Annotation schemes, annotation tools and the question of interoperability: from Typed Feature Structures to XML Schemas

Philippe Blache, Brigitte Bigi, Laurent Prevot, Stéphane Rauzy, Julien Seinturier

To cite this version:
Philippe Blache, Brigitte Bigi, Laurent Prevot, Stéphane Rauzy, Julien Seinturier. Annotation schemes, annotation tools and the question of interoperability: from Typed Feature Structures to XML Schemas. Second International Conference on Global Interoperability for Language Resource, 2010, Hong Kong, China. hal-01393612

HAL Id: hal-01393612
https://hal.science/hal-01393612
Submitted on 12 Dec 2016
Annotation schemes, annotation tools and the question of interoperability: from Typed Feature Structures to XML Schemas

Philippe Blache¹, Brigitte Bigi¹, Laurent Prévot¹, Stéphane Rauzy¹, Julien Seinturier²

(1) LPL, CNRS-Université de Provence  (2) LSIS, CNRS-Université Méditerranée
5, Avenue Pasteur  13288 Marseille Cedex 09
13604 Aix en Provence - France  julien.seinturier@esil.univmed.fr
firstname.name@lpl-aix.fr

1 Introduction

Many annotation schemes have been proposed, answering the needs of specific projects: facial expressions (FACS), gestures (MUMIN), dialogue acts (DAMSL), etc. A first problem comes from the fact that these schemes often have a close relation with concrete encoding: in many cases, the annotation scheme is presented as a tagset (see for example (ISLE, 2002)). The scheme is even in some cases specified in order to fit with the restrictions of annotation tools. For example, encoding recursive structures is not direct due to the fact that most of the tools encode information in terms of layers or tracks. As a consequence, annotation schemes are often considered as annotation guides, and vice-versa (see for example (Dipper, 2007)).

This aspect can be a problem: each specific domain or project lead to a specific annotation scheme according to the granularity of the information, the way it is encoded, etc. This scheme multiplication renders interoperability problematic. On the other hand, it is not realistic to think possible the specification of a unique generic scheme. Our proposal goes in the direction of a higher level of generality when specifying a scheme, independently from the constraints of the coding procedure.

2 Representing information with TFS

Linguistic knowledge is represented by means of three different types of information:
• **Properties**: the set of characteristics of an object

• **Relations**: the set of relation an object has with other objects

• **Constituents**: complex objects are made with other objects, their constituents

This information can be represented by means of typed feature structures (see for example (Carpenter, 1992)): properties are encoded by features, constituency is implemented with complex features, and relations make use feature structure indexing.

We present in the following the main characteristics of a TFS representation of multidomain linguistic annotation. Information representation given here comes from the OTIM project (Blache, 2009) aiming at developing schemes and tools for multimodal annotation.

### 2.1 The object type

The object type is the most general one. All elements of the linguistic information is an object subtype. In terms of location, an object can be situated by means of two different kinds of position, depending on the fact they correspond to an interval (for example a syllable), or a point (e.g. a tone). In the first case, interval boundaries are represented by the features `START` and `END`, with temporal value (which value being label or milliseconds, depending on implicit or explicit time encoding). The following type hierarchy presents the location type and its two subtypes (`interval` and `point`), together with their appropriated features.

### 2.2 Phonetics

The phoneme is used as primary data in most of the cases: this object is at the lowest level of the constituent hierarchy: most of the objects are set of phonemes. The following FS proposes a representation of the main phoneme properties:

### 2.3 Prosody

As seen above, prosodic phrases are of two different subtypes: `ap` (accentual phrases) and `ip` (intonational phrases). The prosodic type hierarchy is represented as follows:

Accentual phrases have two appropriate features: the label, which value is simply the name of the corresponding type, and the list of constituents, in this case a list of syllables. The objects of type `ip` contain the list of its constituents (a set of `aps`) as well as the description of its contour. A contour is a prosodic event, situated at the end of the `ip` and is usually associated to an `ap`.

