NATIONAL CENTER FOR SCIENTIFIC RESEARCH
"DEMOKRITOS"

Institute of Informatics & Telecommunications

Software & **K**nowledge **E**ngineering **L**aboratory

Users Guide to *Ellogon*

Stergos D. Afantenos
George Petasis
Vangelis Karkaletsis

June 2002

Table of Contents

Chapter 1

1 Introduction

Natural Language Processing (NLP) emerged as a mixed field of Computational Linguistics and Artificial Intelligence, during the 1950's. Since then there is a constant outburst of interest, not only from people involved in academic research, but also from companies involved in the production and commercial exploitation of language engineering systems.

What all those people have in common is the need for a tool that will assist them on their research. Here, in the SKEL laboratory of the NCSR "Demokritos"[1], we have developed *Ellogon*, a multi-lingual, cross-platform text-engineering environment developed exactly to aid people who are doing research in Computational Linguistics, as well as companies which produce and deliver language engineering systems. *Ellogon* was developed in an attempt to create the necessary infrastructure to facilitate the development and distribution of various NLP tools.

1.1 What is *Ellogon*?

Before proceeding with what the *Ellogon platform* is, we would like to tell some words about what *Ellogon*, as a *word*, means and how come we chose that particular name. *Ellogon* is composed from the ancient Greek words εν + λόγος, which taken together mean "in accordance with logic". The *Ellogon platform* is actually in accordance with the logic, the logic that lurks underneath each writer of a text, and helps scientists, working in the area of NLP, exploit that logic and make bare all the information that lie inside a passage. But there is more to the story. "Logos", furthermore, can be translated something as oration, talk, utterance or written speech. We chose to name our platform *Ellogon* because it deals with texts, written speeches or passages in other words, and makes the information that lie inside such passages emerge.

Had we to describe the *Ellogon* platform within a few words, we could say that *Ellogon* is a general-purpose text-engineering *platform*. The word *"platform"* was emphasized here, in order to show that *Ellogon* is simply an environment and as such, it does not claim to perform any sort of linguistic processing. This is done with the

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

use of external embeddable components, which may be written either in Tcl/Tk or in C/C++ as described in section 3.6.

1.2 *Ellogon* architecture

The subsystems, that *Ellogon* consists of, are mainly three (see *Figure 1.1*):

1. A highly efficient core, the *Collection and Document Manager (CDM)*. The CDM is developed in C++ and it implements all the basic functionality of *Ellogon*. Its design is based on the TIPSTER[2] architecture but, despite this fact, it is not strictly compliant with it; for example, it does not provide the object-oriented framework defined by the TIPSTER. The main responsibility of CDM is to manage the storage of the textual data and the associated linguistic information. Furthermore, it provides a well-defined programming interface that can be used in order to retrieve or modify the stored information. CDM can be easily embedded into other applications.

2. A powerful and easy to use *Graphical User Interface (GUI)*. This GUI reveals to the developer the vast potential abilities of the CDM and the component system described below. Furthermore, this interface can be easily tailored to the needs of the end user.

3. A modular, pluggable component system. All linguistic processing within the platform is performed with the help of external, loaded at run-time components.

More information on collections, documents and components can be found in the Chapter 2. Chapter 3 describes the *Ellogon* GUI.

1.3 Benefits from using a Text Engineering platform like *Ellogon*

Ellogon is aiming at easing the process of developing and exchanging linguistic information and tools. It provides an infrastructure for managing, storing and exchanging textual data, embedding of text processing components as well as visualization tools. As was mentioned, linguistic processing is achieved through the embedment of external components. *Ellogon* provides the needed interfaces that enable these components to communicate with each other as well as to insert, query and modify the linguistic information that accompanies the textual data.

[2] http://www.gate.ac.uk

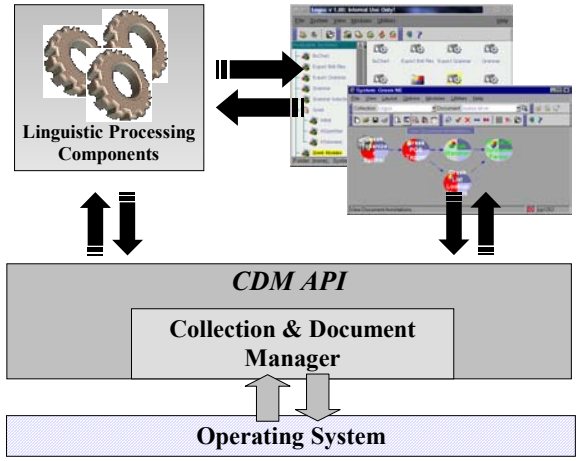

Figure 1.1 Ellogon Architecture

In a more concrete level, the most important benefits of *Ellogon* can be summarized as follows:

- *Ellogon* comes with built-in support for non-English languages. Both the core, as well as its GUI, offers full Unicode support, thus enabling the textual data to contain many different languages at the same time. A wide range of input and output filters exist that can handle a great number of widely used encodings (e.g. iso8859-7, windows 1253 or Macintosh Greek).

- *Ellogon* is extremely portable among different operating systems, as it has native support for the UNIX operating system (SUN Solaris, Linux and others) and Microsoft Windows (95/98, Millennium, NT 4 and NT 2000)

- *Ellogon* has reduced hardware requirements, due to memory compression done internally.

- *Ellogon* is fast.

- *Ellogon* offers an advanced and easy to use graphical interface, which is exactly the same under all supported operating systems. The GUI can easily be customized even by end users and, due to its modular design, can be extended or simplified to suit particular needs. The interface supports localization of all its messages into different languages. Currently, a localized version is available only for the Greek language.

- *Ellogon* offers compatibility with the GATE 1[3] text-engineering platform, offering the user the ability to transfer components easily from GATE 1 to

[3] http://www.itl.nist.gov/iaui/894.02/related_projects/tipster/

Ellogon. GATE 1 components written in the Tcl programming language (a high level scripting language) can run within *Ellogon* without modifications. *Ellogon* can even export/import textual data into/from GATE 1 native formats. Chapter 5 provides more information on GATE 1 compatibility.

Now that we have seen a general overview of the *Ellogon* platform, lets move on to the details. In the next chapter, we shall describe the *Ellogon* Data Model.

1.4 Acknowledgments

We would like to thank our colleagues from the NLP group of the University of Sheffield and especially the development team of the GATE 1 text engineering platform. Our cooperation with the University of Sheffield in the context of the R&D projects ECRAN[4] and GIE[5], motivated us to be actively involved in the area of text engineering platforms and make our first efforts towards the development of *Ellogon*.

[4] ECRAN is an R&D project on Information Extraction, partially funded by the EC (Telematics Applications / Language Engineering Action), 12/1995 – 02/1999. ECRAN partners were Thomson-CSF (France, coordinator), Universita di Ancona (Italy), Universita di Roma "Tor Vergata" (Italy), Smart Information Services GmbH (Germany), NCSR "Demokritos" (Greece), Friburg University (Switzerland), University of Sheffield (UK).

[5] GIE is a bilateral project (English-Greek) on named entity recognition, funded by the Greek General Secretariat of Research & Technology and the British Council, 05/1997-05/1999. GIE partners were NCSR "Demokritos" and the University of Sheffield.

Chapter

2

2 The *Ellogon* Data Model

Before proceeding with how we shall work with *Ellogon*, it is useful, indeed necessary, to understand what kind of information it stores and how it models it internally. It should be mentioned here that *Ellogon* shares the same data model as the TIPSTER architecture, yet it is not strictly compliant with it.

2.1 Collections – Documents

The central element of storing data in *Ellogon* is the *Collection*. A Collection is nothing more than a finite set of *Documents*. In other words, the Collection can be thought of as the corpus we want to process. Then the Documents of that Collection represent the *actual* documents of the corpus, bundled together.

An *Ellogon Document*, though, does not consist merely of text. It consists of *textual data* as well as *linguistic information about the textual data*. By textual data, we mean the actual text, which may come in a variety of encodings. An encoding is the internal representation, which a particular machine has, for representing the characters. For example, ASCII is a well known and used encoding. Linguistic information, on the other hand, comes in the form of *attributes* and *annotations*. When we create a Collection we specify to *Ellogon* which documents that collection will consist of. At that moment, the Documents may be plain text, but later on, after some linguistic processing takes place, they will be enriched with the aforementioned linguistic information, in the form of attributes and annotations.

2.2 Annotations

As was mentioned, linguistic processing is not done by *Ellogon* but by external embeddable modules called *components*. Components will be described shortly, but before, we have to see what those components are able to do on Collections and Documents. A component does not alter the textual data of a Document, but may instead add, modify, alter, delete or replace the linguistic information within it.

The linguistic information comes in the form of *annotations* and *attributes*. An annotation associates arbitrary information (in the form of attributes) with portions of textual data. We name such portions of data as *spans*. A span is characterized by

two byte offsets, denoting the start and the end character of the span. The start and end character are measured from the first character of the textual data.

Annotations typically consist of four elements (see *Figure 2.1*):

- *A numeric identifier:* This identifier is unique for every annotation inside a document and it 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:* This is a set of pairs of two byte offsets, as described above, that denote the range(s) of the annotated textual data.
- *A set of attributes:* These attributes usually contain the necessary linguistic information. Attributes are explained in the next section of this chapter.

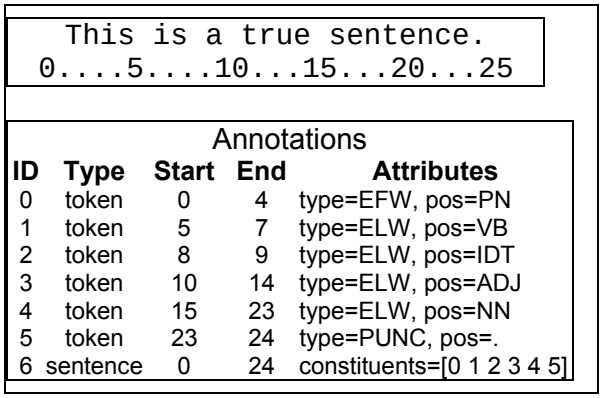

Figure 2.1 Example of a sentence and its relevant annotations

2.3 Attributes

When an annotation is constructed, certain information for it can be defined through its *attributes*. Attributes, actually, associate certain *types* of information with a typed value.

Things will become clearer with an example. Consider the table of *Figure 2.1*. The annotation, say, with numeric identifier (ID) 0, spans positions 0 through 4 of the textual data. Furthermore, its type is a token. Now, we have certain attributes for that annotation. For each token type, we have defined two attributes, namely type and pos (part-of-speech) that ought to take a certain value. In that particular case, they have taken the values type=EFW and pos=PN. In other words, the textual range referred by that annotation, is an English word with the first letter of it being a capital one (EFW). Furthermore, its part-of-speech is a pronoun, as the second attribute denotes.

For the same example, the annotation with ID 6 is a sentence spanning positions 0 through 26. The sole attribute of it is a *set* of other values (integers representing

Annotation IDs). This reveals to us that *Ellogon* does not only support strings as a value for a certain attribute. Currently supported types are *strings, sets of strings* and *annotations*.

2.4 Components

Components, as was mentioned before, are what carry on the linguistic processing in the *Ellogon* platform. Since linguistic processing will be the central point of interest for most users of *Ellogon*, components are an essential part that ought to be understood.

Each component consists of two parts. The first part is responsible for performing the linguistic processing. The second part of a component can be thought of as carrying out the communication between the linguistic subcomponent part and the *Ellogon* platform. In other words, the main responsibility of the second component part is to interface the linguistic processing subcomponent with *Ellogon*, through the provided API.

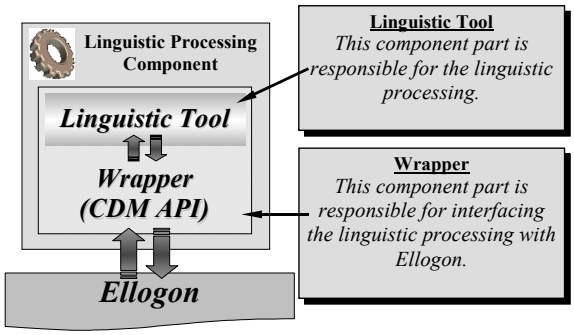

Figure 2.2 Component Structure

Components can appear, either as *wrappers* or as *native components* (see *Figure 2.2*). The job that a wrapper usually carries on is to 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.

Ellogon Components should collaborate with each other and with *Ellogon* platform. In order for this to be achieved, each component should have a set of *pre-conditions* and a set of *post-conditions*. 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. That information is used by *Ellogon* in order to establish relations among various components, and organize them into *Systems*, as we shall describe shortly.

Each component, furthermore, can also specify a set of parameters, as well as a set of viewers. Parameter values 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.

Creating components in *Ellogon* is quite a straightforward task, for which the *Ellogon* GUI aims at assisting a component-writer. Currently, *Ellogon* supports two programming languages in which a component can be written, C++ and Tcl. The *Ellogon* GUI offers a specialized dialog where the user can specify various parameters of the component he or she intends to create, including its pre-conditions and 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++, a *makefile* for compiling the component under UNIX 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.

2.5 Systems

Components are the basic units of *Ellogon* platform that perform a specific task. It would be useful if we could bundle a set of components that collaborate together in order to achieve a specific task. For example, before doing any part-of-speech tagging one usually needs a tokenizer and several other modules, to act on the corpus. Such a facility is supported by the *Ellogon* platform through *Systems*.

A System is nothing more than a set of cooperating components that performs a specific task. The tasks can range from basic linguistic operations, such as part of speech tagging or parsing, to application level tasks, such as information extraction or machine translation, etc.

Ellogon provides all the needed infrastructure for organizing components into Systems. Once several components have already been created and are available to *Ellogon* platform, we can view them all together and select the ones that we believe are appropriate for the task at hand. *Ellogon* then creates the links between the components, showing us in a graphical way the order in which we should run the components of the System.

More information on the aforementioned topics we shall see on the next Chapter, "The *Ellogon* Data Model".

Chapter 3

3 Working with *Ellogon*

In the previous chapter, we saw a general overview of the main elements that constitute the functionality of *Ellogon*. In this chapter we shall see, in more detail, how are we going to use them in order to work with *Ellogon*.

3.1 *Ellogon* GUI

What the casual user of the *Ellogon* platform sees and deals mostly with, is its powerful, multilingual *Graphical User Interface*. *Ellogon* GUI provides users with the ability to manage Collections, Documents and Systems, to visualize linguistic information with an extensible set of visualization tools, to develop and integrate linguistic components, to browse documentation and of course, to do linguistic processing of textual data using various modes.

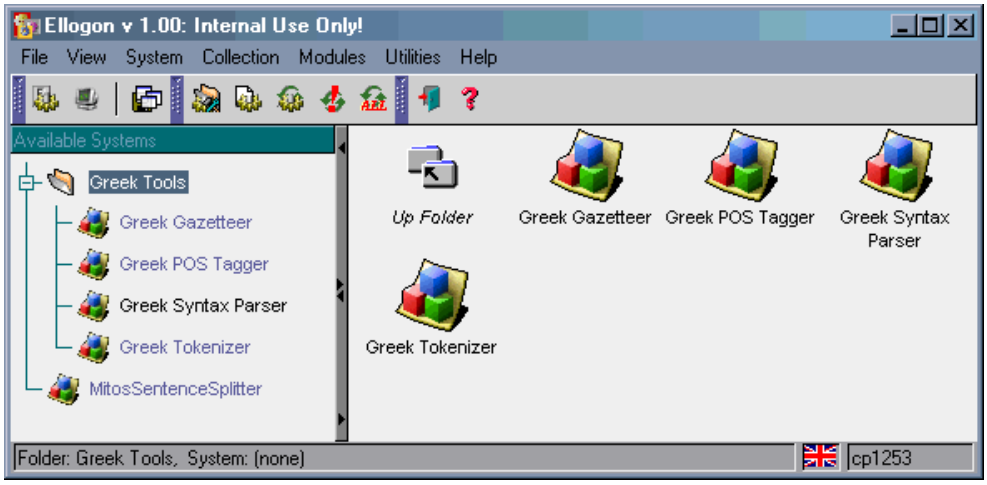

Figure 3.1 Ellogon Main Window

Once you have successfully installed *Ellogon* to your system (see chapter 6 for more information on installing *Ellogon*), you can invoke the platform simply by double-clicking on the *Ellogon* icon on your desktop, if you have a Windows system, or doing something analogous on a Unix platform. This action will open the main window of the *Ellogon platform*, as shown in *Figure 3.1*.

On the left hand side of the window, we can see that the *Ellogon* GUI has a tree. The nodes of the tree are folders which contain the *Systems,* which are represented as the leafs of the tree. We would like to mention that *Ellogon* comes equipped with some predefined Systems on installation which perform various tasks. The folders and Systems of the tree are also shown on the right hand side of the window.

When we double click on a System, a separate window opens, containing the *Components* of the System and the links between them, as shown in *Figure 3.2*. Those links were added by *Ellogon*, for it knows, through the pre-conditions and post-conditions of every component, what relations it should establish among them. Pre-conditions and post-conditions were discussed in the previous chapter under the heading "Components".

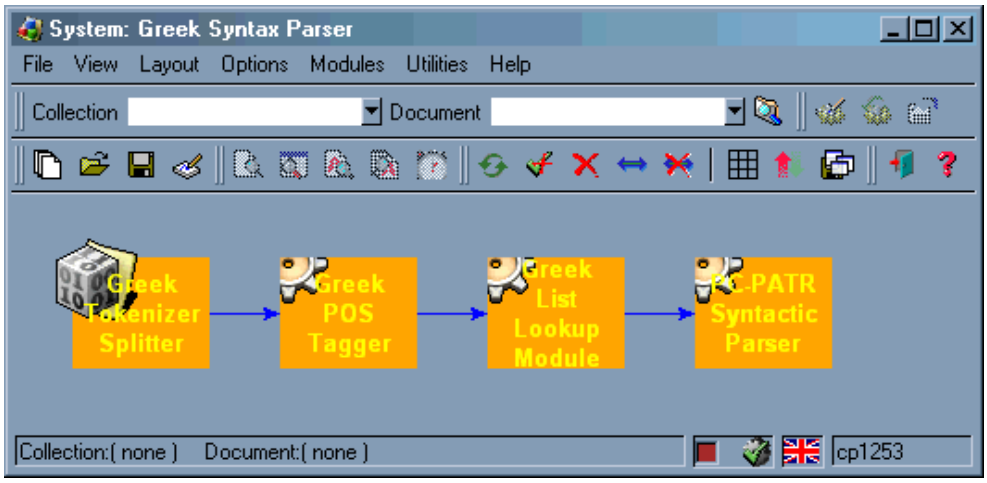

Figure 3.2 Example System

Furthermore, the GUI contains a set of toolbars, a status bar and a menu. The details of them will be discussed in this and the subsequent chapters.

3.2 Creating Collections

Collections were described in the previous chapter. In this section, we shall see how to create and manage Collections.

In order to create a new Collection, from the *Ellogon* main window (*Figure 3.1*) select `Create Collection` from the Collection menu. A new dialog, as shown in *Figure 3.3*, appears.

In this dialog, you can specify various parameters. Collections are usually stored under a folder named "Collections" on the home directory[6] of the user. Under that folder one usually creates a new subfolder that contains the Documents of the Col-

[6] Under a UNIX environment the home directory is the home directory of the user that runs *Ellogon*. On a Windows System *Ellogon* searches for certain environment variables such as `HomeDir` or `Home`. If none is defined, then *Ellogon* assumes `C:\` to be the home directory.

lection. To specify such a folder, you can either specify a Collection name, or push the associated button (see *Figure 3.3*) to browse for a folder for the newly created Collection.

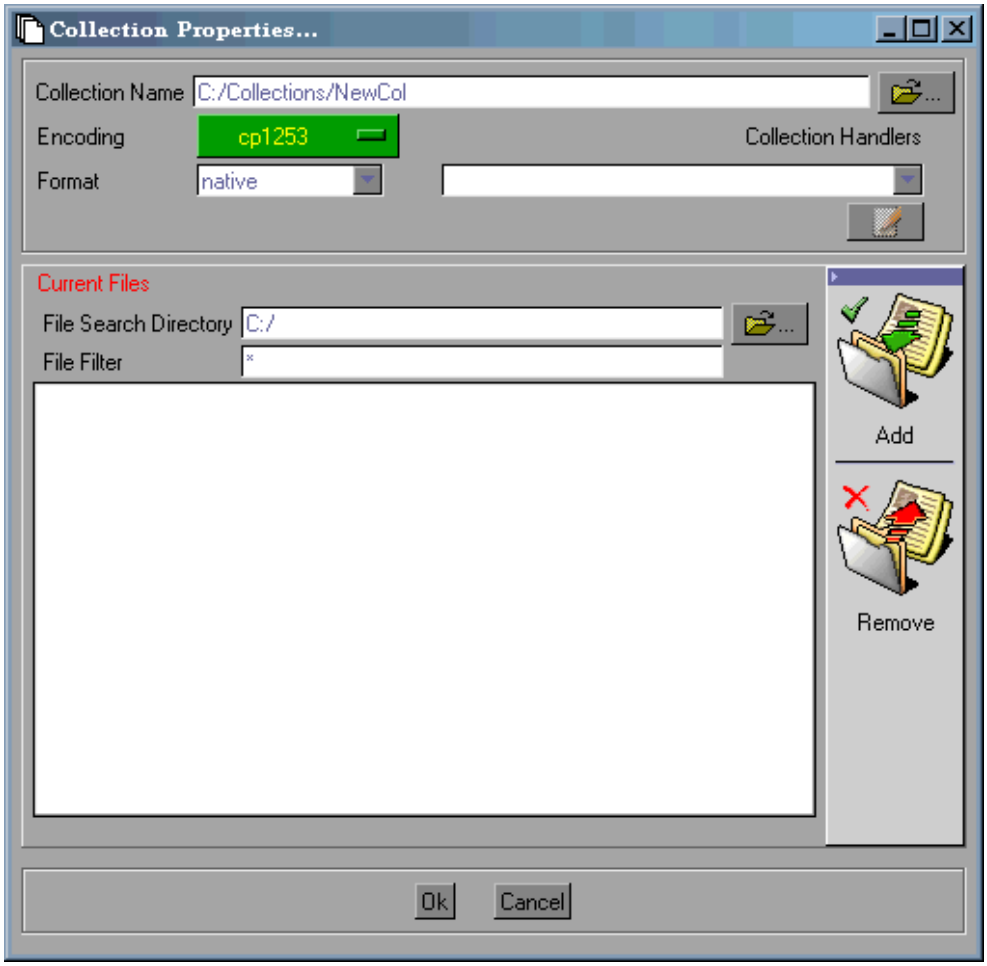

Figure 3.3 Creating a Collection

Immediately below the Collection name, you can choose an encoding for your Collection, from various sets of encodings. Remember that an encoding is the internal representation of a machine for characters. *Ellogon* generally, uses Unicode as the encoding in which it represents the Documents of the Collections. Albeit this fact, several Documents may come in various encodings. *Ellogon* has to know in which encoding the Documents of a Collection are written, so that it will be able to transform that encoding to Unicode.

Next to the encoding you can choose a handler for the Collection. You can imagine a handler as a module which performs some specific job on the Collection *at the creation time*. In other words a handler is a piece of code which is executed on every Document of the Collection once that Document is added to the Collection. Usually, you will have to write your own handlers.

Ellogon comes equipped with two predefined handlers: the "Import HTML images" handler and the "SGML tags to Annotations" handler. The first one is used

for HTML pages, and what it does is to import the images along with the HTML code in the Collection. The second handler converts SGML tags into *Ellogon* Annotations. More information on the handlers and how you can create on, you can find on the *Developers Guide to Ellogon*.

Furthermore, you can specify in what format you want your Collection to be stored (see section "Changing the Collection Format" for more information on what is a Format). Then, you have to specify which texts are going to compose the Collection that you are about to create. This is done, either by specifying a path that the text files are to be found, and adding a *filter* (e.g. *.txt), or simply by selecting the files that are going to compose your Collection from the file manager of your operating system (e.g. Windows Explorer under the Windows platforms) and dragging them over to the Dialog. At that moment, the Dialog changes color, indicating to you that you can drop the files, letting *Ellogon* arrange for you the details (e.g. the path that the files belong to).

If mistakenly you added a file or more to the Collection, that you shouldn't have, then you can easily remove them before creating the Collection. Simply highlight those documents, by pointing your mouse and clicking on them, and then press the "Remove" button on the right hand side toolbar (see *Figure 3.3* again). In case that no files are selected and you press the "Remove" button, then the files will be removed according to the file filter. Furthermore, if you have defined a search path, you can add all the files that exist in that path by simply clicking the "Add" button on the same toolbar.

Once a Collection has been created, the source texts are copied to the new Collection folder. *Ellogon* in no case shall it modify the source files; it will leave them intact in their path.

3.3 Managing Collections

In order to manage a Collection, you choose `Edit Collections` from the `File` menu of a System, or push the associated toolbar button. Then a similar Dialog to that depicted in *Figure 3.3* opens (see *Figure 3.6*). The operations of managing Collections are available from the `Collection` menu or from the associated buttons on the right hand side toolbar. In what follows, we are going to describe those operations in more detail.

3.3.1 Opening Collections

In order to process the Collection you have created, you have to open it. This can be done in two ways. The easiest is to go to the *Collection Toolbar* (see *Figure 3.4*) on the window of a System and select a Collection from the associated drop-down list.

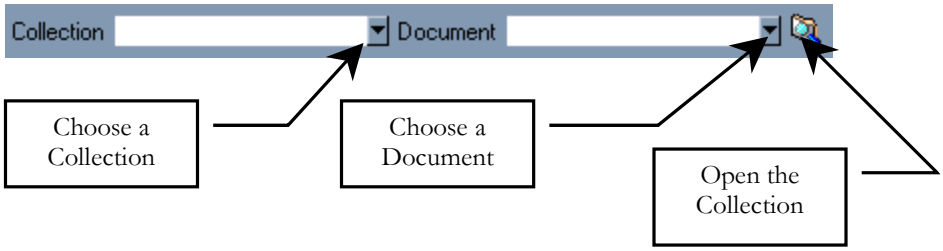

Figure 3.4 The Collection Toolbar

After selecting a Collection, you have to specify to the *Ellogon* platform which Documents of the Collection you want to process. Next to the Collection drop-down list, you can find the Document drop-down list. From that list, you can choose the file you need, or choose all the files of the Collection, which is the default. Afterwards you simply push the button next to the Document drop-down list (see again *Figure 3.4*), so that your choice will be executed. Actually, this toolbar is a shortcut for the operations available in the "Open Collection" dialog, described next.

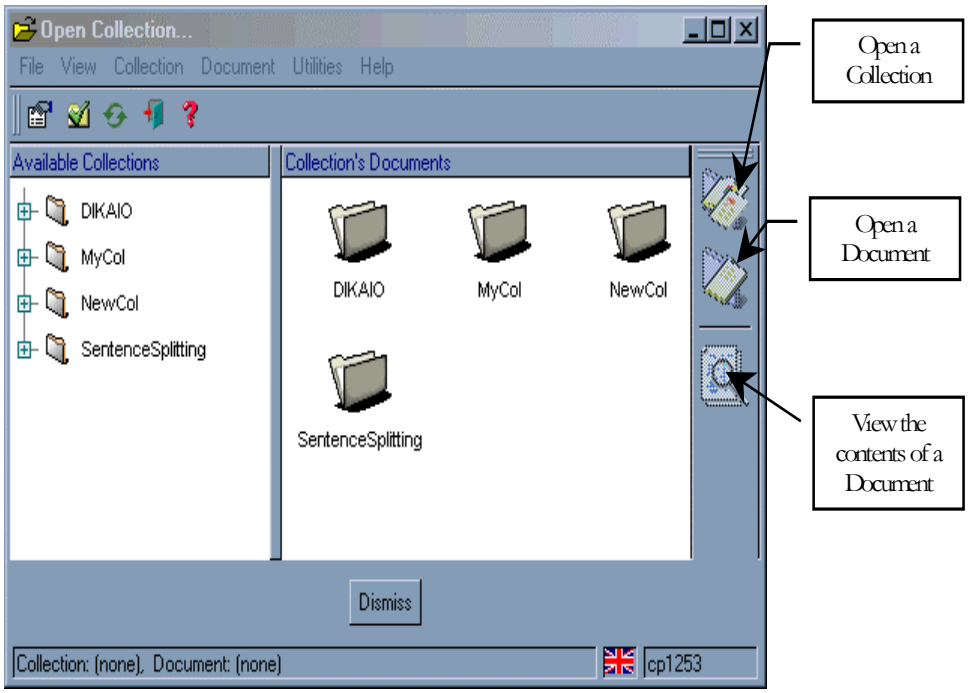

Figure 3.5 The Open Collection Dialog

In order to invoke the "Open Collection" dialog, from the `File` menu of the System that you are currently working, choose `Open Collection`. Once you have done this a new dialog opens, as shown in *Figure 3.5*. In that dialog you can see all the Collections that you have created thus far, and open the one you are interested in. If you want to open a whole Collection select the Collection and either push the associated toolbar button (see *Figure 3.5*), or right click on the Collection and choose `Open the Selected Collection` from the context sensitive menu that appears. If you want to open just a Document of a Collection for proc-

essing, in the same dialog, open the Collection you want, either by double-clicking on it, or by expanding the tree view of the Collection by pushing the + sign. Then, select the Document and push the associated button as shown in *Figure 3.5*, or right click on the Document with your mouse and select `Open the Selected Document`.

