

LREP: A Language Repository Exchange Protocol

Daan Broeder, Peter Wittenburg, Thierry Declerck, Laurent Romary

MPI for Psycholinguistics, Nijmegen, The Netherlands. - DFKI, Saarbruecken, Germany. - LORIA, France
Contact: Peter.Wittenburg@mpi.nl

Abstract

The recent increase in the number and complexity of the language resources available on the Internet is followed by a similar increase of available tools for linguistic analysis. Ideally the user does not need to be confronted with the question in how to match tools with resources. If resource repositories and tool repositories offer adequate metadata information and a suitable exchange protocol is developed this matching process could be performed (semi-) automatically.

1. Introduction

The domain of linguistics is faced with an enormous increase of available language resources and an increasing amount and diversity of available tools that operate on these resources. Also the improved computational facilities have made it possible that many tools and resources do no longer depend on storage with one of the large resource distribution agencies as ELRA and LDC but are stored at the many individual research institutes. Add to this the increasing reliability and bandwidth of the Internet and the availability of high quality Internet connections for even home PC's. There we have a situation that demands the development of an organisational plan together with exploitation tools that offer the user one integrated view of these two complementary domains of tools and resources.

A recent [ACL/EACL] workshop [1] clearly indicated the needs of the community and the potential present in the current registries waiting to be unlocked by appropriate mechanisms and protocols.

This work does not appear out of the void; in place already are the DFKI/ACL tool descriptions [2] and the IMDI [3] and OLAC [4] language resource description environments.

The purpose of this paper is to show the advantages that an integrated approach would offer and the required mechanisms and protocols for this that are needed.

2. Required Functionality

From the view point of the typical user a desirable scenario would be:

1. Locate suitable resources using metadata either by browsing or searching.
2. Find out what kind of suitable tools are available for these particular resources.
3. The user makes a selection of one of these tools. Perhaps after making further inquiries.
4. Start up the selected tool with the resources already specified.

Another less typical scenario would be that of a tool builder who would like to find resources that fit with the particular tool he is testing or developing.

In any case both scenarios require the location of resources using metadata either on the basis of linguistic

requirements or requirements concerning the data-type or format.

Once resources have been identified the next step would be to look what kind of tools are available and see if they would fit the particular type of the resources and needs of the users. That last requirement is something only the user can answer. For the moment we assume he will make a choice from a selection offered by a tool. Preferably this would be the same tool he used to locate the language resources in the first place. Note that the located resources are often not locally available and are identified by an URL. From that point on a number of options and possible complications exist that will need to be addressed by a protocol that regulates the proper exchange of relevant information. Just to give an impression here is a (by no means exhaustive) list:

- a) There can be locally installed tools that might be suitable.
- b) A tools repository might advertise suitable tools.
- c) The tool can be present on the tool repository itself.
- d) The repository might just have a link to its actual (remote) home.
- e) The site where the resource resides does not allow copying of the resource from the site but has itself a suitable tool available. (Statistics)
- f) The tool is not available for download.

3. Current State of Affairs

We will now make a small inventory of the current state of affairs with respect to required technology and start with the available standards that describe (metadata) language resources and linguistic tools. Then we look at possible mechanisms of communications between tool and language resource consumers and providers.

3.1. Language resource descriptions

First we will investigate what the current situation is with respect to language resource storage and description. The for our purposes, that is resource location and access, relevant way of describing language resources is by using metadata that is made available on the Internet.

At the moment there are two initiatives that claim to provide a relevant metadata set for the linguistic domain. First there is the ISLE Metadata Initiative (IMDI) started in 2000 that has tried to develop a metadata vocabulary from the bottom up starting with the requirements of the researcher and finally arrived at a complex set that can be considered the beginning of an ontology for the domain.

IMDI does not claim to describe more than the language resources only, but does offer a complete metadata environment that handles to organise resources in browsable structures and bundle related resources in administrative units. Tools to create manipulate and exploit IMDI tagged resources are also provided. The IMDI project is also relevant because one of the tools developed shows tight functional integration of available linguistic tools and resources. The IMDI-BCBrowser, a tool to navigate the universe of IMDI tagged resources allows a user to map available local linguistic tools on resource types. Whenever a suitable resource comes in focus the applicable tools are available directly from this browser tool. See [5] for more details. This is a local variant of what we would like to achieve with a Language Repository Exchange Protocol (LREP).

