Ellogon: A Natural Language Engineering Infrastructure

Georgios Petasis, Vangelis Karkaletsis, Georgios Paliouras,
and Constantine D. Spyropoulos

Software and Knowledge Engineering Laboratory,
Institute of Informatics and Telecommunications,
National Centre for Scientific Research (N.C.S.R.) "Demokritos",
P.O. BOX 60228, Aghia Paraskevi,
GR-153 10, Athens, Greece.
e-mail: {petasis, vangelis, paliourg, costass}@iit.demokritos.gr

Abstract. This paper presents Ellogon, a multi-lingual, cross-operating system, general-purpose natural language engineering infrastructure. Ellogon was designed in order to aid both researchers in natural language processing, as well as companies that produce language engineering systems for the end-user. Ellogon provides a powerful TIPSTER-based infrastructure for managing, storing and exchanging linguistic data, embedding and managing processing components as well as visualising textual data and their associated linguistic information. Among its key features are complete Unicode support, an extensive multilingual graphical user interface, a modular architecture and the reduced hardware requirements.

1 Introduction

In this paper we describe Ellogon, a natural language engineering platform developed by Software and Knowledge Engineering Laboratory of the Institute of Informatics and Telecommunications, N.C.S.R. "Demokritos", Greece. Ellogon is a multi-lingual, cross-platform, general-purpose language engineering environment, developed in order to aid a wide range of users – from researchers in the natural language field or computational linguistics to companies that produce and deliver language engineering systems.

Being in constant development since 1998, Ellogon is a mature and well tested infrastructure, as it has been used in many national and European funded projects. Its facilities have been used for creating a wide range of applications, from multilingual information extraction systems to controlled language checkers.

Ellogon as a text engineering platform offers an extensive set of facilities, including tools for visualising textual/HTML/XML data and associated linguistic information, support for lexical resources (like creating and embedding lexicons), tools for creating annotated corpora, accessing databases, comparing annotated data, or transforming linguistic information into vectors for use with various machine learning algorithms. Additionally, Ellogon offers some unique features, like the ability to freely

modify annotated textual data (with Ellogon automatically applying the required transformations on the associated linguistic information) and the ability to create stand-alone applications with customised user interfaces that perform specific tasks.

More information about Ellogon can be found at the Ellogon's site, at http://www.iit.demokritos.gr/skel/Ellogon, as well as in (Petasis et al., 2002; 2003).

The rest of this demonstration paper is organised as follows: In section 2 the architecture and data model of Ellogon are briefly described. In section 3 some important aspects of Ellogon are presented and finally section 4 presents some concluding remarks and future plans.

2 Ellogon Architecture and Data Model

Ellogon belongs to the category of referential or annotation based platforms, where the linguistic information is stored separately from the textual data, having references back to the original text. Based on the TIPSTER data model (Grishman, 1997), Ellogon provides infrastructure for:

- Managing, storing and exchanging textual data as well as the associated linguistic information.
- Creating, embedding and managing linguistic processing components.
- Facilitating communication among different linguistic components by defining a suitable programming interface (API).
- Visualising textual data and associated linguistic information.

The architecture of Ellogon, the utilised data model and the linguistic processing components as well as some key features of Ellogon are presented in the following subsections.

2.1 Ellogon Architecture

Ellogon aims to be a versatile infrastructure that can used either as an NLP integrated development environment (IDE) or as a library that can be embedded to foreign applications. As a result, Ellogon proposes a modular architecture with four independent subsystems (fig. 1):

- A highly efficient core developed in C, which implements an extended version of the TIPSTER data model. Its main responsibility is to manage the storage of the textual data and the associated linguistic information and to provide a well-defined programming interface (API) that can be used in order to retrieve/modify the stored information.
- An object oriented C++ API which increases the usability of the C core API. This object oriented API is exposed in a wide range of programming languages, including C++, Java, Tcl, Perl and Python.
- An extensive and easy to use graphical user interface (GUI). This interface can be easily tailored to the needs of the end user.
- A modular pluggable component system. All linguistic processing within the platform is performed with the help of external, loaded at run-time, components. These

components can be implemented in a wide range of programming languages, including C, C++, Java, Tcl, Perl and Python.

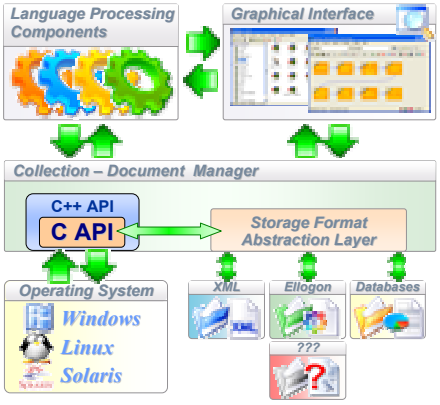

Fig. 1. Ellogon Architecture.

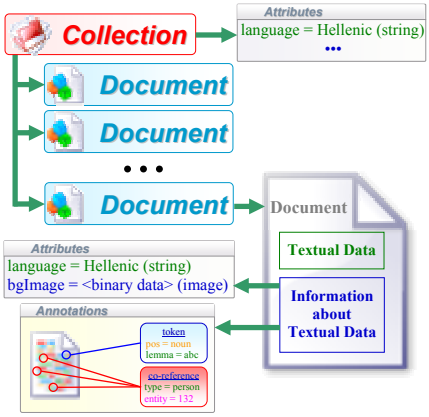

Fig. 2. Ellogon Data Model.

