The EUDICO project, multi media annotation over the internet

Albert Russel, Hennie Brugman,
Daan Broeder, Peter Wittenburg

Max Planck Institute for Psycholinguistics
P.O. Box 310, 6500 AH Nijmegen, The Netherlands
{albert.russel, hennie.brugman, daan.broeder, peter.wittenburg}@mpi.nl

Abstract

In this paper we dsecribe a software environment that facilitates media annotation and analysis of media related corpora over the internet. We will describe the general architecture of this environment and we will introduce our Abstract Corpus Model with which we isolate corpora specific formats from the annotation and analysis tools. The main set of tools is described by giving examples of their usage. Finally we will discuss features regarding the distributed character of this environment.

1. Introduction

The EUDICO (European Distributed Corpora) project began in 1997 at the Max Planck Institute for Psycholinguistics (MPI) as a project to make multi media linguistic resources available to the largest possible scientific audience. The first generation of its tools offers platform independent software that works over the Internet to give researchers all over the world access to linguistic resources with multimedia content[1].

Work on EUDICO was inspired by extensive experience with building software tools for the analysis and annotation of both analog and digital audio and video material over the last ten years (Wittenburg et al., 1998). One of EUDICO's direct predecessors is MediaTagger[2], a multi tier video annotation tool based on Quicktime and running on the Apple Macintosh platform.

The essential design decisions for the EUDICO multimedia annotation and exploitation framework were: (1) distributed and internet capable architecture; (2) annotation format independent operation; (3) highly interactive support for streaming of audio and video data; (4) different fully synchronized viewers presenting annotation and media data to the user; (5) extensible framework, with respect to both corpus formats and software tools; (6) both central and local data storage; (7) centralized software maintenance.

2. Resources and projects

The linguistic resources that we support in the EUDICO project consist of media tracks and separate annotation layers that contain linguistically relevant descriptions of events that occur within the media tracks. Annotations in these layers can be exactly aligned with media time, but may be merely sequential. A mix of time aligned annotations and ordered annotations is possible, even within the same layer.

Since our tools are intended to be used in a wide variety of annotation and analysis tasks, and with many different language resources (current and new), the annotation layers may be configured in a theory-neutral way. EUDICO's annotation type system is flexible enough to allow users to compose their own annotation layers setup, including the definition of new annotation layer types.

Resource formats currently supported are resources in the Childes' CHAT format (MacWhinney, 1995), the Gesture format (a proprietary relational database format used with MediaTagger), and the Tipster[3] format via the GATE[4] API from Sheffield University.

As well as supporting various speech and gesture related projects of scientists at the MPI, EUDICO technology will be applied in the European MUMIS project and for the exploitation of collected corpora in the Spoken Dutch Corpus project[5] (Oostdijk, 2000).

3. The system's architecture

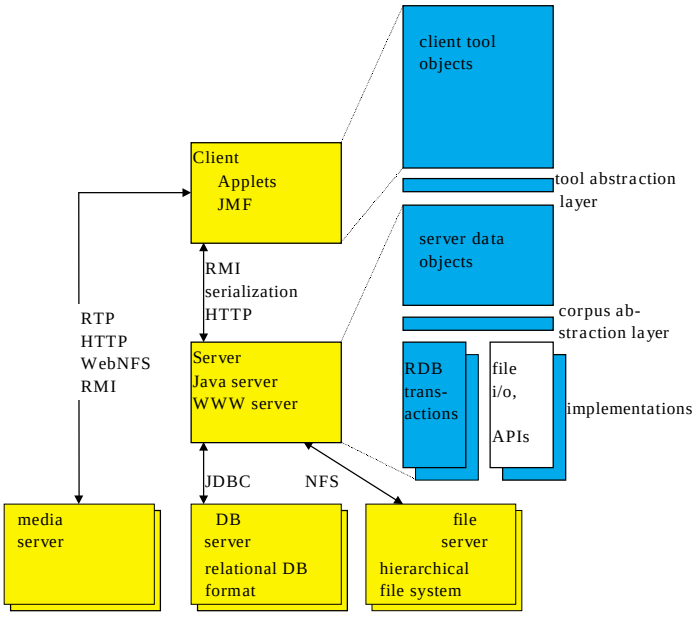

Figure 1: EUDICO's architecture

[1] http://www.mpi.nl/world/tg/lapp/eudico/eudico.html
[2] http://www.mpi.nl/world/tg/CAVA/mt/MtandDB.html
[3] http://www.tipster.org
[4] http://www.dcs.shef.ac.uk/research/groups/nlp/gate
[5] http://lands.let.kun.nl/cgn

EUDICO's architecture was designed to facilitate distributed operation, easy addition and exchange of modules and abstraction from annotation format specifics. Therefore the architecture is mostly three-layered (Figure 1), separating application logic and data representations on the server.

