Using the Ellogon 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

Ellogon is a multi-lingual, cross-operating system, general-purpose natural language engineering infrastructure. Ellogon has been used extensively in various NLP applications. It is currently provided for free for research use to research and academic organisations. In this paper, we outline its architecture and data model, present Ellogon features as used by different types of users and discuss its functionalities against other infrastructures for language engineering.

1. Introduction

Ellogon was developed by the Software and Knowledge Engineering Laboratory of the Institute of Informatics and Telecommunications, N.C.S.R. “Demokritos”, Greece[1]. Its development started in 1998. The motivation for its development, at that time, was the inadequacy of existing platforms to support important features for the NLP applications of the SKEL laboratory. Being in constant development since 1998, Ellogon is now a mature and well tested infrastructure, as it has been used in many national and European projects either by SKEL or its partners in some of these projects. Ellogon facilities have been used for creating a wide range of NLP applications, such as various linguistic tools, information filtering and extraction systems in several European languages, controlled language checkers. Since 2002, Ellogon is provided for free, for research use, to research and academic organisations.

Ellogon as a text engineering platform offers an extensive set of facilities, including tools for processing and 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.

The rest of the paper is organised as follows: In section 2 Ellogon architecture, data model and some important features are briefly described. Section 3 presents some features that facilitate its use by different types of users, whereas section 4 gives examples of certain NLP applications where Ellogon was employed. Section 5 discusses Ellogon functionalities against existing infrastructures. Finally, section 6 presents some concluding remarks and future plans.

2. Ellogon Architecture and Data Model

During the last decade, a large number of software infrastructures aiming at facilitating R&D in the field of natural language processing have been presented. Some of these infrastructures, such as LT-NSL/LT-XML tools (McKelvie et. al. 1997) or GATE[2], have become extremely popular as they have been applied to a wide range of tasks by many institutions around the world.

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 sub-sections.

2.1. Ellogon Architecture

Ellogon can be used either as an NLP integrated development environment (IDE) or as a library that can

[1] http://www.iit.demokritos.gr/skel/Ellogon

[2] Information about GATE can be found at http://gate.ac.uk

be embedded to foreign applications. To achieve this, Ellogon proposes and implements a modular architecture with four independent subsystems:

- 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.

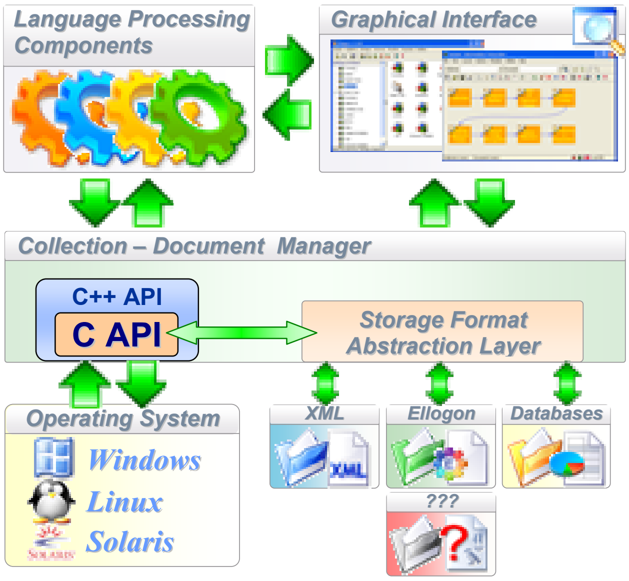

Language Processing Components
Graphical Interface
Collection – Document Manager
C++ API
C API
Storage Format Abstraction Layer
Operating System
Windows
Linux
Solaris
XML
Ellogon
Databases
???

Figure 1: Ellogon Architecture.

2.2. Ellogon Data Model

Ellogon shares the same data model as the TIPSTER architecture. Due to this, it shares some basic features with other TIPSTER-based infrastructures, such as GATE. However, it also offers a large number of features that differentiate it from such infrastructures.

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.