The second relevant initiative is the Open Language Archive (OLAC). OLAC has created a specialisation of the well-known Dublin Core (DC) [6] set for use in the linguistic domain. The specialisation involves the introduction of one new element in addition to the existing 15 of DC and the further adjustment of the other 15 by means of specialised qualifiers. OLAC claims not only to describe language resources but also wants to describe linguistic tools and “best practice” recommendations.

Currently there is an experiment transforming IMDI records to OLAC format in order to allow, although with considerable information loss, OLAC users to locate IMDI tagged resources.

3.2. Linguistic tool descriptions

With respect to tool repositories we refer mainly to the DFKI ACL Natural Language Software registry that maintains an exhaustive list of tools and offers it in a human readable form on the web but also allows export to DC records in an XML format so that its data can be harvested by protocols such as suggested by OAI [7]. It uses a well-described taxonomy [8] that for our purposes will need to be further specialised.

The OLAC proposal uses DC elements to describe tools and although they have made the step to machine readable metadata what we require for our purposes, it remains to be seen if their set can describe these tools in sufficient detail.

3.3. Suitable exchange protocols.

Defining an exchange protocol can capitalize on the experience of other earlier initiatives. There is GATE [9,10] that is presented as a software architecture for NLP components. It enables experienced users to easily configure NLP systems. Although this tightly integrated system is a local one i.e. the software modules exchange information locally, it shows that some experience exists about the nature of the information to be exchanged.

Another example is the Browser tool [11] developed within the IMDI project mentioned before. The Browser facilitates an information exchange between the two domains of “Language Resources” and “Linguistic Tools” by examining a configuration file that maps tools to resource types/formats and offering the user a choice of tools for a specific set of resources. This is limited to locally installed tools only although it allows specification of tools that work on remote resources.

Evidently we need to generalise the described exchange mechanisms to more flexible exchange mechanisms that support distributed systems by nature. These requirements seem to be full-filled by emerging standards such as SOAP [12] but could just as well be supported by exchanging specific XML formatted messages via an existing network protocols as HTTP.

4. Functionality to add

The exact nature of the underlying protocol for communication between users, Resources Repositories and Tool Repositories either is not important. More important is to establish the information content that is to be exchanged. Following the typical usage scenario described above, we summarise the information sent by the Browser or Search tool to the Tool Repository in the form of a Tool Request Packet (TRP) as:

- i. Administrative information
- ii. Function specification (what is the tool needed for)
- iii. A resource type selection (The user might only be interested in audio while there are also annotations available)
- iv. Operating environment for the tool (OS, available execution environment such as Java, Perl, available fonts etc.)

This information is received and validated by a tool repository that will try to match the received information with the tools it describes in its database. After processing it will return a response package with the following information:

- Administrative information
- Error info (if needed)
- List of tool descriptions
 - i. Name
 - ii. Owner/Creator
 - iii. General description
 - iv. *Download/install/execute instructions*
 - v. *Local/remote resource attribute*
 - vi. *Rating*
 - vii. *Warnings*
 - viii. *Availability/access rights*

Most of the items are self-explicitatory, but some of them deserve some explication:

- *Download/install/execute*: The tool is presumed to be (1) automatically downloadable in which case there should be a URL; (2) Downloadable after consultation with the owner, this is indicated in the “Availability/Access rights field”; (3) Not downloadable, but may be executed remotely.
- *Local/remote resource attribute* specifies if the tool is capable of accessing remote resources.
- *Ratings*: Indicates how well the tools fit the list of resource types.
- *Warnings*: Indicates what resource types cannot be included when starting the tools.

4.1. Description of resources

The description of the resources needed by a tool repository to determine if a tool is suitable has until now

been indicated by “resource type”. This will obviously need to be more complex than just a single type specification element. The IMDI project has experimented with mime-types following the convention used on the Internet where resources are characterized by a type and sub-type (see Multipurpose Internet Mail Extensions as described in RFC 2046 [13]). In this way an annotation file in CHAT format would be described as “text/x-chat”. This scheme is however not sufficient to express all necessary information as there is in some cases also need to describe different encoding types. IMDI did recognise this and later added special fields as “content-encoding” as the name of the encoding scheme used for annotation purposes and “character-encoding” to specify the character set used. A CHAT annotation file with morpho-syntax information would be described as:

- Type: annotation-unit/morpho-syntax
- Format: text/chat
- Content-encoding: eurotyp
- Character-encoding: ISO-Latin1

As yet it is uncertain if this is a sufficient set for all the tools we need to administrate with LREP, the elements of the IMDI set for resource file typing are not altogether orthogonal and any new extension will only complicate this further.