3.1. The Abstract Corpus Model

The basis of the EUDICO architecture is what we call the Abstract Corpus Model (ACM) which we created after carefully analyzing a range of annotation formats relevant for our work. This generic model captures the functional equivalence of a range of elements that occur in most corpora (e.g. most corpora contain transcriptions or documents that can have multiple annotation layers/tiers, so it makes sense to have an operation on transcriptions called "getTiers").

The scope of the model was determined by a set of use cases that we wanted to implement in our tools. Therefore we explicitly chose to include annotation types in our model, in addition to modeling the structural elements of language resources (in constrast with Bird and Liberman (1999)).

To the client the common model offers uniform services for each corpus element irrespective of what corpus the element comes from. The portfolio of services offered to clients is called the "tool abstraction layer".

On the server, the services of the "tool abstraction layer" are implemented differently for each supported corpus. Specific implementations of objects can be loaded from widely different existing resources: from relational databases using JDBC, from plain text files on file systems accessible locally or over NFS, or from proprietary systems accessible using a specific API. Currently we have implemented instances of all of these three types. The minimal set of all services that have to be implemented specifically for each corpus format is called "corpus abstraction layer".

The ACM is an object oriented model that is implemented in the form of Java classes and all software tools are designed to work with this ACM. Dealing with a new corpus within the EUDICO framework simply means making a mapping from the specific corpus format to the ACM by implementing the limited range of operations from the corpus abstraction layer. All existing EUDICO tools are thereafter instantly available for the new corpus. For example, if we need an XML based format for some reason (and we will: for data exchange, long term archiving, local persistent storage, etc.) it will be straightforward to extend our ACM to support it.

3.2. Media servers

EUDICO was designed with media integration as the default modus of operation. Since the system is network based it makes use of streaming media. Viewers on the client are able to represent the synchronization of annotation data with media time while the media data is streaming over the network.

The client tools make use of the Java Media Framework (JMF), a standard Java extension made freely available by SUN Microsystems. JMF is an extendible media architecture that supports the transport and display of a number of digital audio and video formats.

One of the main requirements for EUDICO's streaming media solution is the ability to address and transfer specific scenes from media objects on the server. Since JMF's HTTP support only allows 'progressive download' of complete media objects, and it turned out to be practically impossible to configure a suitable RTP server, we were forced to extend JMF with a media transport mechanism of our own. We explored three solutions: (1) an alternative mechanism based on HTTP, but with support for downloading of specific media fragments, (2) a mechanism based on WebNFS, effectively making media objects appear as local files on the client and thus allowing playback of media fragments, and (3) a mechanism that encapsulates media files on the server with remote objects using RMI (Remote Method Invocation). This also makes server media files appear as local files to JMF on the client system.

For the moment the RMI based mechanism turns out to be the most promising approach, both in terms of usability via internet routers and firewalls and in terms of performance. However, if better streaming media solutions become available they could be plugged into EUDICO's modular structure.

4. EUDICO's tool set

A Eudico server supports a set of data and the tools that can operate on that data. The data set consists of corpora that can be described by the ACM. The tool set consists of generic tools that work on the ACM and of proprietary tools that can work directly on a specific corpus format. An example of the last type of tool would be a search engine that had to be able to explore large amounts of corpus data very fast. For both types of tools we provide the client a much as possible with a generic user interface so that they do not have to distinguish between generic and proprietary tools.

As soon as a client connects with the Eudico server they will have to provide a user/password combination to identify themselves. Once the client is identified the appropriate set of available data is presented as a browsable tree. The client can select data items from this tree. As soon as the data item is selected, the available tools for that item are shown in another browsable tree. This available toolset can also be made dependent on the client's identity. For example it would probably not be a good idea to give every client permission to add or change annotations.

The set of available tools is still being expanded. We are getting feedback on tools from a number of linguists involved in a range of projects and working on a number of specific corpora.

At the moment we distinguish three main tool types, i.e. Viewer tools, Annotation tools and Search tools.

4.1. Viewer tools

The Viewer tools facilitate the display of the media data and its annotations. More than one of these viewers can be active on the same data at the same time. In this case they are fully synchronized with one another and with the media data.

For both audio and video media data we provide players that have controls for starting and stopping, playing loops, and jumping to a specific media time. The audio data can be visualized as a wave form.

