

# Linguistic Data Processing for Everyman

Jean Carletta, Amy Isard, and David McKelvie

Language Technology Group and Human Communication  
Research Centre, University of Edinburgh, 2 Buccleuch Place,  
Edinburgh EH8 9LW, Scotland

{J.Carletta,Amy.Isard,David.McKelvie}@edinburgh.ac.uk

Paper presented at the workshop on Web-Based Language Documentation and  
Description, 12-15 December 2000, Philadelphia, USA.

***Abstract.** Current XML technologies can provide a powerful underpinning to any new linguistic data repository. We can minimize cost to the community by capitalizing on external tools intended for XML users in general. We propose that since most XML use privileges element hierarchies by making hierarchical structures easy and fast to navigate, element hierarchies should be used to represent the most important relations in an XML data set. This is the approach taken, for instance, in the TEI. AIF violates this constraint by insisting upon one standardized DTD for all data resources, no matter what their content or natural structure. We support our argument by exploring our understanding of the reasons behind AIF and showing how they are as well met by our counter-proposal. We further describe where our communal interests as linguistic data users are unique and therefore where resources might best be concentrated.*

## 1 Introduction

Edinburgh's Language Technology Group has long been in the business of tool support for working with corpus data (<http://www.ltg.ed.ac.uk>), with the Human Communication Research Centre driving some tool development for their research using spoken dialogue corpora. Most recently, we have completed an analysis of a corpus of 325 telephone dialogues between an operator and a public caller. This work was done for British Telecom and required us to use a wide range of XML tools for both hand and automatic data annotation, as well as data transformation and extraction technologies.<sup>1</sup> Although this work is on data of quite a different kind from that of concern to this group (that is, data sets which document different languages), our experiences on this project have given us insights into data creation, storage, and re-use which may be relevant to others with a common interest in linguistic data resources.

---

<sup>1</sup> Reports describing the analysis and complete instructions for its replication, as well as the XML format annotated corpus, are available from BT Adastral Park to sites which can satisfy certain data protection and ethical constraints on use of the data. It is possible that the data would be of interest to this community because it is, to our knowledge, the best-documented publicly-available data resource to use quite the complexity of hyperlinked XML documents which we advocate in this paper. To obtain copies, or to point out the better exemplars of this style which we think must exist, contact Jean Carletta in the first instance.

The call for papers asks authors to directly address the workshop concerns rather than describe project work. One of our difficulties is that the exact goals of setting up a linguistic data repository (in terms of who the potential users are and what they might wish to do with the data) are not always made explicit, even though the proposals arising from those goals are. Our arguments in this paper are based on the understanding, gleaned from AIF, that even if the actual repository storage format may still be open to revision, it has been proposed that all data should conform to the same XML document type definition (whether that is specified by means of a DTD or a schema, or a set of interlinked DTDs or schemas), XML having been chosen because it provides the most universally accepted way of defining such a format. In this paper, we explore the motivations for and implications of this proposal, contrasting them with one obvious counter-proposal.

Our argument is motivated by one principle we think crucial in any repository design. It is that a repository should place as little additional burden on the data depositor and data user as possible. The better a repository adheres to this principle, the more successful it will be in terms of encouraging data re-use.

## **2 Why a uniform document type definition?**

We find the premise that all data should conform to the same, single XML document type definition somewhat puzzling because, given the nature of linguistic research, it places a large additional burden on the data depositor, with a benefit for the data user which is still, to our minds, somewhat uncertain. Specific data sets always have their quirks, depending on the intentions and level of resource available to their collectors. Even if the goal of the data set is to document a particular language "completely", the data set will still have its own unique and natural structure, depending on the properties which the data creator sees as unique and natural to that language. Whether or not one agrees with the data creator, forcing data sets to fit a standardized DTD or schema obscures the way the data is really organized, making the data set inherently more difficult to understand.

Presumably it is felt that the benefits to data re-users outweigh these difficulties. One possible benefit to the community, which we have heard aired in communities dealing with other types of corpus data, is that it would allow for comparison among different data sets. This is patently not the case for this community, at least with the AIF formalism as it currently stands. Note that because AIF uses element names such as "corpus" and "annotation", with freedom for how to name annotations and how to structure the relationships among different annotations on the same corpus, it provides no guidance about which parts of different data sets are comparable. This may be just as well, since such comparisons are often spurious; different data sets quite often mean different things by "word" or "lexical entry". Without an understanding of the data sets in the repository as a whole, which individual depositors might well not have, facilitating meaningful comparisons is difficult.

