows for the tasks of corpus acquisition, corpus clean-up and normalization, and text processing. These workflows will be realized in the PANACEA Factory by a set of tools that will be deployed as web services. Table 8 lists these tools and the functionalities they will offer. The table includes information about the partner that will deploy and host each tool, and the month the tool will be available as a web service, ready to be integrated in the PANACEA factory. All tools must meet the general requirements stated in section 3.1 of *D8.1 User Requirements*. Moreover, the table includes references to specific requirements from section 3.2 of *D8.1* that each tool addresses. Finally, it should be noted that during the project's timeline some web services will be available to PANACEA partners only. Availability of the web services to users outside the consortium will be the subject of task WP2.1, *Exploitation*.

Functionality	**Tool to be deployed as a WS**	**Host of the WS**	**Month of delivery as a WS**	**License**	**Corresponding D8.1 requirements**
Monolingual Crawler	Monolingual Crawler	ILSP	T12	Proprietary	Req-TOL-100, Req-TOL-111, Req-TOL-120
Bilingual Crawler	Bilingual Crawler	ILSP	T12	Proprietary	Req-TOL-100, Req-TOL-110, Req-TOL-111, Req-TOL-120
Boilerplate removal	Boilerplate removal tool	ILSP/DCU	T14	Proprietary	Req-TOL-100, Req-TOL-121
Duplicate detection	Duplicate detection tool	ILSP/DCU	T14	Proprietary	
Sentence Splitter for EL	ILSP Sentence Splitter and Tokenizer	ILSP	T14	Proprietary	Req-TOL-100, Req-TOL-130
Tokenizer for EL	ILSP Sentence Splitter and Tokenizer	ILSP	T14	Proprietary	Req-TOL-100, Req-TOL-150
POS Tagger and Lemmatizer for EL	ILSP FBT Tagger & ILSP Lemmatizer	ILSP	T14	Proprietary	Req-TOL-100
Chunker for EL	ILSP Chunker	ILSP	T20	Proprietary	Req-TOL-100
Sentence Splitter for DE	LT-SentenceSegmentiser	LT	T14	Proprietary	Req-TOL-100, Req-TOL-130

Sentence Splitter for EN	LT-SentenceSegmentiser	LT	T14	Proprietary	Req-TOL-100, Req-TOL-130
Tokeniser for DE	LT-Tokeniser	LT	T14	Proprietary	Req-TOL-100, Req-TOL-150
Tokeniser for EN	LT-Tokeniser	LT	T14	Proprietary	Req-TOL-100, Req-TOL-150
Lemmatiser for DE	LT-Lemmatiser	LT	T14	Proprietary	Req-TOL-210
Lemmatiser for EN	LT-Lemmatiser	LT	T14	Proprietary	Req-TOL-210
Decomposer for DE	LT-Lemmatiser	LT	T14	Proprietary	Req-TOL-210
TopicIdentifier for DE	LT-TopicIdentifier	LT	T14	Proprietary	Req-TOL-210
Term Extraction (mono) DE	LT-TermExtract	LT	T22	Proprietary	Req-TOL-210
Term Extraction (mono) EN	LT-TermExtract	LT	T22	Proprietary	Req-TOL210
Term Extraction (biling.) EN/DE	LT-BiExtract	LT	T30	Proprietary	Req-TOL-230
NE recognition DE	LT-Namer	LT	T30	Proprietary	Req-TOL-260
NE recognition EN	LT-Namer	LT	T30	Proprietary	Req-TOL-260
Document and sentence segmentation for IT	Syn SG **	CNR-ILC	T14	Proprietary	Req-TOL-100, Req-TOL-130
Tokenizer for IT	Syn SG **	CNR-ILC	T14	Proprietary	Req-TOL-100, Req-TOL-150

POS Tagger and Lemmatizer for IT	Syn SG **	CNR-ILC	T14	Proprietary	Req-TOL-100
Chunker for IT	Syn SG **	CNR-ILC	T20	Proprietary	Req-TOL-100
Dependency Parser for IT	Syn SG **	CNR-ILC	T20	Proprietary	Req-TOL-100
Sentence Splitter for ES	IULA Preprocessing tool	UPF	T14	free of charge, non-commercial	Req-TOL-100
POS Tagger and Lemmatizer for ES	IULA POS Tagger	UPF	T14	free of charge, non-commercial	Req-TOL-100
Sentence splitter and tokeniser for EN, FR, DE, ES	Europarl tools	DCU	T18	free of charge, non-commercial	Req-TOL-100, Req-TOL-130, Req-TOL-150
PoS tagger and parser for EN, FR, DE	Berkeley	DCU	T18	free of charge, non-commercial	Req-TOL-100
Tokenizer for EN	RASP	UCAM	T18	Free for research purposes	Req-TOL-100, Req-TOL-150
POS tagger for EN	RASP	UCAM	T18	Free for research purposes	Req-TOL-100
Lemmatiser for EN	RASP	UCAM	T18	Free for research	Req-TOL-100

				purposes	
Parser for EN	RASP	UCAM	T18	Free for research purposes	Req-TOL-100

Table 8 List of tools that will be deployed as web services

8 Workplan

Following a) the PANACEA Description of Work document and b) the evaluation workplan as detailed in D7.1 *Criteria for evaluation of resources, technology and integration*, the workplan for WP4 will include the tasks below:

- T13: **D4.2.** Initial functional prototype and documentation. The initial prototype will consist of a web service for monolingual and a web service for bilingual web crawling. The two web services will include modules for language identification and boilerplate removal. As required for *D7.2 First evaluation report* (T14).

- T13: **D4.3.** Version 1 of the monolingual corpora of English, Spanish, Italian, French and Greek in the environment and labor legislation domain. Each corpus will contain raw text, acquired from the web and cleaned. The texts will be stored in the common encoding format documented in *D3.1 Architecture and Design of the Platform.* As required for *D7.2 First evaluation report* (T14). We expect the final version of each corpus to consist of 1M tokens.

- T18: **Internal deliverable.** Partners adapt NLP tools focusing on sentence splitting/tokenization and POS tagging/lemmatization for EN, DE, EL, ES, IT, FR as detailed in Table 8. Partners will make sure that I/O of the tools is conformant with the common encoding format documented in *D3.1 Architecture and Design of the Platform.* These tools will be used in the production of the second version of the monolingual corpora.

- T20: **Internal deliverable**. Version 2 of the monolingual corpora of English, Spanish, Italian, French and Greek annotated for POS and lemma. Result of the 2nd development cycle after the first evaluation cycle.