Sometimes you may not be sure as to which Documents you want to open because you are ignorant of the contents of those Documents. Fortunately, *Ellogon* gives you the opportunity of having a glance at the contents of the Documents you may want to open. This is done by selecting the Document and then pushing the appropriate button as shown in *Figure 3.5*.

3.3.2 Deleting a Collection

In case you want to delete a Collection, you must first choose the Collection to be deleted, in the Edit Collections Dialog (see *Figure 3.6*), and then select `Delete the Selected Collection` from the `Collection` menu. This operation is also available as a shortcut from the right hand toolbar of the Dialog.

Once you have selected the menu item or pushed the associated toolbar button, *Ellogon* will ask if you are sure you want to delete that Collection. If you answer yes, then you must know that the entire Collection, including its Documents will be deleted. That is, any work you have done within a Collection will be lost, including the attributes and annotations of the Documents. Nevertheless, the source files, from which the Collection was created, will remain intact.

3.3.3 Editing and Resetting a Collection

Editing a Collection means you can add or remove Documents from it. This is accomplished by choosing a Collection and then selecting `Edit the Selected Collection` from the `Collection` menu, or by pushing the associated toolbar button. A dialog similar to that in *Figure 3.3* appears.

If you want to edit a Collection, bear in mind that all the attributes and annotations of the Documents of the Collection will be deleted. *Ellogon*, before doing anything disastrous, asks you whether you want to proceed, reminding you what the consequences of your action will be.

3.3.4 Creating New Documents for a Collection

When we described how to edit a Collection we implied that you can add new Documents to that Collection, only that the attributes and annotations of the *previous* Documents would be lost. If you do not want to loose any important information for you, but you want to enrich your Collection with new Documents, then you should simply create new Documents for your Collection. This is done by choosing a Collection and then selecting `Create a new Document in Selected Collection` from the `Collection` menu, or by pushing the associated toolbar button.

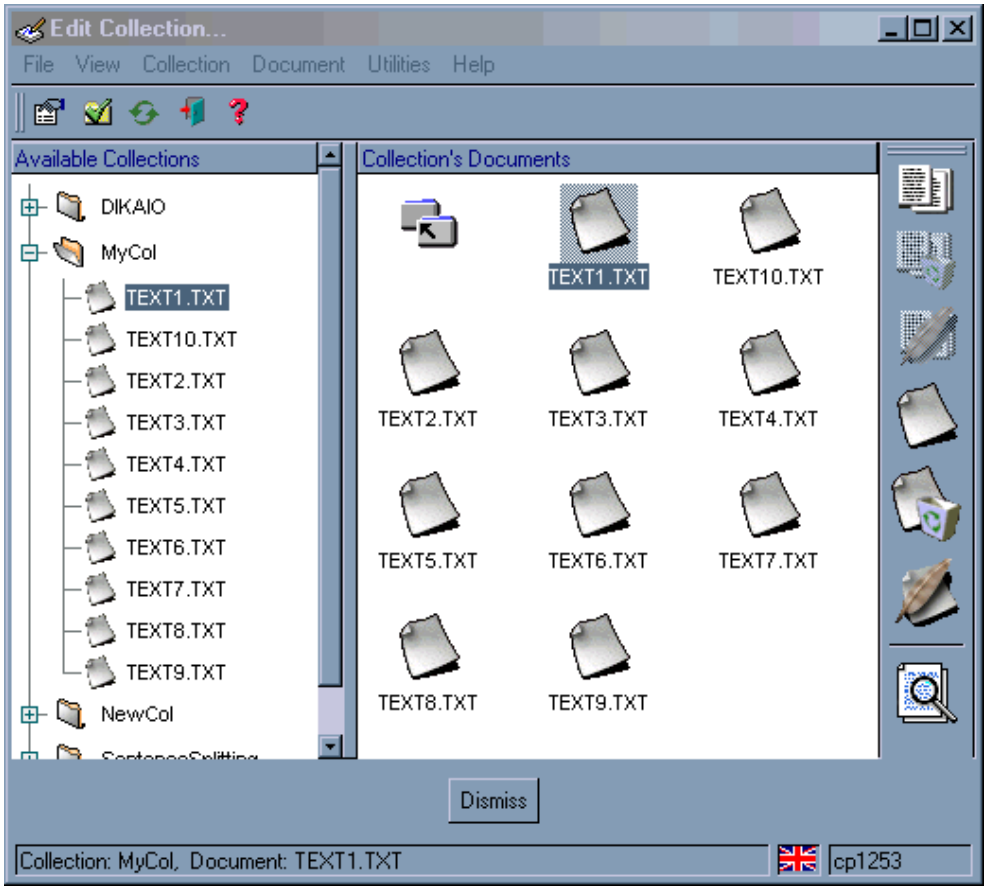

Figure 3.6 Managing Collections

Once you have performed that action, a dialog appears that enables you to search for a Document in your hard disk. After selecting the file that you want to insert as a new Document to your Collection, another Dialog appears, depicted in *Figure 3.7* that prompts you to choose an encoding for your new Document. You may choose an encoding for various predefined sets of encodings. In that dialog, you are able, furthermore, to see a part of the contents of your new Document.

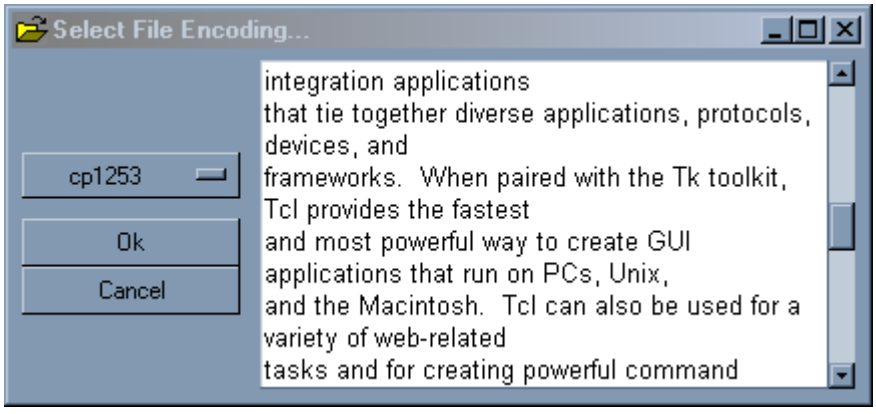

Figure 3.7 Selecting Encoding for a New Document

3.3.5 Deleting Documents from a Collection

Again, when we described how to edit a Collection, we implied that you may delete some Documents of your Collection, but you would loose the attributes and annotations that the rest of the Documents contained. If you do not want that to happen, you should choose to simply delete some Documents from your Collection. This is done by choosing a Collection and then selecting `Delete Documents from Selected Collection` from the `Collection` menu or by pushing the associated toolbar button.

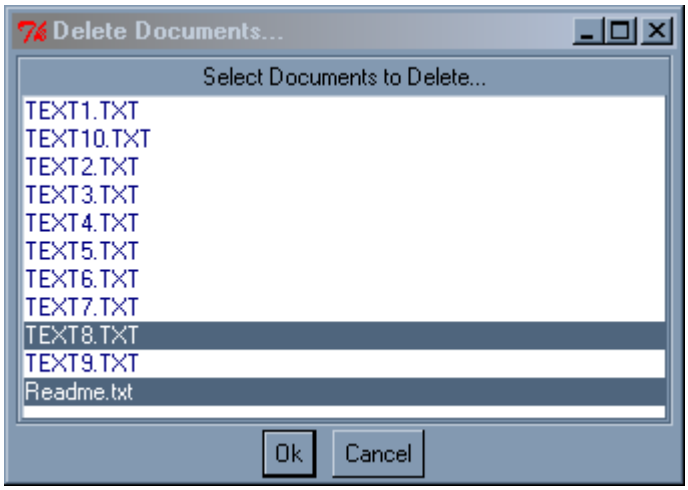

Figure 3.8 Deleting Some Documents From a Collection

Once you have performed that action, then a Dialog is prompted as in *Figure 3.8*, which lets you select one or more Documents from your Collection and remove them. When you have selected the Documents, simply press OK. *Ellogon* will ask you if you are sure, in which case it deletes the selected Documents with no further ado.

3.3.6 Changing the Collection Format

Before describing how you will be able to change the Format of a Collection, it would be useful to explain what we mean by Format. The Format of a Collection is simply the internal representation of a Collection by which it is going to be stored on your hard disk. The details of a representation are not needed for the purposes of this manual, so we won't describe them here. Currently, *Ellogon* supports two kinds of Formats. The first is the *native* format, the format that *Ellogon* stores Collections. The second Format that *Ellogon* can handle is the GATE 1 format. This Format is supported, for compatibility purposes with the GATE 1 platform. In other words, if you have already worked with a Collection on the GATE 1 platform and you now want to move on the *Ellogon* platform, you can easily do it, since *Ellogon* supports the GATE 1 format.

Now we can describe how to change the Format of your Collection. From the "Edit Collection" Dialog (*Figure 3.6*), choose the Collection whose Format you want to change. Then from the `Collection` menu, choose `Change Collection Format`. A Dialog will appear as shown in *Figure 3.9*. Then from the

Format drop-down menu (encircled in *Figure 3.9*) choose the Format you wish, either *native* or *gate* and then press the Ok button.

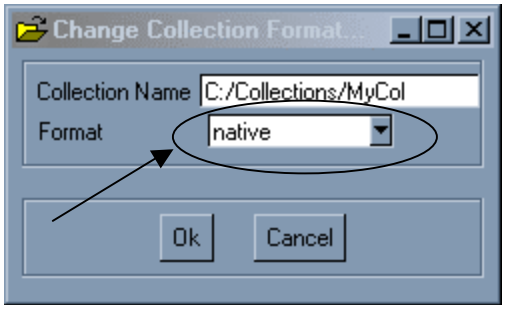

Figure 3.9 Changing the Format of a Collection

3.3.7 Renaming a Collection

When you created a Collection, you provided a name for it. If you decide at a later stage that this name was not descriptive, you may change it. This is done by choosing a Collection and then selecting `Rename the Selected Collection` from the `Collection` menu.

When you choose that option a dialog as shown in *Figure 3.10* will appear. As you can see you do not simply rename your Collection, but you may also provide a new folder for your Collection to be stored in. The name of the folder will be used as the new name for your Collection. The old folder and all of its contents will be deleted from your hard disk.

Figure 3.10 Renaming a Collection

3.4 Creating Systems

Ellogon comes with a variety of Systems preinstalled, but most probably, you will need to create your own Systems. In order to create a new System, from the `System` menu choose `Create New System` or push the corresponding toolbar button, as shown in *Figure 3.11*.

Once you have done this, a new Dialog Box, depicted in *Figure 3.12*, appears. Apart from the Menu items and the toolbar, that dialog is split up into two regions. In the top region, you can see the components you have chosen so far. In *Figure 3.12* there is no component to be shown since we have not selected any so far. In the bottom region we have a tabbed pane in which the components that are available to the *Ellogon* platform are listed.

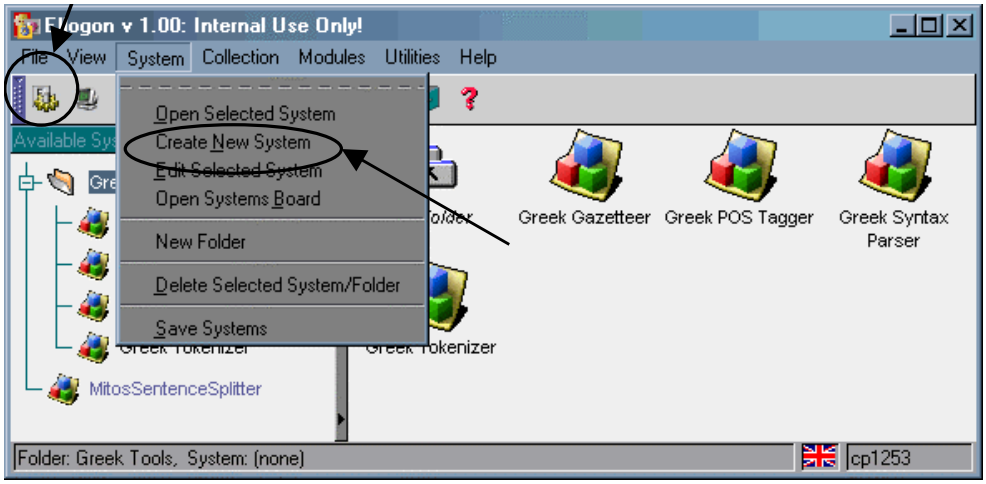

Figure 3.11 Creating a New System

To create a new System, you simply choose the components that your System is going to consist of, by clicking on them. *Ellogon* then, links the components in the order they should be called. The information for linkage is taken from the pre-conditions and post-conditions of every component, in case they exist. If pre-conditions or post-conditions do not exist, then no links are added among the components.

An example System has been created in *Figure 3.13*. As you can see, we have simply selected the components of that example System and all the connections between the components has been provided by *Ellogon*, so that we have a graphical way of representing our System. Once we have chosen all the components that are going to compose our System, we simply push the Save button on the toolbar, or choose `Save` from the `System` menu of the Dialog. Then a new Dialog opens, as shown in *Figure 3.14*, asking us to enter a name for our System and a Folder that it will be saved in. Be careful to select a name for your System that it will be descriptive of the purpose that it was created for. Nevertheless, you can change the name of a System as described in section 3.5.3.

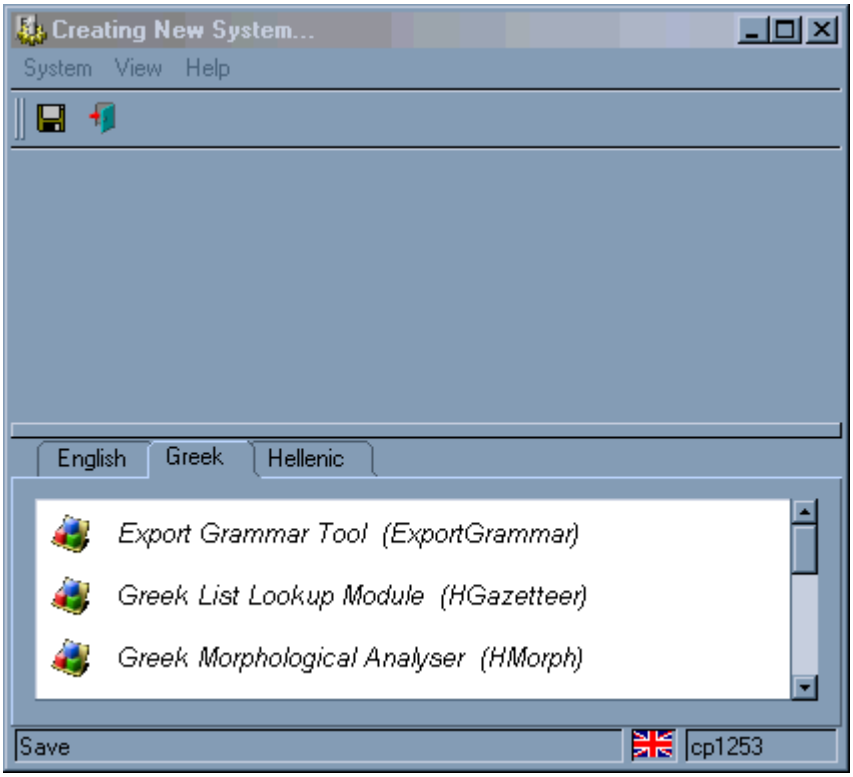

Figure 3.12 A Dialog for Creating a New System

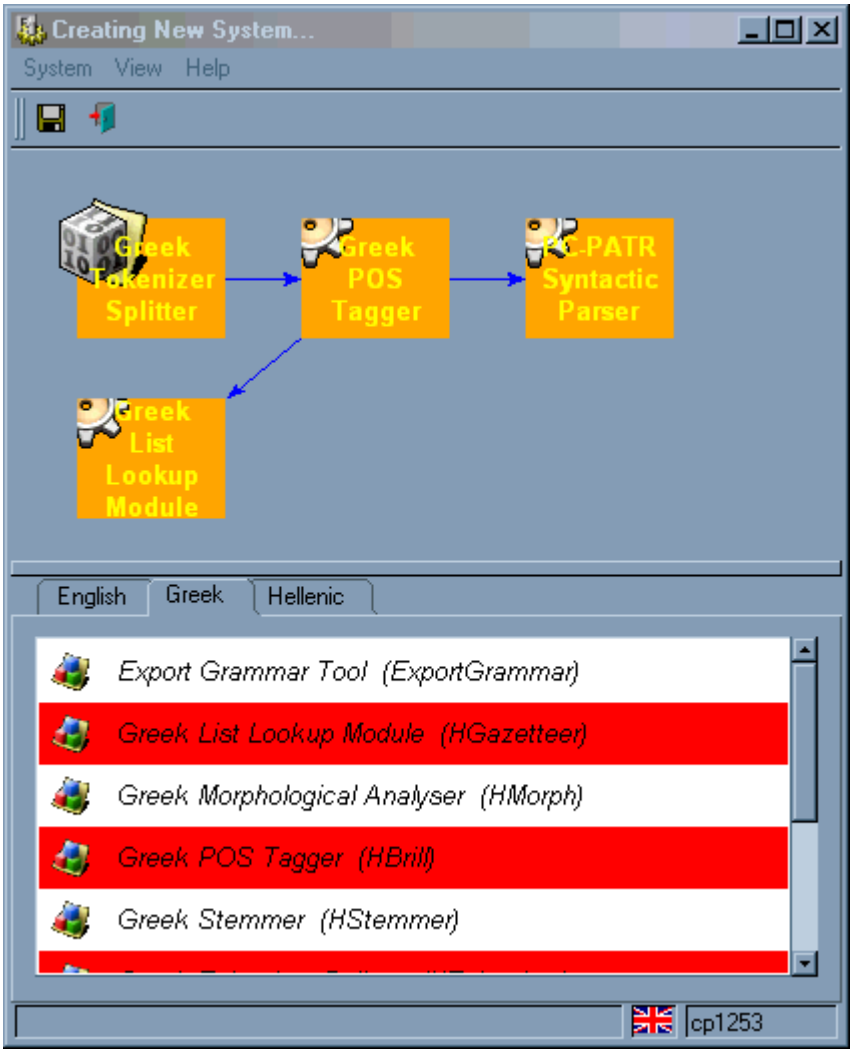

Figure 3.13 Creating a New System

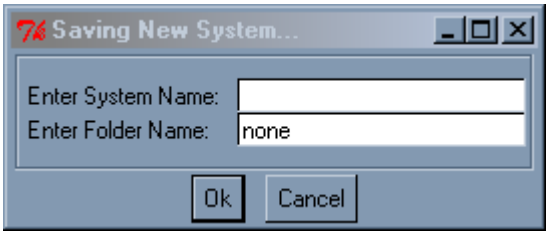

Figure 3.14 Saving a New System

3.5 Managing Systems

If you have created several Systems on the *Ellogon* platform, then you will most probably want to manage them and organize them in a way that it will be easy for you to navigate through them and use them. In this section we shall describe how to organize and manage your Systems.

3.5.1 Organizing systems into Folders

The most obvious way to organize your Systems is to create several *Folders*, each of which will contain various Systems designed for various tasks. Creating a folder is very easy in *Ellogon*. In the main window of the *Ellogon* platform (see *Figure 3.1*) simply choose `New Folder` from the `System` menu. A dialog box appears, depicted in *Figure 3.15*, asking you to provide a name for your new folder. Once you have entered a name for the folder press Ok.

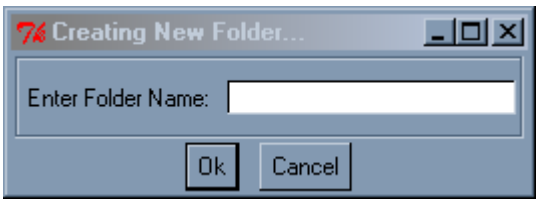

Figure 3.15 Creating a New Folder

In *Figure 3.16* we have created a new folder named "My systems". In that folder we have placed a System that we already have created. In order to place a System into a folder of *Ellogon* you may either drag and drop that System into that folder from another one, or you may create a new System into that folder.

In this section we shall only describe how to drag and drop Systems into folders; for information on how to create new Systems see section 3.4 "Creating Systems", above. If you are familiar with the Windows platform and especially with the file manager of Windows, the Windows Explorer, then you need no explanations on dragging and dropping; the process is exactly the same. What you simply have to do, is to click on a System from another folder and, having your mouse button pressed, *drag* your mouse on the folder you want, carrying the System with the mouse pointer. Once your mouse pointer is on top of the folder you want to place the new System, simply release the mouse button, thus *dropping* the System into the folder you desire. Exactly the same process as the drag-and-drop of the Windows platform. All along the way, you may notice your mouse pointer changing appearance and color. This is to indicate to you where you can drop your System and were not, i.e. if you are on top of a folder or not. If you want to drop your System on the root folder of the *Ellogon* platform, you have to drop it on the "Available Systems" label (see *Figure 3.1*)

When you right-click with your mouse on a folder, a context-sensitive pop-up menu appears with two options. The first one is for creating a new System, and it is actually a shorthand for creating a new System as was described in section 3.4. The new System then, will be placed inside that folder. The other option of the pop-up menu is to delete your folder. If you press it *Ellogon* will ask you if you want to delete that folder *and all of its contents* (i.e. all of the Systems that lie in that folder); it then proceeds according to your choice.

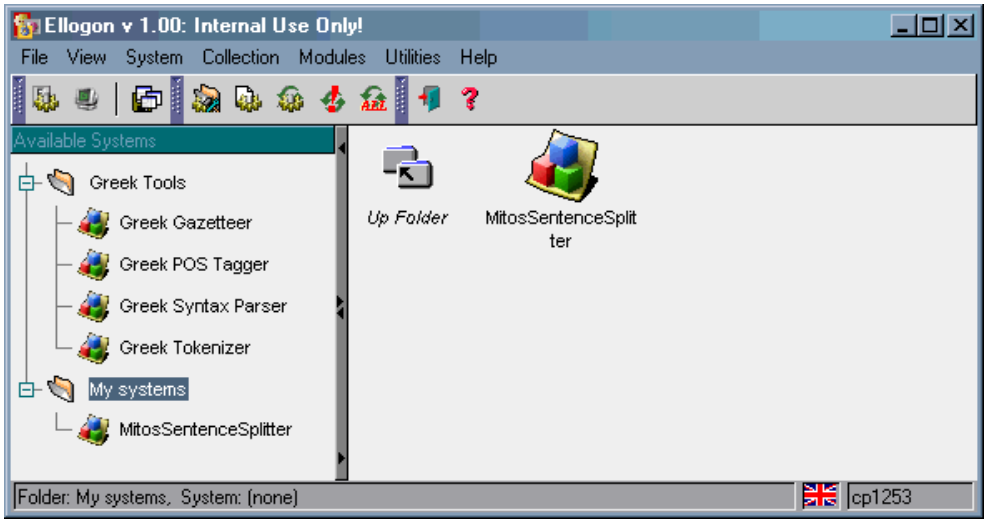

Figure 3.16 Having Created a New Folder

3.5.2 Editing a System

If you have created a System, then you may realize after some work with it, that you want to modify that System, for example by adding a new component to it or changing the relative placement of the components, etc.

Adding/Removing Components from a System

To edit a System simply choose the System you want to edit from the *Ellogon* main window, by clicking on it, and then choose `Edit Selected System` from the `System` menu. What appears is a dialog, similar to that in *Figure 3.13*. The process of modifying a System is exactly the same as the one for creating a new System. You can add or remove components, observe the connections that *Ellogon* establishes between those components and arrange the relative placement of the components into the System window, for your optical convenience. Once you have finished with the modification of a System you have to save your changes. This is done, either by pushing the Save button of the Toolbar, or by selecting `Save` from the `System` menu. Once you close that dialog, the System window opens up.

Changing the Appearance of a System

In case a System is composed of many Components, then the placement that the *Ellogon* will perform on them may not be very helpful, and you may want to modify it. This is easy to do. Simply open the System you want and, having the control button (Ctrl) of your keyboard pressed, click on a Component of the System and drag it. While you move the Component a tool-tip will appear showing to you the coordinates of the Component, so that you can you them for fine adjustment of the placement. If you have made your mind on where to place the component, simply release the mouse button and the Ctrl button.

Furthermore, you can modify the appearance of the arrows indicating the dependencies among the Components or you may even add more dependency arcs or remove any one. In *Figure 3.17* you can see the toolbar that will assist you on those operations. But let's take things one by one.

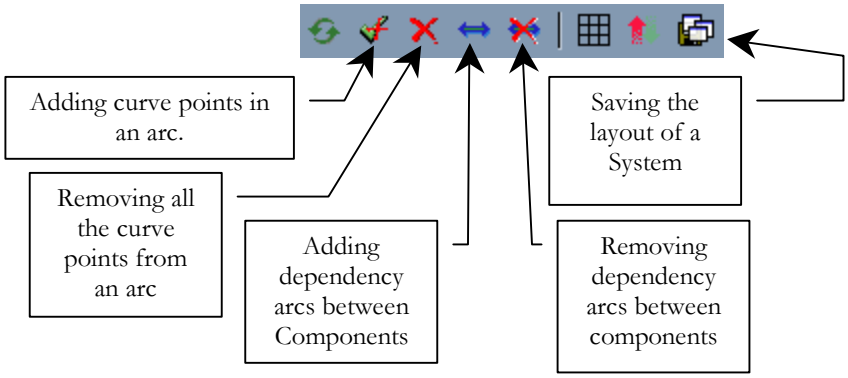

Figure 3.17 The Toolbar for Formatting the Appearance of a System

Adding Curve Points in Arcs

The dependency arcs are represented as straight lines when you firstly create a System, but you may change their appearance add make them appear as curve lines. In order to do so you have to add a *curve point* on an arc. On an open System choose the arc you want (you will understand that the arc has been chosen if it is more bold than usual). Then from the toolbar press the appropriate button, as shown in *Figure 3.17*. A curve point will appear on the arc you selected, which you may drag all aver the System so that you can adjust the way that the arc will look. Once you have done it once, you may want to do it again. No problem; you can add as many curve points as you wish on an arc. Simply repeat the same process. A funny example is shown in *Figure 3.18*. In that figure we have added three curve points in the wavy arrow.

Removing Curve Points From Arcs

In case you have overdone it with the curve points there is no reason to worry. You can remove all of them quite easily. Choose the arc with the many curve points and push the appropriate button on the toolbar, as shown in *Figure 3.17*. All the curve points from that arc will disappear, and your arc will look normal again.

Adding Dependency Arcs between Components

Dependency arcs are added between Components form the *Ellogon* platform, after examining the pre-conditions and post-conditions of the various components that compose the System. In case *Ellogon* didn't insert a dependency arc, but you still want to have one between two components, you can easily insert one. Push the appropriate button on the toolbar, as shown in *Figure 3.17*. A tool tip will appear asking you to select the component that the arrow will leave from; select a component. Another tool tip will appear, asking you to select the source component; select another component. In case the arc already exists, *Ellogon* will inform you of this fact; otherwise it will insert the arc.