Another possibility could be that the uniform XML structure is intended to facilitate the production of meta-data about what kinds of annotations the data set includes. It is not

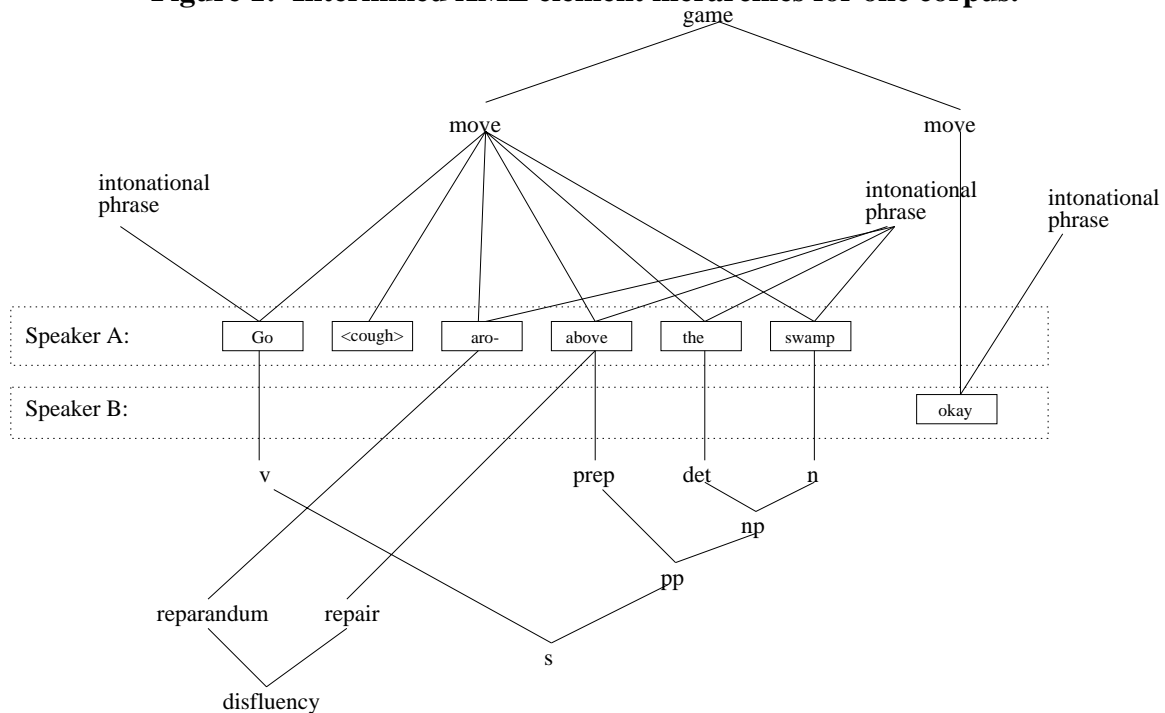
entirely clear yet what scope of meta-data is envisioned here. If this is the motivation for the uniform document type definition, than one possibility is that the amount of effort required to create and/or work with that format outweighs the effort savings of being to obtain this information automatically from the corpus. At any rate, if this is the reasoning, then the merit is in making the meta-data extraction fairly simple and not necessarily in having a uniform XML structure itself.

The reasoning may be that if there will be new tools which are unique in facilitating research arising from these data sets, then of course it is better to be able to use these tools. However, when one considers the nature of linguistic research, where every user asks a unique and unpredictable question of the data, it is hard to predict what the processing requirements of the community will be. One thing is certain — there will always be aspects of the deposited data, and of the questions which users ask, which are not served by either the standardized format or the specialist processing mechanisms. Thus if the repository is to support research fully, it must facilitate the answering of other kinds of questions outside the main processing model as well. Otherwise we risk always looking for our lost coins under lamp posts. The easiest way to do this is by making sure that the repository, whatever else it does, facilitates the kind of processing which XML users in the greater world perform on their data. This covers a wide and growing range of processing models and requires no extra implementation effort on our part.

### **3 A counter-proposal**

AIF is novel in proposing uniform tag names over disparate data sets. Instead, developments such as the TEI have encouraged document structures which match the inherent structure of the data. Similar data sets might use the same elements; where data creators feel existing element definitions are insufficient for representing their data, they are encouraged to create new ones which make sense for them. For instance, consider Figure 1, which gives a graphical representation of the XML structure of one of our corpora. In the structure, transcribed words are linked to speech signal, which, for simplicity, is not shown. Different trees of annotation interlink, in this case, at the level of the transcribed word. If this structure seems overly complicated in involving so many different annotations of the same source, remember that one thing data re-use is likely to encourage is new annotations on old sets. That is, one of the points of a data repository is to encourage structure overlaying in precisely this manner.