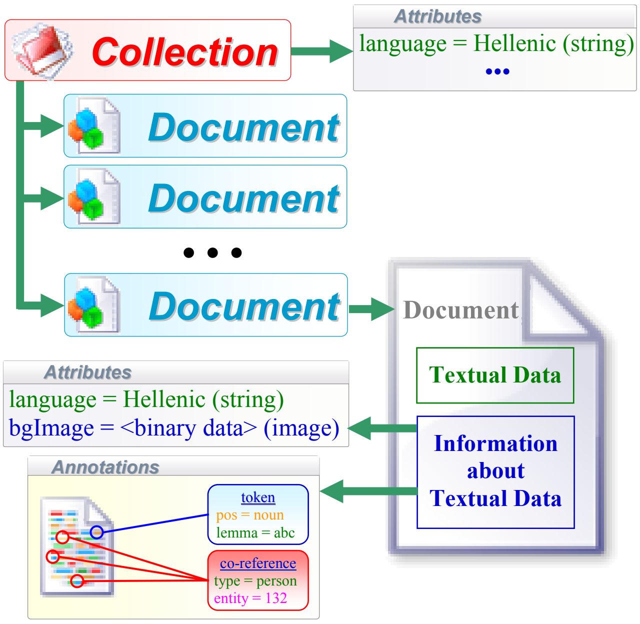


Figure 2: Ellogon Data Model.

2.3. Ellogon Components

For most users of Ellogon, the central point of interest is the linguistic processing that can be carried out within it. Ellogon provides a generic framework where external components can be easily embedded. As Ellogon follows a modular paradigm, it utilises components of various types, with each type specialising in a specific processing task. A taxonomy of the currently defined component types are shown in figure 3.

The most important component type from the user's point of view is of course the linguistic processing component, as natural language processors usually belong to this component type. These components (along with components of the machine-learning processing type) can be organised into Systems for performing some specific task. The tasks can range from basic linguistic tasks, such as part-of-speech tagging or parsing, to application level tasks, such as information extraction or machine translation.

A linguistic processing component consists mainly of two parts. The first part is responsible for performing the desired linguistic processing while the main responsibility of the second component part is to interface the linguistic processing sub-component with Ellogon, through the provided API. Components can appear either as *wrappers* or as *native components*. Wrappers usually provide the needed code in order to interface an existing independent implementation of a linguistic processing tool to the Ellogon platform. Native components on the other hand are processing tools specifically designed for use within the Ellogon platform. Usually, in such components the two component parts cannot be easily identified or separated.

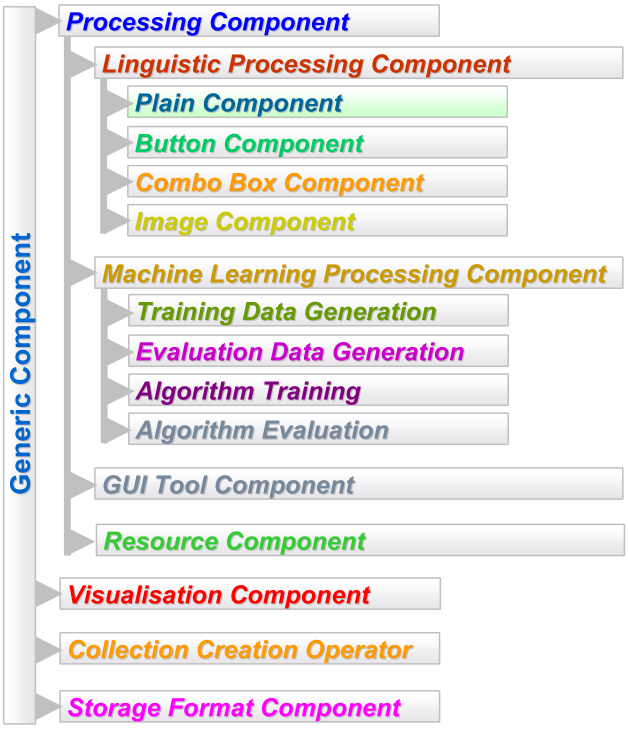
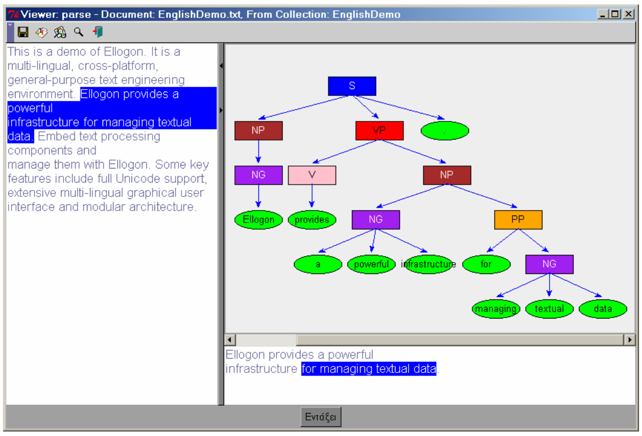


Figure 3: The various component types.