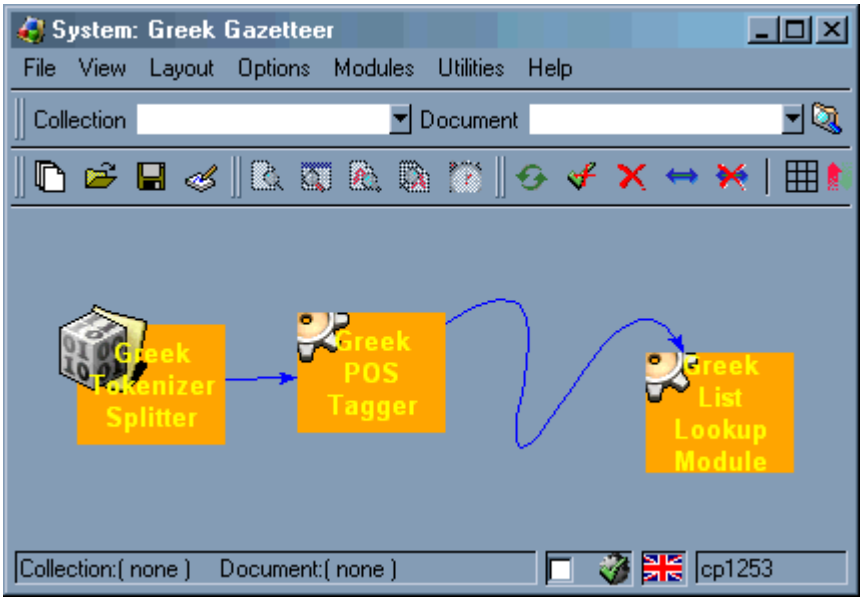

Figure 3.18 The result of adding three curve points in an arc

Removing Dependency Arcs between Components

In case a wrong dependency arc was inserted, or you wouldn't like to have the ones already inserted by *Ellogon* during the creation process of a new System, you may remove as many dependency arcs as you wish. In *Figure 3.17* you can see the button that once you push it will remove the selected arc. In case no arc wad been selected *Ellogon* will remove no arcs.

Saving the Layout of a System

After making all the necessary visual modifications on a System, it is *crucial* to save the layout of the System. This can be done in two ways. The first is to push the appropriate button on the toolbar, as shown in *Figure 3.17*. The second is to select `Save Layout` from the `Layout` menu. Once you have done it, the System in the future will look exactly as it looks at the time you saved it. If you do not save the layout all the changes you have made will be lost.

3.5.3 Renaming a System

If you have created a System and later decide that the name you provided to it was not descriptive for the purpose you designed it, then you may change its name. This is done, either by right clicking on the System and selecting `Rename Selected System` or by selecting the System and choose the same option from the `System` menu of the *Ellogon* main window, depicted in *Figure 3.1*.

3.5.4 Deleting Systems and Folders

In order to delete a System or a folder simply right click with your mouse on the System or Folder you want to delete. On the pop-up menu that appears click Delete System or Delete Folder. Alternatively, you may select the System or Folder in the *Ellogon* main window (*Figure 3.1*) and then from the `System` menu select `Delete Selected System/Folder`.

3.6 Creating Components

Ellogon comes equipped with a set of components, but most probably you shall need to create your own ones, in order to perform some sort of linguistic, or other, processing on Collections. In this section we shall describe how to create components from scratch. In the section that follows we shall describe how to edit already defined components.

Components, in the jargon of *Ellogon* are called *modules*. In order to create a new module, from the *Ellogon* main window (*Figure 3.1*) select `Create New Module` from the `Modules` menu. A new dialog, depicted in (*Figure 3.19*) will appear. As you can see from the figure, that dialog enables you to insert various information for your component. For example you can give it a name, define the pre-conditions and post-conditions, associate various viewers with the component and set any parameters that you may wish.

But, let's take things from the beginning. *Ellogon* supports two programming languages in which you can write your modules. The first programming language is Tcl and the second is C++. No matter which language you will write your component in, you must provide a name for it; this is done by writing the name of your module into the text area entitled "Module Name" (see *Figure 3.19*). That name will be used as the name of your file containing your source code. Valid names do not contain spaces and some other characters such as (; , : . -) In general valid component names are valid names of C++ functions.

Next to that text area is a drop down list with two options. The first option creates Tcl code whilst the second creates C++ code. In case you choose to write your module in Tcl, E*llogon* will create two files. The first is named `creole_config.tcl`. The name of the second is composed from the name you provided appended with the `.tcl` extension. The `creole_config.tcl` file contains several useful pieces of information about your module and it is enriched analogously, according to the information you provide during the creation of a new module. More details we shall see shortly.

If, on the other hand, you choose to create C++ module, *Ellogon* will create three files for you. The first is the `creole_config.tcl` file and contains exactly the same information as before. The second is your source code and its name composed from the name you provided plus the `.cpp` extension. The third file that is created is a *makefile* for compiling the source code of the component.

Once you have provided the information about the name of your module and the programming language that it will be written in, you have to specify a folder in which the above mentioned files will be stored. This is done either by explicitly writing the path, or by pushing the "Select Directory" button (see *Figure 3.19*) and choosing the directory or even by dropping a folder from the file manager. Note that once you have specified a folder for storing the source code of your component, *Ellogon* will create a new folder *under* the one you specified named according to the name of the module you gave. For example, if you have created a module

named `Test` under the directory `C:/modules`, *Ellogon* will put the necessary files into the folder `C:/modules/Test`.

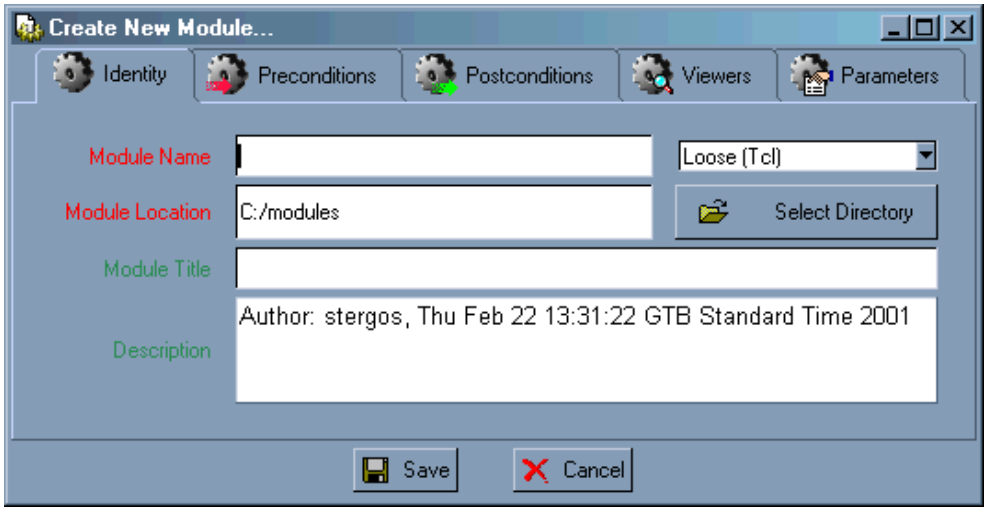

Figure 3.19 Creating a New Module

Providing a name and a folder for your module is mandatory information. Apart from that information, you may as well provide some *optional* information. Optional information include a *title* for your module and a short description. The title you provide is actually what you see on the components of a System. The short description you provide is inserted to the source code of the module as a comment. It can contain several information such as the author of the module, for example, his or her institute etc.

3.6.1 Setting Pre-conditions and Post-conditions

In order to insert pre-conditions and post-conditions into a newly created module, click on the Pre-conditions or Post-conditions tab of the dialog depicted in *Figure 3.19*. Pre-conditions and post-conditions are optional information; if you wish you may ignore them altogether. Bear in mind though that pre-conditions and post-conditions are used by *Ellogon* in order to establish dependencies among the modules.

Pre-conditions and post-conditions come in three forms:

1. Collection attributes
2. Document attributes
3. Annotations

Collection attributes are intended for the Collection as a whole. In other words if you have defined a Collection attribute as a pre-condition to a module, then that module, before processing the Collection, will search for that pre-condition; if a Collection attribute is defined as a post-condition, then the module will save that information to the Collection, after it process the Collection. Document attributes, on the other hand are intended just for a single Document.

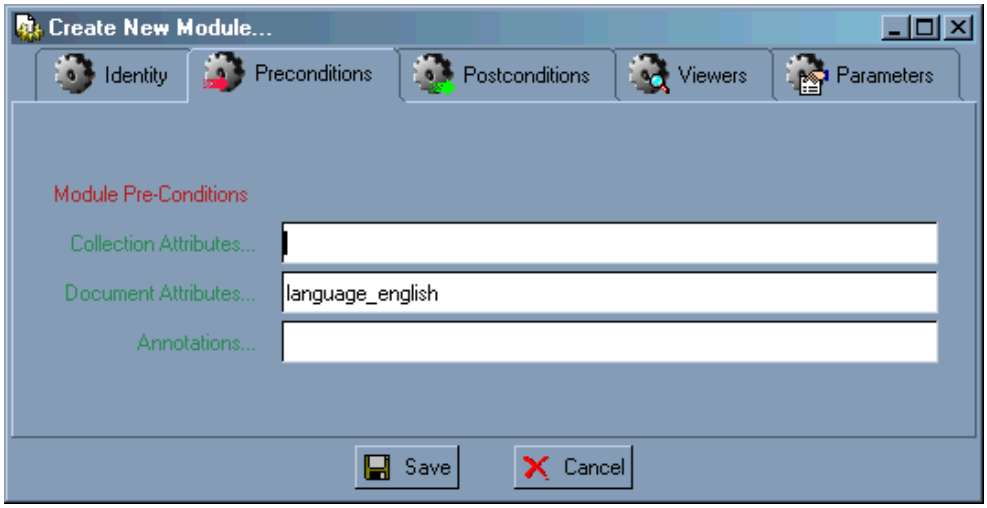

Figure 3.20 Setting Pre-Conditions on a New Module

Annotations, if set as pre-conditions, describe the information that a module needs to find in Document in order to run. If they are set as post-conditions, they describe the information that the module will write on the Documents that it will process. For example, if you want to say that the module you are about to create needs to find the annotations `token` and `sentence`, with the second annotation having the attribute `constituents`, then you shall write: `token {sentence constituents}` on the Annotation field of the dialog depicted in *Figure 3.20*.

3.6.2 Associating Viewers with a Module

With a viewer you are able to see several pieces of information of the processed Documents graphically. Currently *Ellogon* supports three kinds of viewers: Single Span Viewer, Raw Viewer and Tree Viewer; they are described in section 3.8.3. If you want to associate a viewer with a module click on the "Viewers" tabbed pane (see *Figure 3.21* taken from a UNIX platform this time).

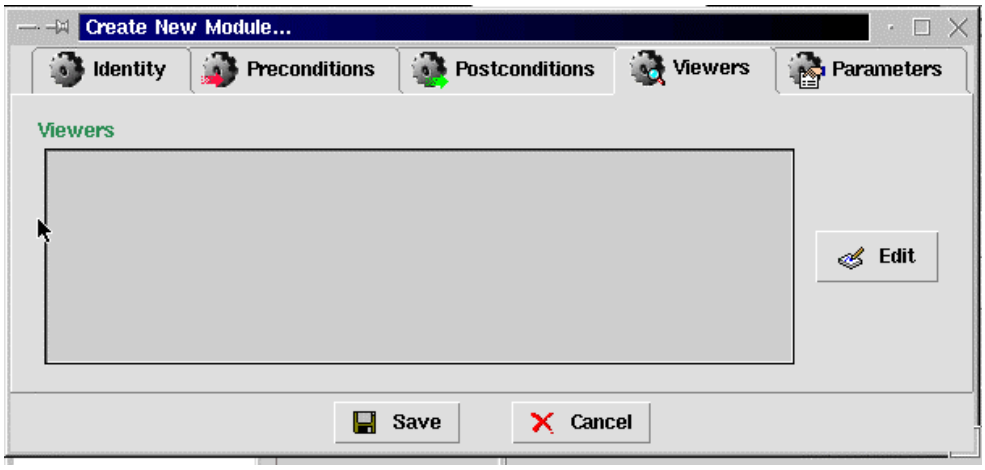

Figure 3.21 The Viewers Tabbed Pane

If you push the Edit button on the right of that tabbed pane, a new dialog opens, as depicted in *Figure 3.22*. From that dialog you are able to choose the viewer that you want for your module, and associate with it several pieces of information. The first thing that you should do is to select a type of viewer, as shown in *Figure 3.22*. Then you have to tell the viewer which annotations it is going to graphically represent. In order to do so, simply write the annotations in the appropriate text field of *Figure 3.22*. Apart from that information, you have furthermore to provide a name for your viewer so that the user of your module will be able to invoke it. The name of the viewer is written on the right hand side text field shown in the same figure.

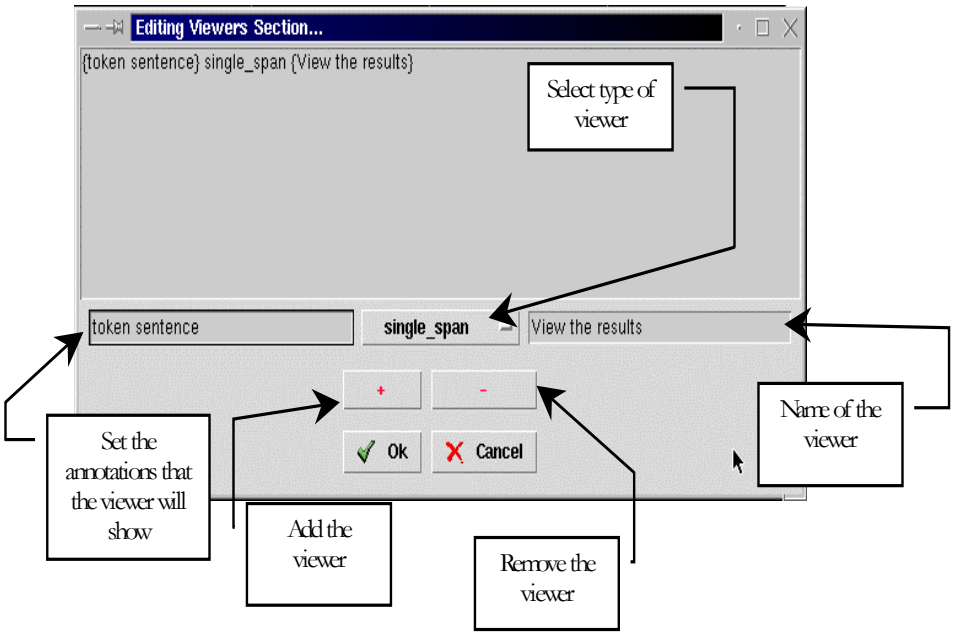

Figure 3.22 Associating Viewers with a New Module

Once you have finished simply push the add button (indicated by a plus sign). The result will be written in the text area of the dialog of *Figure 3.22*. Note that this is a text area, so you are able to go there, after having added a viewer and change any information you need. If you want to remove a viewer select it with your mouse and push the remove button, indicated by a minus sign (see *Figure 3.22*)

Once you have added to your module all the necessary viewers, simply push the Ok button. If you want to exit from that dialog without saving any changes push the Cancel button.

3.6.3 Setting Parameters

Some modules, in order to run, may need several parameters. For example a module may need to find some information located on a file, etc. While creating a module you are able to specify such parameters. From the "Create New Module" dialog, select the "Parameters" tabbed pane. What appears is shown in *Figure 3.23*.

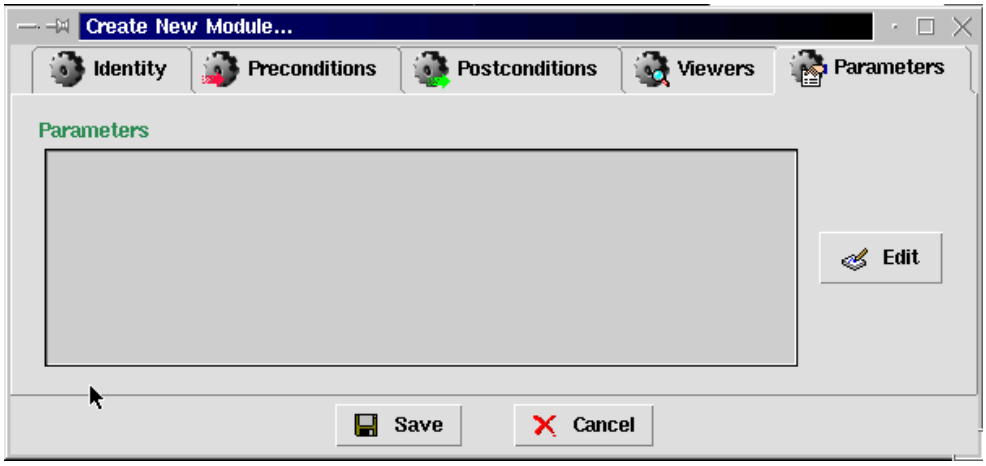

Figure 3.23 The Parameters Tabbed Pane

On the right hand side of that dialog push the Edit button in order to define any parameters to your new module. It will appear a new dialog, as shown in *Figure 3.24*. *Ellogon* supports eight types of parameters: File, Directory, Number, Combo Box, Encoding, Command, Boolean and Other. You select the type of parameter from the button shown in *Figure 3.24*. Then you should supply a name for the parameter, so that the users of the module will be able to understand what that parameter is about, and an optional default value, in appropriate text areas, shown in *Figure 3.24*. When you are finished push the add button; this will insert the information on the text area of the dialog, so that you may modify them manually. If you want to remove a parameter, select it from the text area, and push the remove button.

If you supply a Directory or File parameter to your new module, then, when you edit later the parameters, a text area containing the default value and a button for browsing the file or the directory will appear. If you provide a Number parameter, only a text area and the default value will appear. Providing a Boolean parameter only a check box will appear, with the name you supplied. If you provide a Combo Box, a combo box will appear with some predefined values for the user to select. In case you select Encoding, the user will be able to select an encoding. Finally if you provide a Command parameter, a comment and an OK button will appear. If the user pushes the button, the Tcl command you provided will be executed. Note that this Tcl command can take advantage of the whole *Ellogon* API (see *Developers Guide to Ellogon* for more information on the API). In *Figure 3.26* you can see the result of adding some parameters in a module.

3.7 Editing Components

Once you have created a module you have to edit it; to provide some code for it so that it will perform the desired processing on Collections. If you open a System that contains the desired module you want to edit, click on it and from the toolbar for editing the modules select the appropriate button as shown in *Figure 3.25*. This action will open up your default editor for editing the module. If you want to

change the default editor, from the *Ellogon* main window (*Figure 3.1*) select `Options...` from the `File` menu. In the dialog that will open select the Operating System tabbed pane. There you can specify manually the directory of the executable file of your favorite editor, or browse for it.

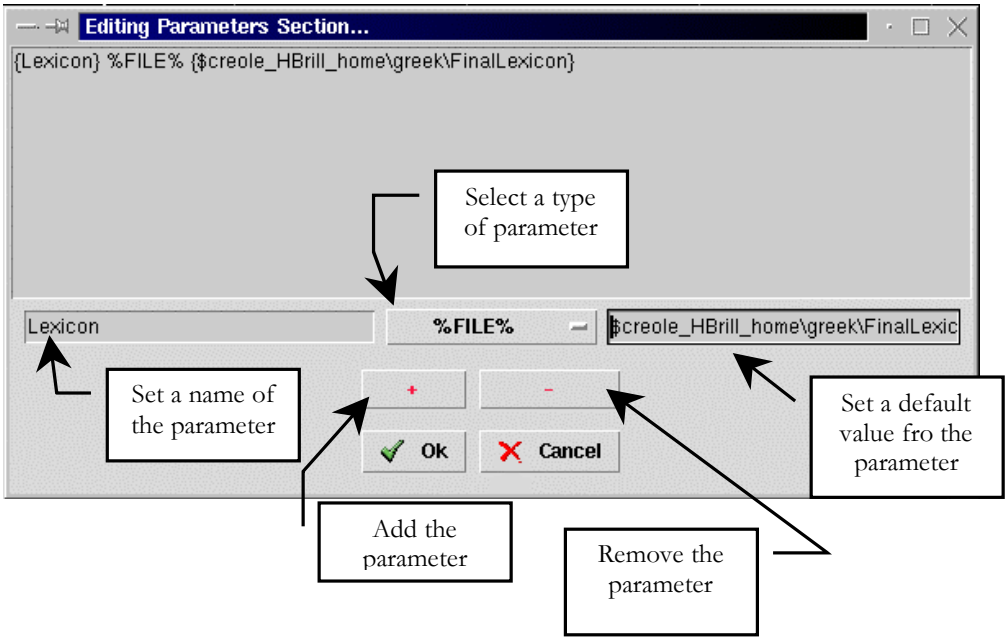

Figure 3.24 Setting Parameters on a New Module

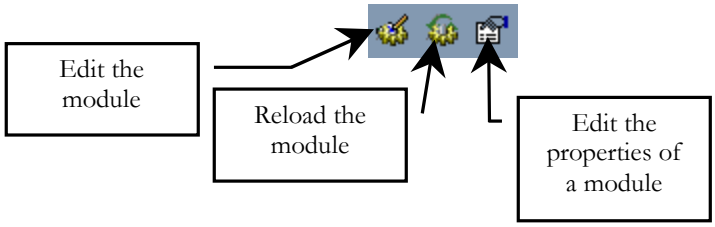

Figure 3.25 The Toolbar for Editing components

Reloading Components

Once you have edited the source code of your component, you have to reload it in order for *Ellogon* to work with the new code of the component, instead of the old one. This is done by pushing the reload button on the edit components toolbar, as shown in *Figure 3.25*. In case you want to reload all the modules of *Ellogon*, you may select `Reload All Modules` from the `Modules` menu, either from the System that you are currently working, or from the *Ellogon* main window. Bear in mind though, that only the modules written in Tcl can be reloaded. Modules written in C++ cannot be reloaded after they have been recompiled, as a program cannot discard a loaded dynamic library and load it again. The user has to exit *Ellogon* and restart it in order to use a changed dynamic component in C++.

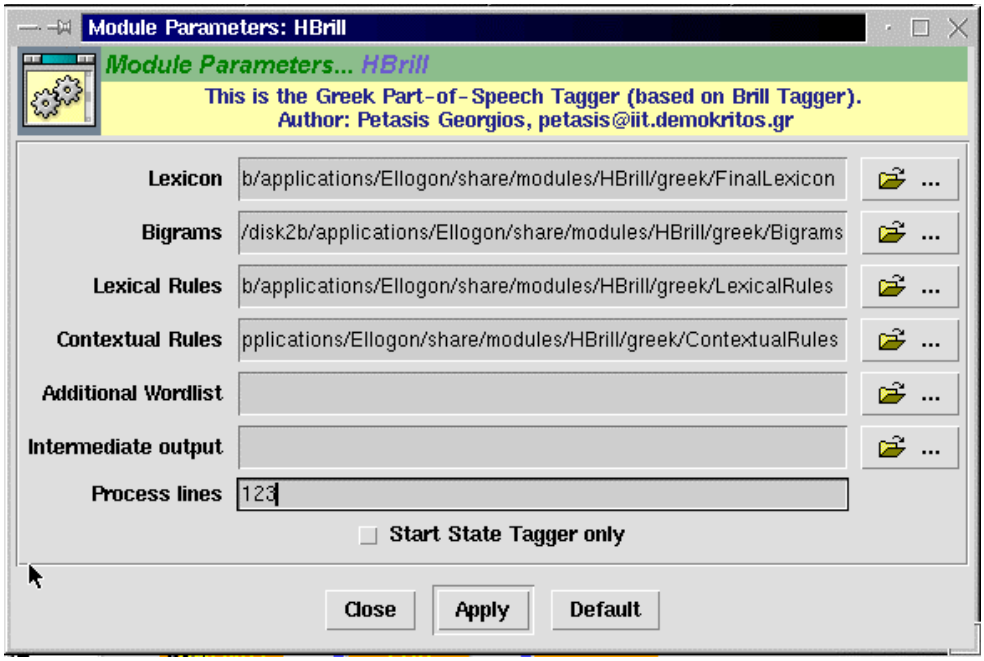

Figure 3.26 Editing the Parameters of a Component

Editing the Parameters of Components

When you created a component (section 3.6), you provided some parameters for it. If you want to provide values to those parameters, instead of the default ones, you can do it by pushing the Edit Parameters button, as shown in *Figure 3.25*. A new dialog will open similar to the one shown in *Figure 3.26*.

In that dialog, as you can see, we had declared the first six parameters as File or Directory parameters, so apart from the name for them and the default value (empty on the last two) a button appears so that we cam browse for the file or the directory. The "Process lines" parameter was declared a Number parameter, so we can insert a numerical value into it. The last parameter was declared as Boolean, so there appears a checkbox for it.

3.8 Running Systems

In order to run a System you have firstly to open it from the *Ellogon* main window. This is done by either double-clicking on the System, or selecting it and choosing `Open Selected System` from the `System` menu. Then you have to open a Collection for processing, as described in section 3.3.1.

When you open the Collection, then you shall notice that several Components alter color. This is to indicate to you that they are ready for processing the Documents of the Collection, i.e. all the information that the Components need (made known to *Ellogon* via their pre-conditions), is available to the Collection.

Then you have to select the components that you want to process the Documents with. Their color will alter once again indicating that you have selected them and

are ready to run; possibly the color of the Components that are connected with them with dependency arcs will change. The *order* by which you select Components matters. *Ellogon* remembers that order and if you later run the components, they will run according to the order you selected them.

If you make a second click on the Components you want to run, then they will process the Documents of the Collection you have selected. As mentioned, they will run in the order you selected them. If you decide to run a component that is connected with several other Components that haven't yet processed the Documents, then *Ellogon* will first run all the Components that are depended with the Component you selected, going back as many levels as needed.

When a component finishes processing the Documents, then its color alters once again, showing that it has finished processing the Documents. Furthermore, during the process time, an indicator shows you the progress made so far by the Component, so that you can estimate the time needed in order for the Component to finish. Several other useful information may appear as well.

In *Figure 3.27* we have opened a System that contains three Components. Then we opened a Collection and chose just a single Document of the Collection to be processed by the System.

Then we chose to run the first Component on the Document. The processing time (in user time, not CPU time) that the Component needed in order to process the Document is shown to us in the lower left hand side of the System (see *Figure 3.27*). The second Component of the System is selected and so its color is altered, indicating to us that with another click it will start processing the selected Document. Finally, the last Component of the System has not been selected, and thus has its initial color.

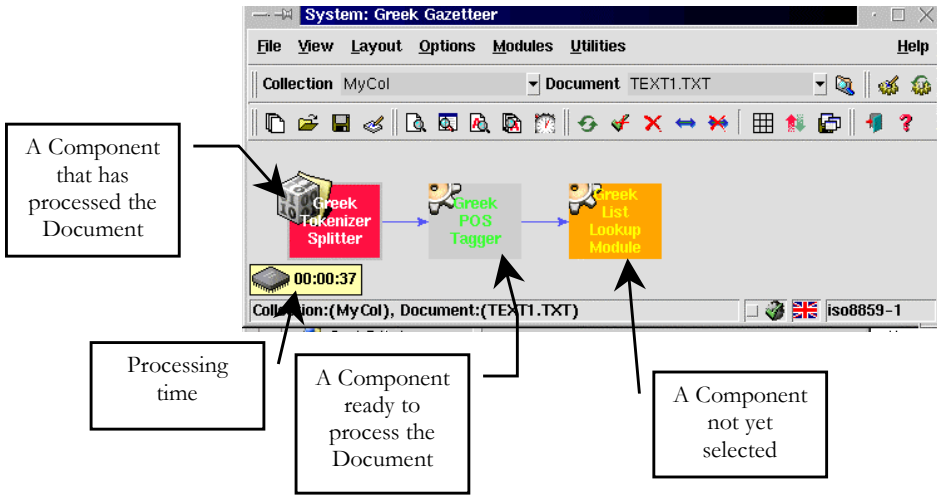

Figure 3.27 Running a System

3.8.1 Resetting Modules

In case you have run a Component and you do not want the results of its processing the Collection to be stored in the Documents, you can undo that processing.