2.2 Ellogon Data Model

Ellogon shares the same data model as the TIPSTER architecture (fig. 2). The central element for storing data in Ellogon is the *Collection*. A collection is a finite set of *Documents*. An Ellogon document consists of *textual data* as well as *linguistic information about the textual data*. This linguistic information is stored in the form of *attributes* and *annotations*.

An attribute associates a specific type of information with a typed value. An annotation associates arbitrary information (in the form of attributes) with portions of textual data. Each such portion, named *span*, consists of two character offsets denoting the start and the end characters of the portion, as measured from the first character of some textual data. Annotations typically consist of four elements:

- *A numeric identifier*. This identifier is unique for every annotation within a document and can be used to unambiguously identify the annotation.
- *A type*. Annotation types are textual values that are used to classify annotations into categories.
- *A set of spans* that denote the range of the annotated textual data.
- *A set of attributes*. These attributes usually encode the necessary linguistic information.

3 Ellogon key features

In the following paragraphs, we briefly present some of the most important aspects of the Ellogon language engineering platform.

3.1 Support of multiple languages

The fact that Ellogon offers complete Unicode support (in both its core unit CDM as well as in its GUI) provides the ability to properly support a wide range of languages. Ellogon includes a large number of input/output filters for various encodings, such as the ISO-8859-* encodings or the encodings used under Microsoft Windows or Apple Macintosh. Additionally, components can be classified according to the language they support and can utilise the utilities provided by the API in order to convert textual data among various encodings. Finally, Ellogon provides an internationalised GUI that has been designed to facilitate the integration of additional languages, even by the end user.

3.2 Portability

Supporting all the major operating systems has always been a shortcoming of many of the existing language engineering platforms. Ellogon on the other hand, offers native ports to many operating systems and has been extensively used and tested under Unix (Solaris 2.6, 7, 8 & 9, Red Hat Linux 6.x, 7.x, 8.0 & 9.x) and Microsoft Windows (95, 98, Me, NT 4.0, NT 2000 & XP). Additionally, Ellogon aims to provide a unified view of various operating system specific tasks under all supported operating systems. For example, pipelines and file redirections are emulated under Microsoft Windows or filenames can be specified using the Unix notation under all supported operating systems. Finally, the provided graphical interface provides exactly the same functionality under the various supported operating systems.

3.3 Advanced GUI

Ellogon offers an extensive and powerful multi-lingual user interface. This GUI provides users with the ability to manage Collections/Documents/Systems, to visualise linguistic information with an extensible set of visualisation tools, to develop and integrate linguistic components, to browse documentation and of course, to do linguistic processing of textual data using various modes. Finally, the user interface can be adapted to meet specific needs, such as systems dedicated to specific linguistic processing tasks.

3.4 Modular Architecture

Ellogon is based on a modular architecture that allows the reuse of Ellogon subsystems in order to ease the creation of applications targeting specific linguistic needs.

Ellogon's core component – CDM – is implemented as a separate library that can be dynamically loaded if the underlying operating system offers such ability. This library can be independently embedded inside any application that can call functions from libraries, following the C++ naming conventions. Examples of embedding CDM under various applications include Microsoft Word as well as the Tcl, Java, Perl and Python interpreters.

3.5 Memory compression

The use of memory by a text engineering platform is a very important aspect, as it usually determines the size of textual data that can be processed under this platform. Under Ellogon, this requirement is far more important, as the use of Unicode can increase memory requirements by simply changing from a language that requires fewer bytes per character (like English) into a language needing more bytes per character (like Greek). Ellogon tries to decrease its memory requirements by incorporating a memory compression scheme. Initial measurements have shown that Ellogon uses less memory for performing the same tasks than other (TIPSTER-based) platforms.

4 Future Plans

We are continuously working to improve Ellogon along many directions. Although Ellogon is already highly optimised, we still try to further reduce the memory requirements. Currently, we try to enhance CDM with the ability to selectively load only the needed information from a document in memory instead of the whole document. We are also working towards improving the user interface by adding new features and improving existing ones. Future versions of Ellogon will provide more ready to use tools as well as more linguistic processing components. Finally a specialised extension is under development, which will provide better support for relations among annotations (inside the same document or across multiple documents) as well as better handling of hierarchical data, like ontologies.

References

(Grishman, 1997): Grishman, R. 1997. "TIPSTER Architecture Design Document Version 2.3". *Technical Report, DARPA*. Available at:
`http://www.itl.nist.gov/div894/894.02/related_projects/tipster`
and
`http://www.tipster.org.`

(Petasis et al., 2002) G. Petasis, V. Karkaletsis, G. Paliouras, I. Androutsopoulos and C. D. Spyropoulos, "Ellogon: A New Text Engineering Platform". *In Proceedings of the 3rd International Conference on Language Resources and Evaluation (LREC 2002)*, Las Palmas, Canary Islands, Spain, pp. 72-78, May 2002.

(Petasis et al. 2003) Petasis G., Karkaletsis V and Spyropoulos C. D., 2003. "Cross-lingual Information Extraction from Web pages: the use of a general-purpose Text Engineering Platform", In *Proceedings of the RANLP'2003 Conference*, Borovets, Bulgaria, 2003.