# Managing Metadata Variability within a Hierarchy of Annotation Schemas

Ionuţ Cristian Pistol[1] and Dan Cristea[1,2]

[1] Faculty of Computer Science, University “Al. I. Cuza” of Iaşi, Romania

[2] Institute for Computer Science, Romanian Academy, Iaşi, Romania

**Abstract.** The paper describes the theoretical basis of the ALPE[1] model, a hierarchy of annotation formats used to guide the automatic computation of processing flows capable of performing complex linguistic processing tasks. The hierarchy is comprised of a core, which is a direct acyclic graph whose nodes represent XML annotation formats, and a halo which contains additional annotation formats. The core hierarchy also serves as a standardization hub for annotated documents. The focus of the paper is the description of the new additions to the model, allowing the integration and usage of non-XML formats in processing flows and new equivalence relations between XML formats.

## 1 Introduction

In the latter years, the field of Natural Language Processing witnessed the emergence of a significant effort concerning the standardization and usability aspects of developed processing tools and resources. Projects such as CLARIN[2] and FLaReNet[3], among others, intend to offer both developers and users of language resources and tools a management solution for the growing set of resources available. The primary objectives of these projects are to provide reusability in new contexts for existing resources and to guarantee maximum visibility and reusability for newly developed resources. An easy widening of the original setting of usage means a multiplication of the visibility of a tool and, finally, of the productivity of the research activity. In terms of managing linguistic processing tools, previous efforts lead to the development of linguistic processing meta-systems, the most significant ones being GATE[4] [3] and UIMA[5] [4].

ALPE [1, 2] is another such system, intended to define a framework which facilitates the integration of processing tools of different origins. ALPE offers several advantages over existent systems with a similar goal, as it is able to identify the annotated format of the input file, then to automatically compute the processing steps re-

[1] Automated Linguistic Processing Environment
[2] http://www.clarin.eu/
[3] http://www.flarenet.eu/
[4] http://www.gate.ac.uk/
[5] www.research.ibm.com/UIMA/

quired to bring an input file to the required output format, and, eventually to run this chain if costs/IPR conditions are fulfilled.

Section two of this paper briefly describes the base ALPE hierarchy and section three describes the enhancement of the base hierarchy with processing power and "clouds" of equivalent formats. The conclusions, as well as the further planned developments are described in section four.

## 2 The Base Hierarchy

### 2.1 The Core Hierarchy

The basis of the core hierarchy is a directed acyclic graph which configures the metadata of linguistic annotation in a hierarchy of XML schemas. Nodes in this graph are called core nodes. Each core node corresponds to a single XML annotation format.

We note as T(A) , where A is a core node, the set of elements (tags) defined in the XML annotation format corresponding to the core node A. We note as $t_a(A)$, where A is a core node and $t_a \in T(A)$ is the set of attributes belonging to the element t as it appears in the core node A. Edges connecting core nodes are called core edges. If there is a core edge linking a core node A with a core node B (we will say also that A is formally subsuming B, noted as AsB) then the following conditions holds simultaneously:

- any element (tag-name) of A is also in B: $T(A) \subseteq T(B)$;
- any attribute in the list of attributes of a tag-name in A is also in the list of attributes of the same tag-name of B: $ta(A) \subseteq ta(B)$ for all $t \in T(A)$.

The direction of the core edge connecting nodes A and B is given by the subsuming relation, with the subsuming node being the origin of the core edge and the subsumed node the destination.

### 2.2 The Haloed Hierarchy

In addition to the core nodes and edges, which strictly observe the specified restrictions, we can include in an extension to the core hierarchy, called a halo, other types of nodes and edges such as:

- Nodes representing other annotation format than XML. We can consider each node in the core hierarchy representing not just a specific format, but rather a class of annotation formats, whose representative is an XML format. These formats can be represented in the hierarchy as halo nodes;
- Edges originating or ending in halo nodes. These edges can either originate or end in the core hierarchy, or they can be completely outside the core hierarchy. The semantic value of these edges is to mark the semantic subsumption between the source and destination nodes, relation considered at an abstract level as opposed to the formalized subsumption relation. Semantic subsumption means that the information encoded in the origin node's format is part of the information encoded in the destination node's format, but this inclusion cannot be strictly formalized using XML elements and attributes.