4.2. Groups of Coupled Resources

Often it will be the case that the user wants to operate on a group of linked resources. Think of the example that a multimedia/multi-modal resource has tightly coupled files for two video cameras, an internal and an external microphone, a track for laryngoscopic data and various annotations in separate files (see Figure 1).

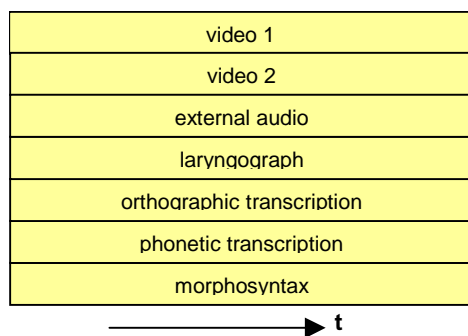


Figure 1 Coupled MM resources

In the IMDI project this situation was understood from the beginning and the term “session” was introduced [14]. A session bundles various related data files and the corresponding metadata description contains links to all of them.

It is the task of the tool to use this information appropriately. When using the resource repository browser the user will select the session and not a particular file. When connecting to the tool repository he could want to operate on all or on a selection of them.

The resource type list field in the Tool Request Package (TRP) specifies if a subset of the included resources was selected, i.e. the user has to have the

possibility to not only specify the function of the operation (in the shown example a wish to start an image processing tool would automatically imply that only the mpg files are relevant), but also the resource type to specify whether for example video or audio should be used¹. Of course, there is a link between applications and resource types. However, it is expected that for example including all resource types of a metadata description as indicated above would result in offering almost all tools that are in the repository. Therefore, the user should be able to define a function - it should be an optional parameter – that indicates an analysis or display function beforehand. This information is present in the TRP. Questions whether for example video1 or video2 or both should be included in an operation have to be answered by the browser who will offer a list of all resources selected. The user should be able to tune his list.

All information is sent to the Tool Repository which will check on which formats the available tools can operate and will calculate a rating and generate warnings. The ratings will specify the degree of match and the warnings will say which types and formats cannot be included when using a certain tool. When a user selects one of the tools only those files that are compatible will be included in the start script. For the moment we assume a framework of separately executable tools that can be called from a script. More advanced schemes such as remotely executable tools are conceivable and need to be provided for.

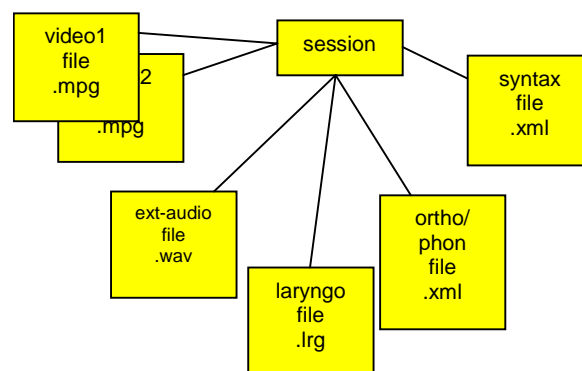


Figure 2. The session bundling resources

4.3. Group of Unrelated Resources

The user may have selected various resources, which for example can be various text files only taken from various subjects, but it also could be that various sessions with mixed file types were selected. Operations like this could be wanted for example to train speech recognizers on a whole set of speech files together with their annotation files or a search on textual data which could be in various formats such as a certain XML format or an older legacy format such as CHAT.

¹ Another option would be to present both filenames to the tool such that the tool will ask the user which one to select. But this will require more intelligence at the side of the tools. It cannot be assumed that builders of already existing tools want to modify their code extensively.