This is done by right clicking with your mouse on a Component. A dialog box will appear asking you whether you want to reset the module in which you right clicked. If you answer "No", nothing happens; if you answer "Yes", then *Ellogon* will undo all the changes that the component made to the Collection. Furthermore, if that Module is followed by other components that have already processed the Document\Collection, it will undo the changes that those modules had as well done to the Collection. You have to bear in mind that this undo action is based on post-conditions of the module.

Right clicking on a module that was selected, but has not yet run, has the effect of unselecting that module.

3.8.2 Running modes

When running Systems, *Ellogon* offers you the ability you to run the System in various modes, each one providing a specific functionality. You can enable those modes from the `Options` menu of a System.

- ***Ignore Changes:***
  When you enable that option, then whatever changes a module does on a Document of a Collection, they are discarded once the selected for processing modules finish processing that Document. In other words, each module is processing the Document as it ought to have done, but after all selected modules have finished processing the module, the memory that was allocated for that Document is freed and the modifications are lost.

  The reason that one might want to discard the changes is that some modules while processing a Document create several other auxiliary files containing useful information; one might want just to keep those files without creating any changes to the Documents of the Collection and without overloading the memory.

- ***Log Errors:***
  When a syntactic, or other, error occurs in a module, *Ellogon*'s main policy is to stop running the module and inform you that an error has occurred. This policy can be turned off, by selecting the option `Log Errors` from the `Options` menu of a System. In that case when an error occurs, *Ellogon* informs you of that error, but it, nevertheless, continues with the processing of the next Document of the Collection. The errors that occur are kept in a Log, so that you can later track them. This Log is accessible from the `Error Log` option of the `View` menu in a System.

- ***Auto-Save Collections:***
  This option has effect only when you have opened a whole Collection, instead of just a single Document of the Collection (see also section 3.3.1). When *Ellogon* processes a Document or a whole Collection, all the changes that the various modules make on that Collection are being kept in memory. They are actually written on your hard drive only when you explicitly tell *Ellogon* so (saving the results from the Collections' processing was described in section 3.8.4).

If you enable the Auto-Save Collections option, then after all selected for processing modules have run on a Document of a Collection, the changes are saved in your hard drive and the memory used for storing the Document and its annotations and attributes, is freed before *Ellogon* retrieves the next document from the Collection for processing. In other words when you have that option enabled you are gaining in memory space.

3.8.3 Viewing Results

Once you have processed the Documents of a Collection with a Module, then you most probably want to see the results of the processing on the Documents. As we described in section 3.6, when you create a new module, you are able to associate several viewers with that module. With the viewers, you are able to see the results of processing a Document with a module, in a graphical manner.

In order to see the results with a viewer, click on a module that has processed a Document or a Collection with your mouse. A context sensitive pop-up menu will appear, which lists all the viewers that are associated with that particular component. All you have to do is to select a viewer from the list, by clicking on it. If you have selected just a single Document for processing, then a new window with the viewer will appear. In case you chose a whole Collection for processing, then *Ellogon* will inform you that you cannot open a viewer for a whole Collection and it will prompt you to choose a Document from the available documents of the Collection. Once you choose the Document, the viewer will appear.

When you associated a viewer with a module, you gave it a description and explicitly said to *Ellogon* which annotations that viewer will graphically show. The description of the viewer you provided appears in the context sensitive pop-up menu, while the annotations will appear on the viewer window. While you open that viewer, *Ellogon* retrieves from memory the relevant annotations that the viewer is going to show and then presents them graphically to you.

The first viewer we shall describe is the Annotations Explorer viewer, which is depicted in *Figure 3.28*. As you can see from that figure the Annotations Explorer consists of three parts. In the upper left is the raw text of the Document you have opened. On the left you can see all the annotations with a checkbox on their left and a plus/minus sign on their right. On the bottom of the viewer you can see the annotations along with several information about them. The information consist of the annotation ID, its type, the start and end spans and the attributes it may have.

When you click on the checkbox, on a particular annotation the text spans corresponding to that annotation will be highlighted in the raw text. Each annotation will be highlighted with a different color. Deselecting a checkbox will disable highlighting for that particular annotation.

Clicking on a plus sign, makes it become a minus sign and adds all the annotations of that type into the lower part of the viewer along with their information. Clicking on a minus sign removes all the annotations from there.

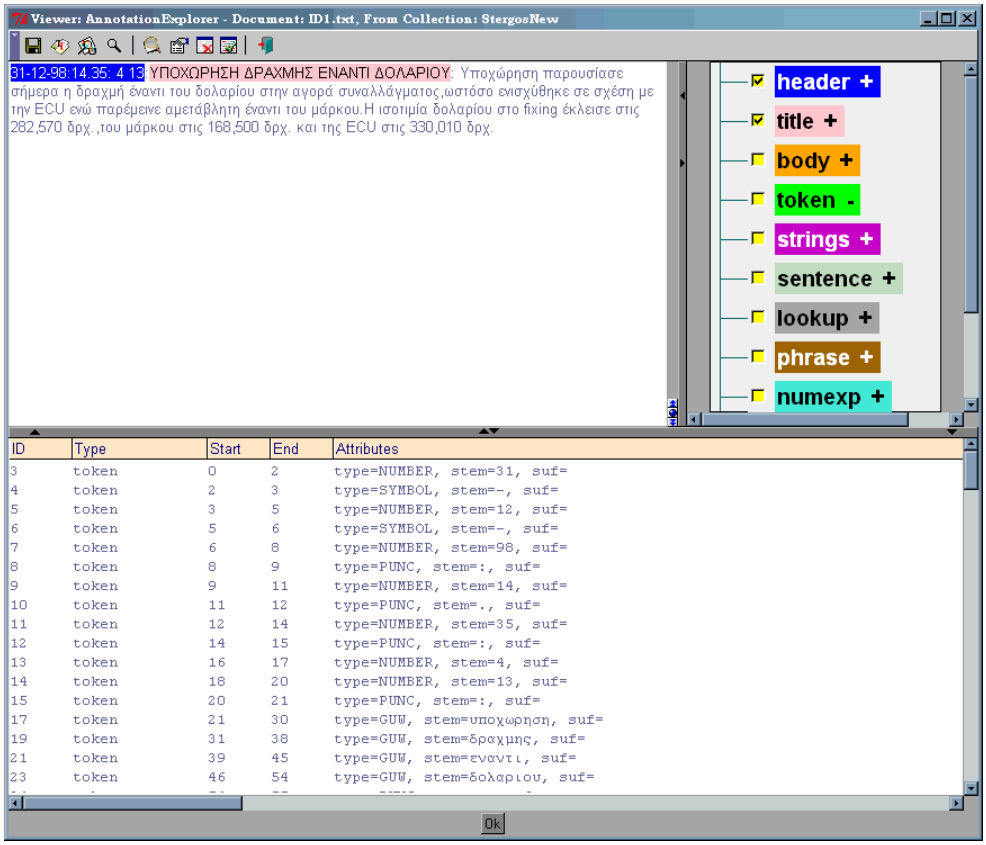

Figure 3.28 The Annotations Explorer Viewer

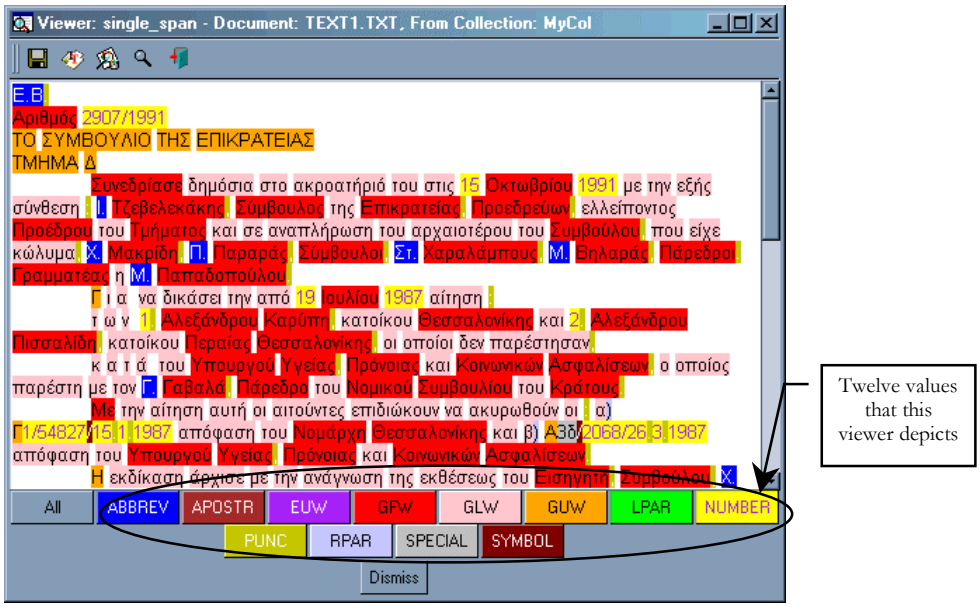

Figure 3.29 A Single Span Viewer

In *Figure 3.29* we have an example of a single span viewer. That viewer has twelve buttons, each one associated with an attribute that the viewer should show. When we created that viewer, we did not explicitly told which attribute values it would depict; we just said to it the annotation type and the attribute contained in this annotation type, that it would show and then it found which attribute values that annotation type contained. Each button has a different color, so that we will be able to discriminate among the attribute values. Furthermore, there is one more button with the caption "All", that enables us to see all the attribute values. If we push a button then only the corresponding attribute will be highlighted in the text area of the viewer. If we push the "All" button, then all the attributes associated with the viewer will be highlighted. We can toggle between the attributes being on or off by right clicking on a button.

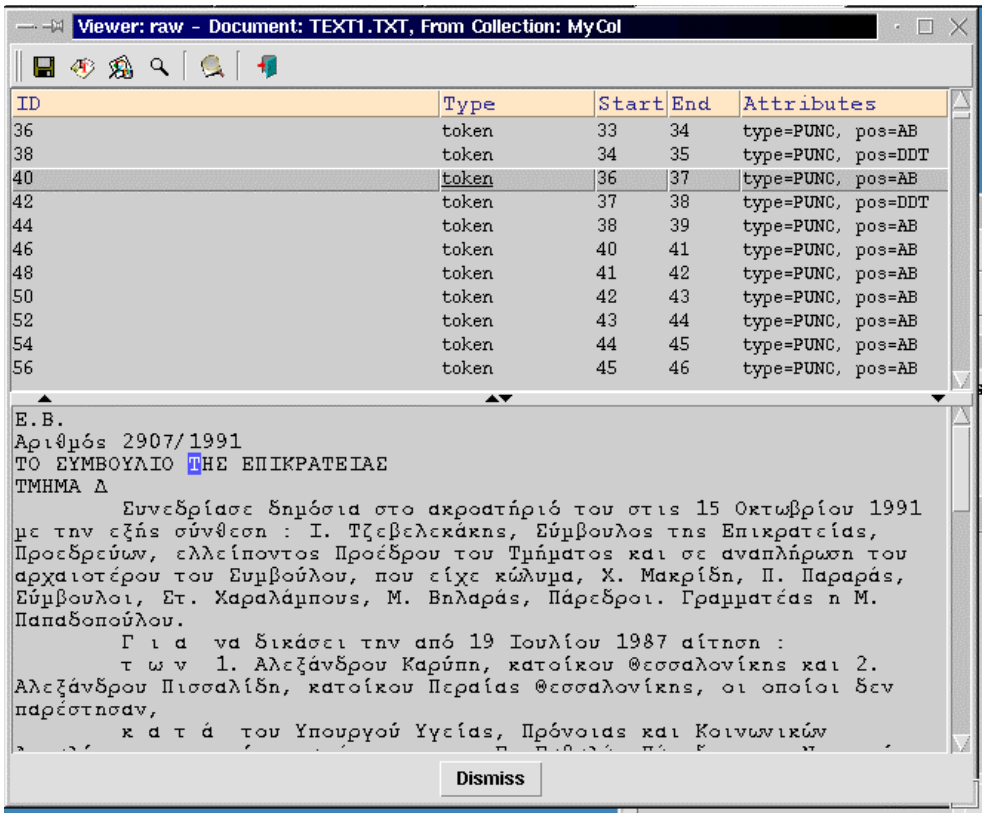

Figure 3.30 A Raw Viewer

In *Figure 3.30* we can see yet another viewer; the Raw Viewer. While associating that particular viewer with a Component, we said that it will show annotations of type `token`. The viewer is split in two regions. The bottom region shows the Document we chose and is highlighting the annotation we have selected on the top region. The top region contains the numerical identifier (ID) of the annotation, the type of the annotation, the starting point and ending point of its first span and finally all the attributes of the annotation. If you choose an annotation, then its span will be highlighted in the Document.

Another type of viewer is depicted in *Figure 3.31*. This viewer shows the graphical representation of a set of annotations that describe a tree structure, like the syntac-

tic parse tree of a sentence. The window of that viewer is divided into three regions. On the left hand side, we have the text of the Document that the module processed. If we place our mouse upon that region, then the sentences of the Documents are highlighted. Now, if we select a sentence then its syntactic tree, the parse tree of the sentence in other words, is shown on the right hand side of the viewer window. Furthermore, the selected sentence is shown in the bottom of the window (see again *Figure 3.31*). As you can see from the figure, the part-of-speech tags are depicted as rectangles and are the non-leaf nodes of the parse trees. On the other hand, the terminals, the actual words of the sentence, are depicted with oval shapes.

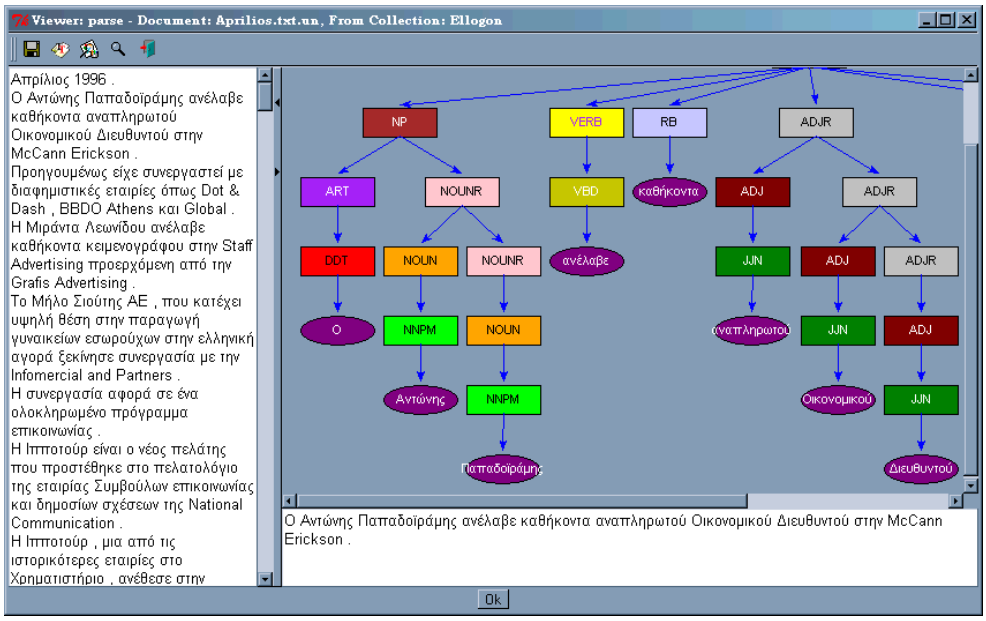

Figure 3.31 A Tree Viewer

The last type of viewer is depicted in *Figure 3.32*. This viewer needs XML Documents to be opened, in order to function. What this viewer does is to parse the XML Document and create the tree. Then the viewer graphically depicts that tree. The elements that are expandable are assigned a plus sign; if you push it the element expands revealing the structure of the tree.

3.8.4 Saving Results

When we process a Collection with a System then the various modules composing the System add, remove or modify the annotations and the attributes of the Documents of the Collection. The altered information is not stored immediately after processing on the hard disk. Instead, it is stored in the memory of your computer. If you want to save the results of processing some Documents, you have to do that explicitly. From the `File` menu of a System, choose `Save Collection`. A dialog box will appear prompting you to confirm if you want to save the Collection or not. Only if you choose "Yes" will the changes be stored on your hard disk.

Furthermore, in case you exit a System without saving the Collection, *Ellogon* will prompt you once again to save the Collection.

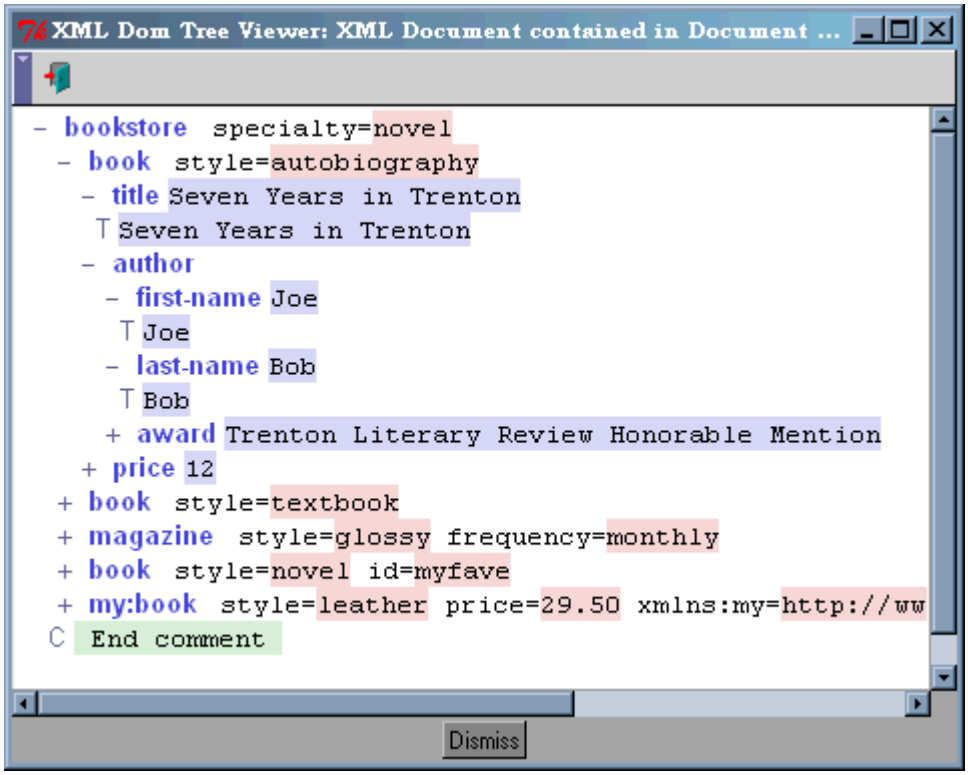

Figure 3.32 The XML Dom Tree Viewer

3.9 Getting help

Help in *Ellogon* is available through the `Help` menu on the *Ellogon* main window (see *Figure 3.1*). If you choose the option `Help Contents` a new window opens up. This is the Help System of the *Ellogon* platform. It is a browser-like environment, so you can browse through the information you need. It is depicted in *Figure 3.33*.

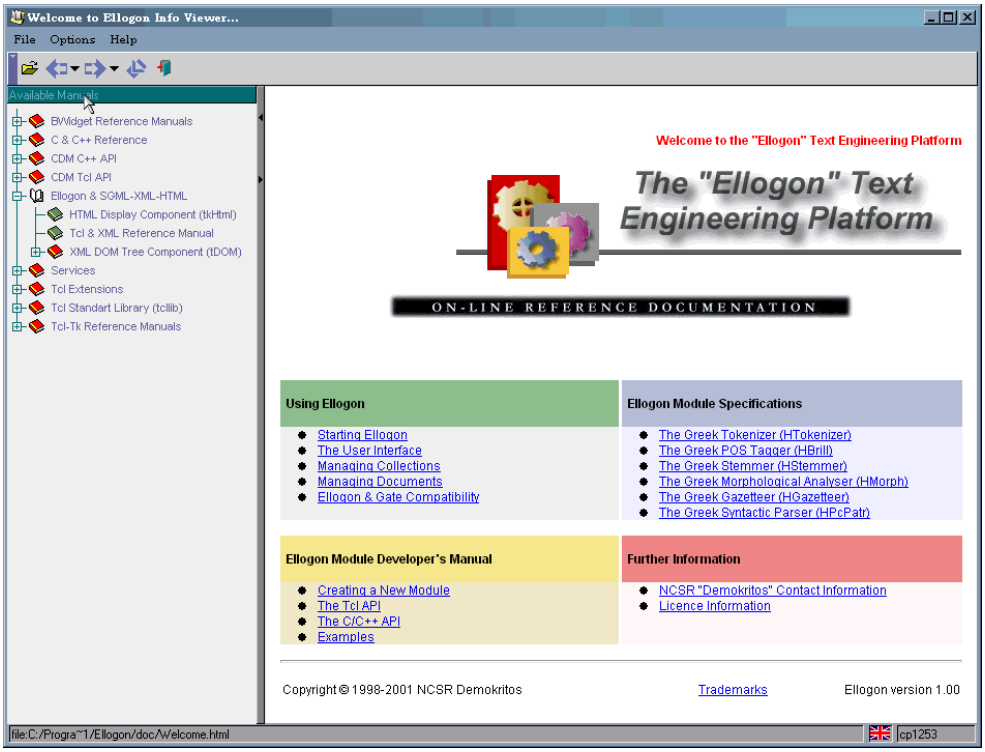

Figure 3.33 The Help System

Chapter

4

4 Using the Provided Tools

The *Ellogon* text-engineering platform comes equipped with a variety of tools. The tools are accessible from the `Utilities` menu of the *Ellogon* main window, or from the `Utilities` menu of a system window. Before proceeding with the explanation of the functionality of each tool, we have to explain some things. Some tools, in order to run, they need a Collection or just a single Document of a Collection to be opened. The process of opening a Collection or a Document was described in section 3.3.1, so we won't describe it here again. In what follows, while describing a tool, we shall mention that it needs a Collection or a Document to be opened in order to function. What that means is that you will have to open a System and from there open the Collection or Document that you are interested in, in order to run the tool. In case a tool does not need a Collection or Document, you can simply access it from the `Utilities` menu of the *Ellogon* main window.

4.1 Collection and Document Statistics Tool

Every Document and Collections carry with them a host of information, apart from the explicit linguistic information within the texts. Such information, for example, is the encoding of the Document or its format.

Ellogon stores such information and enables you to easily view them through a dialog. In order to view this dialog, open a Collection or a single Document through a System. From the "Annotation Tolls" of the "Utilities" menu of the System, choose "Collection/Document Statistics". In case you have opened a Collection a dialog, similar to the one in *Figure 4.1*, will appear. The information that this dialog contains consist, firstly from the name of the Collection and that path on the disk that this Collection is stored. Next, you can see the encoding and the type format that it is saved in. It shows also the Tcl command for that Collection (see *Developers Guide to Ellogon*) and whether that Collection is modified or not. On the attributes section, you can see the attributes associated with that Collection. On the documents information, you can see the number of the Documents composing that Collection, and the ID and External ID (i.e. the name on the disk) of every Document.

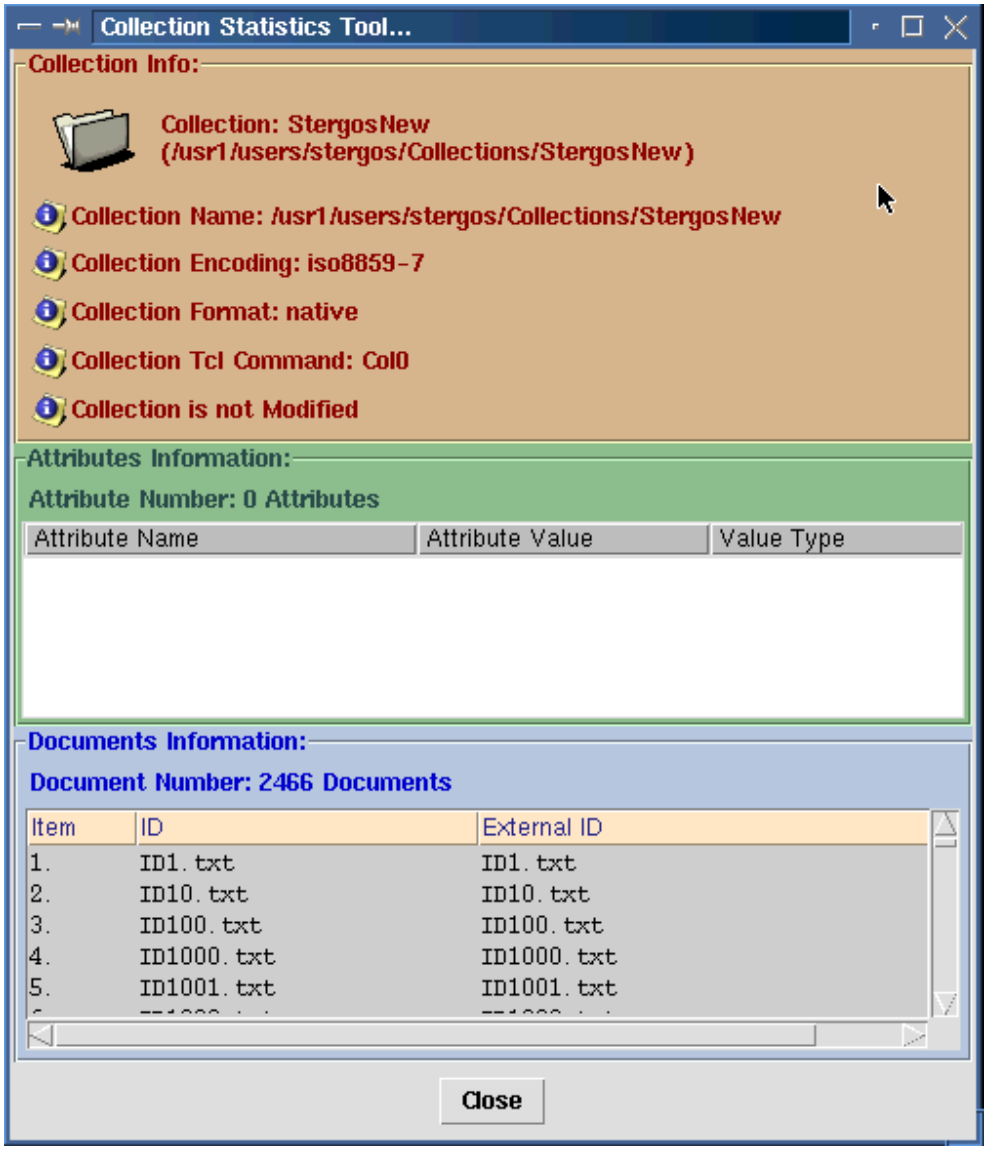

Figure 4.1 Collection Statistics

Now, if you had opened a single Document, a similar dialog will appear which will contain information about the Document (see *Figure 4.2*). That dialog contains the same information, as previously, about the Document this time.

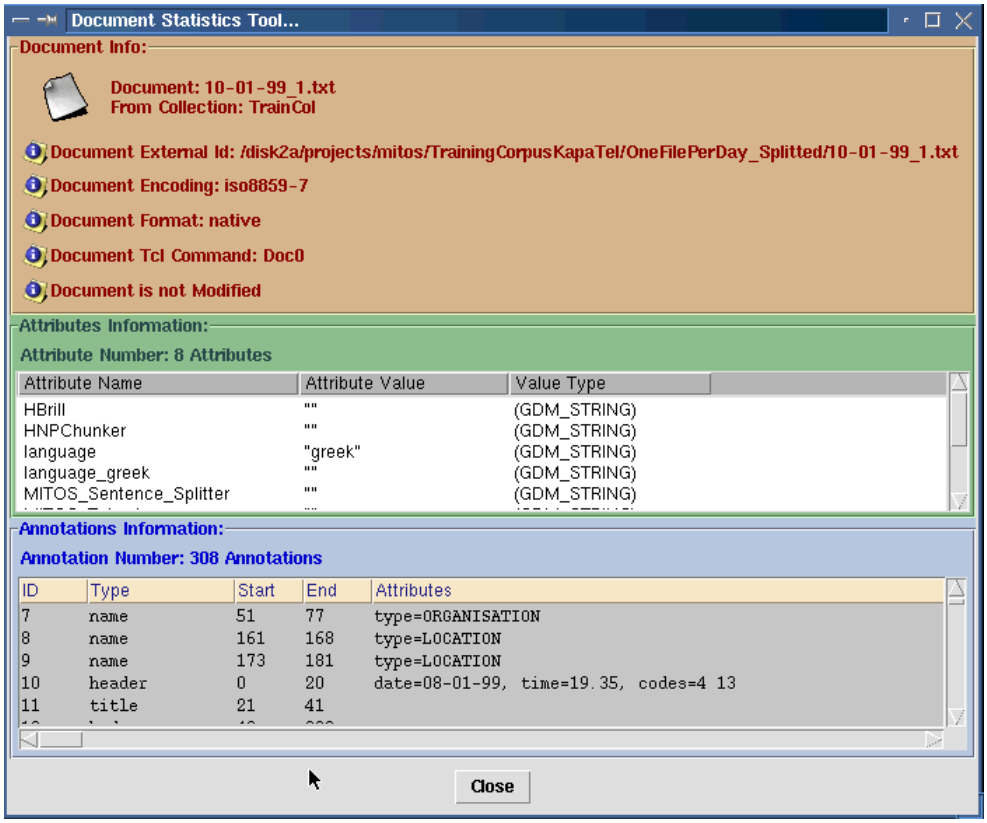

Figure 4.2 Document Statistics

4.2 Run Component Tool

A System is composed from a variety of Components, each one performing a specific task on a Document or a whole Collection. Nevertheless, there are times that you would wish to run one or more Components on a Document or a Collection and those Components are *not* included in the System's Components or, if the Component is on the System, it is not activated.