The prosodic phrases are defined as set of syllables. They are described by several appropriate features: the syllable structure (for example CVC, CCVC, etc.), the position of the syllable in the word, their possibility to be accented or prominent:
Syllable constituents, objects of type `const_syl`, are described by two different features: the set of phonemes (syllable constituents), and the type of the constituent (onset, nucleus and coda). Note that each syllable constituent can contain a set of phonemes.

\[
\text{const_syl} = \left[ \text{PHON list(phon)} \right]_{\text{const_type}} \{ \text{onset, nucleus, coda} \}
\]

3 From TFS to XML via a graphical representation

The TEI consortium proposes guidelines for implementing feature structures in XML (see TEI Guidelines P5). However, we decided not to encode TFS this way for several reasons. First, type hierarchies and inheritance are not easily and directly represented. Second, it is not realistic, or even possible, to encode all information by means of a feature structure, that rapidly becomes huge and intractable. This last argument is important in the perspective of interoperability: annotation tools mainly focus on properties annotation (the encoding of object characteristics). Only some of them propose solution for constituency representation (e.g. in terms of primary/secondary tracks). None implement typing machinery. We think then preferable a decentralized representation in which objects are represented separately, their hierarchization being encoded independently from their properties. Such an encoding offers the advantage to be close to the traditional way of encoding annotations without losing the richness of TFS representation. Our proposal, presented in this section, relies on an object-oriented representation, capable of generating automatically the XML schema of the annotation scheme.

3.1 Motivations

As seen above, TFS proposes to represent a precise level of annotation for the different domains and modalities (phonetics and phonology, prosody, morphology, syntax, discourse, gesture, etc). It is well suited to take into account the heterogeneous characteristics of annotated data. Each domain is represented as a hierarchical model and the global view comes from the various hierarchical organizations proposed. Unfortunately, representing with TFS the entire multimodal annotation schema is difficult, and the global interdependencies are hard to visualize. We propose then to complete this representation by means of a graphical view.

Such a graphical representation makes the reading less time consuming and easily understandable. Its disadvantage lies in its lack of details: for example, TFS describes the object’s types (boolean, string, integer, etc.) which are omitted in the graphics. Concretely, we chose as modeling language UML (Unified Modeling Language). The idea consists in translating automatically the representation into a XML schema following specifications proposed by TFS and UML model. The XML implementation and, more specifically, the XML schema, is then a specific rendering of a very detailed model.

3.2 UML representation

UML relies on two different views. The static one describes the static structure in terms of objects, attributes, operations and relationships. This view includes the class diagram. The dynamic view emphasizes the dynamic behavior of a system. One of its advantages is that the language includes a set of graphical notation techniques to create and share intuitive visual models.

We can represent a TFS description by a set of UML class diagrams by means of the following mapping:

- to each complex TFS corresponds a class;
- to each TFS atomic attribute corresponds a class attribute;
- the inheritance relationship defined on the TFS is represented by an inheritance relationship on the classes;
- the constituency relationships between TFS are represented by aggregation relationships on classes.

Figures 1 shows the UML representation of phonetics and prosody domains (for clarity, the articulation gesture feature has not been developed). This graphical representation provides a global view of the two domains and suitable way for experts to share their knowledge. Some classes are represented in the two figures, their background is white (others are gray).

3.3 From UML to XML

In our approach, the process generates an XML schema for each domain such as phonetics and
prosody. Data are then represented in different document structures, which is motivated by two important points: (i) this representation provides a high level of modularity for applicative requirements and enables to modify only the structure / schema needed; (ii) recent works on multi-structured documents open a way for dynamically linking and aggregating various documents with different structure. Within such a framework, it is possible to deal with a TFS-like XML representation and to process data as standard XML documents.

Our XML schema, generated from TFS via UML, can be seen as the "third component" mentioned as a perspective in (Schmidt, 2009). This component, besides a basic encoding of data following AIF, encode all information concerning the organization as well as the constraints on the structures. In the same way as TFS are used as a tree description language in theories such as HPSG, the XML schema generated from our TFS representation also plays the same role with respect to the XML annotation data file. On the one hand, basic data are encoded with AIF, on the other hand, the XML schema encode all higher level information. Both components (basic data + structural constraints) guarantee against information lost that otherwise occurs when translating from one coding format to another (for example from Anvil to Praat).

4 Conclusion

We propose in this paper the use of a general description level, representing the annotation scheme by means of typed feature structures. Such a representation is independent from coding languages and tools. It guarantees the definition of a precise semantics in which type and constituent hierarchies are clearly distinguished. Moreover, such a representation makes it possible to propose a strict conversion towards an XML schema: TFS are directly coded into UML structures which are automatically translated into an XML schema.

This schema is then used as a description language over the coding scheme itself: it contains all information relative to the organization, the hierarchies as well as feature values restrictions that the annotated data have to follow. As a result, this approach, on top of providing a semantics to the schemes, completes the concrete encoding: it makes it possible to describe data at a low level (for example using AIF) and complete the description in giving the entire structuration following the TFS definitions.

References


Ide N. and K. Suderman 2007 “GrAF: A Graph-based Format for Linguistic Annotations” in proceedings of the Linguistic Annotation Workshop (LAW-07)


Wegener Knudsen M.and al. 2002 Survey of Multimodal Coding Schemes and Best Practice, ISLE