Each component is associated with metadata, which include a set of pre-conditions and a set of post-conditions among other information. Pre-conditions declare the linguistic information that must be present in a document before this specific component can be applied to it. Post-conditions describe the linguistic information that will be added in the document as a side effect of processing the document with this specific component. Ellogon uses these two sets in order to establish relations among the various components or to “undo” the results of a component application on a corpus.

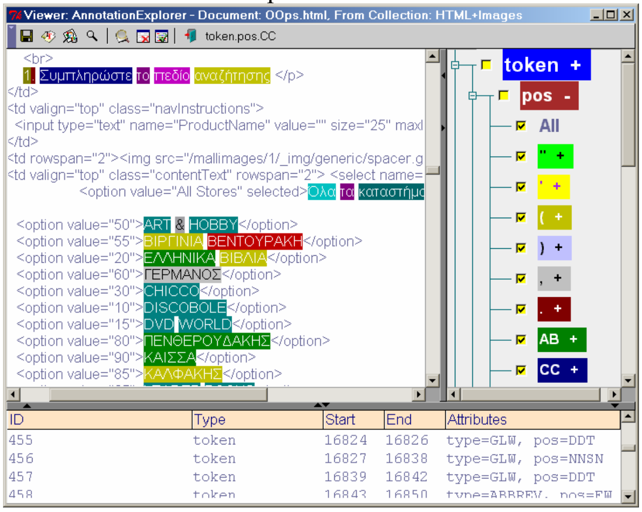
Each component can also specify a set of parameters, as well as a set of viewers (components of type “visualisation component”). Parameters represent various run-time dependent options (such as the location of a file containing the grammar of a syntactic parser). They can be edited by the end user through the graphical interface and are given to the component every time it is executed. A component can also specify a set of predefined viewers, in order to present in a graphical way the linguistic information produced during the component execution. Examples of available viewers are shown in figures 4 and 5.


Viewer: parse - Document: EnglishDemo.txt, From Collection: EnglishDemo

This is a demo of Ellogon. It is a multi-lingual, cross-platform, general-purpose text engineering environment. Ellogon provides a powerful infrastructure for managing textual data. Embed text processing components and manage them with Ellogon. Some key features include full Unicode support, extensive multi-lingual graphical user interface and modular architecture.

Ellogon provides a powerful infrastructure for managing textual data.

Figure 4: An Ellogon viewer showing a parse tree of a simple sentence.


Viewer: AnnotationExplorer - Document: OOps.html, From Collection: HTML+Images

token.pos.CC

```
  <br>
  1. Συμπληρώστε το πεδίο αναζήτησης </p>
</td>
<td valign="top" class="navInstructions">
 <input type="text" name="ProductName" value="" size="25" maxl
</td>
<td rowspan="2"><img src="/mallimages/1/_img/generic/spacer.g
<td valign="top" class="contentText" rowspan="2"> <select name=
         <option value="All Stores" selected>Όλα τα καταστήμα

 <option value="50">ART & HOBBY</option>
 <option value="55">ΒΙΡΓΙΝΙΑ ΒΕΝΤΟΥΡΑΚΗ</option>
 <option value="20">ΕΛΛΗΝΙΚΑ ΒΙΒΛΙΑ</option>
 <option value="60">ΓΕΡΜΑΝΟΣ</option>
 <option value="30">CHICCO</option>
 <option value="10">DISCOBOLE</option>
 <option value="15">DVD WORLD</option>
 <option value="80">ΠΕΝΘΕΡΟΥΔΑΚΗΣ</option>
 <option value="90">ΚΑΙΣΣΑ</option>
 <option value="85">ΚΑΛΦΑΚΗΣ</option>
```

ID	Type	Start	End	Attributes
455	token	16824	16826	type=GLW, pos=DDT
456	token	16827	16838	type=GLW, pos=NNSN
457	token	16839	16842	type=GLW, pos=DDT
458	token	16843	16850	type=ABBREV, pos=FW

Figure 5: An Ellogon viewer presenting the output of a part-of-speech tagger applied on an HTML document.