The "Run Component Tool" enables you to do exactly that, to run one or more Components either they are present on a System or not. This tool needs a Collection to be opened in order to function.

In case you have already opened a Collection or a Document, a dialog box, similar to that in *Figure 4.3* will appear. That dialog box contains all of the Components available to the *Ellogon* platform and it urges you to select the Components you want to run. When you select the Components you wish, simply push the OK button so that the Components will process the Collection or the Document you have opened.

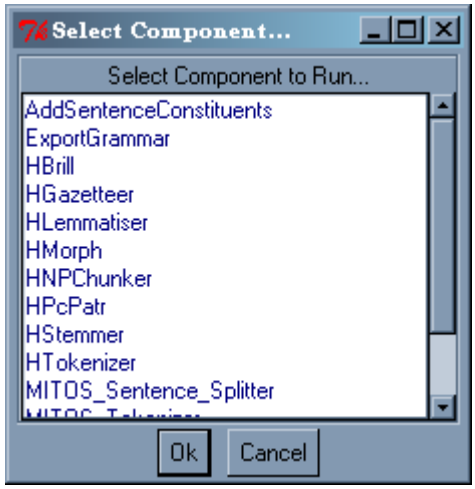

Figure 4.3 The Run Component Tool

4.3 Copying Annotations between Collections

It may be the case that you want to copy annotations from some Documents of a Collection to some other Documents of another Collection. *Ellogon* gives you the opportunity to easily do just this through the "Copy Annotations Between Collections" tool under the submenu "Annotation Tools" of the "Utilities" menu.

This tool has a very informative graphical interface, which easily guides you through the process. You can see it in *Figure 4.4*. This process has six steps. In the first step you select a Collection from the drop-down list which contains the annotations to be copied; its Documents will appear below that list. Next you choose a Collection in which the annotations will be copied. Similarly, you choose the Collection from a drop-down list, and the Documents of the Collection appear below that list. On the third step you select the Documents from the first Collection that contain the annotations to be copied and on the fourth the Documents from the second Collection in which the annotations will be copied. In order to choose more than one Documents hold down the Control (Ctrl) button on your keyboard. In the fifth step you can select the annotation types that you want to be copied; again you must hold the Ctrl key to choose more than one annotation type. On the sixth and final step you simply push the Copy button to complete the process.

Note that once you move your cursor on a step, the appropriate area changes color (in *Figure 4.4* this is done on step 5), helping you thus locate what you have to do where.

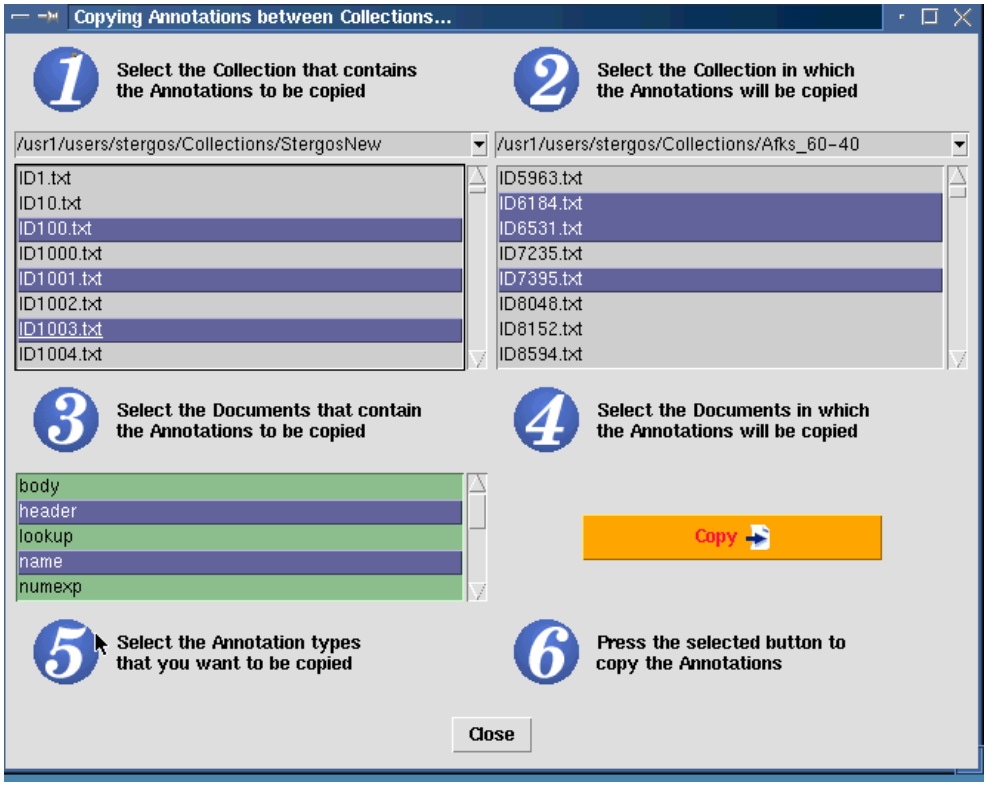

Figure 4.4 The Copy Annotations between Collections Tool

4.4 Tool for Editing Annotations

There are many cases in which you would like to manually change the attributes from annotation types in a Document, either because you want to check the correctness of a tool or because you want to manually tag a corpus in order for a tool to be trained, or for a plethora of other reasons. The "Tool for Editing Annotations" allow you to open a Document and manually edit it, i.e. add, remove or modify its annotations.

This tool is accessible from the `Annotation Tools` submenu of the `Utilities` menu and it does not need any Collection to be opened. Once you select that item, a dialog box, as in *Figure 4.5* will appear.

In that dialog box you can choose the annotation type and the attribute that you want to modify, from the "Edit Annotation" drop down list. *Ellogon* has a variety of annotations that you might want to modify. According to the annotation you choose, you then can choose the type of the annotation, from the "Annotation Type" drop-down list. Not all of the annotations, of course, have an annotation type to select. The last option you can to provide is the language that the Document is written in. In case an annotation or an attribute is not present in the top down lists, you can manually provide it, since all the fields are editable.

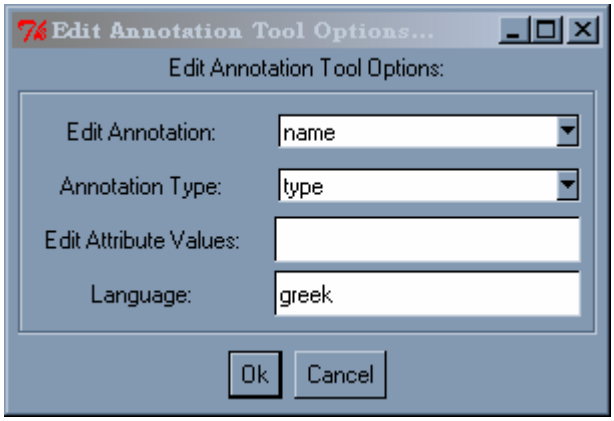

Figure 4.5 The Edit Annotations Tool

When you are finished providing those details push the OK button. A new window will appear, as shown in *Figure 4.6*. On the upper side of that window you will find two drop-down menus, one for choosing a Collection and one for choosing a Document of a Collection. Once you choose the Document, which you want to modify, press the "Open" button.

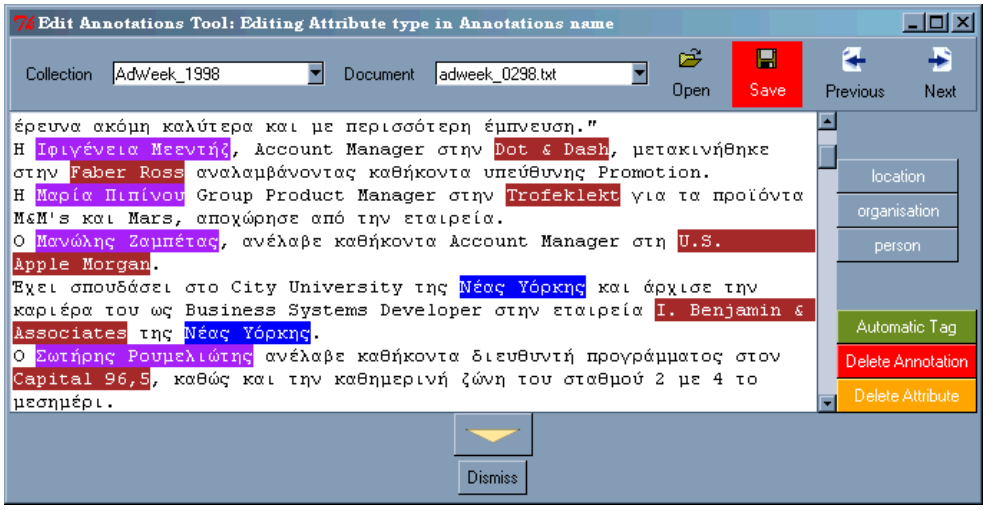

Figure 4.6 Manually Editing Annotations

In order to annotate a Document you simply select with your mouse the textual range you wish to annotate and then you push the appropriate button, which corresponds to the attribute you want to place to that annotation. Each different attribute is coloured differently in the text.

In case you have done a mistake and you want to remove the whole annotation, click on that annotation inside the text, and then click the "Delete Annotation" button. The annotation will be deleted from the text. If you do not want to delete the annotation, but only the specific attribute, click again the annotation and then click the "Delete Attribute" button. In this case only the attribute will be deleted, but the annotation will not.

When you finish annotating a Document, press the "Save" button on the upper side of the window. Next to that button, there are two more buttons for navigating through the Documents of the opened Collection. Instead of closing the Document and opening a new one for annotating you can push the "Previous" or "Next" buttons, and you will be moved to the previous or the next Document of the Collection, accordingly. Of course, in case you forgot to save any changes, *Ellogon* will remind you of this fact and it will prompt you to save the changes you have made to the Document.

Note that in case that tool is opened from within a System, which has already opened a Document from a Collection, then this "Navigation Toolbar" will not be there. You will be able to edit only the Document that was opened from the System.

4.5 Remove Annotations/Attributes Tool

There are certain cases in which you would like to remove some of the attributes of an annotation or all the instances of an annotation from a Collection or a Document. The "Remove Annotations/Attributes Tool" does exactly this. It needs a Collection or a Document to be opened in order to run. Once you run that tool, a dialog box, similar to that in *Figure 4.7*, will appear. In case that only a Document of a Collection has been opened the "Remove Collection Attribute" option of that dialog will be absent.

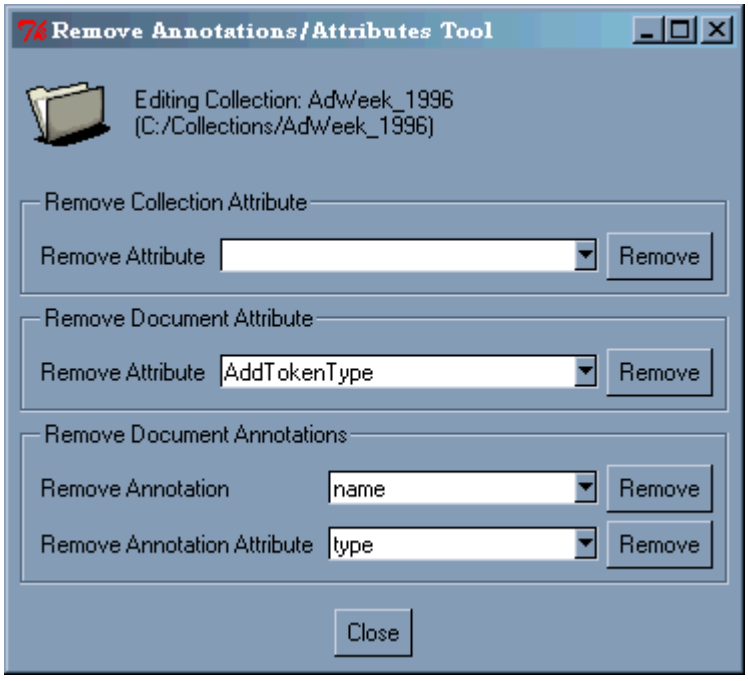

Figure 4.7 The Remove Annotations/Attributes Tool

This dialog box collects all the annotations and the attributes that the opened Document or Collection contains. From the "Remove Collection Attribute" drop-

down list you can select the attribute of the Collection you want to remove. From the "Remove Document Attribute" drop-down list you are able to select the attribute of the Document that you want to remove. From the "Remove Annotation" drop-down list you can select the annotation you want to remove, and from the "Remove Annotation Attribute", the attribute of the annotation. Once you have finished selecting the items you want to be removed, push the corresponding "Remove" button. Note also that all the fields are editable, so that if a value is not present on the drop-down lists, you can manually provide it by writing it down on the field.

When you finish, push the "Close" button for the dialog box to close.

4.6 Collection Comparison Tool

The "Collection Comparison Tool", as its name suggests, enables you to compare two Collections, or, more precisely, to compare their annotations and attributes. This tool does not need a Collection to be opened in order to run. Once you open that tool a new window, as depicted in *Figure 4.8*, will appear.

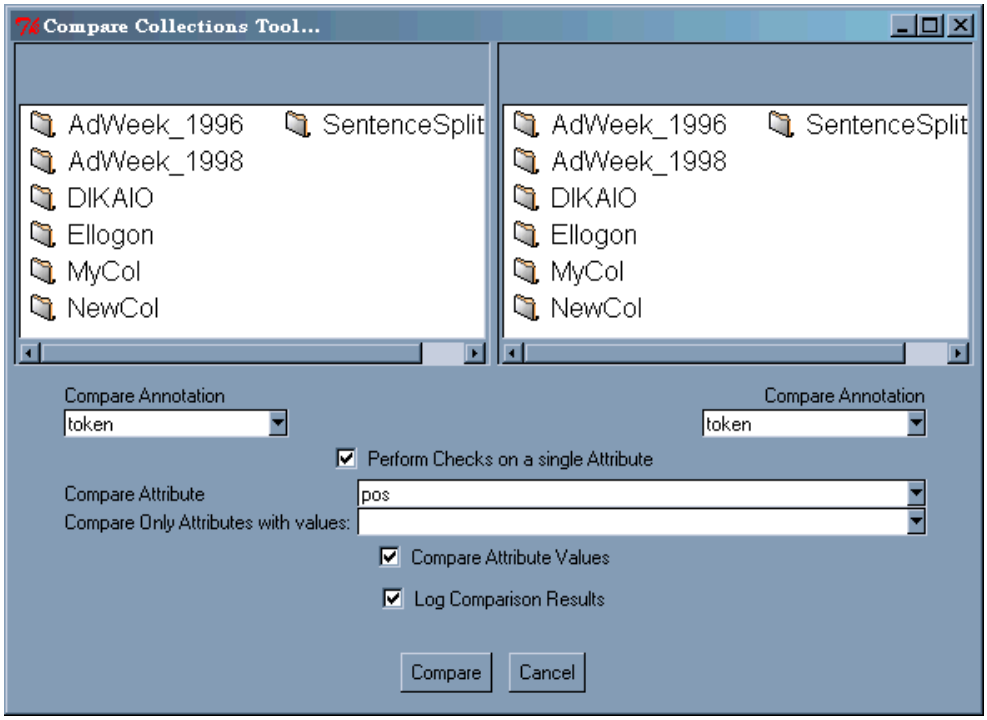

Figure 4.8 The Compare Collections Tool

As you can see from the figure, this tool asks you to choose two Collections to compare, the "Key Collection" and the "Response Collection". Note that in order for the tool to work properly, the Collection should contain at least the same number of Documents; usually, comparing only Collections that have exactly the same textual data, has meaning. After selecting the key and response Collections that are to be compared, you have to choose the annotations that those Collections will be

compared against. You can either choose the annotations from the corresponding "Compare Annotation" drop-down lists, or provide it manually by writing in those text fields.

After providing the annotation, there are several options for you to choose among, depending on what you have in mind to do. In case you simply want to check whether the *spans* of the annotations are the same, ignoring the possible attributes and the values that they may have, then you should uncheck the "Perform Checks on a single Attribute" check button. This will cause the next two drop-down lists and the following check box to be disabled. In that case the tool will only compare whether two annotations have the same span, regardless of the, possible different, attributes and values they may have.

If this is not what you intend to do, and you want apart from the spans of the annotations to compare their attributes as well, then you should check the "Perform Checks on a single Attribute" check button, and then provide the attribute to be checked. Again, this can be done either by selecting the attribute from the "Compare Attribute" drop-down list, or by writing it in the text field. Of course, you can provide only one attribute; there is not yet any option for comparing two attributes.

Furthermore, if you want to compare only attributes with certain values, you can provide the name of the attribute value in the "Compare Only attributes with values" text field, and checking the "Compare Attribute Values" check box (both things have to be done). Note that in the "Compare Only attributes with values", you can use wildcards, e.g. the * wildcard.

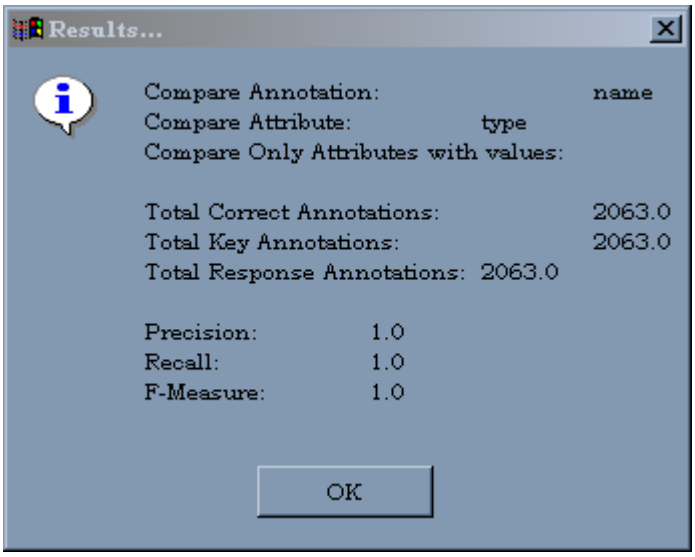

Figure 4.9 A Synoptical View of the Results of the Compare Annotations Tool

The last check box, the "Log Comparison Results" if checked, will open a new window (see *Figure 4.10*) once the tool stops comparing the Collections. In any case, after the tool stops comparing the Collections a window, similar to that in *Figure 4.9* will open. This dialog box synoptically describes which annotations, attributes and values it has compared; the total key and response annotations; the

total correct annotations (that is, the total annotations in the response Collection, that were present in the key Collection as well) and, finally, the Precision, Recall and F-Measure.

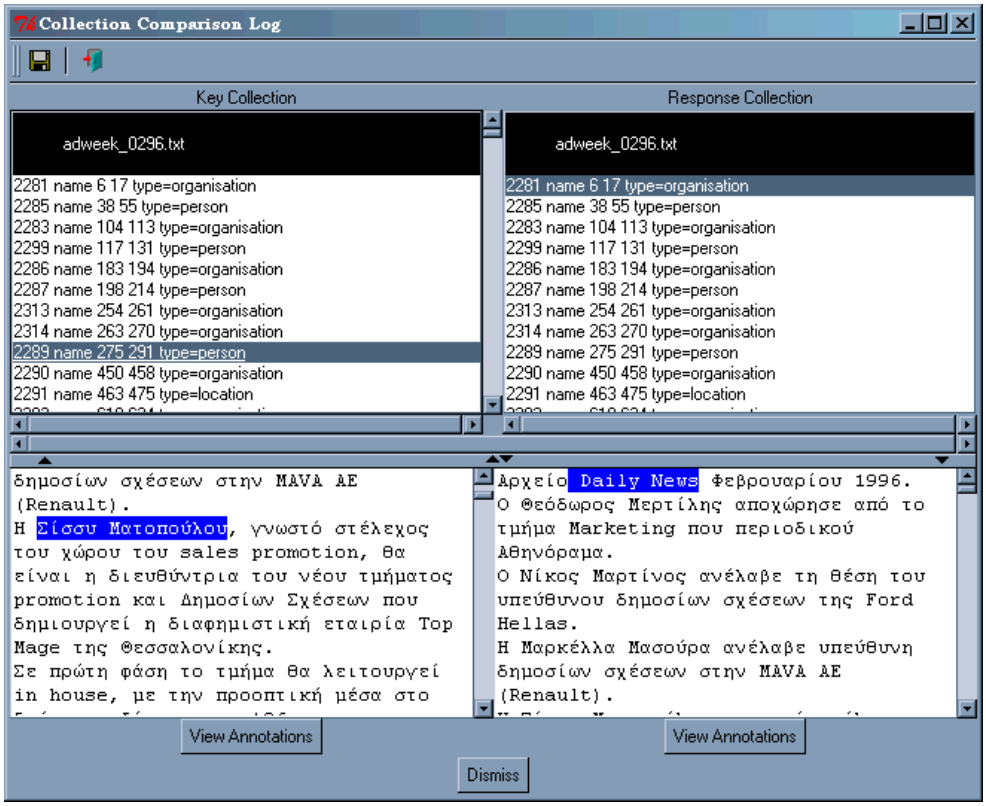

Figure 4.10 Collection Comparison Tool Log

Now, in case you have checked the "Log Comparison Results" check box, another window will appear as well, as shown in *Figure 4.10*. In that window you can see the key and response Collection's annotations and their attributes and values, if you have chosen the attributes and their values to be compared. Selecting an annotation, either from the key or the response Collection, will cause that annotation to appear highlighted within its Document, as you can see in the figure. If you, furthermore, push the "View Annotation" button, a new window will open (which is similar to the raw viewer depicted in *Figure 3.30*), which enables you to see the whole Document along with it's annotations and several information about it, such as the span of the annotations, their attributes and their values, etc.

4.7 Create Vectors Tool

Annotations and attributes in the *Ellogon* Documents may be stored either in its native format or in a format compatible to that of the GATE 1 platform (see chapter 5). Nevertheless, many algorithms, such as Machine Learning algorithms, require their data to be in a different format. Usually, such algorithms require their data to be stored into vectors, containing, apart from the information about the annotations, and *meta-information* such as the class that the vector itself is classified.

Ellogon provides a tool, the "Create Vectors" tool, which enables you to transform the *Ellogon* annotations into vectors, and add to it whatever other information or meta-information you want. This tool does not need a Collection to be opened in order to function properly. Once you open that tool a new window, as depicted in *Figure 4.11*, opens. Note that, in order for that window to open, you have to have opened a Collection or a Document beforehand. In case you haven't, *Ellogon* will remind you of this fact with a dialog box.

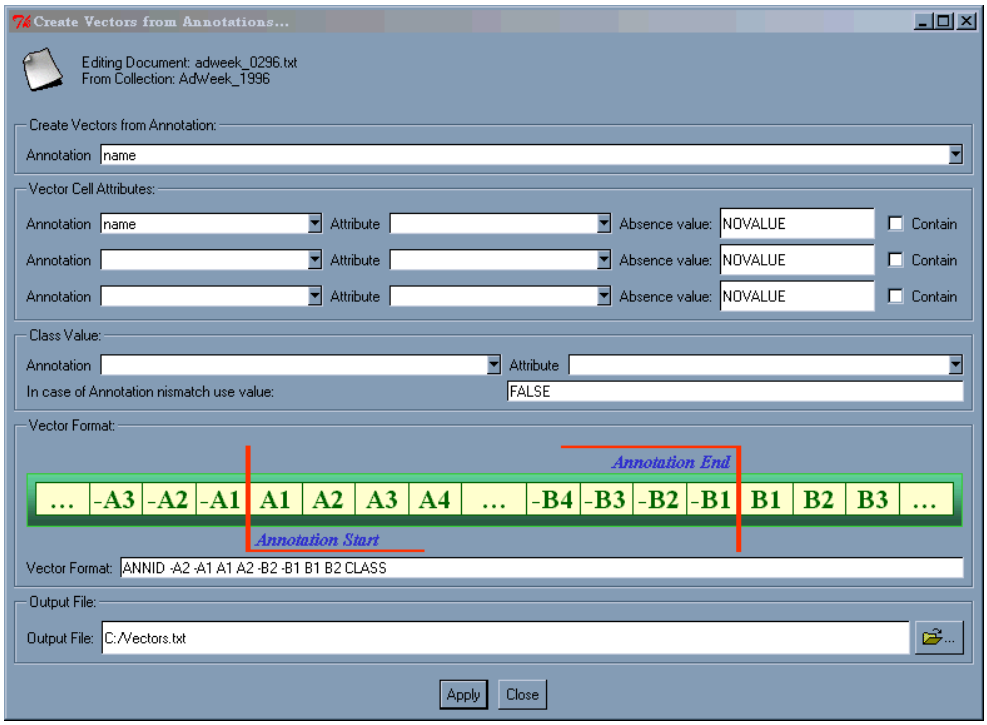

Figure 4.11 The Create Vectors Tool

Before proceeding with how to create a vector using this tool, let us a notation on how to access the elements that span an annotation, and place that information on the vector, we are going to create. Each annotation spans positions, say, x_1 through x_n. Position x_1 is denoted as A_1, position $x_1 + k$ is denoted as A_k if $k > 0$ or as $-A_k$ if $k < 0$. Similarly, taking the elements from the end of the vector, position $x_n + k$ is denoted as $-B_k$ if $k > 0$ or as B_k if $k < 0$. With this notation now, we can specify how many elements the vector will contain and, of course, we are free to add what other information we want.

Now, in order to create a vector from the annotations contained in a Document or in a whole Collection, the first thing you have to do is to provide the annotation from which you want the vector to be created. This is done by writing the annotation in the "Create Vectors from Annotations" text area, or by choosing an annotation from the drop-down list.

Next, you have to provide a name for the elements of the vectors. In the "Vector Cell Attributes" area, you can see three rows. If you fill in all of the rows with information, then each element of the vector will contain three pieces of informa-

tion, as we shall explain; if you fill in only two rows, every element of the vector will contain two pieces of information, etc. In the "Annotation" text field, you should specify the information, which is a string of characters, which will be placed whenever a certain attribute in the annotation is found. That attribute you specify in the "Attribute" text field. In case that attribute is absent, specify the information that will be placed in the vector cell, in the "Absence value" text field. The "Contain" check box, if checked, will place the information, if the attribute is contained within an other. If it is unchecked, the tool will not place that information in the vector cell, if the attribute is contained within another.

As we have said, many algorithms require their vectors to contain some meta-information, as to the classification of the specific vector. In the "Class Value" area you can specify, if you want, that information. If you want to give an arbitrary name to your class, we shall see shortly how to do it, but if you want to give the class name the combination of an annotation along with the attribute that it has, *Ellogon*helps you to easily manage that information. In the "Annotation" text field, write down the annotation that you want, and in the "Attribute text field the attribute you want. After the tool runs it will place that information in the CLASS cell of the vector, if that combination is found. In case the combination is not found, then you can place another string of characters denoting the fact that that combination wasn't found, in the text field "In case of Annotation mismatch use value:"

Now, let's see how to format the vector; that is how many cells it will contain, and what information will be placed inside the cells. If you want your vector to have a fixed length, then in the "Vector Format" text field write `ANNID` as the first thing. This implies, apart from the fact that your vector will have a fixed length, that the first cell of your vector will contain a unique identifier composed of three things: the Collection id, the Document id and the id of the annotation. This is useful when you want to trace back where inside your Collections that vector refers to. Next, through the notation A_k and B_k described some paragraphs before, you can specify as many cells as you want, each containing the information about that position in the original annotation, as you have provide it in the "Vector Cell Attributes" area.