The annotation data that has to be visualized consists of tiers that contain tags that describe events in the media data. A simple example of such a tier could be an orthographic transcription of the utterances that are spoken in the media data. More complex tiers can contain for example tags that describe certain gestures that are made by a speaker in the media data. In figure 2 a viewer is shown in which tags on these two tiers are displayed as subtitles to the video image. When the video plays, the subtitles play along in sync with the media time.

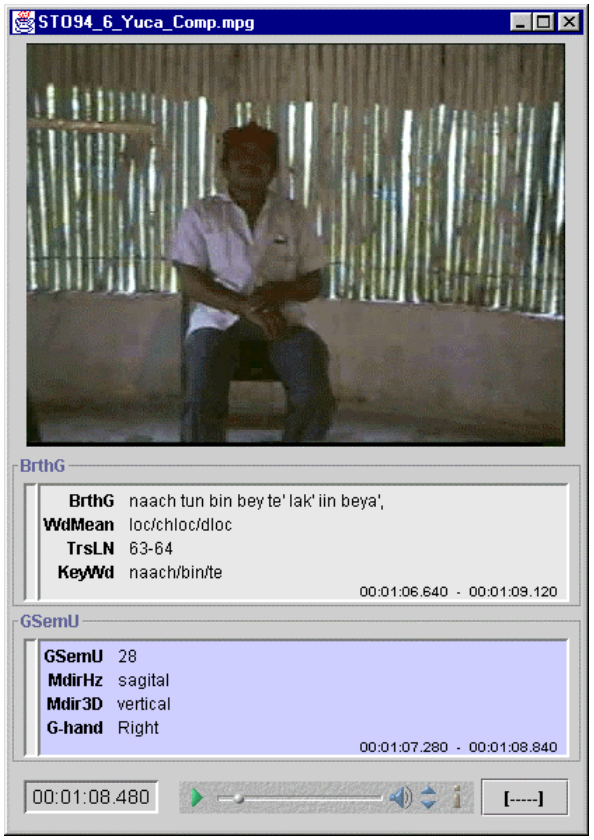

Figure 2: Video window with subtitle.

Other viewers can show the tags of a tier as a time ordered scrollable list. When a tag is selected from this list all other viewers, including the media player, jump to the start time of that tag. When visual information regarding the relative time position of annotations to each other is needed, we offer a time line view (figure 3) where the content and duration of annotations on selected tiers are indicated along the time line.

This makes it possible to easily detect overlap of segments from different annotation layers. If the user scrolls to a certain position (media time) in the time line view, all other viewers follow and also display that position.

The modular design of EUDICO's synchronized viewers makes it very easy to come up with alternative representations of annotation data and to couple them to the other viewers.

4.2. Annotation tools

When the user wants to make new annotations, tools are offered for selecting time intervals within the media data and for entering the annotation content. By selecting a segment in the media data, the annotator defines the begin time and the end time of a new tag. This time selection can be made with the help of tools that are optimized for either the audio signal (with the help of the audio wave form) or the video signal (by specifying in- and outpoints). After defining this time information the user can enter the tag's value(s). One of the advantages of explicitly supporting a type system for annotations is that it can be used to guide the user when entering annotation values thus ensuring consistency and correctness. If, for example, a certain tier can only contain tags with a limited number of different values, we can offer the annotators a drop down menu by which they can choose one of these values, thereby assuring that they cannot erroneously enter an undefined value. It is also possible to define certain constraints between tags on different tiers. In this way we can guide the annotators while they are entering data in a dependent tier by checking continuously to see if the constraints are being adhered to.

4.3. Search tools

If annotated corpora are available, researchers may want to inspect only the parts of that corpora that meet specific search criteria. One main type of query is for combinations of annotation values in a specific ordering or time relation to one another. For example, they could be interested in all segments where a speaker raises both his right and left hand within a 2 second time interval followed by the word 'naach'. We are currently implementing tools for specifying and executing such queries, as well as viewers for inspecting the query results. Another type of query that we want to support regards statistical questions about a data set. A user should be able to ask questions like "how many times does the speaker raise his left hand in this transcription".