Creating components can be easily done through the Ellogon GUI. Currently, Ellogon components can be developed in five languages, C/C++, Tcl, Java, Perl and Python. The Ellogon GUI offers a specialised dialog where the user can specify various parameters of the component he/she intends to create, including its pre/post-conditions. Then Ellogon creates the *skeleton* of the new component that will handle all the interaction with the Ellogon platform. If the language of the component is C/C++ or Java, proper *Makefiles* for compiling the component under Unix and Windows will also be created. Besides creating a skeleton, Ellogon tries to facilitate the development of the component by allowing the developer to edit the source code and re-load the specific component into Ellogon from its GUI.

2.4. Ellogon key features

In the following paragraphs, we briefly present some of the most important aspects of the Ellogon language engineering platform. For more details, see (Petasis et al. 2002) as well as the Ellogon documentation at the Ellogon site.

2.4.1. 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[3] that has been designed to facilitate the integration of additional languages, even by the end user.

2.4.2. 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.

2.4.3. Advanced GUI

Ellogon offers an extensive and powerful multilingual 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.

2.4.4. Modular Architecture

Ellogon is based on a modular architecture that allows the reuse of Ellogon sub-systems 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[4] as well as the Tcl, Java, Perl and Python interpreters.

2.4.5. 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.

3. Supporting Ellogon Users

The users of Ellogon can be roughly classified in three major categories: computational linguists, language engineers and end users. In this section we try to investigate how Ellogon can facilitate each user group work.

3.1. Facilitating Computational Linguists

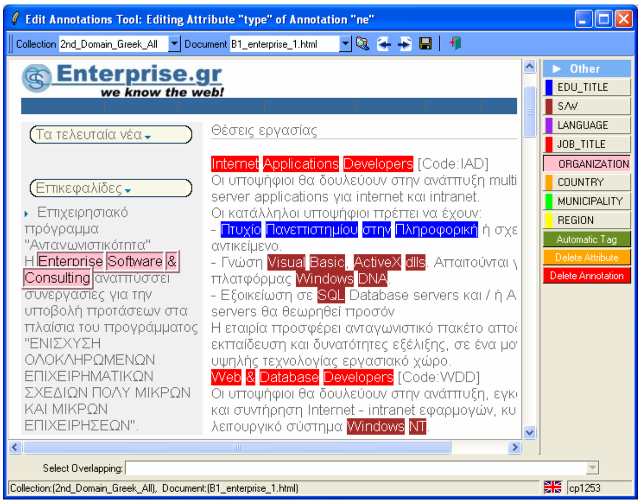
Ellogon tries to facilitate many aspects of the tasks computational linguists usually perform within the platform, especially if the task involves annotated corpora creation, linguistic processing component adaptation or various evaluation tasks.

Providing a wide range of highly customisable and easy to use annotation tools, Ellogon is an ideal environment for annotated corpora construction.

[3] Currently, the provided GUI languages include only English and Greek.

[4] In order to embed CDM under Microsoft Word we utilise the Active-X technology, with CDM exported as an Active-X component.

Available annotators support regular marking (e.g. part of speech tagging or named entities annotation) as well as annotation of hierarchically information (i.e. syntactic relation annotation) on plain as well as HTML corpora. (Two annotation tools are shown in figures 6, 7).



Edit Annotations Tool: Editing Attribute "type" of Annotation "ne"

Collection 2nd_Domain_Greek_All Document B1_enterprise_1.html

Enterprise.gr
we know the web!

Τα τελευταία νέα

Επικεφαλίδες

Επιχειρησιακό πρόγραμμα "Ανταγωνιστικότητα"
Η Enterprise Software & Consulting αναπτύσσει συνεργασίες για την υποβολή προτάσεων στα πλαίσια του προγράμματος "ΕΝΙΣΧΥΣΗ ΟΛΟΚΛΗΡΩΜΕΝΩΝ ΕΠΙΧΕΙΡΗΜΑΤΙΚΩΝ ΣΧΕΔΙΩΝ ΠΟΛΥ ΜΙΚΡΩΝ ΚΑΙ ΜΙΚΡΩΝ ΕΠΙΧΕΙΡΗΣΕΩΝ".