If you want your vector to have a variable width length, then write on the vector format the string `ANN` as the first thing. That will cause the number of the vector cells to vary according the *span* of the annotation. That is, if a certain annotation has length x then the vector will have x cells corresponding to that length. Of course you are free to add any other information you may want. For example you may want to add the previous two elements and the next two elements of an annotation in every place in the vector format you can write the string `CLASS` which will place to corresponding vector cell the information you provided in the "Class Value" area. Furthermore, in any place in the vector format, you can place any string of characters you want, without spaces. That string will be placed as is in the corresponding cell in the vector.

Finally, in the "Output File" text filed you can specify the path of the file that those vectors will be stored, either by manually writing it, or by pushing the associated button, and searching through the dialog that opens for that file.

4.8 Export Annotations Tool

Each Document of a Collection apart from the textual data, it also contains linguistic information, in case it has been processed from a module. In some cases though, you may need some of the linguistic information (i.e. annotation types along with their attributes) contained in your Documents, to be exported into a file. *Ellogon* offers you exactly this capability through the "Export Annotations Tool". It can be found in the `Export` submenu of the `Utilities` menu. This tool needs a Collection or a Document to be opened.

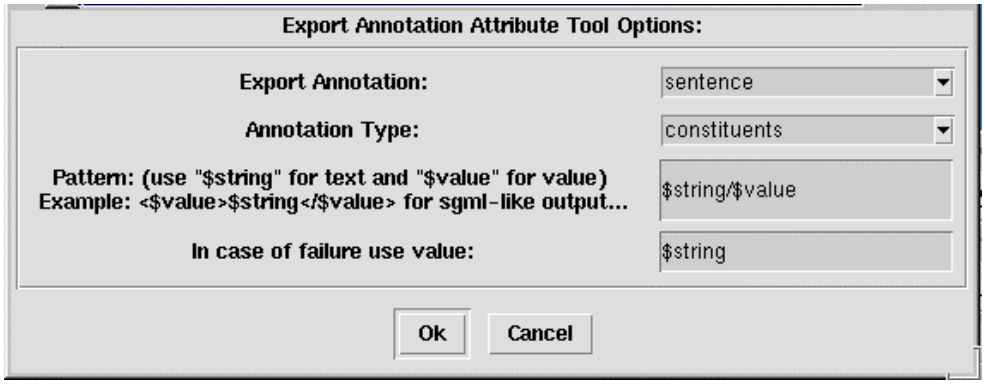

Figure 4.12 The Export Annotations Tool

In *Figure 4.12* you can see the window of the tool. In the "export Annotation" drop down list you can select the annotation that you want to export; if the annotation is not in the list, you can provide your own annotation by writing it into the field. Then you can select the annotation type you want to export from the "Annotation Type" text field; again, if the annotation type is not in the list, provide it manually.

Then you have to provide a pattern, according to which, the annotations will be exported. The pattern can contain any arbitrary text you may wish, but two symbols have a special meaning. `$string` will replace the whole annotation with its textual value. `$value` will be replaced by the attribute value. In case that an attribute value is not found into the annotation, you can provide yet another pattern that will indicate this fact. You can provide this pattern in the text field labelled "In case of failure use value:"

Once you are finished press the OK button. The tool will export the annotations and annotation types into a file according to the pattern (and the failure pattern) you provided. The file will be shown into a text viewer.

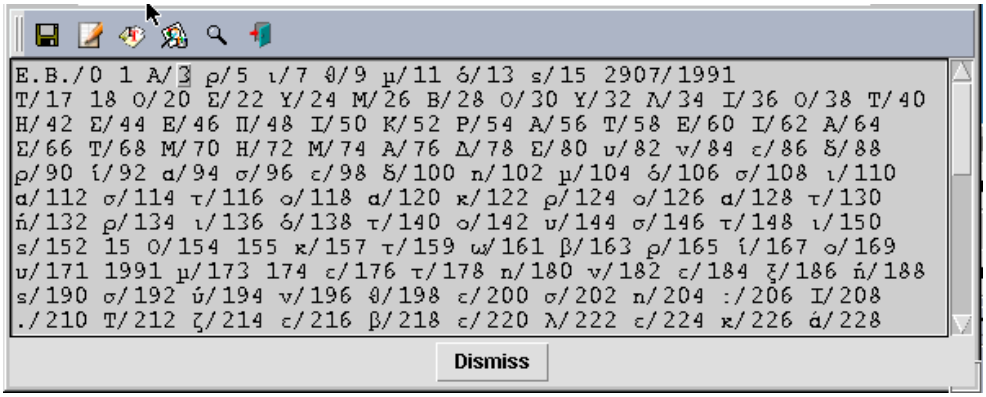

Figure 4.13 The result of the Export Annotations Tool

In *Figure 4.13* you can see the result from a run of the "Export Annotation Tool" in the text viewer. On the upper of the viewer you can see a toolbar. This toolbar enables you to save what you see into a file by pushing the first button. The second button lets you edit the results.

4.9 Export SGML Tool

Your documents contain annotation types and attributes which are stored in *Ellogon*'s native format. However there exist applications that need their data to be in the SGML format. *Ellogon* enables you to export the linguistic information contained in your Documents into SGML format through the "Export SGML Tool", which can be found in the `Export` submenu of the `Utilities` menu. This tool needs a Collection or a Document to be opened.

If you open that tool, a window similar to that in *Figure 4.14* will open. The tool, as soon as it opens, it goes through the Document(s) and collects all the attribute types that exist in there. Those attributes are then listed in the right hand side of the window of the tool (see *Figure 4.14*). In order to export the annotations (along with the attributes they may have) you have to select them from that list, by clicking them, and then pushing the associated green button on the bottom of that list. The selected annotation will be moved on the list, which is entitled "Annotations NOT to be renamed", which means that those annotations will be exported to a file with the same name. Now, if there are some annotations, which you wish to export, but with a different name, you can do it by selecting an annotation from the list "Annotations NOT to be renamed" and pressing the green button, on the bottom of that list. A dialog will open asking you to provide a new name for the annotation. This will move that annotation to the list entitled "Annotations to be renamed".

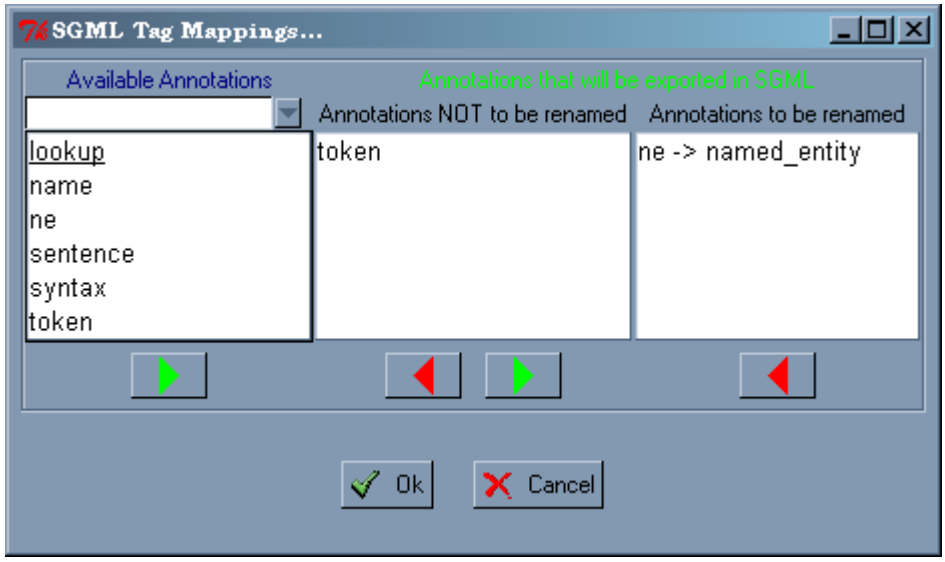

Figure 4.14 The Export SGML Tool

In every step, in case you have mistakenly added an attribute to a list, you can easily remove it from that list. Simply select the wrongly placed annotation and then press the red button associated with every list. This will remove the selected annotation from the list. Furthermore, in case the tool failed to find an annotation type, you can manually provide it. Above the list containing the annotations there is a text field. Write down there the annotation you wish and press the "Enter". This will place the annotation type you provided into the list with all the annotations.

Now, if there is an annotation type you don't like to export into SGML, you can do it by not including that annotation to any list. That is, b leaving the annotation in the initial list.

Once you have finished with all this procedure, push the OK button. Your Document(s) will be processed and the exported SGML will be shown into a Text Viewer. In case you processed a whole Collection, the SGML code for the Documents will be placed sequentially one after the other. You can save the results by pressing the "Save" button of the viewer.

4.10 Create Stand-Alone Application Wizard

Applications that are developed under *Ellogon* need the *Ellogon* platform in order to successfully run. But sometimes you might need to create a stand-alone application to run in another machine (*e.g.* your client's) without copying the whole *Ellogon* infrastructure (for example, for copyright reasons). *Ellogon*, allows you to do just that, through the "Create Stand-Alone Wizard", which you can find under the "Utilities" Menu.

This wizard, whose first screen you can see in *Figure 4.15*, allows you to built two kinds of applications:

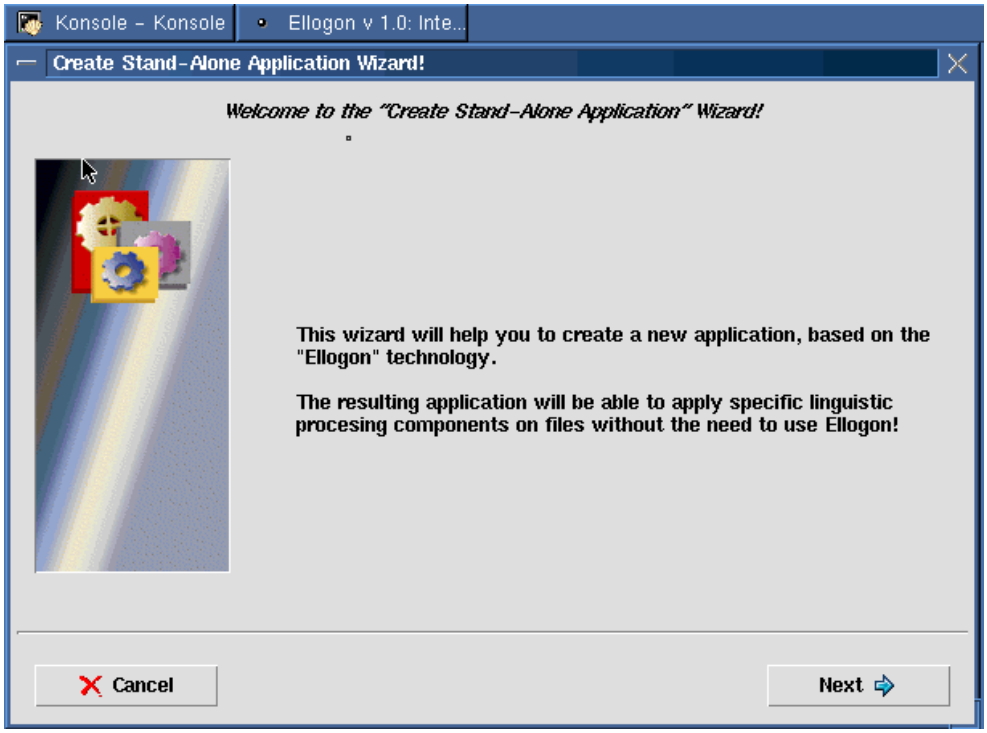

Figure 4.15 The first screen of the Create Stand-Alone Application Wizard

1. Graphical Applications

2. Console Applications

The processes of building graphical applications or console applications are, essentially, the same. In order to build an application, push the "Next" button of the first screen. In the next screen that appears (*Figure 4.16*) you can select between graphical and console application. Once you have chosen between graphical and console application, push the "Next" button.

The screen that appears (*Figure 4.17*) prompts you to specify the type of application. This can be either a script application or a complete application. The difference between the two is that in the script application, a script will be created which will be given as argument to *Ellogon* for batch processing. Instead, if you choose to create a complete application then the generated application will run without requiring *Ellogon*.

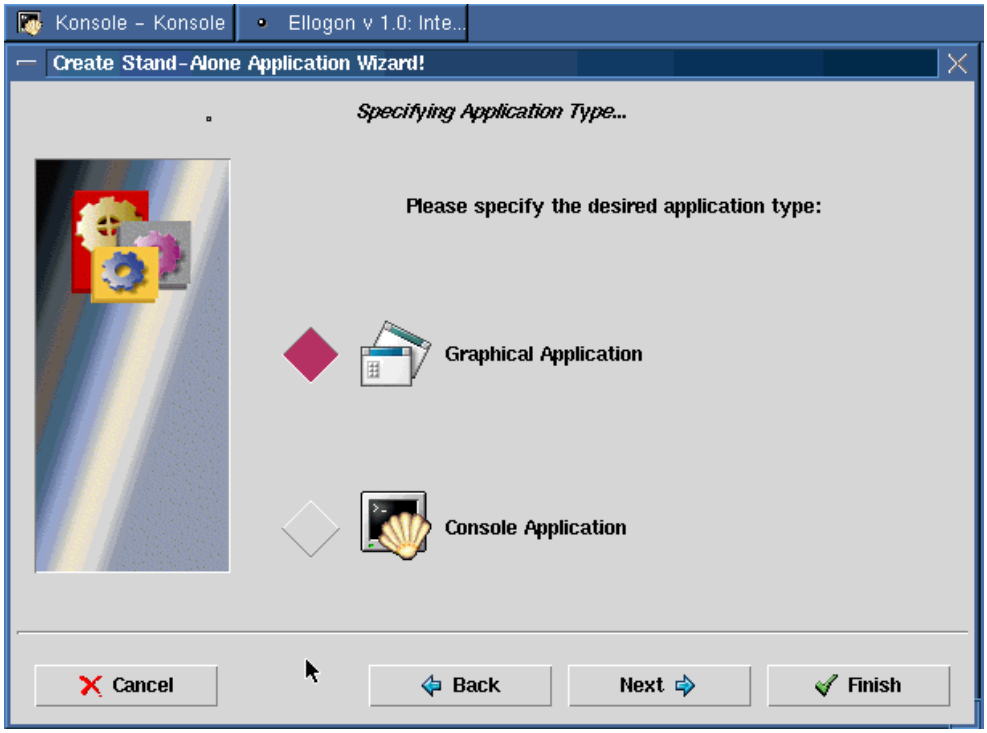

Figure 4.16 Selecting the type of Application

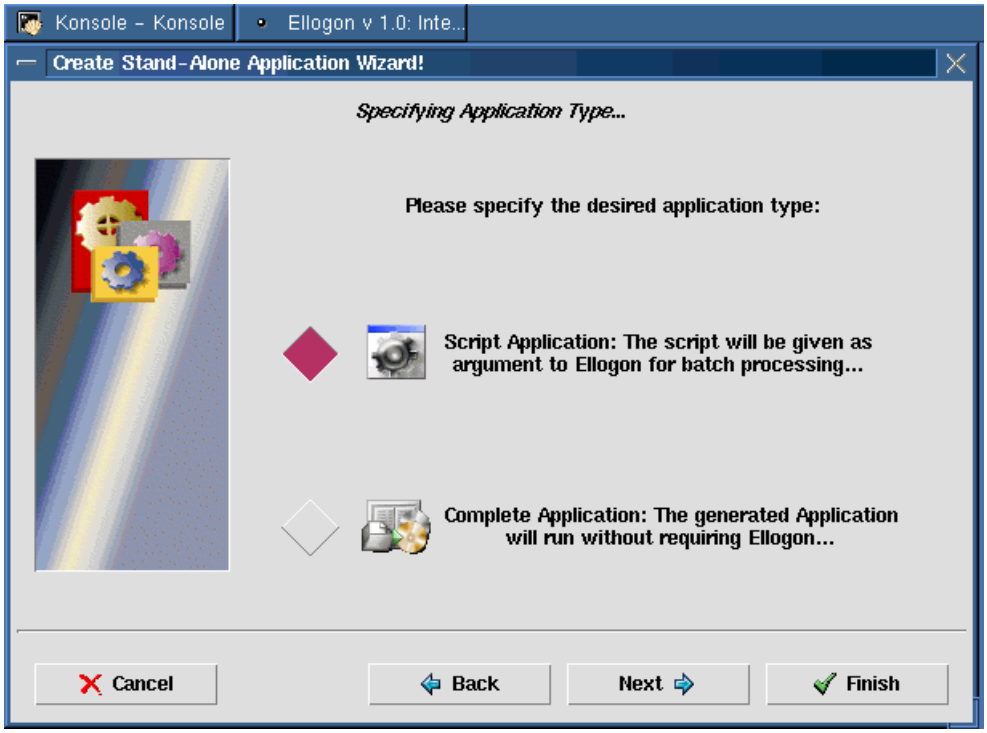

Figure 4.17 Selecting between Script Application and Complete Application

Once you have chosen the type of application that you want to build, push the next button. A different screen will appear, according to the choice you made on the previous step. The only difference between the two is that if you choose a script

application, a text box will appear which will enable you to insert the name and the location of the script that will be created (see *Figure 4.18*). On the other hand, if you choose a complete application the screen that will appear will require of you to specify a path.

Note that in specifying a path or a file name, the contents of the folder that the application will be stored will be erased, so be careful with what you specify. After you have finished push the "Next" button.

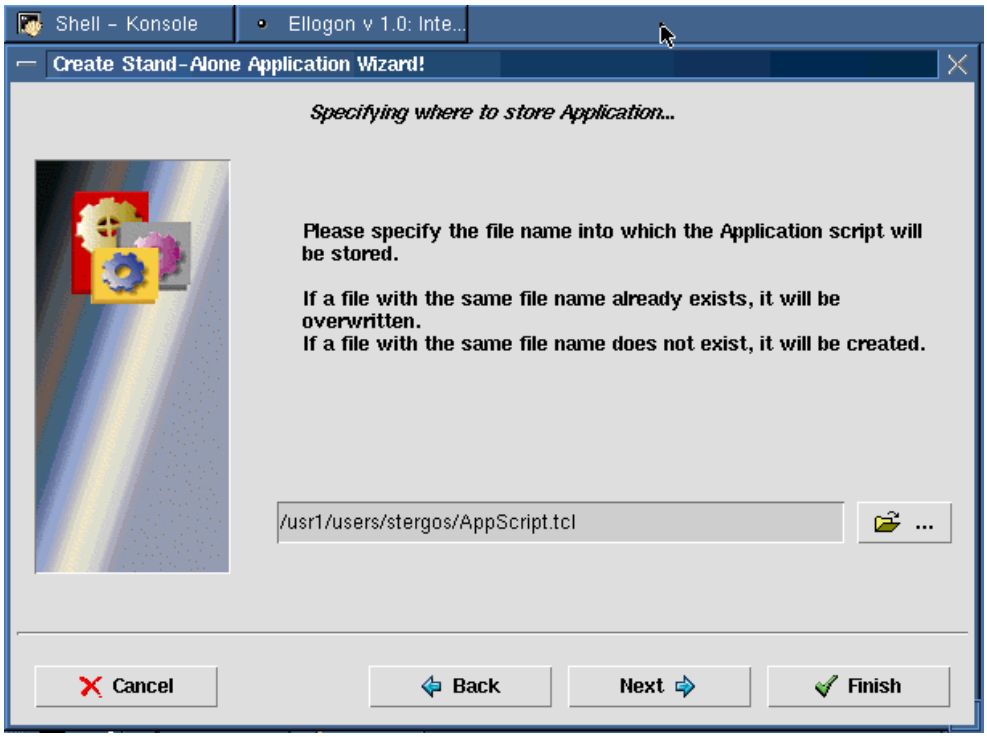

Figure 4.18 Selecting the name and the location of the script

In the next screen that appears (see *Figure 4.19*) you can make three choices concerning three parameters of the Collection that your application will process. The first choice you can make concerns the folder in which the Collection that your application is going to process, will be stored. You can provide either the path of the Collection on the designated text area, or choose it from the associated button which opens a file browser.

The next options concern the saving of the Collection to your hard disk. If you choose the "Sync Collection" option, your Collection will be saved to the disk, after each module has finished processing. If you do not choose that option, the Collection will be saved to the disk only after all the modules have finished processing.

Of course, it may be the case that you do not want the Collection to be stored into your hard disk. For example your modules may create several auxiliary files based on the Collection. You might not want to save the Collection but just the auxiliary files. In such a case select the "Delete Collection" option.

After you have finished push the "Next" button.

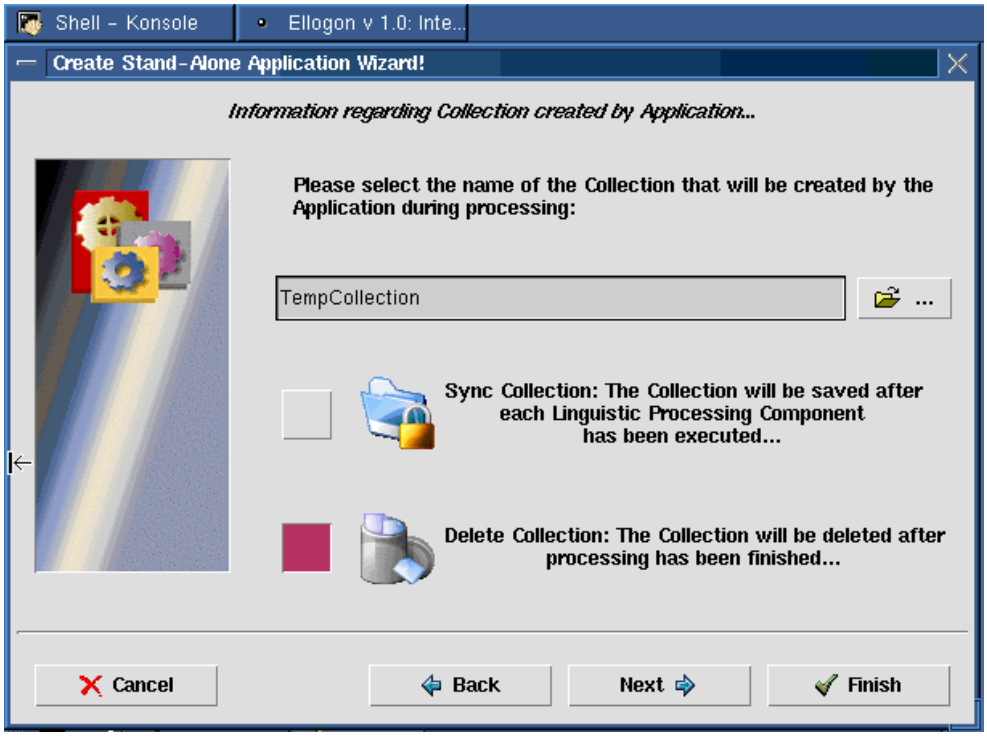

Figure 4.19 Specifying the information regarding the Collection

The next step (*Figure 4.20*) concerns the Documents of the Collection. That step enables you firstly to choose an encoding for the Documents of the Collection that will be created. Then it lets you decide between giving the Documents of the Collection as arguments to the script, or providing to the wizard the files. This is done by deselecting the associated checkbox. If you choose to provide the Documents to the wizard, you can either drag and drop them into the designated area, or you can simply push the associated button and browse towards those files.

When you finish push the "Next" button.

The next step concerns the selection of the modules that will run. In the screen that appears (*Figure 4.21*) lists all the modules that exist in the module search path of the *Ellogon* and groups them in categories. All you have to do is to select the modules you want to comprise your application.

After that, push the "Next" button.

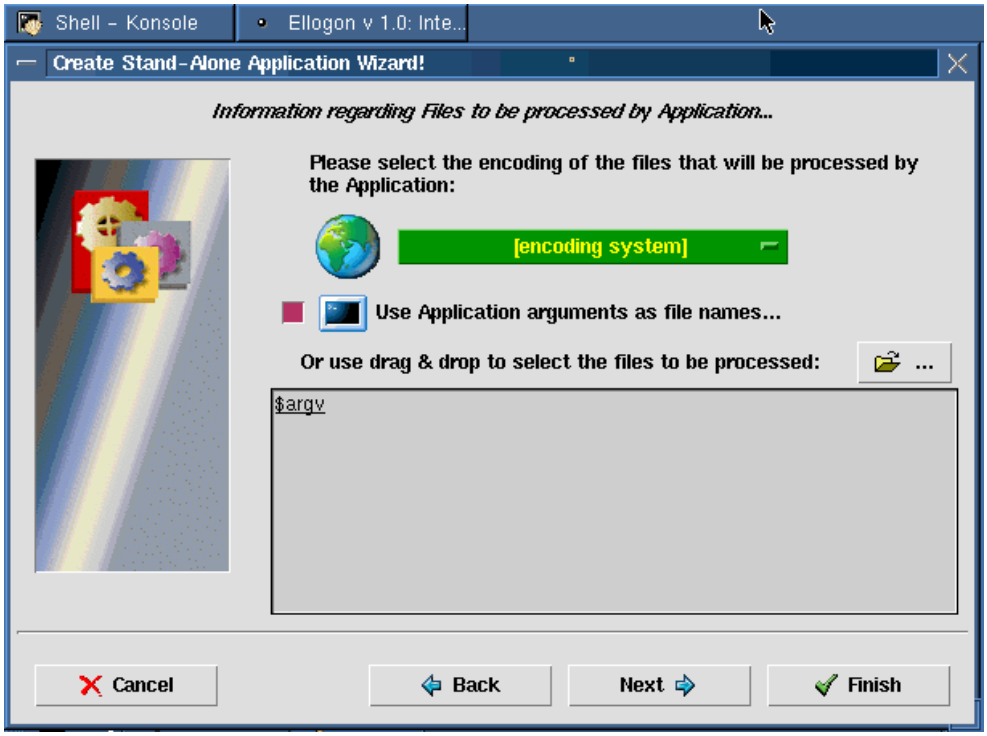

Figure 4.20 Selecting the Documents to be processed

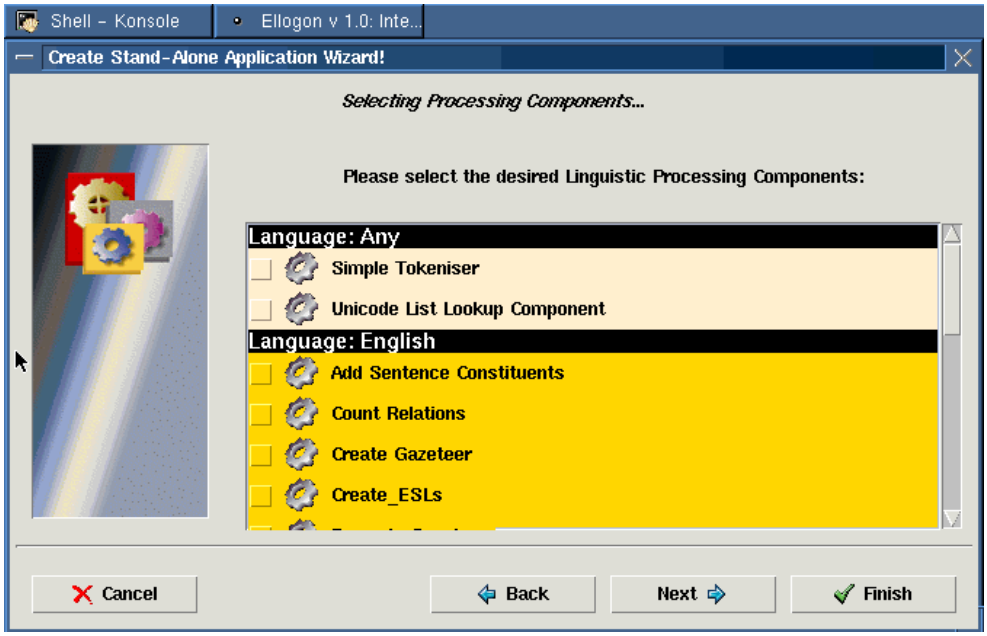

Figure 4.21 Selecting the modules of the application

The next step (*Figure 4.22*) concerns the *order* in which the selected modules will run. The selection of the order is very simple. You just have to drag each module to the appropriate position. The topmost module will run first, the second next, etc.