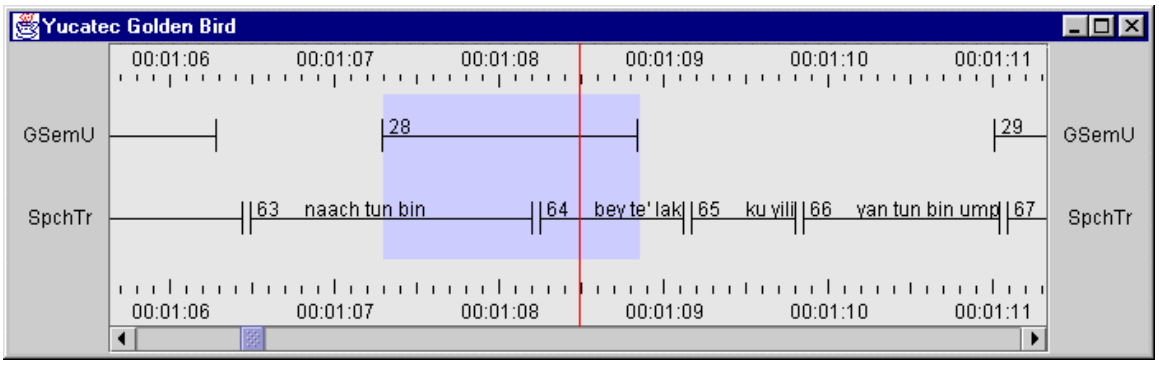

Figure 3: Time line view

As with other EUDICO tools, these query specification tools will be corpus format independent as far as the user interface is concerned, though the actual search engines will be corpus specific. We made this choice because it is not efficient to translate a complete corpus into the ACM. We always translate only that part of the corpus that is actually visualized. This approach also allows us to make use of highly optimized corpus-specific search mechanisms like relational database search engines and appropriate indexes.

5. Distributed and Local usage

As can be seen in Figure 1, the EUDICO architecture is based on the client-server model. Both the media data and the annotation data are made available to a client process by specific media and annotation data servers. These servers can be situated on different computers anywhere in the world as long as the client can connect with them through the internet. If necessary, it is also possible to run these servers on the client machine. We will discuss four different setups that we will support. All four setups require that the client machine supports the Java Runtime Environment and the Java Media Framework software library.

5.1. Remote media server and remote data server

This setup gives clients all over the world the possibility to share corpora. The media data and annotation data can be served by dedicated high performance machines that have large amounts of storage available. When a client adds annotations to a corpus these new annotations can be made available immediately to a world wide scientific community. It is also possible for more than one annotator to work on the same corpus element at the same time. Thus it is possible for specialist linguists to cooperate without needing to be at the same location. We have designed our annotation software such that this concurrent annotation mode is safe.

In this setup the client machine must have a connection to the internet that is fast enough to support MPEG1 streams for video data. Audio data and annotation data are less demanding of bandwidth. The requested parts of the media data are streamed to the client's computer when they need to be rendered. There is no need for temporary storage of the media data on the client's machine.

5.2. Local media server and Remote data server

If for some reason a client machine has not got a fast enough connection to the internet to support MPEG1 streams it is possible to install a light weight media server on the client machine. The media data will then be accessed as files from the client's local file system. The media data then must be sent to the client, for example on CD/DVD ROM. This could be done for several clients with limited bandwidth internet connections all over the world. In this setup they still share the annotation data from one common server with all the above mentioned advantages.

5.3. Remote media server and local data server

This setup can be desirable if the internet connection is fast enough but the client wants to make annotations that are not to be disclosed or of interest to others. The client can use media data that is made available for general usage, but he can do research on that media data without sharing his annotations. In order to support this setup we will provide a data format in which the annotations can be saved on the local file system of the client. This data format will probably be XML based.

5.4. Local media server and local data server

This setup requires no connection to the internet. Modern laptops facilitate digitizing of audio/video data. Such a laptop could be used by a scientist doing field work for annotating the media data he just collected.

6. Conclusion

EUDICO is an open software framework that will be progressively extended with further features to make it a major platform at the MPI and in the Dutch language research community during the coming decade. Because of the highly modular and object oriented architecture of the software it is easy to expand the set of tools, the set of supported corpus formats and the set of media formats and media server architectures. This could make EUDICO valuable to a much larger community.

7. References

Bird, S., Liberman, M., 1999. *A formal framework for linguistic annotation.* Technical Report MS-CIS-99-01, Department of Computer and Information Science, University of Pennsylvania.

MacWhinney, B., 1995. *The CHILDES Project: Tools for analyzing talk*. Second ed. Hillsdale, NJ: Lawrence Erlbaum.

Oostdijk, N., 2000. The Spoken Dutch Corpus, Overview and first evaluation. *To be published in the LREC2000 proceedings.*

Wittenburg, P., Brugman, H.G. & Broeder, D., 1998. Impact of modern information technology on corpus-based research. In A. Trapp, N. Hammond & C. Manning (eds.) *CIP98 Conference Proceedings* (pp. 91-92). York: CTI.