- T21: **D4.4.** 2nd version of the prototype and documentation. The revised prototype will integrate dedicated web services for normalization (including boilerplate removal and duplicate document detection). As required for *D7.3 Second evaluation report* (T22).

- T21: **Internal deliverable.** Version 1 of the bilingual corpora of EN-FR, EN-EL in the environment and labor legislation domain, annotated for POS and lemma. We expect the final version of each corpus to contain enough data so that parallel sentences of 250-500 K tokens for each language of each pair can be extracted. As required for the second evaluation cycle detailed in *D7.1 Criteria for evaluation of resources, technology and integration*.

- T22: **Internal deliverable.** Partners adapt NLP tools focusing on parsing and/or chunking for DE, EN, EL, ES, IT, FR as detailed in Table 8. Partners will make sure that I/O of the tools is conformant with the common encoding format documented in *D3.1 Architecture and Design of the Platform.* These tools will be used in the production of the third version of the monolingual corpora and will be part of the final version of the WP4 prototype.

- T28: **Internal deliverable.** Version 3 of the monolingual corpora of English, Spanish, Italian, French and Greek with syntactic annotations. Result of the 3rd development cycle after the first evaluation cycle.

- T29: **Internal deliverable.** Version 2 of the bilingual corpora of EN-FR, EN-EL in the environment and labor legislation domain, annotated with syntactic annotations. As required for the third evaluation cycle detailed in *D7.1 Criteria for evaluation of resources, technology and integration*.

- T29: **D4.5.** Final version of the prototype and documentation. The revised prototype will integrate NLP tools for a) sentence splitting/tokenization b) POS tagging/lemmatization and c) parsing for EN, DE, EL, ES, IT, FR as detailed in Table 8. As required for *D7.4 Third evaluation report* (T30).

9 References

9.1 Corpus acquisition and clean-up components

Ardo, A. 2005. Combine web crawler, Software package for general and focused Web-crawling, http://combine.it.lth.se/.

Ardo, A., and Golub, K. 2007. Documentation for the Combine (focused) crawling system, http://combine.it.lth.se/documentation/DocMain/

Aslam, J. A., and Frost, M. 2003. An information-theoretic measure for document similarity. In Proceedings of the 26th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR 2003).

Baroni, M., and Bernardini, S. 2004. BootCaT: Bootstrapping corpora and terms from the web. In Proceedings of LREC 2004. 1313-1316.

Baroni, M., Kilgarriff, A., Pomikalek J., and Rychly. P. 2006. WebBootCaT: a web tool for instant corpora. In Proceedings of Euralex 2006. 123-132.

Bauer, D., Degen, J., Deng, X., Herger, P., Gasthaus, J., Giesbrecht, E., Jansen, L., Kalina, C., Krüger, T., Märtin, R., Schmidt, M., Scholler, S., Steger, J., Stemle, E., and Evert, S. 2007. Fiasco: Filtering the internet by automatic subtree classification. In Proceedings of the Web as Corpus Workshop (WAC3), Cleaneval Session.

Bergmark, D., Lagoze, C., and Sbityakov, A. 2002. Focused crawls, tunneling, and digital libraries. In Proc. of the 6th European Conference on Research and Advanced Technology for Digital Libraries. 91–106.

Brin, S., and Page, L. 1998. The anatomy of a large-scale hypertextual Web search engine, Computer Networks and ISDN Systems. 30, 1–7, 107–117.

Broder, A. Z., Glassman, S. C., Manasse, M. S., and Zweig, G. 1997. Syntactic clustering of the Web. Computer Networks. 29, 8-13.

Cavnar, W., and Trenkle, J. M. 1994. N-gram-based text categorization. In Proceedings of Symposium on Document Analysis and Information Retrieval.

Chakrabarti, S., Joshl, M. M., Punera, K., and Pennock, D. M. 2002. The structure of broad topics on the Web. In Proceedings of the 11th International Conference on World Wide Web (WWW). ACM Press, New York, NY, 251–262.

Chen, L,. Ye, S., and Li, X. 2006. Template detection for large scale search engines. In Proceedings of the 2006 ACM Symposium on Applied computing.

Cho, J., Garcia-Molina, H., and Page, L. 1998. Efficient crawling through URL ordering, Computer Networks and ISDN Systems. 30, 1–7, 161–172.

Chowdhury, A., Frieder, O., Grossman, D., and McCabe, M. C. 2002. Collection statistics for fast duplicate document detection. ACM Transactions on Information Systems. 20, 2, 171-191.

Cooley, R., Mobasher, B., and Srivastava, J. 1999. Data preparation for mining world wide web browsing patterns. Knowledge and Information Systems. 1, 1, 5-32.

Davison, B. D. 2000. Topical locality in the Web. In Proceedings of the 23rd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval. ACM Press, New York, NY, 272–279.

Diligenti, M., Coetzee, F., Lawrence, S., Giles, C. L., and Gori, M. 2000. Focused crawling using context graphs, in Proceedings of the 26th International Conference on Very Large Databases, 527–534.

Dorado, I. G. 2008. Focused Crawling: algorithm survey and new approaches with a manual analysis. Master thesis.

Dunning, T. 1994. Statistical identification of language. Technical Report CRL MCCS-94-273, Computing Research Lab, New Mexico State University.

Espla-Gomis, M., and Forcada, M.L. 2010. Combining Content-Based and URL-Based Heuristics to Harvest Aligned Bitexts from Multilingual Sites with Bitextor. The Prague Bulletin of Mathematical Linguistics. 93, 77-86.

Evert, S. 2008. A lightweight and efficient tool for cleaning Web pages. In Proceedings of the 6th International Conference on Language Resources and Evaluation (LREC 2008).

Gionis, A., Indyk, P., and Motwani, R. 1999. Similarity search in high dimensions via hashing. In Proceedings of the 25th International Conference on Very Large Data Bases. 518-529.

Golub, K., and Ardo, A. 2005. Importance of HTML structural elements and metadata in automated subject classification. In Proceedings of the 9th European Conference on Research and Advanced Technology for Digital Libraries (ECDL). Lecture Notes in Computer Science, vol. 3652. Springer, Berlin, pp. 368–378.

Grefenstette, G. 1995. Comparing two language identification schemes. In Proceedings of Analisi Statistica dei Dati Testuali (JADT).

Hersovici, M., Jacovi M., Maarek, Y. S., Pelleg, D., Shtalhaim, M., and Ur S. 1998. The sharksearch algorithm—An application: TailoredWeb site mapping, Computer Networks and ISDN Systems, 30, 1–7, 317-326.