**Figure 1: Interlinked XML element hierarchies for one corpus.**



This sort of document structure certainly makes it easier to work with generic XML processing techniques, such as XSLT. The core of any XML processing is the method for writing queries which find matches within the XML document. These queries lie at the heart of mechanisms for transducing data into other forms, displaying it (which is a special form of transduction into, say, HTML), extracting information, and counting phenomena so that they can be analysed statistically. XML query languages tend to make it easy to specify paths from the root of the document down the tree-structure, so that, for instance, the query for "find me all substitution disfluencies containing the word 'above'" might be something like this:

```
./disfluency[@type='substitution']/./word/#above
```

The construction of an XML query for any AIF representation of the same basic structure is left as an exercise for the reader, partly because AIF is a moving target. Such a query would doubtless be complicated, since it would require one to trace through pointers for each of the different annotation types; XML query languages tend to make navigation through element hierarchies easy but other kinds of navigation more difficult. It would also be very slow in any straightforward implementation; because implementors assume element hierarchies are the paramount concern, optimization concentrates on them. Of course, the XML queries become more complicated when they require one to thread through the element hierarchies in the various interlinked XML documents. They remain an order of magnitude less complicated if one makes element hierarchies mirror the structure of the data than if one assumes an AIF-like format. Although our example is from a different domain from what the repository would contain, the problems are almost certainly the same.

## 4 Tool support

Processing requirements for linguistic data are unpredictable. Therefore it is likely that at times, data users will need to resort to generic XML processing techniques. If an AIF-like data storage format were adopted, the community might well need specialist tools for constructing XML queries which get from some natural idea of the structure of the data set to the hierarchy of AIF elements used to represent it. On the other hand, the more our concerns match those of the general XML community by making element hierarchies paramount, the more we can expect general improvements in XML support to help linguistic data users. Tools for authoring DTDs and/or schemas are irrelevant for AIF users, but essential when document type definitions match the data's natural structure. We can expect more graphically-oriented, easier to use authoring tools to arise as general XML demand increases. Tools for writing queries and the XSL stylesheets which form the most common generic processing mechanism are bound to improve quickly without implementation effort of our own. To some extent, so are facilities for links to signal, since these are driven by multimedia web applications — although one thing a data repository would need to do, no matter the format, is standardize how access to signal is specified. If there is anything which is particular to linguistic data, and therefore requires our special attention, it is the degree to which we rely on pointers, either to interlink hierarchies in the naturalistic data representation, or to link everything in AIF. Designing such linking structures can be tricky when authoring DTDs directly.

One possible answer is a specialist authoring tool which allows one to draw a picture similar to that in Figure 1, and creates the interlinked set of DTDs for the user. At the same time, the tool could encourage the addition of comments about, for instance, the meaning of an element name, or further restrictions on the content model in practice which cannot be formally expressed, which are clearly useful but do not necessarily fit into standard DTD writing practice. This picture, with the links to the additional information, then of course becomes an important resource for data users.<sup>2</sup> The picture might also form the basis for an appropriate graphical query language interface, although one which allows for all of the queries that might be asked will always be rather complicated.

---

<sup>2</sup> Note the similarity to the potential meta-data argument in support of AIF. However, is not just easier to read the picture than its AIF equivalent. This picture theoretically defines the structure of the corpus in a way that is missing in AIF; in AIF it is not possible to know whether a particular arrangement of pointers is missing because no examples occur in the corpus, or because it is theoretically malformed. Thus data sets in AIF will require further information which a natural structuring gives for free. Conversely, in AIF it is not possible to know whether a particular arrangement of pointers is present because an example occurs incorrectly. This means that XML parsing can not be used to find "errors" in the traditional way.

## 5 Discussion

Although we have phrased our suggestions as a counter-proposal in order to highlight the differences in approach, recent developments in XML mean that probably the best approach would draw from both of them. XML Schemas allow elements to inherit properties from each other within hierarchical classes, in much the same way as objects inherit properties and methods within object-oriented programming methodologies. This means, for instance, that it is possible to make XML structure mirror data structure whilst still marking elements with the categories inherent in the AIF element names. If this kind of inheritance is used, it may also be possible to translate into an AIF-like format from a more natural representation. Translation in the other direction is more difficult because it relies on additional information about what the primary structures are. Given the demonstrated utility of making XML element hierarchies mirror the natural structure of a data set, and the uncertain benefits but certain costs arising from any AIF-like representation it may be more appropriate to consider the former representation as primary within a data repository. If AIF has advantages in terms of common processing needs not supported in the wider XML world, then perhaps it is best seen as a useful derivative format, and not the other way around.