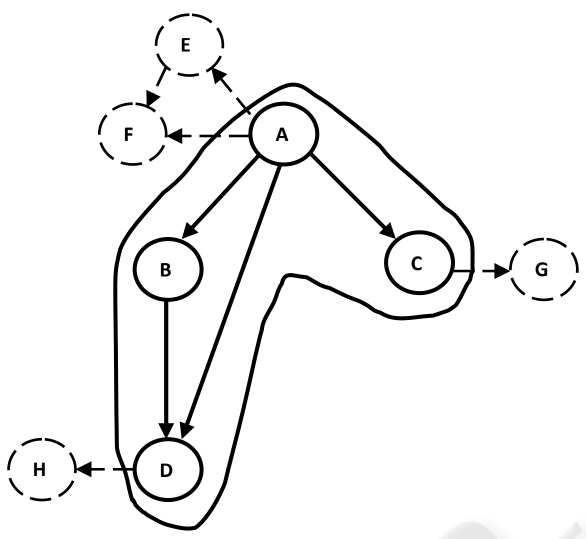

**Fig. 1.** A full ALPE hierarchy (core and halo).

The core hierarchy and the halo form a full ALPE hierarchy. The halo nodes and edges expand the core hierarchy outside the limits of XML, allowing other types of annotation formats to be represented in the full hierarchy. In Figure 1 is shown an example of a full hierarchy. With continuous lines are marked core nodes and edges (and the core hierarchy), and with interrupted lines halo nodes and edges. Nodes A, B, C and D are core nodes and nodes E, F, G and H are halo nodes.

As for the core edges, the following hold true:

**Axiom**: There exists at most one edge between two halo nodes.

**Axiom**: There exists at most one edge between any two nodes in the full hierarchy.

These axioms say that between any two nodes in the full hierarchy there is at most one subsumption relation, either formal or semantic.

In all core hierarchies we introduce an obligatory root core node. This node corresponds to the basic XML format, with only a root element. The definition of this node leads to the following theorem:

**Theorem**: The root core node of a hierarchy subsumes all nodes in the hierarchy.

**Theorem**: The core hierarchy is a connected graph (disregarding the edges orientation).

In order to guarantee the connectivity of the full hierarchy, we introduce the following axiom:

**Axiom**: From each halo node there is at least one core node which can be reached (disregarding the orientation of the edges).

This axiom basically says that no halo nodes are disconnected from the core hierarchy: in order for an annotation format to be included as a node in the hierarchy it has to have at least one other node already in the hierarchy which it subsumes (or is subsumed by) - either formal subsumption (introduced in 2.1) or semantic subsumption. The previous theorem and this axiom lead to the next theorem:

**Theorem**: The full hierarchy is a connected graph (disregarding orientation of the edges).

The proof is direct: the core nodes are connected (as proven by the previous theorem) and each halo node is connected to at least one core node. This means that from each halo node all core codes can be reached. Since all halo nodes can be reached from at least one core node, as a corollary to the previous axiom, this means that from each core node all other nodes can be reached. Also, the axiom and the following conclusion lead to the fact that from each halo node all other nodes can be reached. Thus, the full hierarchy is a connected graph.

## 3 The Hierarchy Augmented with Processing Power

### 3.1 Adding Processing Power to the Hierarchy

If there is an edge (either core or halo) between nodes A and B, there should be a process which takes as input a file observing the restrictions imposed by the node A and produces as output a file observing the restrictions imposed by the node B. While doing this type of processing the module might make use also of some additional resources outside the hierarchy, such as language models and lexicons. A graph of annotation schemas on which processing modules have been marked on edges is called augmented with processing power (or simply, *augmented*).

An edge to which there is at least one processing module attached will be called a processing edge. A single edge in the graph can have multiple processing modules attached to it, if those modules observe the same restrictions regarding their input and output formats. If there is no known processing module attached to an edge, that edge is called a *carrier edge*. For a more detailed and commented description, please consult [1, 2].

### 3.2 Conversion Edges and Synclouds

An edge connected a halo node with another node of the full hierarchy is called a *conversion edge*. All conversion edges have at least one attached processing module. That module is basically a wrapper capable of converting a format into another. Part of the encoded information in the source node is rewritten in another format in the destination node. No new information is added. If one of the connecting nodes is in the halo, the module attached to the conversion edge rewrites either XML into a non-XML format or the other way around, depending on the direction of the edge.

All nodes that can be reached from the same node using only conversion edges (disregarding the direction of the edges) form a **syncloud** (synonymy clouds). All nodes in the same syncloud contain the same information, but encoded differently.

## 3.3 Flows and Synclouds

The main benefit of the introduction of synclouds in the ALPE hierarchy model is the accommodation of various available processing modules using non-XML documents as either input or output. In previous papers [1,2] we introduce processing flows as combinations of basic operations on an ALPE augmented hierarchy. These flows are computed with regards to specified input and output nodes and generate actual processing sequences (workflows) capable of transforming an input document corresponding to the input node to the output format by applying individual processing modules attached to edges in the hierarchy.