Hofmann, K., and Weerkamp, W. 2007. Web corpus cleaning using content and structure. In Proceedings of the Web as Corpus Workshop (WAC3), Cleaneval Session.

Hughes, B., Baldwin, T., Bird, S., Nicholson, J., and MacKinlay, A. 2006. Reconsidering language identification for written language resources. In Proceedings of the 5th International Conference on Language Resources and Evaluation (LREC2006).

Joachims T. 1997. A probabilistic analysis of the Rocchio algorithm with tfidf for text categorization. In International Conference on Machine Learning (ICML).

Joachims, T. 1998. Text categorization with support vector machines: learning with many relevant features, in Proceedings of the 10th European Conference on Machine Learning. 1398, 137–142.

Kohlschütter, C., Fankhauser, P., and Nejdl, W. 2010. Boilerplate Detection using Shallow Text Features. The Third ACM International Conference on Web Search and Data Mining.

Kwon, O.-W., and Lee, J.-H. 2003. Text categorization based on k-nearest neighbor approach for Web site classification. Information Processing and Management. 29, 1, 25–44.

Lee, M.-L., Ling, T. W., and Low, W. L. 2000. Intelliclean: a knowledge-based intelligent data cleaner. In Knowledge Discovery and Data Mining.

Lin S.-H., and Ho J.-M. 2002. Discovering informative content blocks from web documents. In Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD-2002).

Marek, M., Pecina, P., and Spousta, M. 2007. Web Page Cleaning with Conditional Random Fields. In Proceedings of the Web as Corpus Workshop (WAC3), Cleaneval Session.

Menczer, F. 2005. Mapping the semantics of Web text and links. IEEE Internet Computing 9, 3, 27–36.

Menczer, F., Pant, G. and Srinivasan, P. 2004. Topical Web Crawlers: Evaluating Adaptive Algorithms, ACM Transactions on Internet Technology, Vol. 4, No. 4, pp. 378–419.

Mohr, G., Stack, M., Ranitovic, I., Avery, D., and Kimpton, M. 2004. An Introduction to Heritrix, 4th International Web Archiving Workshop.

Passerini, A., Frasconi, P., and Soda, G. 2001. Evaluation methods for focused crawling, Lecture Notes in Computer Science. 2175, 33–45.

Pinkerton, B. 1994. Finding what people want: Experiences with the Web Crawler. In Proceedings of the 2nd International World Wide Web Conference.

Poutsma, A. 2001. Applying Monte Carlo Techniques to Language Identification. In Proceedings of Computational Linguistics in the Netherlands (CLIN 2001).

Qi, X., and Davison, B. D. 2009. Web Page Classification: Features and Algorithms. ACM Computing Surveys. 41, 2, 12.

Quinlan, J. R. 1993. C4.5: Programs for Machine Learning. Morgan Kaufmann.

Roche, X. 2007. http://www.httrack.com.

Spousta, M., Marek, M., and Pecina, P. 2008. Victor: the Web-Page Cleaning Tool. In Proceedings of the 4th Web as Corpus Workshop (WAC4), - Can we beat Google? (LREC 2008).

Sun, A., Lim, E.-P. and Ng, W.-K. 2002. Web classification using support vector machine. In Proceedings of the 4th International Workshop on Web Information and Data Management (WIDM). ACM Press, New York, NY, 96–99.

Teytaud, O., and Jalam, R. 2001. Kernel-based text categorization. In Proceedings of the International Joint Conference on Neural Networks (IJCNN’2001).

Theobald, M., Siddharth, J., and Paepcke, A. 2008. SpotSigs: Robust and Efficient Near Duplicate Detection in Large Web Collections. In: 31st annual international ACM SIGIR conference on Research and development in information retrieval (SIGIR 2008).

Utard, H., and Furnkaranz, J. 2005. Link-local features for hypertext classification. In Semantics, Web and Mining: Joint International Workshops, EWMF/KDO. Lecture Notes in Computer Science, vol. 4289. Springer, Berlin, Germany, 51–64.

Yang Y. 1997. An evaluation of statistical approaches to text categorization. Technical Report CMU-CS-97-127, Carnegie Mellon University.

Yi, L., and Liu, B. 2003. Web page cleaning for web mining through feature weighting. In Proceedings of 18th International Joint Conference on Artificial Intelligence (IJCAI-03).

Yi, L., Liu, B., and Li X. 2003. Eliminating noisy information in web pages for data mining. In Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD-2003).

Ziv, B.-Y., and Rajagopalan, S. 2002. Template detection via data mining and its applications. In Proceedings of the 11th international conference on World Wide Web. 580-591.

9.2 Text Processing Component

English

Atserias, J., Casas, B., Comelles, E., González, M., Padró, L., and Padró, M. 2006. FreeLing 1.3: Syntactic and semantic services in an open-source NLP library. In the Proceedings of LREC. Genoa, Italy.

Briscoe, T., Carroll J., and Watson, R. 2006. The second release of the RASP system. In the Proceedings of the COLING/ACL interactive presentation sessions. Sydney, Australia.

Cer, D., de Marneffe M.-C., Jurafsky D., and Manning C. D. 2010. Parsing to Stanford dependencies: Trade-offs between speed and accuracy. In the Proceedings of LREC.

Clark, S., Copestake, A., Curran, J. R., Zhang, Y., Herbelot, A., Haggerty, J., Ahn, B.-G., Wyk, C. V., Roesner, J., Kummerfeld, J., and Dawborn, T. 2009. Large-scale syntactic processing: Parsing the web. Final report of the JHU CLSP Workshop, Johns Hopkins University.

Curran, J.R., and Clark, S. 2003. Investigating GIS and smoothing for maximum entropy taggers. In the Proceedings of EACL, pp.91-98.

de Marneffe, M.-C., MacCartney, B., and Manning C. D. 2006. Generating Typed Dependency Parses from Phrase Structure Parses. In the Proceedings of LREC.

Elworthy, D. 1994. Does Baum-Welch re-estimation help taggers? In the Proceedings of the Fourth Conference on Applied Natural Language Processing, pp. 53-58.

Grover, C., Matheson, C., Mikheev, A., and Moens, M. 2000. LT TTT - A Flexible Tokenisation Tool. In the Proceedings of LREC.

Jurafsky, D., and Martin, J. 2000. Speech and Language Processing, Prentice-Hall.