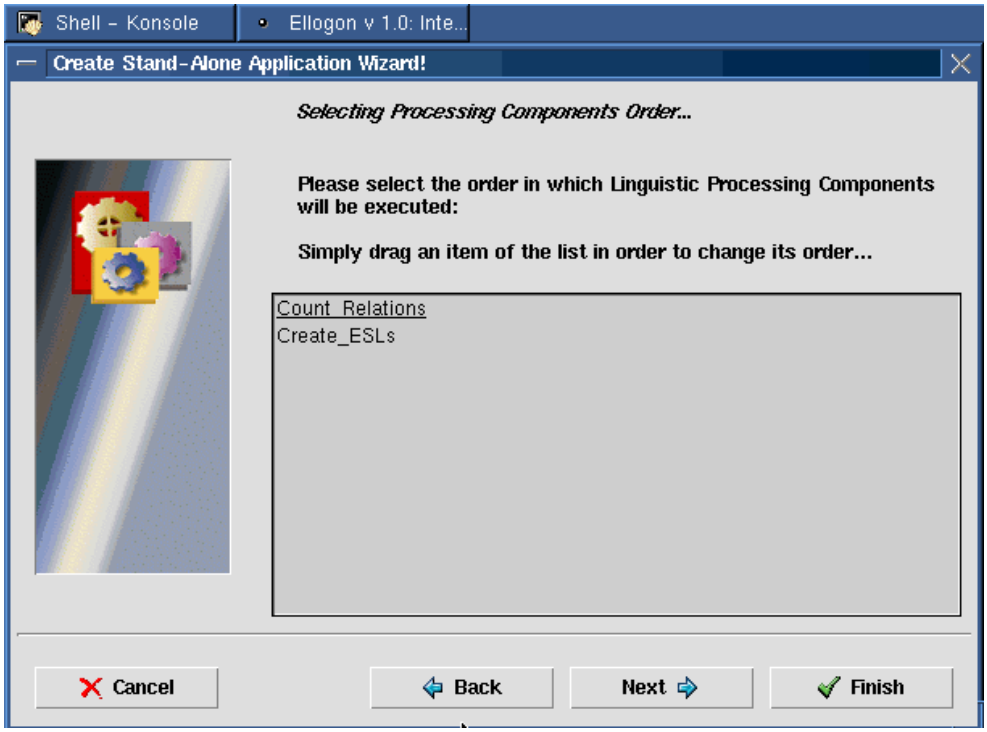

Figure 4.22 Selecting the order in which the modules will run

Finally, after all those steps, the wizard will have gathered all the necessary information to create your application. The last screen (*Figure 4.23*) is a summary of your steps. You can view what you have told the wizard to do and maybe backtrack, with the "Back" button to make alterations. If you are satisfied with what you have selected, push the "Finish" button.

Depending on the choices you made, a script will be created with the aid of which you will be able to run your stand alone application. Information of how to run are contained into that script. Usually, you will have just to run *wish* with the name of the script given as argument. In the Windows platforms, most probably you have just to double-click to that file.

Note, that in some occasions, you might want to edit the created script, which must be done with caution. Information on how to edit that screen, you can find in the *Developers' Guide to Ellogon*.

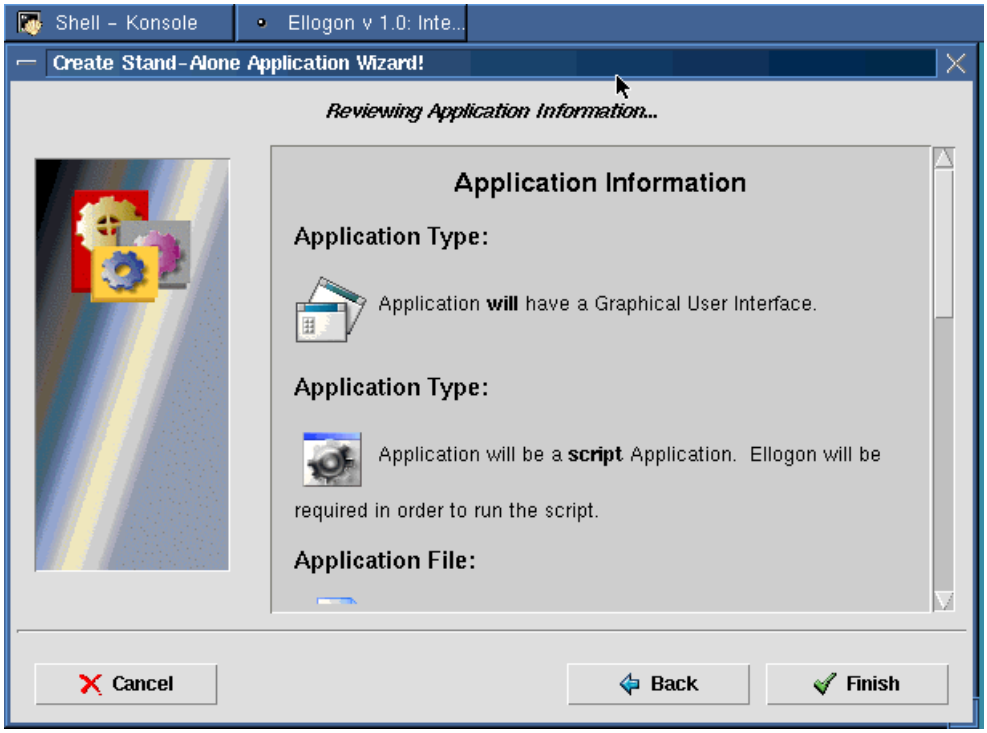

Figure 4.23 Summary of the choices made

Chapter

5

5 Advanced Topics

In this chapter, we shall describe some of the advanced topics of *Ellogon*. More specifically, we begin by examining several issues on the Compatibility of *Ellogon* and the well-known GATE 1 platform, and describe how to convert Collections from the *Ellogon* format to the GATE 1 format and vice versa. Then we describe the Console of *Ellogon* and the HTTP Daemon. Finally, we describe various options that you can set on *Ellogon*.

5.1 GATE 1 Compatibility

GATE 1 is a well known and used text-engineering platform developed in the Sheffield University. *Ellogon* is compatible with GATE 1; in other words, modules written for GATE 1 can run in the *Ellogon* platform as well. The opposite is not true; GATE 1 is not compatible with *Ellogon*.

When you are writing Components, *Ellogon* performs more strict tests on the functions you are using. More specifically the variables that you pass on several functions, are more strictly examined, and if not an appropriate type is found, the *Ellogon* prompts you an error message. On the other hand, GATE 1 does not perform so many tests on what type of variables you pass on the functions of its API. Instead, it leaves it up to you to be careful with the code you are writing.

Many modules have so far been written for GATE 1. This led us to enforce compatibility with GATE 1. That is, we devised an option that allows *Ellogon* not to perform all those strict tests, *exactly* in the manner that GATE 1 does. By this, Components written for GATE 1 are now able to run under *Ellogon*.

In order to turn on full GATE 1 compatibility, from the `File` menu of the *Ellogon* main window, choose `Options`. On the dialog that opens click on the GATE 1 tabbed pane and then check the `Enable Full GATE 1 Compatibility` checkbox, as shown in *Figure 5.2*. This will enforce *Ellogon* to be fully compatible with GATE 1. The same function is provided by clicking on the shortcut placed on the status bar of a System (see *Figure 5.1*)

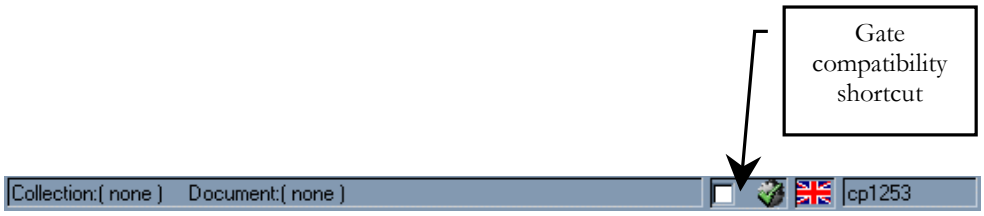

Figure 5.1 GATE 1 Compatibility Shortcut on the Status Bar of Ellogon

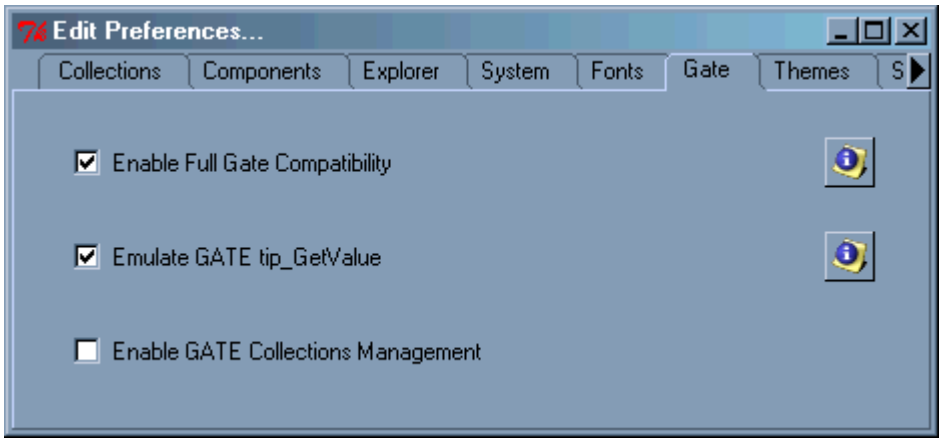

Figure 5.2 GATE 1 Compatibility

5.2 Converting Collections' format from GATE 1 to *Ellogon* and vice versa

In section 3.3.6 we mentioned that *Ellogon* supports two kinds of formats, internal representations by which it stores Collections on the hard drive. The one was a native to *Ellogon* format, and the other one was a format compliant to that of GATE 1. Furthermore, we described how you are able to convert a Collection from the one format to the other.

Now, if you have several GATE 1 Collections in your Collections' Path (see section 5.3.1), *Ellogon* will not be able to handle those Collections, unless you explicitly tell it so. In order for *Ellogon* to understand and handle Collections in the GATE 1 format exactly as it handles Collections in its native format, click on the "Enable GATE 1 Collections Management", shown in *Figure 5.2*

5.3 Setting Various Paths

When the *Ellogon* platform starts up, it seeks on specific paths for modules and Collections. Those paths can be modified so that you will be able to store your Collections and Modules on arbitrary folders.

5.3.1 Modifying the Search Path for Collections

In order to modify the path that *Ellogon* searches for Collections, from the main window of the *Ellogon* platform (see *Figure 3.1*) select `Modify Collection Search Path` from the `Collection` menu. A new dialog box, depicted in *Figure 5.3*, appears.

Figure 5.3 Modifying the Collections' Search Path

In that module, you can add the paths of the folders that you want *Ellogon* to search for Collections. This is done by either pushing the add button, indicated by the plus sign (+) button and browsing for folders, or simply by dragging those folders into the dialog from your document manager. In case you drag a *file* instead of a *folder* into that dialog, it will be accepted; this dialog will only accept folders that will be dragged on it. In order to remove a path, select it from the dialog and push the remove button indicated by the minus sign (-).

5.3.2 Modifying the Search Path for Modules

The process for modifying the search path for modules is exactly the same as that for modifying the search path for Collections. From the `Modules` menu, select `Modify Modules Search Path` in the *Ellogon* main window. A similar dialog to that depicted in *Figure 5.3* will appear. You can add or remove paths exactly as was described before. Keep in mind that if you drag files on that dialog, instead of folders, it will not accept them.

5.4 The Console

The *Ellogon* console can be launched by clicking on the `Open Console` item of the `File` menu on the *Ellogon* main window. That console is depicted in *Figure 5.4*, and it provides all the functionality of the Tcl/Tk language along with the API of the *Ellogon* platform.

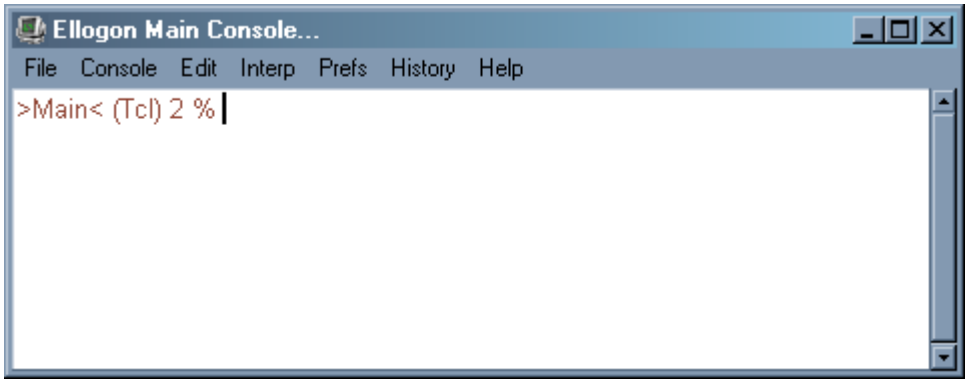

Figure 5.4 The Ellogon Console

5.5 Options

In this section, we shall describe the various options you can set on the *Ellogon* platform. From the `File` menu of the *Ellogon* main window, choose `Options`. A new dialog will open, similar to the one depicted in *Figure 5.5*.

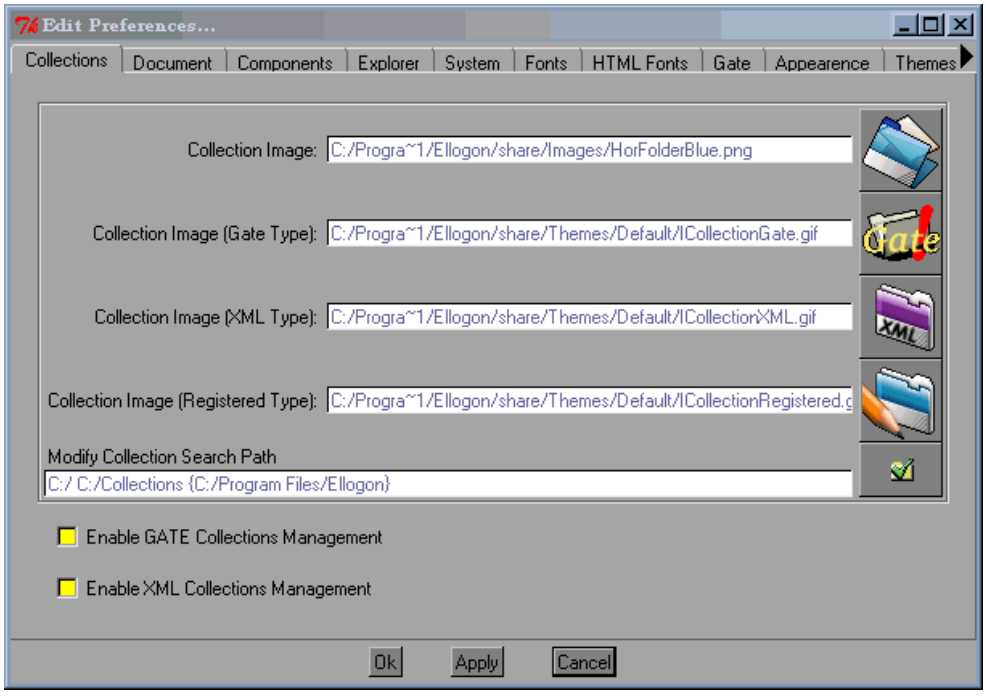

Figure 5.5 The Options Dialog

5.5.1 Collections' Options

This dialog contains several tabbed panes. The first one named "Collections" contains information about the Collections, that you can modify. For example, you are able to change the images used for the Collections and Documents, whether they are in the native format, XML or the GATE 1 format. In order to do so, specify, in the appropriate text area, the path of the image, or press the associated button and search for the image through the dialog box that opens (see *Figure 5.6*)

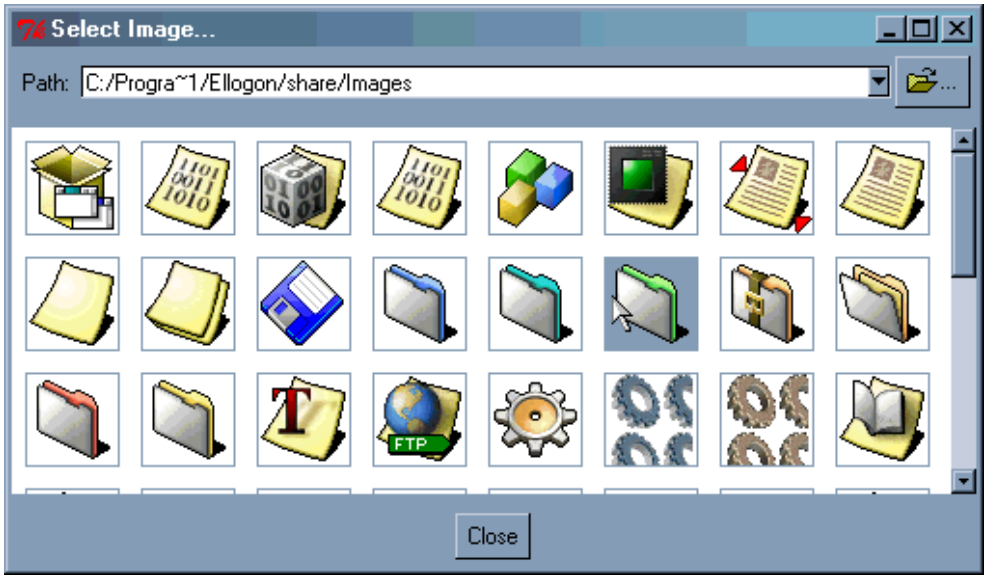

Figure 5.6 Select an Image

Furthermore, you can modify the search path that *Ellogon* looks for Collections. Simply push the button for modifying the search path of a Collection (see again *Figure 5.5*). A dialog similar to that depicted in *Figure 5.3* will appear. In that dialog you are able to add or remove paths, as described in section 5.3.1.

Finally, from the "Collections" tabbed pane of the Options dialog, you can instruct *Ellogon* to manage all the Collections in the GATE 1 format. Simply push the associated button, shown in *Figure 5.5*. The effect of this action is that *Ellogon* will treat all the Collections in the GATE 1 format, as was described in section 5.2. A similar effect has the "Enable XML Collection Management", only that *Ellogon* will treat all the Collections on the XML format.

5.5.2 Documents' Options

On the Document pane, shown in *Figure 5.7*, you can change the images of the documents used for various formats. In order to do so, specify either the full path, or click on the button. On the latter case you shall be prompted with a dialog, similar to that in *Figure 5.6*, which will enable you to choose an image.

5.5.3 Components' Options

If you press the Components tabbed pane the Options dialog will become as shown in *Figure 5.8*.

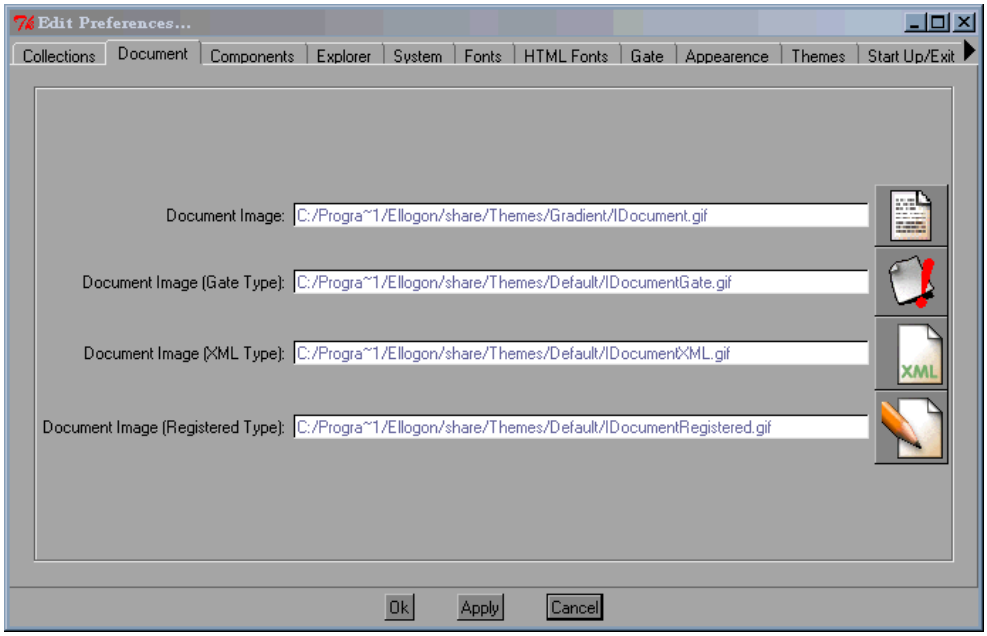

Figure 5.7 Changing the Options for Documents

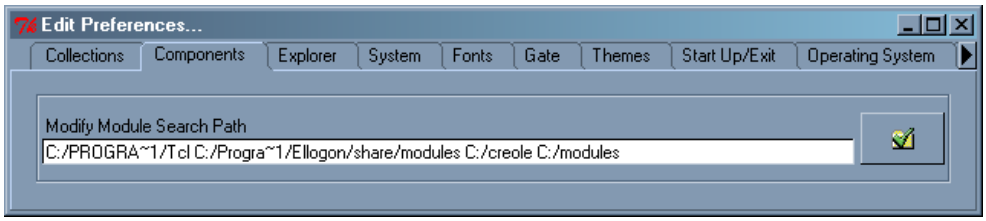

Figure 5.8 Changing the Options for Components

From that dialog you are able to change the search path for Modules. Simply press the associated button and a dialog box will open, allowing you to add or remove paths from the Modules' search path. We described the process in section 5.3.2, as well.

5.5.4 The Explorer's Options

In the *Ellogon* jargon, by "Explorer" we mean the tree view of the Systems, and their folders, that appear in the *Ellogon* main window (see *Figure 3.1*). If you push the Explorer's tabbed pane in the Options dialog, it will become as shown in *Figure 5.9*.

That dialog, enables you to change several of the images and the colors that it uses. In order to change one of the images either you specify the path for the image in the text area, or you push the associated button, and search for an image, through a similar dialog to that shown in *Figure 5.6*.

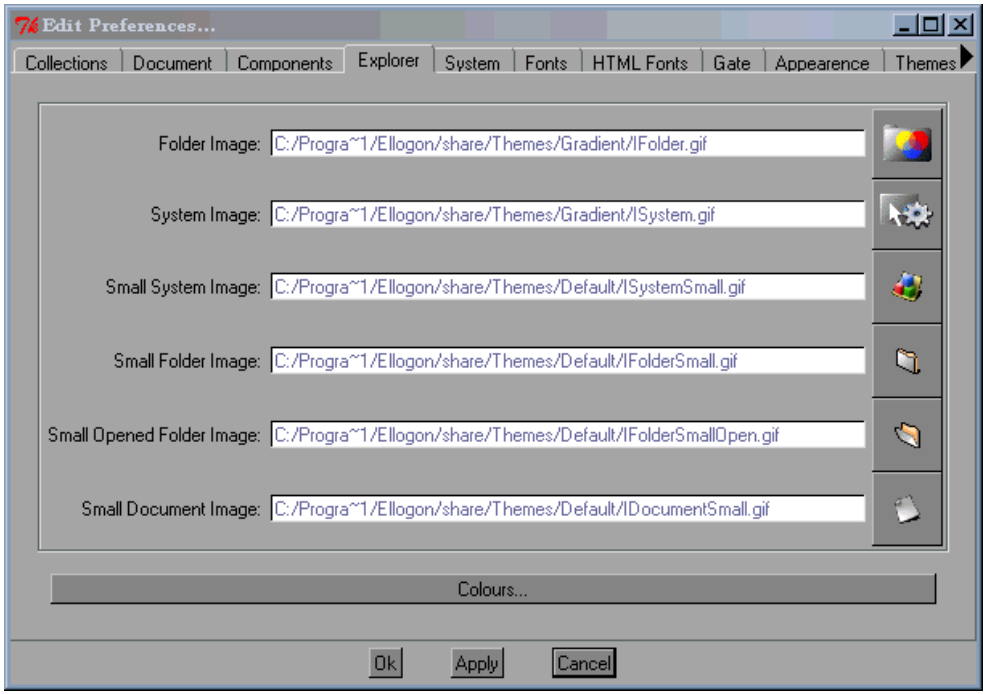

Figure 5.9 The Explorer's Options

5.5.5 The System's Options

Pushing in the System tabbed pain the Option's dialog will become as shown in *Figure 5.10*

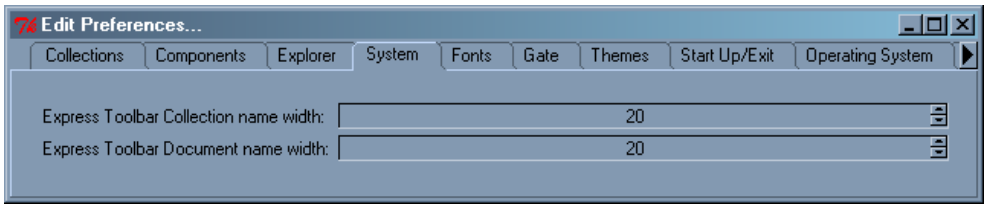

Figure 5.10 The System's Options

For the moment the only parameters you can change is the length for the name of the Collections and the Documents of the Collections toolbar depicted in *Figure 3.4*. By pushing the up and down arrows you can adjust those lengths, which actually correspond to character length. For example, in the figure, the number 20 corresponds to a length of 20 characters.

5.5.6 Changing Fonts

Ellogon enables you to change the fonts used for its interface, as well as the fonts used in the HTML pages it uses.

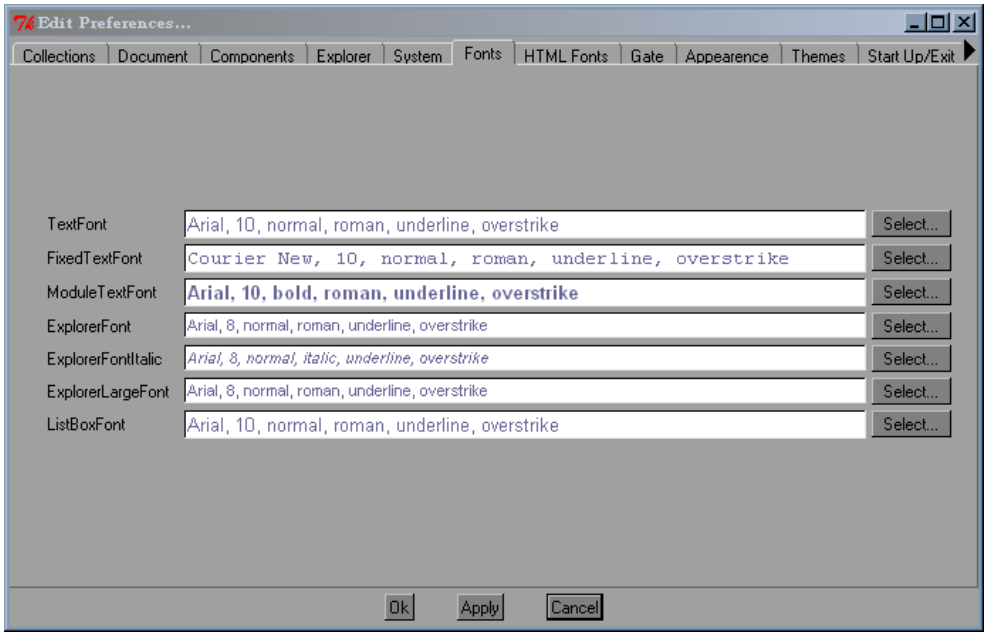

Figure 5.11 Changing Fonts for the Ellogon interface

To change the Fonts for the Interface of the *Ellogon* push on the Fonts tab (see *Figure 5.11*). On this tab, you can change several of the fonts used for the interface of the *Ellogon*. On the font you want to change, click the "Select" button. A dialog will appear which will allow you to choose the fonts of your liking.