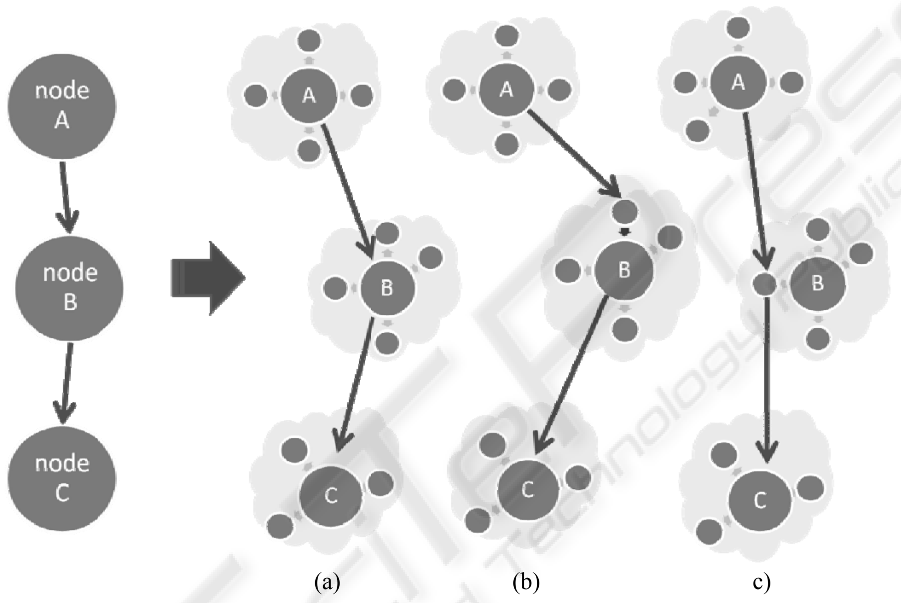

**Fig. 2.** Flows in synclouds.

A simple computed flow, on the left of figure 2, shows only the general picture of an ALPE hierarchy, where a node identifies a distinct annotation format. Figure 2a depicts the case of the processing modules considered on the left being actual modules between the core nodes of the synclouds. Figure 2b shows that a flow can include processing edges between XML and non-XML formats in a syncloud. These modules can be pipelined with XML processing modules using wrappers attached to edges between the core nodes and other nodes in the syncloud. Also, as in 2c, flows can actually exist outside core nodes and include only processing modules using non-XML intermediate formats, allowing the straightforward integration in ALPE of flows produced by other systems, such as GATE or UIMA, with the only change being the addition of the input/output wrappers.

## 4 Conclusions and Further Work

Adopting ALPE as a management and access environment for the resources employed and developed in a computational linguist project proposing the development of multilingual resources and tools, such as CLARIN [6], has the potential of benefiting both the project and the interested user. One important further development of ALPE will be a web-service allowing users to build, configure and use ALPE hierarchies on the web, either as a limited password-protected resource or a global linguistic resources collection. Since UIMA is the prominent system comparable to ALPE and since both GATE and UIMA are now open-source, we also study the possibility of integrating ALPE in either system.

Standards usually appear late. In order for an annotation convention to become a standard it should be adopted by a community of people. Therefore, there is a strong need for accepting new formats, which should work together with well accepted ones. We need a mechanism able to "understand" the notations, to detect the semantics beyond the notations, to infer the meaning of notations and to establish semantic links between new formats before standards appear. The first step would be the clear definition of the semantics of a standard. A promising new model of describing annotation semantics, the Linguistic Annotation Format [5], has the potential of clearly defining semantic links between various annotation formats. We are currently in the process of integrating a version of this model as a way to formally describe nodes in the ALPE core hierarchy and as a possible base for an automatic detection of semantic links between formats.

## References

1. Cristea D., Forăscu C., Pistol I. : Requirements-Driven Automatic Configuration of Natural Language Applications. In Bernadette Sharp (Ed.): Proceedings of the 3rd International Workshop on Natural Language Understanding and Cognitive Science - NLUCS 2006, in conjunction with ICEIS 2006, Cyprus, Paphos, May 2006. INSTICC Press, Portugal. 92006) ISBN: 972-8865-50-3. (2006).
2. Cristea, D., Pistol, I. : Managing Language Resources and Tools Using a Hierarchy of Annotation Schemas. Proceedings of the Workshop on Sustainability of Language Resources, LREC-2008, Marakesh. (2008).
3. Cunningham H., Maynard D., Bontcheva K., Tablan V. : GATE: A framework and graphical development environment for robust NLP tools and applications. In Proceedings of the 40th Anniversary Meeting of the ACL (ACL'02). Philadelphia, US. (2002).
4. Ferrucci D. and Lally A. : UIMA: an architectural approach to unstructured information processing in the corporate research environment, Natural Language Engineering 10, No. 3-4, 327-348. (2004).
5. Romary L., Ide N. : International Standard for a Linguistic Annotation Framework, Natural Language Engineering 10, 3-4 (09/2004) 211-225 (2007).
6. Váradi T., Krauwer S., Wittenburg P., Wynne M. and Koskenniemi K. : CLARIN: Common Language Resources and Technology Infrastructure, Proceedings of LREC-2008, Marakesh (2008).

[6] http://www.mpi.nl/clarin/