Kiss, T., and Strunk, J. 2006. Unsupervised multilingual sentence boundary detection. Computational Linguistics 32(4):485-525.

Klein, D., and Manning, C. D. 2003. Accurate Unlexicalized Parsing. In the Proceedings of ACL, pp. 423-430.

Minnen, G., Carroll, J., and Pearce D. 2000. Robust, applied morphological generation. In the Proceedings of INLG.

Petrov, S., Barrett, L., Thibaux, R., and Klein, D. 2006. Learning accurate, compact, and interpretable tree annotation. In the Proceedings of COLING-ACL.

Toutanova, K., Klein, D., Manning, C., and Singer, Y. 2003. Feature-Rich Part-of-Speech Tagging with a Cyclic Dependency Network. In Proceedings of HLT-NAACL, pp. 252-259.

French

Adda, G., Lecomte, J., Mariani, J., Paroubek, P., and Rajman, M. 1998. The GRACE French Part-of-Speech Tagging Evaluation Task. In the Proceedings of LREC. Granada, Spain.

Allauzen, A., and Bonneau-Maynard, H. 2008. Training and evaluation of POS taggers on the French multitag corpus. In the Proceedings of LREC.

Brill, E. 1995. Transformation-based error-driven learning and natural language processing: a case study in part-of-speech tagging. Computational Linguistics. 21 (4)

Villemonte de la Clergerie, E., Hamon, O., Mostefa, D., Ayache, C., Paroubek, P. and Vilnat, A. 2008. PASSAGE : from French Parser Evaluation to Large Sized Treebank. In Proceedings of LREC.

Jongejan, B., and Dalianis, H. 2009. Automatic training of lemmatization rules that handle morphological changes in pre-, in- and suffixes alike. In the Proceedings of ACL-AFNL. Suntec, Singapore.

Paumier, S. 2010. UNITEX 2.1 User Manual. http://www-igm.univ-mlv.fr/~unitex/UnitexManual2.1.pdf. Visited 2010-06-10

Paroubek, P., Robba, I., Vilnat, A., and Ayache, C. 2006. Data Annotations and Measures in EASY the Evaluation Campaign for Parsers of French. In the Proceedings of LREC. Genoa, Italy.

Schmid H. 1997. Probabilistic part-of-speech tagging using decision trees. In D. Jones & H. Somers (eds.) New Methods in Language Processing Studies in Computational Linguistics.

German

Benden, C., and Hermes, J. 2004. Präprozessierung mit Nebenwirkungen: Dynamische Annotation. In Buchberger, Ernst (ed.), Konferenz zur Verarbeitung natürlicher Sprache (KONVENS), Wien

Brants, T. 2000. TnT - A Statistical Part-of-Speech Tagger. In the Proceedings of the Sixth Applied Natural Language, Processing Conference ANLP-2000, Seattle, WA.

Florian, R., Ittycheriah, A., Jing, H, and Zhang, T. 2003. Named entity recognition through classifier combination. In the Proceedings of the Seventh conference on natural language learning at HLT-NAACL. Edmonton, Canada.

Fraser, A., Wang, R., and Schütze, H. 2009. Rich bitext projection features for parse reranking. In EACL.

Gurevych, I., Mühlhäuser, M., Müller, C., Steimle, J., Weimer, M., Zesch, T. 2007. Darmstadt Knowledge Processing Repository Based on UIMA. In the Proceedings of the First Workshop on Unstructured Information Management Architecture at Biannual Conference of the Society for Computational Linguistics and Language Technology.

Jongejan, B., and Dalianis, H. 2009. Automatic training of lemmatization rules that handle morphological changes in pre-, in- and suffixes alike. In Proc of ACL-AFNL. Suntec, Singapore.

Jongejan, B., and Haltrup, D. 2005. The CST Lemmatiser. Center for Sprogteknologi, University of Copenhagen

Kübler, S., and Penn, G. (eds). In the Proceedings of the Workshop on Parsing German. 2008. Columbus, Ohio.

Kübler, S., and Prokic, J. 2006. Why is German Dependency Parsing more Reliable than Constituent Parsing? In the Proceedings of the Fifth International Workshop on Treebanks and Linguistic Theories, Prague, Czech Republic.

Kübler, S., Hinrichs, E. W., Maier, W. 2006. Is it Really that Difficult to Parse German? In the Proceedings of the Conference on Empirical Methods in Natural Language Processing, EMNLP 2006, Sydney, Australia.

Palmer, D.D., and Hearst, M. A. 1997. Adaptive multilingual sentence boundary disambiguation. Computational Linguistics, 23, 2, 241–269.

Perera, P., and Witte, R. 2005. A Self-Learning Context-Aware Lemmatizer for German. Human Language Technology Conference/Conference on Empirical Methods in Natural Language Processing (HLT/EMNLP 2005). Vancouver, Canada. pp. 636–643.

Petrov, S., and Klein, D. 2007. Improved Inference for Unlexicalized Parsing, In the Proceedings of HLT-NAACL.

Petrov, S., Barrett, L., Thibaux, R., and Klein, D. 2006. Learning accurate, compact, and interpretable tree annotation. In the Proceedings of COLING-ACL.

Schmid, H. 1994. Probabilistic Part-of-Speech Tagging Using Decision Trees. In Proceedings of the International Conference on New Methods in Language Processing (NeMLaP-1), pages 44–49.

Schmid, H. 1995. Improvements in Part-of-Speech Tagging with an Application to German. Proceedings of the ACL SIGDAT-Workshop.

Schmid, H. 2000. LoPar: Design and Implementation. Number 149 in Arbeitspapiere des Sonderforschungsbereiches 340. Institute for Computational Linguistics, University of Stuttgart.

Schmid, H. 2004. Efficient parsing of highly ambiguous context-free grammars with bit vectors. In COLING.

Schmid, H., and Laws, F. 2008. Estimation of Conditional Probabilities with Decision Trees and an Application to Fine-Grained POS Tagging, COLING 2008, Manchester, Great Britain

Schneider, G., and Volk, M. 1998. Adding Manual Constraints and Lexical Look-up to a Brill-Tagger for German. In*:* ESSLLI-98 Workshop on Recent Advances in Corpus Annotation.

Greek

Boutsis, S., Prokopidis, P., Giouli, V., and Piperidis, S. 2000. A Robust Parser for Unrestricted Greek Text. In the Proceedings of the 2nd LREC. Athens, Greece.

Dermatas, E., and Kokkinakis, G. 1995. Automatic stochastic tagging of natural language texts Computational Linguistics, MIT Press, 21, 137-163.