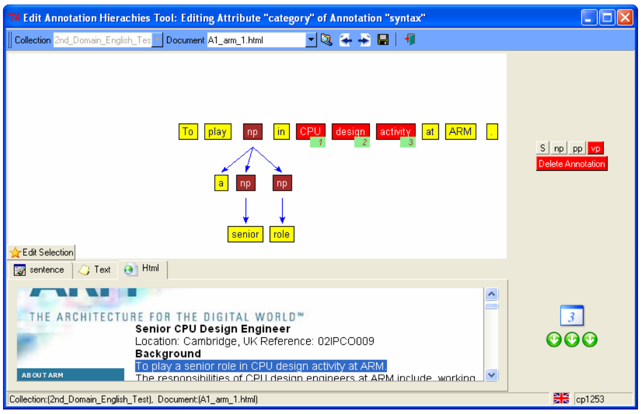
Θέσεις εργασίας

Internet Applications Developers [Code:IAD]
Οι υποψήφιοι θα δουλεύουν στην ανάπτυξη multi
server applications για internet και intranet.
Οι κατάλληλοι υποψήφιοι πρέπει να έχουν:
- Πτυχίο Πανεπιστημίου στην Πληροφορική ή σχε
αντικείμενο.
- Γνώση Visual Basic, ActiveX dlls. Απαιτούνται γ
πλατφόρμας Windows DNA.
- Εξοικείωση σε SQL Database servers και / ή A
servers θα θεωρηθεί προσόν
Η εταιρία προσφέρει ανταγωνιστικό πακέτο αποι
εκπαίδευση και δυνατότητες εξέλιξης, σε ένα μο
υψηλής τεχνολογίας εργασιακό χώρο.
Web & Database Developers [Code:WDD]
Οι υποψήφιοι θα δουλεύουν στην ανάπτυξη, εγκ
και συντήρηση Internet - intranet εφαρμογών, κυ
λειτουργικό σύστημα Windows NT.

Select Overlapping:

Collection:(2nd_Domain_Greek_All), Document:(B1_enterprise_1.html) cp1253

Figure 6: Annotating named entities on HTML pages.



Edit Annotation Hierachies Tool: Editing Attribute "category" of Annotation "syntax"

Collection 2nd_Domain_English_Tes Document A1_arm_1.html

THE ARCHITECTURE FOR THE DIGITAL WORLD™
Senior CPU Design Engineer
Location: Cambridge, UK Reference: 02IPCO009
Background
To play a senior role in CPU design activity at ARM.
The responsibilities of CPU design engineers at ARM include working

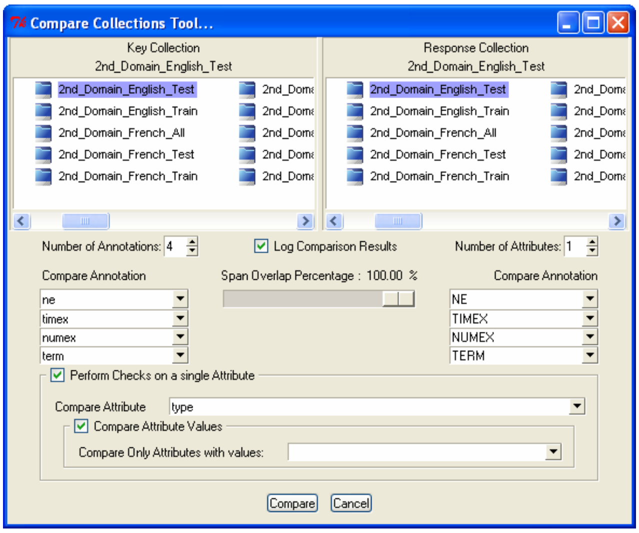
Collection:(2nd_Domain_English_Test), Document:(A1_arm_1.html) cp1253

Figure 7: Annotating syntactic relations.