We will discuss the most general case first. In the case of unrelated resources the user can only execute mass operations such as search, statistics, speech recognizer training etc. on them. Currently, there are already tools, which operate on different input formats, which means that a variety of formats is not problematic and the tool repository can match tools for them. In the future we can expect cases where search operations can be executed on texts as well as on sound or video files, i.e. there also a variety of file types does not have to be excluded. The user though, should have the possibility to make selections to decrease the complexity of the task.

It is therefore suggested that the Resource Repository Browser determines a list of file types and formats by reading all metadata descriptions of the selected resources and by interacting with the user.

All other configurations are specializations of this general one and can be handled in the same way.

5. Tool Execution

The usual working environment will need to have the standard tools already present on the local machine and the Resource Browser aware of this. Preferably these local tools are obtained from tool repositories during a previous work session. That way their configuration and start-up information can be in the same format as used for remote tools. Only if the user requires functionality not available from the locally configured tools or enters an explicit command, remote tool repositories will be contacted and queried.

We can distinguish between a number of cases with respect to the place where the tool and resources remain.

Local tool & local resources. The tool may have been downloaded first or have been present at the local site before. Important is that the Resource Browsing Tool has knowledge how to startup the tool and how to specify the resources the tool works on when executing the tool.

Local tool & remote resources. As above the Browse Tool knows how to start the tool. The tool itself must be able to handle remote resources (specified as an URL). Since not many tools are able to deal with remote resources the Browser can implement a mechanism to automatically download the resources to the local machine. This reduces to case 1. In case of multi-media resources this can be a slow affair.

Remote tool & remote resources at same site. Some sites offer resources as well as the tools that work on them. These sites have already integrated their resources localisation service with tools that work on these resources. It should be considered for each such a site what added value could be generated by opening up their resources to be used with other tools. As an example of an integrated service look at LACITO [15].

Remote Tool (not downloadable) & resources at different site. The only way to exploit such a tool would be if the tool in question would be remote executable and would work on remote resources or the site would allow the download of resources. This is certainly an interesting possibility when a site has a great quantity of statistic and search engines that depend on optimised machine configurations.

6. Future Developments

The development of the LREP protocol and working environment are part of the INTERA project that seems to start later this year.

7. References

- [1] Thierry Declerck, Steven Krauwer and Mike Rosner (chairs): Proceedings of the ACL/EACL Workshop *Sharing tools and Resources*, Toulouse, 2001.
- [2] The ACL Natural Language Software Registry <http://registry.dfki.de>
- [3] IMDI, ISLE Metadata Initiative. <http://www.mpi.nl/ISLE>
- [4] OLAC, Open Language Archive Community <http://www.language-archives.org>
- [5] The IMDI Metadata Set, it's Tools and accessible Linguistic Databases. Daan Broeder, Freddy Offenga, Don Willems, Peter Wittenburg. Proceedings of the IRCS Workshop on Linguistic Databases, Philadelphia 2001.
- [6] DC or DCMES Dublin Core Metadata Element Set <http://dublincore.org/documents/dces/>
- [7] OAI, Open Archives Initiative <http://www.openarchives.org/>
- [8] Survey of the State of the Art in the Human Language Technology edited By G.B. Varile and A. Zampolli, *Linguistica Coputazionale*, volume XII – XIII.
- [9] Gate – General Architecture for Text Engineering: <http://gate.ac.uk>
- [10] Hamish Cunningham, Kevin Humphreys, Robert Gaizauskas and Yorick Wilks: Software Infrastructure for Natural Language Processing, in *Proceedings of the 5th conference on Applied Natural Language Processing*, Washington D.C., 1997.
- [11] Daan Broeder and Peter Wittenburg: Interaction of Tools and Metadata-Descriptions for Multimedia Language Resources, in Proceedings of the ACL/EACL Workshop *Sharing tools and Resources*, Toulouse, 2001.
- [12] SOAP – A Simple Object Access Protocol: <http://www.w3.org/TR/SOAP/>
- [13] [RFC2046] Multipurpose Internet Mail Extensions (MIME) Part Two: Media Types <http://www.ietf.org/rfc/rfc2046.txt>
- [14] Broeder, D.G., Brugman, H., Russel, A., and Wittenburg, P., (2000), A browsable Corpus: accessing linguistic resources the easy way. In Proceedings LREC 2000 Workshop Athens.
- [15] Linguistic Data Archiving Project (LACITO), <http://lacito.archivage.vjf.cnrs.fr/>