Giouli, V., Konstadinidis, A., Desipri, E., Papageorgiou, H., and Piperidis, S. 2006. Multi-domain Multi-lingual Named Entity Recognition: Re-visiting & grounding the resources issue. In the Proceedings of the 5th LREC. Genoa, Italy.

Karkaletsis, V., Paliouras, G., Petasis, G., Manousopoulou, N., and Spyropoulos, C.D. 1999. Named-Entity Recognition from Greek and English Texts. Journal of Intelligent and Robotic Systems, 26, 123-135.

Lucarelli, G.; Vasilakos, X. & Androutsopoulos, I. 2007. Named Entity Recognition in Greek Texts with an Ensemble of SVMs and Active Learning International Journal on Artificial Intelligence Tools, 16, 1015-1045.

Maragoudakis, M., Kermanidis, K. & Fakotakis, N. 2003. A Bayesian Part-Of-Speech and Case Tagger for Modern Greek. Proc. of the 6th International Conference of Greek Linguistics. Rethymno, Greece.

Maragoudakis, M.; Fakotakis, N. & Kokkinakis, G. 2004. A Bayesian Model for Shallow Syntactic Parsing of Natural Language Texts. Proc of the 4th LREC. Lisbon, Portugal.

Orphanos, G. S. & Christodoulakis, D. N. 1999. POS disambiguation and unknown word guessing with decision trees. Proc. of the 9th EACL. Bergen, Norway.

Papageorgiou, H., Prokopidis, P., Giouli, V. & Piperidis. S. 2000. A Unified Tagging Architecture and its Application to Greek. Proc. of the 2nd LREC. Athens, Greece.

Petasis, G., Paliouras, G., Karkaletsis, V., Spyropoulos, C. D. & Androutsopoulos, I. 1999. Resolving Part-of-Speech Ambiguity in the Greek Language Using Learning Techniques. The Computing Research Repository, cs.CL/9906019.

Stamatatos, E.; Fakotakis, N. & Kokkinakis, G. 1999. Automatic Extraction of Rules for Sentence Boundary Disambiguation. Proc. ACAI Workshop on Machine Learning in Human Language Technology.

Italian

Aliprandi, C., Carmignani, N., Deha, N., Mancarella, P., and Rubino, M. 2007. UniPiSynthema POS Tagger Description and Evaluation. In Intelligenza Artificiale, Special Issue on NLP Tools for Italian, IV-2.

Attardi, G. 2006. Experiments with a multilanguage non-projective dependency parser. In the Proceedings of the Tenth CoNLL.

Attardi, G. et al. 2009. Tanl (Text Analytics and Natural Language Processing): Analisi di Testi per il Semantic Web e il Question Answering, http://medialab.di.unipi.it/ wiki/SemaWiki. Evalita 2009.

Baroni, M., Schmid, H., Zanchetta, E., and Stein, A. 2007. The Enriched TreeTagger System. In Intelligenza Artificiale, Special Issue on NLP Tools for Italian, IV-2: 22-23.

Ciaramita M., Atserias, J. 2007. Named Entity Tagging with a PoS Tagger. In Intelligenza Artificiale, Special Issue on NLP Tools for Italian, IV-2.

Dell'orletta, F. et al. 2007. Maximum Entropy for Italian Pos tagging. Evalita 2007

Halácsy, P., Kornai, A., Oravecz, C. 2007. HunPos – an open source trigram tagger. In the Proceedings of the Demo and Poster Sessions of the 45th Annual Meeting of the ACL, pp. 209-212.

Lavelli, A., and Corazza, A. 2009. The Berkeley Parser at the EVALITA 2009 Constituency Parsing Task.

Lavelli, A., Hall, J., Nilsson, J., and Nivre, J. 2009. MaltParser at the EVALITA 2009 Dependency Parsing Task.

Lesmo, L., and Lombardo, V. 2002. Transformed subcategorization frames in chunk parsing. In the Proceedings of the 3rd Conference on Language Resources and Evaluation, Las Palmas.

Mehdad, Y., Scurtu, V., and Stepanov, E. 2009. Italian Named Entity Recognizer Participation in NER task @ Evalita 09.

Pianta, E., and Zanoli, R. 2007a. TagPro: A system for Italian PoS tagging based on SVN. In Intelligenza Artificiale, Special Issue on NLP Tools for Italian, IV-2.

Pianta, E., and Zanoli, R. 2007b. Exploiting SVM for Italian Named Entity Recognition. In Intelligenza Artificiale, Special Issue on NLP Tools for Italian, IV-2.

Romagnoli, S. 2007. C4: HMM POS Tagger with POS guesser and external lexicon. Evalita 2007.

Sangati, F. 2009. A simple DOP model for constituency parsing of Italian sentences. Evalita 2009.

Schmid, H. 1994. Probabilistic Part-of-Speech Tagging Using Decision Trees. In: Proceedings of the International Conference on New Methods in Language Processing, pp. 44-49.

Shen, L. 2006. Statistical LTAG Parsing Ph.D. thesis. University of Pennsylvania.

Tamburini, F. 2007. CORIStagger: a high-performance Pos tagger for Italian. In Intelligenza Artificiale, IV(2), 14-15.

Spanish

Ferrández, A., Palomar, M., Moreno, L.: Anaphor Resolution in Unrestricted Texts with Partial Parsing. 1998. In the Proceedings of COLING¬ACL

Ferrández, S., and Peral, J. 2005. Investigating the Best Configuration of HMM Spanish PoS Tagger when Minimum Amount of Training Data Is Available. In the Proceedings of the 10th International Conference on Applications of Natural Language to Information Systems, Alicante, Spain.

Giménez, J., and Màrquez, L. 2004. SVMTool: A general POS tagger generator based on Support Vector Machines". In the Proceedings of the 4th International Conference on Language Resources and Evaluation. Lisbon, Portugal, 2004

Iglesias, A, Castro, E, Pérez, R., Castaño, L., Martínez, P., Pérez, J. M. G., Kohler, S., and Melero, R. 2008. MOSTAS: Un Etiquetador Morfo-Semántico, Anonimizador y Corrector de Historiales Clínicos. In the Proceedings of SEPLN.

Kozareva, Z, Ferrández O., Montoyo, A., Muñoz R., Suarez A. and Gómez J. 2007. Combining data-driven systems for improving Named Entity Recognition. Data & Knowledge Engineering. 61:3

Martínez, H., Vivaldi, J, and Villegas, M. 2010. Text handling as a Web Service for the IULA processing pipeline. In the Proceedings of LREC.