Adapting linguistic processing components into a new domain is another frequent task. Usually it involves modifications to domain specific resources used internally by the processing components. Ellogon facilitates the adaptation process as the modified component can be applied immediately and the user can very easily identify the effect of his/her modifications, through the comparison facilities offered by the platform. Ellogon provides significant infrastructure for comparing the linguistic information associated with the textual data. The Collection Comparison tool (figures 8, 9) can be used for comparing the linguistic information stored in a set (or collection) of documents. Various constraints regarding the information that will be compared can be specified through the graphical user interface of the comparison tool and the comparison results are presented by utilising standard figures, like recall, precision and F-measure. Additionally, the comparison tool can present a comparison log. This log is a graphical representation of the differences found during the comparison process and can provide valuable help to the user in order to locate and possibly correct the errors.

3.2. Facilitating Language Engineers

One of the most frequent tasks performed by language engineers inside Ellogon is of course the development of processing components. Significant infrastructure is provided in order to facilitate component development, from the very first step of writing the component to ensuring that the component works as expected. Operating as an integrating environment (IDE), Ellogon allows the creation of components in a wide range of programming languages (C, C++, Tcl, Java, Perl, Python): all the needed code of the component structure is automatically generated during the initial construction of a component while a component can be compiled, linked, loaded and tested from inside Ellogon. For some specific languages (all supported ones except Java) a component can be even unloaded, modified, compiled and re-loaded, in order to quickly test the effect of desired modifications.



Compare Collections Tool...

Key Collection 2nd_Domain_English_Test		Response Collection 2nd_Domain_English_Test	
2nd_Domain_English_Test	2nd_Doma	2nd_Domain_English_Test	2nd_Doma
2nd_Domain_English_Train	2nd_Doma	2nd_Domain_English_Train	2nd_Doma
2nd_Domain_French_All	2nd_Doma	2nd_Domain_French_All	2nd_Doma
2nd_Domain_French_Test	2nd_Doma	2nd_Domain_French_Test	2nd_Doma
2nd_Domain_French_Train	2nd_Doma	2nd_Domain_French_Train	2nd_Doma

Number of Annotations: 4 ☑ Log Comparison Results Number of Attributes: 1

Compare Annotation: ne, timex, numex, term

Span Overlap Percentage : 100.00 %

Compare Annotation: NE, TIMEX, NUMEX, TERM

☑ Perform Checks on a single Attribute

Compare Attribute: type

☑ Compare Attribute Values

Compare Only Attributes with values:

Figure 8: Specifying comparison parameters.

Developing components for Ellogon is a fairly easy process, as a high level API is provided both as a set of functions or as an object oriented hierarchy of classes, if the programming language allows it. Additionally, Ellogon is distributed with a small set of components whose source code can be used as an example on how to perform some commonly needed tasks.

The fact that almost everything in Ellogon is defined in terms of components, offers a large degree of flexibility to component developers. Combined with its modular architecture, Ellogon offers the ability to be tailored in order to meet specific needs. For example, particular Ellogon parts can be wrapped along with specific processing components to form a stand-alone application that performs a specific processing task (having possibly a specifically-made graphical interface). Such an application[5] will even ran without requiring the installation of Ellogon.

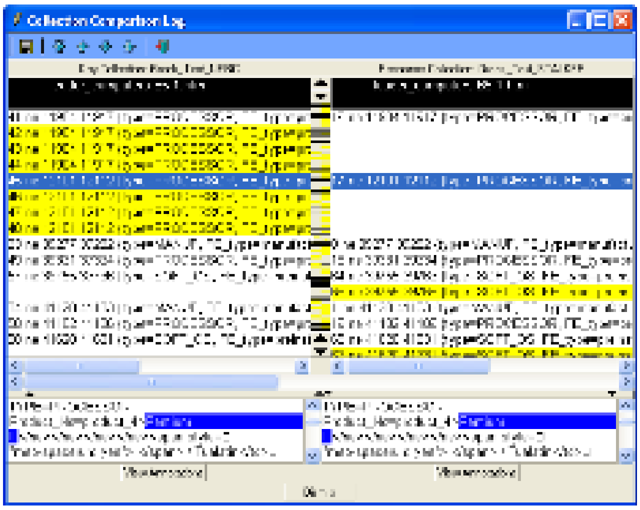


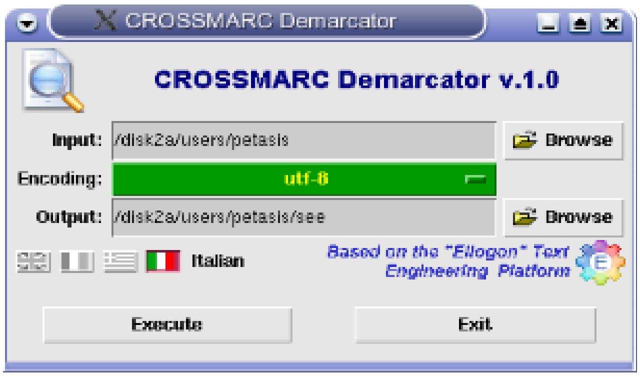
Figure 9: The Ellogon Corpora Comparison tool.