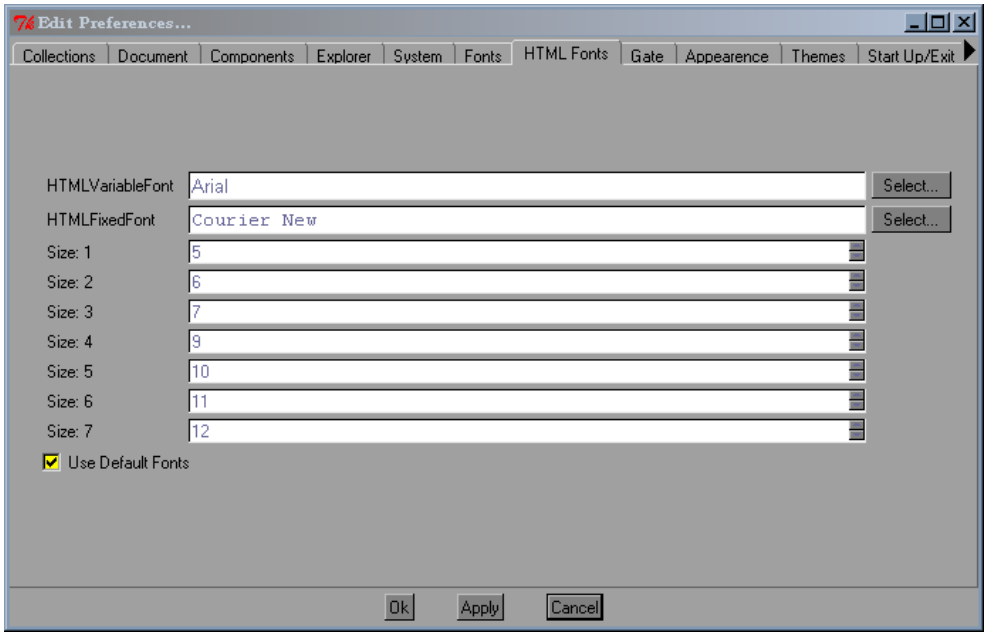

Figure 5.12 Changing the Fonts for the HTML

On the HTML Fonts tab, depicted in *Figure 5.12*, you can change the fonts for the HTML pages in *Ellogon*. The things you can change are the type of the variable and the fixed size fonts, as well as their sizes. For the types of fonts click on the "Se-

lect" button. A dialog will appear on which you can choose the fonts that suit you better. For the size of the fonts, you have to select a number from the corresponding list.

5.5.7 The GATE 1 Compatibility Options

By pushing the Gate tabbed pane you can modify the parameters for GATE 1 compatibility. Actually this tabbed pane was discussed in sections 5.1and 5.2 so we will not describe it here.

5.5.8 Appearance Options

The only option you have on the Appearance tab is the "Enable Tear Off Menus". If you enable this option you an drag the toolbars of the *Ellogon* interface and keep them as separate windows on your desktop. This especially helpful you want quick and frequent access on a toolbar.

5.5.9 The Themes' Options

By pushing the Themes tabbed pane you are able to change the appearance of *Ellogon* by selecting among three themes. This tabbed pane is shown in *Figure 5.13*. *Ellogon* offers three themes: the Default, the Gradient and the Moz. Before you choose a theme, select it from the drop down list (see *Figure 5.13*) and have a preview of the theme in the area below the drop down list.

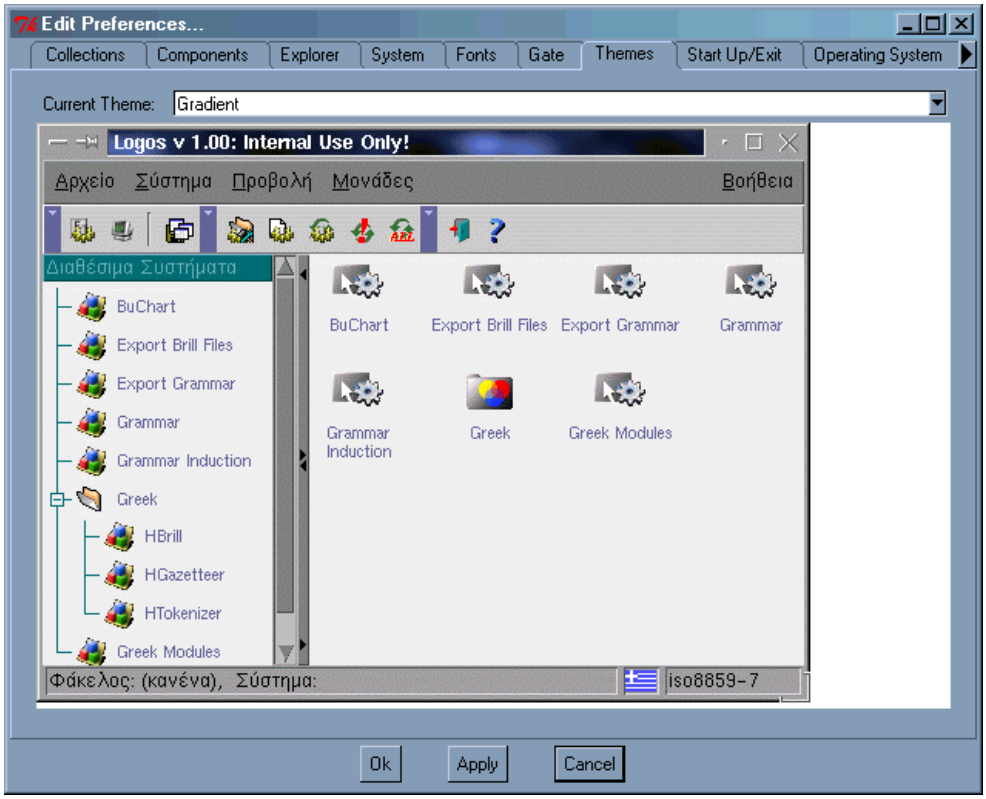

Figure 5.13 The Themes' Options

5.5.10 The Start Up and Exit Options

In *Figure 5.14* you can see the Start Up and Exit tabbed pane of the Options dialog box. As you can see, there are two parameters that you can modify. The first, "Im-

pressive Start-up" causes *Ellogon* to use a different image, a little more impressive than the default, during start up.

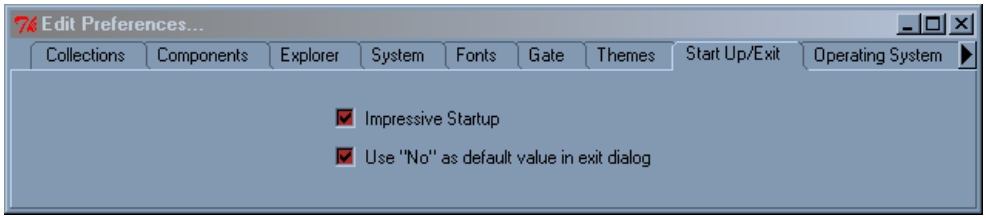

Figure 5.14 The Start Up and Exit Options

When you exit *Ellogon*, and have several Systems open, it will ask whether you want to save the changes you have made thus far to the Collections or not. This dialog has two options: Yes and No. You can specify what the default value of the Dialog will be from the second option. If you have it clicked, No will be the default value, otherwise it will be Yes.

5.5.11 The Operating System Options

If you push the Operating System tabbed pane on the Options dialog, you will be able to modify various parameters. For the moment, the only parameter that you are allowed to change is the editor that will show the modules source code. In section 3.7 we described how to edit the Components source code. Also you can enable the Execution Server, through the associated button.

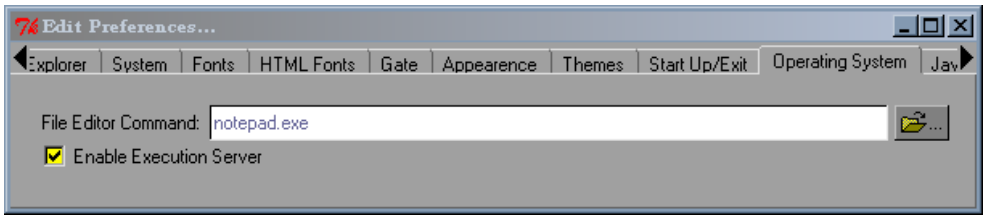

Figure 5.15 The Operating System's Options

Chapter

6

6 Installing *Ellogon*

In this chapter, we shall describe the process of installing *Ellogon*.

6.1 Requirements

The following list is the minimum requirements that your system should meet in order for *Ellogon* to be installed and run properly.

- Pentium II, 300 MHz or higher.
- 64 MB of memory.
- 110 MB free space on the hard disk.
- Tcl/Tk 8.4.0.1b3, or a newer version, already installed. (You can obtain Tcl from http://www.tcl.tk/)
- A C++ compiler (under Unix, gcc is preferred)

6.2 Set-up

Ellogon can be installed in three platforms for the moment: Windows, Linux and Solaris. The steps that you follow are exactly the same, independently of platform, although the internal process that *Ellogon* follows to install itself varies, according to the platform.

In order to initiate the process of installing *Ellogon* you have to run the Installation Program. The first dialog that you will be prompted is shown in *Figure 6.1*. The next screen, *Figure 6.2*, is informing that you are about to install *Ellogon* and that this program requires a license agreement. If you want to install *Ellogon*, push the Next button.

The Next dialog, shown in *Figure 6.3*, prompts you to read the License Agreement. Please, read it carefully, and if you agree with it push the Yes button in order to continue with the installation of *Ellogon*. If you do not agree push the No button; the *Ellogon* installation will be cancelled in this case.

Figure 6.1The First Screen of the Instalation Dialog

If you accept the License Agreement, then the Installation wizard will prompt you a screen in which you have to choose the directory in which the *Ellogon* will be installed. This screen is depicted in *Figure 6.4*. Depending on the platform that you are installing *Ellogon* the wizard will propose a default directory. Of course, if you do want *Ellogon* to be installed on that default directory, you can specify your own one, by pushing the Browse button. Once you are finished with choosing a directory, push the Next button.

The screen that appears next, *Figure 6.5*, prompts you to choose between a typical installation and a custom installation. If you push the typical installation you will be shown the screen depicted in *Figure 6.7*. If you choose a custom installation, on the other hand, you will be shown the screen in *Figure 6.6*.

In the screen depicted in *Figure 6.6* you can choose to install or not install some of the components that *Ellogon* comes equipped with. After you have chosen the components you want to install push the Next button. You will be shown the screen depicted in *Figure 6.7*.

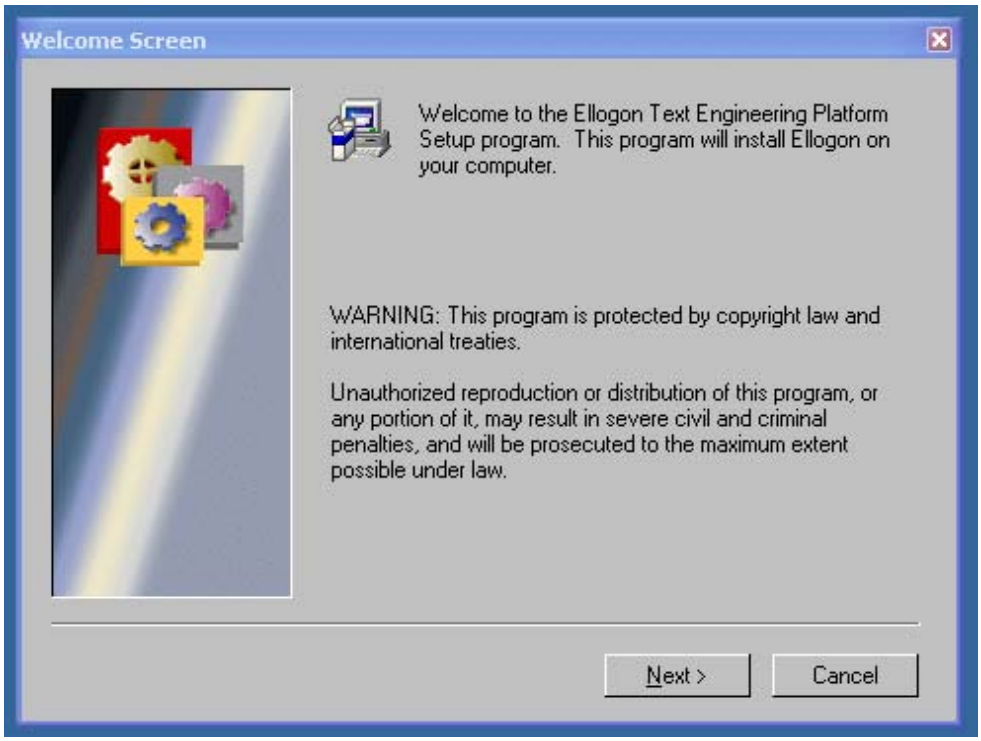

Figure 6.2 Prepering to Install Ellogon

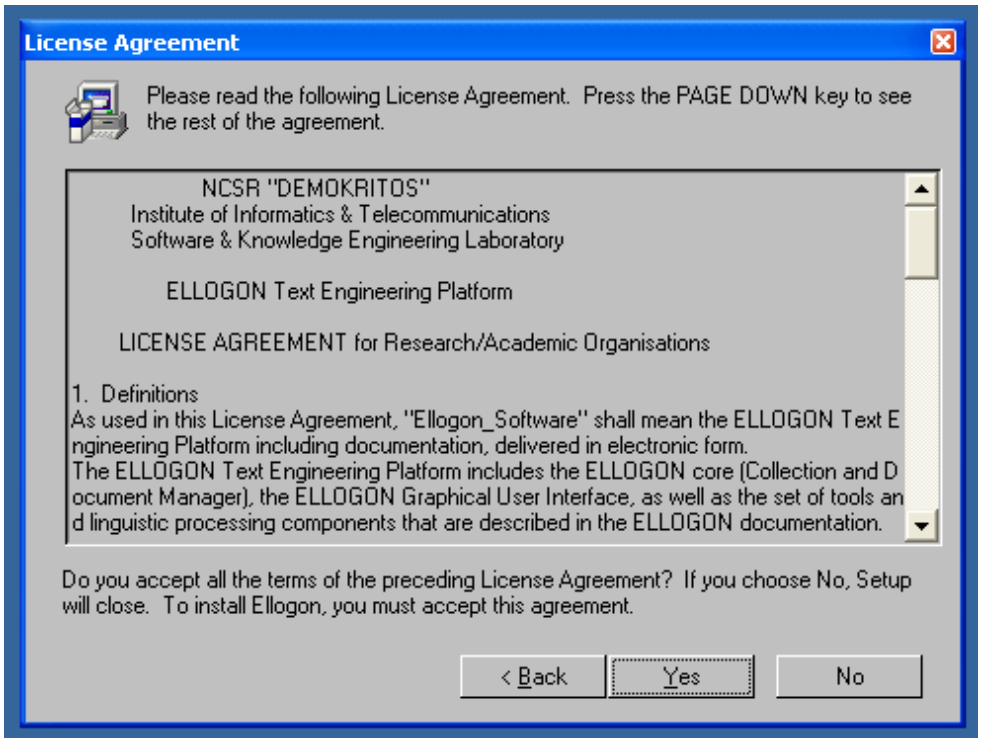

Figure 6.3 The License Agreement

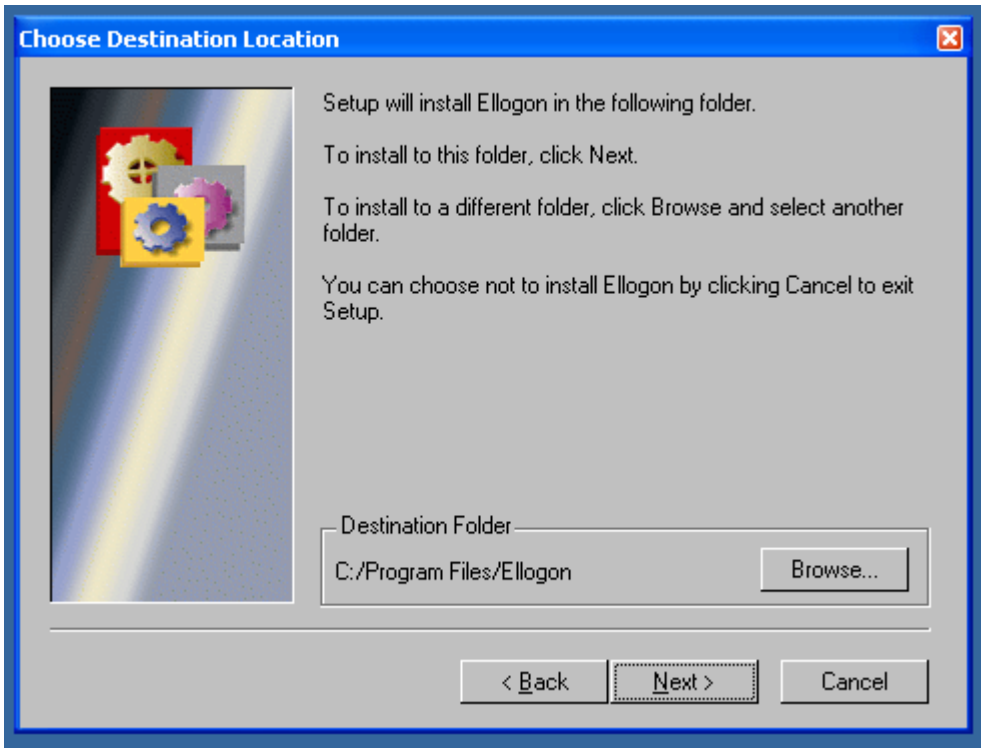

Figure 6.4 Choosing a Directory for Ellogon to be Installed

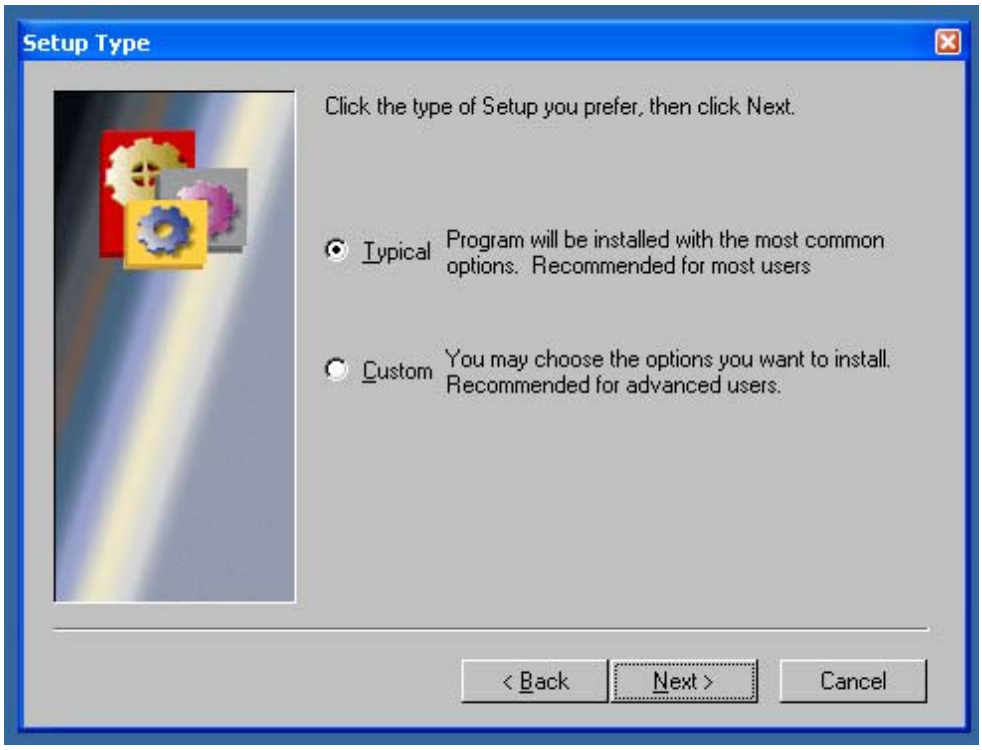

Figure 6.5 Choosing between Typical and Custom Setup

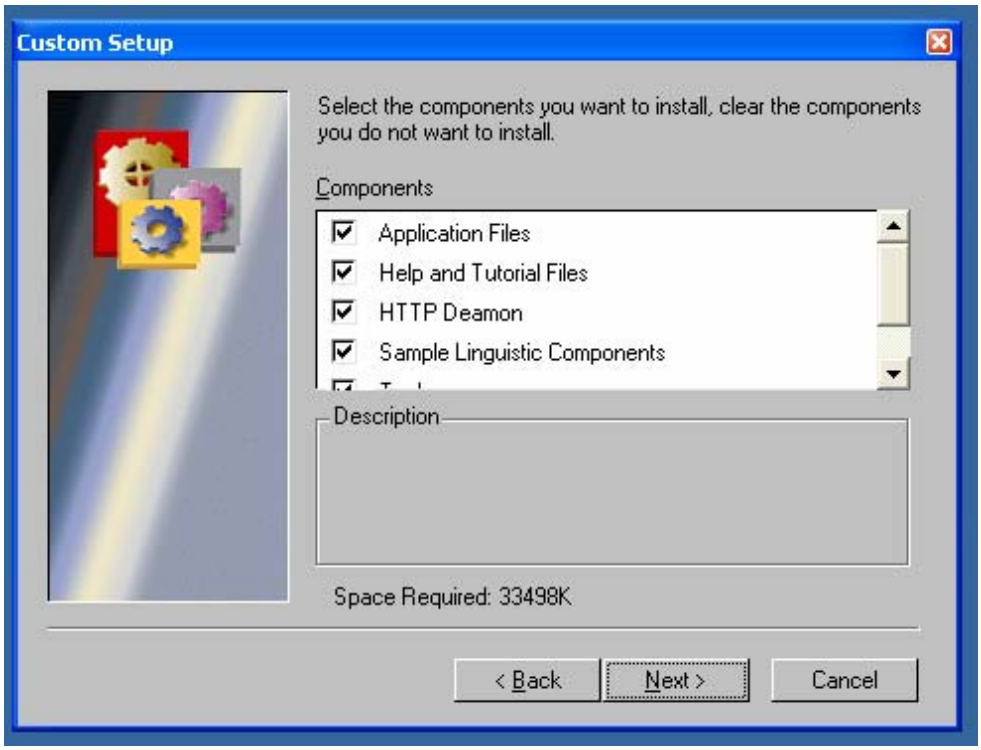

Figure 6.6 Choosing the Components of Ellogon you want to install

Figure 6.7 Selecting a folder for the Ellogon Icons to be installed

In the screen depicted in *Figure 6.7* you can choose a folder in which the icons for the *Ellogon* will be placed. If you are finished press Next.

The screen that appears next, shown in *Figure 6.8*, constitutes in fact a summary of your choices so far. It lists the directory in which *Elllogon* will be installed and the type of setup you have chosen. If you are satisfied with your choices push the Next button; if you are not you can push the back button to track back a step and change the choice you had made.

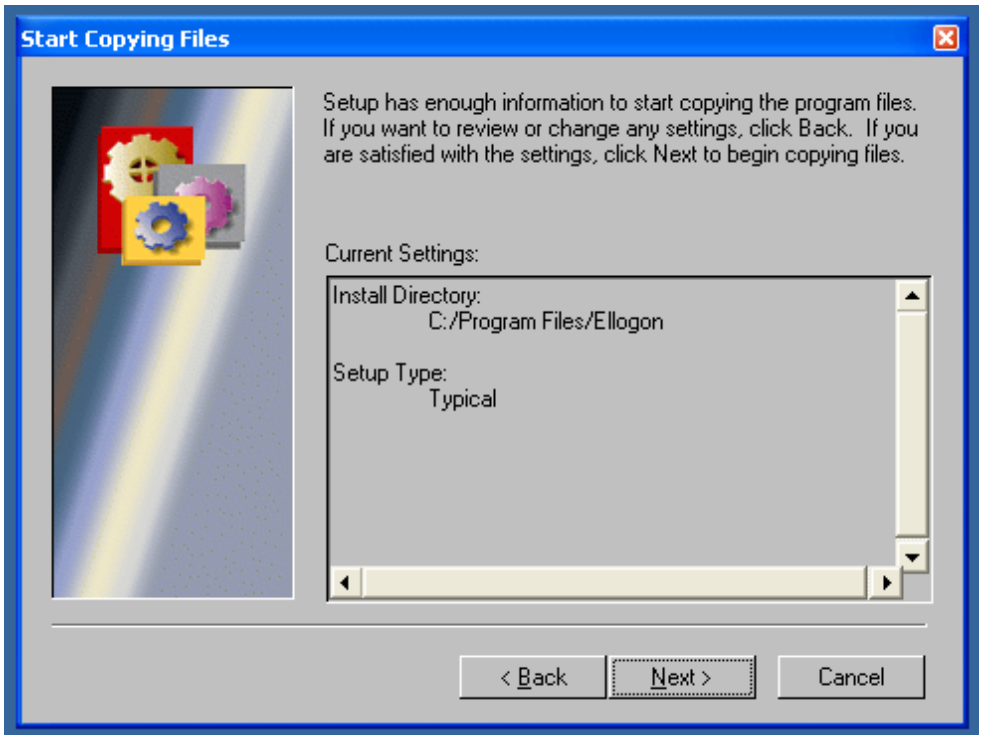

Figure 6.8 A summary of the Installation

If you push the next button on the screen depicted in *Figure 6.8*, then the *Ellogon* installer will start copying the files on your hard disk, as depicted in *Figure 6.9*.

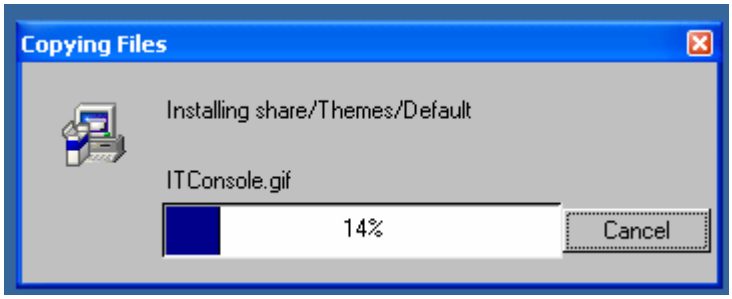

Figure 6.9 Installing Ellogon on your hard disk

Once the process of copying *Ellogon's* files on your hard disk has finished, a dialog will prompt to you, as shown in *Figure 6.10*. On that dialog you can choose to start the *Ellogon* and/or to create a shortcut of *Ellogon* on your desktop. Once you have finished with your choices push the Finish button. *Ellogon* setup will be now completed.

Figure 6.10 Completing the installation process

6.3 Uninstalling *Ellogon*

The process of uninstalling *Ellogon* varies slightly according to the Operating System you have.

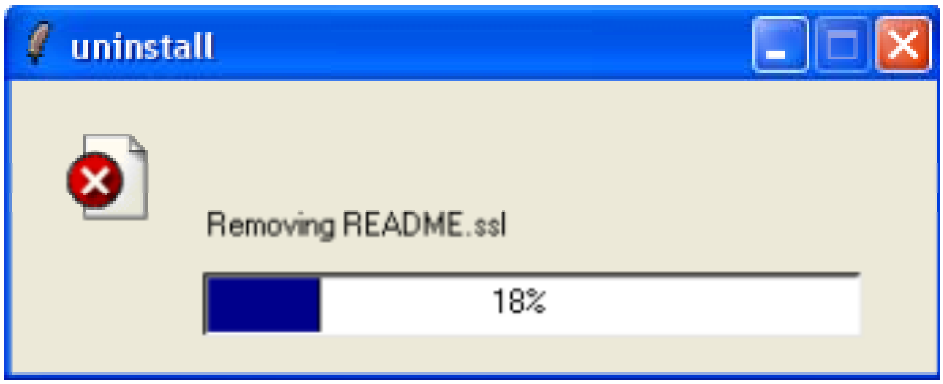

Figure 6.11 Uninstalling Ellogon

6.3.1 Windows Systems

On Windows Systems the process of uninstalling *Ellogon* is the familiar one of uninstalling any other program. In the Control Panel choose the Add or Remove Programs. In the list that appears choose *Ellogon* and push Uninstall. *Ellogon* will be uninstalled.

6.3.2 UNIX Systems

On UNIX systems the process is a bit more manual. In the folder that you have installed *Ellogon* you will find a file with the name "Uninstall". You have to run this file in order to uninstall *Ellogon*.