Moreno, A., and Guirao, J.M. 2006. Morpho-syntactic Tagging of the Spanish C-ORAL-ROM Corpus: Methodology, Tools and Evaluation. In Spoken Language Corpus and Linguistic Informatics, John Benjamins.

Schmid, H. 1994. Probabilistic Part-of-Speech Tagging Using Decision Trees. In Proceedings of the International Conference on New Methods in Language Processing (NeMLaP-1), pp. 44–49.

Vivaldi P.J. 2009. Corpus and exploitation tool: IULACT and bwanaNet. In *A survey on corpus-based research*. Universidad de Murc

Appendix

A. Template used for documenting NLP tools

Administrative information	
* Tool name	The full name of the tool.
Short name	
* Short description	Provide a short description of a) what the tool does b) the type of data it processes (raw/annotated, mono/bi/multi-lingual, parallel data) and c) which technologies and approaches it is based on.
* Organization name	
* Latest version number and release date	
Tool web page	
* Contact person in the context of PANACEA (i.e. person responsible for the integration of the tool in the factory)	
* Contact person's email	
* Technical report or publication relevant to the application	A URL to a document providing detailed documentation of the tool
Relevant project(s)	Optional information on projects related to the development and expansion of the tool
* License and availability	Provide a reference to a well-known (e.g. GPL) or proprietary license that applies to the tool. Optionally, provide also an availability description (e.g. free for research purposes, free for development of new commercial and non-commercial resources and applications, etc.)
Descriptive information	
* Languages covered	English French German Greek Italian Spanish Other: specify

* Character encoding (input)	For example, Unicode (UTF-8) Unicode (UTF-16) ISO-8859-7 …
* Character encoding (output)	For example, Unicode (UTF-8) Unicode (UTF-16) ISO-8859-7 …
* Format (input)	Txt, HTML, XML (specify schema if a standard is used), …
* Format (output)	Txt, HTML, XML (specify schema if a standard is used), …
* Compatibility of the input and/or output data with national/international standards/common practices	Include information on encoding and linguistic representation of I/O data (e.g. CLAWS, EAGLES, MULTEXT tagsets for morphosyntactic annotation, TMX format for alignment etc.) - specify name of standard and degree of compatibility.
* Language resources required for the operation of the application	For example, 1) annotated data users must provide to train the tool 2) Resources built-in in the tool like a precompiled model or a lexicon in a database that the tool has access to 3) Combinations of 1 and 2 above. For example, the tool has its internal resources, which can be complemented with additional user resources Please provide information on size, coverage, and any other relevant details concerning these data.
* Operating system	OS independent Linux/Unix Mac OS

	Windows …
* Implementation language	C/C++ Java Perl Python Ruby …
* Other software requirements	A list of any other libraries needed by this tool
URL providing access to the tool as a web service	
URL providing guidelines on accessing the tool as a web service	
Hardware requirements	Provide an indicative system configuration**, in case** this tool requires more than a typical modern workstation to operate reasonably fast
* Processing speed	Provide an indication like tokens/sec, etc.
Evaluation information	
* Methodology and reference data	Free text. Provide information on the methodology and the metrics used for the tool evaluation. Also, provide information on the type and size of reference data used.
* Results	Free text.

Starred fields in the template denote obligatory elements.

Apart from the free-text descriptions on input and output in the form above, please provide a small set of sample output from the relevant tool. For WP4.3 tools, we propose using the following (quasi-parallel) sentences from the http://europa.eu site.

Input

English

```
Haiti on our minds.
Commission calls for €90m more in aid for the quake-stricken
country, to be drawn from EU emergency funds.
```

French

```
L'Union n'oublie pas Haïti.
```

```
La Commission souhaite octroyer une nouvelle aide de 90 millions
d'euros à ce pays ravagé par un tremblement de terre.
```

German

```
Wir haben Haiti nicht vergessen.
Kommission fordert weitere 90 Millionen Euro aus EU-Soforthilfefonds
für Hilfsmaßnahmen in dem erdbebengeschädigten Land.
```

Greek

```
Δεν ξεχνάμε την Αϊτή.
Επιπλέον βοήθεια 90 εκατ. ευρώ από τα κονδύλια έκτακτης ανάγκης της ΕΕ
προτείνει η Επιτροπή για τη σεισμόπληκτη χώρα.
```

Italian

```
Non ci dimentichiamo di Haiti.
La Commissione propone di attingere ai fondi di emergenza
dell'UE per inviare altri 90 milioni di euro al paese colpito
dal sisma.
```

Spanish

```
La UE enviará más ayuda a Haití.
La Comisión pide otros 90 millones de euros de los fondos de
emergencia europeos.
```

B. Linguatec Decomposer for German Compounds

Administrative information	
* Tool name	LT-Decomposer
Short name	Decomposer
* Short description	Decomposes German compounds into their parts
* Organization name	Linguatec
* Latest version number and release date	--
Tool web page	--
* Contact person in the context of PANACEA (i.e. person responsible for the integration of the tool in the factory)	Gr. Thurmair
* Contact person's email	g.thurmair@linguatec.de
* Technical report or publication relevant to the application	(Thurmair 1986)
Relevant project(s)	PANACEA, ACCURAT, EASTIN-CL
* License and availability	on a case-by-case basis
Descriptive information	
* Languages covered	German
* Character encoding (input)	Unicode (UTF-8)
* Character encoding (output)	Unicode (UTF-8)
* Format (input)	a word (textform) / compound candidate. (Function words etc. would have been filtered out by previous components)
* Format (output)	decomposition result(s), one line per decomposition, giving infomraiton on compound parts
* Compatibility of the input and/or output data with national/international standards/common practices	simple POS tags (standard tagset, open classes)

* Language resources required for the operation of the application	a. decomposer dictionary, similar to lemmatiser dictionary, a compiled form of <textform, lemma, POS> triplets (POS = standard tagset>, plus some compounding information (infixes etc.). Several 100K entries. b. rule set for disambiguation of multiple decompositions
* Operating system	Windows
* Implementation language	Java
* Other software requirements	--
URL providing access to the tool as a web service	--
URL providing guidelines on accessing the tool as a web service	--
Hardware requirements	Standard PC
* Processing speed	30K words per second
Evaluation information	
* Methodology and reference data	Manual comparison of decomposition results.
* Results	Previous version was evaluated best by a German publishing company.

Short comment

1. The decomposer usually results in several decomposition options. It applies heuristics in such cases to select the best alternative.