3.3. Facilitating end users

End users of Ellogon can be roughly distinguished in two categories: users that use applications or services based on Ellogon and users that use Ellogon as a "black box" in order to process corpora and collect the results.

Regarding the first category of end users, Ellogon provides many facilities for creating stand alone applications with customised graphical interfaces that are extremely easy to use. Such an application is shown in figure 10, where all the complexity of creating collections, applying the required processing components and exporting the processing results is hidden behind a simple graphical interface. In addition to creating specialised applications, Ellogon can be instrumented through the use of services, like ActiveX, DDE, HTTP or SOAP, which allow other applications to use Ellogon facilities in a way transparent to the end user.

The second category of end users characterises users who want to perform some sort of linguistic processing by simply applying the components available through Ellogon on a corpus. For this category of users, Ellogon is a toolbox of "black boxes": for example users may want to apply a named-entity recognition system operating within Ellogon or use more primitive components like a syntactic analyser. Ellogon tries to facilitate this category of users by providing an easy to use graphical interface that can be used to create collections from a wide variety of sources and easily apply on them any available processing component. Processing results can be examined through the large set of available viewers or even exported to widely used formats, such as SGML or XML. Finally, Ellogon offers the ability to automate tasks through the definition of "macro" commands, which can be useful especially in tasks that must be repeated multiple times.


CROSSMARC Demarcator
CROSSMARC Demarcator v.1.0
Browse
Browse
Execute
Exit
Based on the "Ellogon" Text Engineering Platform

Input: /disk2a/users/petasis

Encoding: utf-8

Output: /disk2a/users/petasis/see

Italian

Figure 10: A stand alone application automatically generated by the relevant wizard of Ellogon.

4. Systems that use Ellogon

Ellogon has been constantly used by the SKEL laboratory in many national and European research projects (some of which are presented below). Ellogon was also used by some of SKEL partners in the context of these projects. Additionally, Ellogon is currently being used by other research organisations for many tasks, including annotated corpora creation, corpora comparison and evaluation as well as for various educational purposes. We hope to collect valuable information about its use that will enable us to further improve Ellogon along many directions.

4.1. MITOS

MITOS is an R&D project[6] that combines techniques from information filtering to classify incoming news articles, as well as techniques from information extraction to extract factual information from financial news articles, which is then stored into a database (e.g. buyer, company bought). Ellogon was used as the development platform for the linguistic processing and information extraction components. It was also used to develop user-friendly applications for information extraction and for annotating training data. These applications are currently used successfully by users with no linguistic or NLP background.

4.2. SCHEMATOPOIESIS

In the context of the Greek R&D project SCHEMATOPOIESIS, Ellogon was used to develop the first prototype controlled language checker for Greek in

[5] As the creation of stand-alone applications is frequently needed, Ellogon provides a relevant wizard to facilitate this task.

[6] See also at: http://www.iit.demokritos.gr/skel/mitos/.

order to assist Greek technical writers as well as to facilitate translation from Greek to other languages[7]. The project covered technical documents from the domain of computer equipment. Ellogon was used not only as the development platform for the checker, but also as *a mean for embedding the checker under Microsoft Word*, allowing the user to check his/her documents in a similar way as a spell/syntax checker.

4.3. CROSSMARC

The "CROSS-lingual Multi Agent Retail Comparison" (CROSSMARC[8]) project develops technology for information filtering and extraction from the Web.