2. Sources of error are proper names which happen to be decomposable, and notoriously tricky cases involving ambiguous affixes ('*-los*', '*-bar*'). The more frequent ones of those are treated in a lexicon-based way. However, they can always be solved by putting the thing into the decomposer dictionary.

3. There is an uncertainty where the boundary between composition and derivation should be drawn, esp. wrt verb prefixes, but also very productive suffixes like '*un*-', '*-bar*', '*-gemäß*' etc.

Sample Input

```
Wir haben Haiti nicht vergessen.
Kommission fordert weitere 90 Millionen Euro aus EU-Soforthilfefonds
für Hilfsmaßnahmen in dem erdbebengeschädigten Land.
```

Some examples:

Hilfsflugplatz	NOUN	[hilfe (N) + flugplatz (N)]
Hilfsflugzeug	NOUN	[hilfe (N) + flugzeug (N)]
Hilfsfonds	NOUN	[hilfe (N) + fonds (N)]
Hilfsformat	NOUN	[hilfe (N) + format (N)]
Hilfsförster	NOUN	[hilfe (N) + förster (N)]
Hilfefunktion	NOUN	[hilfe (N) + funktion (N)]
Hilfsfunktion	NOUN	[hilfe (N) + funktion (N)]
Hilfsmaschinen	NOUN	[hilfe (N) + maschine (N)]
Hilfsmaß	NOUN	[hilfe (N) + maß (N)]
Hilfsmaßnahme	NOUN	[hilfe (N) + maßnahme (N)]
Hilfsmaßnahmen	NOUN	[hilfe (N) + maßnahme (N)]
Hilfsmaßstab	NOUN	[hilfe (N) + maßstab (N)]
Brandbeschädigt	ADJECTIVE	[brand (N) + beschädigen (V)]
steinschlagbeschädigt	ADJECTIVE	[stein (N) + schlag (N) + beschädigen (V)]
Regenbeschädigt	ADJECTIVE	[regen (N) + beschädigen (V)]
Unbeschädigt	ADJECTIVE	[un (F) + beschädigen (V)]
wasserbeschädigt	ADJECTIVE	[wasser (N) + beschädigen (V)]
Feuerbeschädigt	ADJECTIVE	[feuer (N) + beschädigen (V)]
Lärmgeschädigt	ADJECTIVE	[lärm (N) + schädigen (V)]
bombengeschädigt	ADJECTIVE	[bombe (N) + schädigen (V)]
katastrophengeschädigt	ADJECTIVE	[katastrophe (N) + schädigen (V)]

strahlengeschädigt	ADJECTIVE	[strahl (N) + schädigen (V)]
hirngeschädigt	ADJECTIVE	[hirn (N) + schädigen (V)]
ungeschädigt	ADJECTIVE	[un (F) + schädigen (V)]
hörgeschädigt	ADJECTIVE	[hören (V) + schädigen (V)]

C. ILSP FBT POS Tagger

Administrative information	
* Tool name	ILSP Feature-Based multi-Tiered POS Tagger
Short name	ILSP FBT
* Short description	The FBT POS Tagger is an adaptation of the Brill tagger trained on Greek text. It uses a PAROLE compatible tagset of 584 different tags which capture the morphosyntactic particularities of the Greek language.
* Organization name	ILSP
* Latest version number and release date	1.2 – 2009-01-29
Tool web page	
* Contact person in the context of PANACEA (i.e. person responsible for the integration of the tool in the factory)	Prokopis Prokopidis
* Contact person's email	prokopis @ilsp.gr
* Technical report or publication relevant to the application	See http://sifnos.ilsp.gr:8080/tpc/ LREC_2000_unified_pos_tagging_architecture_for_Greek.pdf .
Relevant project(s)	
* License and availability	Available as a free service for research purposes.
Descriptive information	
* Languages covered	Greek
* Character encoding	Unicode (UTF-8)

(input)	ISO-8859-7
* Character encoding (output)	Unicode (UTF-8) ISO-8859-7
* Format (input)	XML files with tokens and sentence boundaries.
* Format (output)	XML files with POS-tagged tokens. The XML files adhere to a UIMA-compatible annotation type system that is based on the JULIE Lab Type Sytem 2.1 available from http://www.julielab.de. Optionally, inline-annotation versions of the output are also generated.
* Compatibility of the input and/or output data with national/international standards/common practices	The ILSP POS Tagset used is Parole-compatible (see http://sifnos.ilsp.gr:8080/tpc/tagset_examples/tagset_en/ for a description of the tagset) .
* Language resources required for the operation of the application	ILSP FBT assigns initial tags by looking up tokens in a lexicon created from a manually annotated corpus of approx. 455K tokens. A suffix lexicon is used for initially tagging unknown words. 799 contextual rules are then applied to correct initial tags.
* Operating system	OS independent
* Implementation language	Java
* Other software requirements	Java VM 1.5+, UIMA (http://incubator.apache.org/uima/)
URL providing access to the tool as a web service	Making all ILSP tools available as web services is under development. All tools are currently available as UIMA-AS services that use the Java Message Service (JMS) API, and will be converted to UIMA-agnostic web services in the context of PANACEA.
URL providing guidelines on accessing the tool as a web service	See above

Hardware requirements	Typical modern workstation.
* Processing speed	620 tokens/sec
Evaluation information	
* Methodology and reference data	We have tested the ILSP FBT Tagger accuracy against a 90K corpus with manually annotated POS tags.
* Results	The tagger’s accuracy reaches 97.46 when only basic POS is considered. When all features (including, for example, case for nouns and tense for verbs) are taken into account the tagger’s accuracy is 92.29.

Sample output data

The output format of the ILSP FBT Tagger is essentially a tab-separated file with offset, token type, token and POS tags columns. Lemmas from the ILSP Lemmatizer are also included in this example.

```
            (SENT <S>
1\1   TOK   Δεν   δεν   PtNg
1\5   TOK   ξεχνάμε     ξεχνάω      VbMnIdPr01PlXxIpAvXx
1\13  TOK   την   ο     AtDfFeSgAc
1\17  TOK   Αϊτή  Αϊτή  NoPrFeSgAc
1\21  PTERM_P     .     .     PTERM_P
      )SENT </S>
      (SENT <S>
1\23  TOK   Επιπλέον    επιπλέον    AdXxBa
1\32  TOK   βοήθεια     βοήθεια     NoCmFeSgAc
1\40  DIG   90    90    DIG
1\43  ABBR  εκατ. εκατ. ABBR
1\49  TOK   ευρώ  ευρώ  NoCmNeSgAc
1\54  TOK   από   από   AsPpSp
1\58  TOK   τα    ο     AtDfNePlAc
1\61  TOK   κονδύλια    κονδύλι     NoCmNePlAc
1\70  TOK   έκτακτης    έκτακτος    AjBaFeSgGe
1\79  TOK   ανάγκης     ανάγκη      NoCmFeSgGe
1\87  TOK   της   ο     AtDfFeSgGe
1\91  ABBR  ΕΕ    ΕΕ    ABBR
1\94  TOK   προτείνει   προτείνω    VbMnIdPr03SgXxIpAvXx
1\104 TOK   η     ο     AtDfFeSgNm
1\106 TOK   Επιτροπή    επιτροπή    NoCmFeSgNm
1\115 TOK   για   για   AsPpSp
1\119 TOK   τη    ο     AtDfFeSgAc
1\122 TOK   σεισμόπληκτη      σεισμόπληκτος     AjBaFeSgAc
1\135 TOK   χώρα  χώρα  NoCmFeSgAc
1\139 PTERM_P     .     .     PTERM_P
      )SENT </S>
```

D. Sample XML output from WP4

In this section we briefly discuss a proposal for an XML encoding of the WP4 output. This proposal is based on the XCES Corpus Encoding Standard[24]. The full discussion of the proposal is included in section 6 of the *D3.1 Architecture and Design of the Platform*.

For illustration purposes, we assume as input a web page in Spanish referring to the EU aid for Haiti after the 2010 earthquake. In the example XML file below, we include the output of WP4 tools for corpus acquisition, normalization and text processing. Brief comments on the XML structure follow and refer to this example.

```
<?xml version="1.0"?>
<cesDoc id="news_20100514_haiti_es" version="0.4"
xmlns="http://www.xces.org/schema/2003">
  <cesHeader version="0.4">

    <fileDesc>
      <titleStmt>
         <title>La UE enviará más ayuda a Haití</title>
            <respStmt>
              <resp>
              <type>Crawling</type>
              <name>Panacea partner</name>
              </resp>
            </respStmt>
            <respStmt>
              <resp>
              <type>Boilerplate removal, text extraction,
                    paragraph detection, etc.
              </type>
              <name>Panacea partner</name>
             </resp>
            </respStmt>
            <respStmt>
              <resp>
              <type>Sentence splitting, tokenization,
                    POS tagging, lemmatization, parsing
              </type>
              <name>Panacea partner</name>
             </resp>
            </respStmt>
      </titleStmt>
      <sourceDesc>
            <biblStruct>
             <monogr>
             <author>EU web author if available</author>
             <imprint>
             <publisher>EU</publisher>
             <pubDate>2010-02-20</pubDate>
             <eAddress type="web">

http://ec.europa.eu/news/external_relations/100218_es.htm
             </eAddress>
             </imprint>
             </monogr>
            </biblStruct>
```

[24] http://www.xces.org

```
    </sourceDesc>
  </fileDesc>

  <profileDesc>
    <langUsage>
      <language iso639="es"/>
    </langUsage>
    <textClass>
          <keywords>
            <keyTerm>Comisión</keyTerm>
            <keyTerm>Haití</keyTerm>
            <keyTerm>terremoto</keyTerm>
            <keyTerm>. . .</keyTerm>
           </keywords>
           <domain>International News</domain><!--  or automotive,
                    environment,  legal -->
           <subdomain>Optional information on subdomain</subdomain>
           <subject>Optional information on the subject</subject>
    </textClass>
    <annotations>
       <annotation ann.loc="news_20100514_haiti_es.html"
                   type="htmlsource"/>
    </annotations>
  </profileDesc>
</cesHeader>
<text>
<body>
  <p id="p1" >
    <s id="s1">
    <graph root="s1_500">
      <terminals>
        <t id="s1_1" tag="AFS" lemma="el" word="La"/>
        <t id="s1_2" tag="N4666" lemma="UE" word="UE"/>
        <t id="s1_3" tag="VDU3S-" lemma="enviar" word="enviará"/>
        <t id="s1_4" tag="D" lemma="mas" word="mas"/>
        <t id="s1_5" tag="N5-FS" lemma="ayuda" word="ayuda"/>
        <t id="s1_6" tag="P" lemma="a" word="a"/>
        <t id="s1_7" tag="N4666" lemma="Haiti" word="Haiti"/>
      </terminals>
      <non-terminals>
        <nt cat="S" id="s1_500">
          <edge idref="s1_501" />
          <edge idref="s1_502" />
          <edge idref="s1_503" />
          <edge idref="s1_504" />
        </nt>
        <nt  cat="NP" id="s1_501">
          <edge idref="s1_1" />
          <edge idref="s1_2" />
        </nt>
        <nt  cat="VP" id="s1_502">
          <edge idref="s1_3" />
        </nt>
        <nt  cat="NP" id="s1_503">
          <edge idref="s1_4" />
          <edge idref="s1_5" />
        </nt>
        <nt  cat="PP" id="s1_504">
          <edge idref="s1_6" />
          <edge idref="s1_505" />
          </nt>
          <nt  cat="NP" id="s1_505">
            <edge idref="s1_7" />
          </nt>
        </non-terminals>
      </graph>
      </s>
    </p>
  </body>
</text>
</cesDoc>
```

The root of this XML document is a <cesDoc> element. All metadata are contained inside a <cesHeader> element. The header contains a <fileDesc> element that can be used for information about the title of the document and any annotations added by Panacea tools. The <sourceDesc> subelement can be used for information on the author and publication date of the original web document, the publisher of the document, and the URL it was downloaded from. One or more <respStmt> subelements can be used to describe operations and Panacea groups responsible for operations on this particular document. The <profileDesc> element groups information describing the language of the document (<langUsage>) and the nature or topic of the text (<domain>, <subdomain>, <subject>, <keywords>). The <annotations> subelement of the <profileDesc> can be used for storing links to other documents relevant to this basic version. In the example above, a link to the original HTML document is included.

Linguistic annotations are generated by text processing tools including sentence splitting, tokenization, POS tagging, lemmatization and constituency parsing. Each sentence is included in a graph element that contains a terminals and a non-terminals element. Tokens are represented as terminal t elements with attributes for POS tags and lemmas. Non-terminal nt elements correspond to phrases in a contituency parse tree and include attributes describing the constituent category. They also include edge elements that point to terminal or non-terminal children nodes of each nt.