A large number of the functionalities provided by Ellogon have been exploited by SKEL and its partners, in the context of the CROSSMARC project. Supporting Java as a component development language has enabled the integration of the web site crawling and spidering agents as Ellogon components, thus easing their development as well as their deployment. The ability of Ellogon to automatically extract systems of components as stand-alone applications that run unmodified under different operating systems (Windows, Linux, Solaris, etc.) has been used extensively in CROSSMARC since the partners were using different operating systems for the development of their systems. Furthermore, the Ellogon tools for creating vectors in various formats (e.g. WEKA ARFF, C4.5 vector format, etc.) for use with machine learning algorithms have significantly facilitated the process of training of some of the CROSSMARC components based on machine learning. Additionally, the extensive set of annotation tools offered by Ellogon (either for plain text or HTML documents) has played a central role in corpora annotation in CROSSMARC, as these tools have been used for annotating part-of-speech, named-entity, noun phrase and syntactic information on the collected web pages. Finally, functionalities like the processing and display of HTML documents, XML, DOM and XSLT support as well as the various viewers created a "comfortable" environment for CROSSMARC developers (Petasis at al. 2003).

5. Discussion

The motivation behind the development of Ellogon (which started in 1998) was the inadequacy of existing platforms to support, at that time, some essential properties, such as the ability to

- support a wide range of languages through Unicode,
- function under all major operating systems,
- have as few hardware requirements as possible in terms of processing speed and memory usage,
- be based on an embeddable – decomposable architecture that enables parts to be embedded in other systems and
- provide an extensible, easy to use and powerful user interface.

Ellogon in its present form satisfies all of these requirements.

As Ellogon is based on the TIPSTER architecture, it shares many basic properties with other TIPSTER-based infrastructures like GATE. However, Ellogon offers several important features that differentiate it from similar infrastructures. In the following subsections some of these features are briefly presented.

5.1. Easy Component Development

It is fairly easy to understand the process of developing new components and develop them using the functionalities provided by Ellogon. Additionally, a wide range of programming languages for component development are supported, including C, C++, Java, Tcl, Perl and Python.

5.2. Integrated Development Environment

Ellogon operates as an integrated development environment, as it provides complete support to the development cycle of a component. Components can be created, edited, compiled and linked (whether applicable) from inside Ellogon. Furthermore, C/C++/Java components can be unloaded, modified, compiled and re-loaded into Ellogon without having to quit from Ellogon. The ability to unload or re-load all components is essential as it can significantly reduce development cycle, since component modifications can be immediately evaluated.

5.3. A ready to use component "toolbox"

Ellogon is equipped with a large number of ready-to-use tools for performing tasks like annotated corpora creation, vector generation or data comparison.

Additionally, several sample components are provided that can be adapted to various domains and languages, which perform some basic tasks like tokenization, part-of-speech tagging or gazetteer list lookup. Finally, Ellogon offers several data visualisation tools, ranging from simple viewers for the annotation database to viewers able to display hierarchical information, like syntax trees.

5.4. Easy deployment

As Ellogon implements a decomposable architecture, it is extremely easy to create an easy to use product from a set of components that perform a specific

[7] See also at: http://www.iit.demokritos.gr/skel/en/Projects/SCHEMATOPOIESIS.htm.

[8] See also at: http://www.iit.demokritos.gr/skel/crossmarc/.

task. All the components along with the needed Ellogon parts can be packaged either in a single executable (which needs no installation) or as an application (which can be ran unmodified under multiple operating systems). These specialised applications can be distributed and used in any system, even if Ellogon has not been installed to the system.

5.5. Features offered by similar platforms

Although Ellogon offers many features, the provided facilities may not cover specific needs that are possibly covered by other platforms. For example, GATE – another TIPSTER-based platform – offers a complete information extraction system. As such a system is not delivered by Ellogon, it may be less appropriate to use Ellogon in cases where a ready-to-use information extraction system is required.

6. 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.

7. 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.`

(McKelvie et. al. 1997): McKelvie D., Brew C., Thompson H. S. 1997. "Using SGML as a Basis for Data Intensive Natural Language Processing". *Computers and the Humanities*, 31(5): 367-388.

(Petasis et al. 2001): Petasis G., Karkaletsis V., Farmakiotou D., Samaritakis G., Androutsopoulos I., Spyropoulos C.D., 2001 "A Greek Morphological Lexicon and its exploitation by a Greek Controlled Language Checker". In *Proceedings of the 8th Panhellenic Conference on Informatics*, Nicosia, Cyprus, 8-10 November 2001.

(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", Proceedings of the RANLP'2003 Conference, Borovets, Bulgaria, 2003.