Middleware for Creating and Combining Multi-dimensional NLP Markup

Ulrich Schäfer
German Research Center for Artificial Intelligence (DFKI)
Language Technology Lab
Stuhlsatzenhausweg 3, D-66123 Saarbrücken, Germany
email: ulrich.schaefer@dfki.de

Abstract

We present the Heart of Gold middleware by demonstrating three XML-based integration scenarios where multi-dimensional markup produced online by multilingual natural language processing (NLP) components is combined to deliver rich, robust linguistic markup for use in NLP-based applications like information extraction, question answering and semantic web. The scenarios include (1) robust deep-shallow integration, (2) shallow processing cascades, and (3) treebank storage of multi-dimensionally annotated texts.

1 Introduction and Motivation

Heart of Gold is a middleware architecture for creating and combining markup produced by multiple natural language processing components in multilingual environments. It was initially developed for a special sort of multi-dimensional annotation, namely application-oriented, XML- and XSLT-based online integration of various shallow NLP components with a deep HPSG parser for increased robustness in the *hybrid* natural language processing paradigm (Callmeier et al., 2004).

The middleware, however, can also be used for various other online and offline tasks related to multi-dimensional markup creation and integration. These comprise automatic corpus annotation, incorporation of multi-dimensional markup into a single XML representation, and NLP component cascades interleaved with XSL annotation transformation. The middleware provides XML-RPC interfaces for simple, networking-enabled and programming language-independent application and component integration. Heart of Gold is available as one of the DELPH-IN open source tools available from http://www.delph-in.net[1].

2 Middleware Architecture

Fig. 1 gives a schematic overview of the middleware server in between applications (above) and external NLP components (below). When a new application session in Heart of Gold is started, it takes a configuration specifying NLP components to start for the session. Each component is started according to its own parameterized configuration. The client can send texts to the middleware and the NLP components are then queried in a numerically defined processing order ('Depth' in Fig. 4). The shallowest components (e.g. tokenizer) are assigned a low number and are started first etc. The output of each component must be XML markup. Each component gets the output of the previous component as input by default, but can also request (via configuration) other annotations as input. Components may produce multiple output annotations (e.g. in different formats). Thus, the

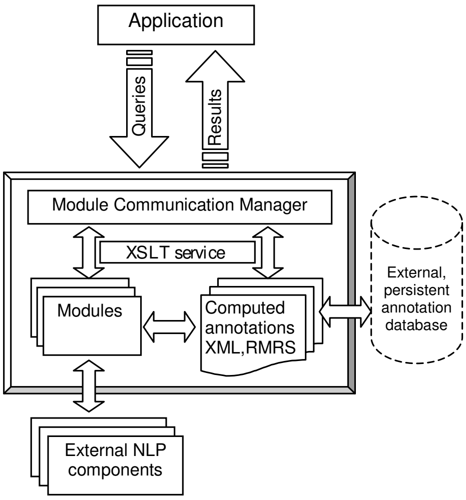

Application
Queries
Results
Module Communication Manager
XSLT service
Modules
Computed annotations XML,RMRS
External, persistent annotation database
External NLP components

Figure 1: Middleware architecture

[1] I would like to thank Robert Barbey, Özgür Demir and Thomas Klöcker for their implementation work, Bernd Kiefer and the co-authors in the cited papers for fruitful cooperation, and the reviewers for valuable comments. This work has been supported by a grant from the German Federal Ministry of Education and Research (FKZ 01IWC02).

component dependency structure in general forms a graph.

2.1 Session and multi-dimensional annotation management

The resulting multi-dimensional annotations are stored in a per-session markup storage (Fig. 2) that groups all annotations for an input query (a sentence or text) in *annotation collections*. The markup storage can also be made persistent by saving it to XML files or to an XML database. Annotations can be accessed uniquely via a URI of the form `hog://sid/acid/aid` in XPath expressions where `sid` is a session ID, `acid` is an annotation collection ID and `aid` is an annotation identifier typically signifying the name of the producing component. Structured metadata like configuration and processing parameters (e.g. processing time and date, language ID etc.) are always stored within the annotation markup as first root daughter element.

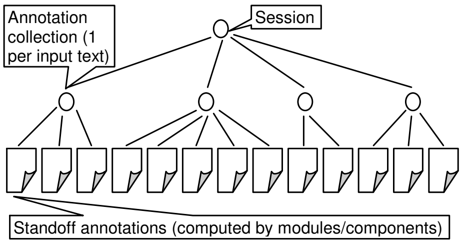

Annotation collection (1 per input text)
Session
Standoff annotations (computed by modules/components)

Figure 2: Session and multi-dimensional markup storage

2.2 XML standoff markup as first-class citizen

Unlike other NLP architectures (e.g. GATE (Cunningham et al., 2002) etc.), Heart of Gold treats XML standoff annotations (Thompson and McKelvie, 1997) as first class citizens and natively supports XML (and only XML) markup of any kind. Moreover, Heart of Gold does not prescribe specific DTDs or Schemata for annotations, provided that the markup is well-formed. In this sense, it is a completely open framework that may however be constrained by requirements of the actually configured components. The advantage of this openness is easy integration of new components. Mappings need only be defined for the immediately depending annotations (see next section) which is by far not an n-to-n mapping in practical applications.

However, the fact that a specific DTD or Schema is not imposed by the middleware does not mean that there are no minimal requirements. Linking between different standoff annotations is only possible on the basis of a least common entity, which we propose to be the character spans in the original text[2]. Moreover, we additionally propose the use of the XML ID/IDREF mechanism to facilitate efficient integration and combination of multi-dimensional markup.

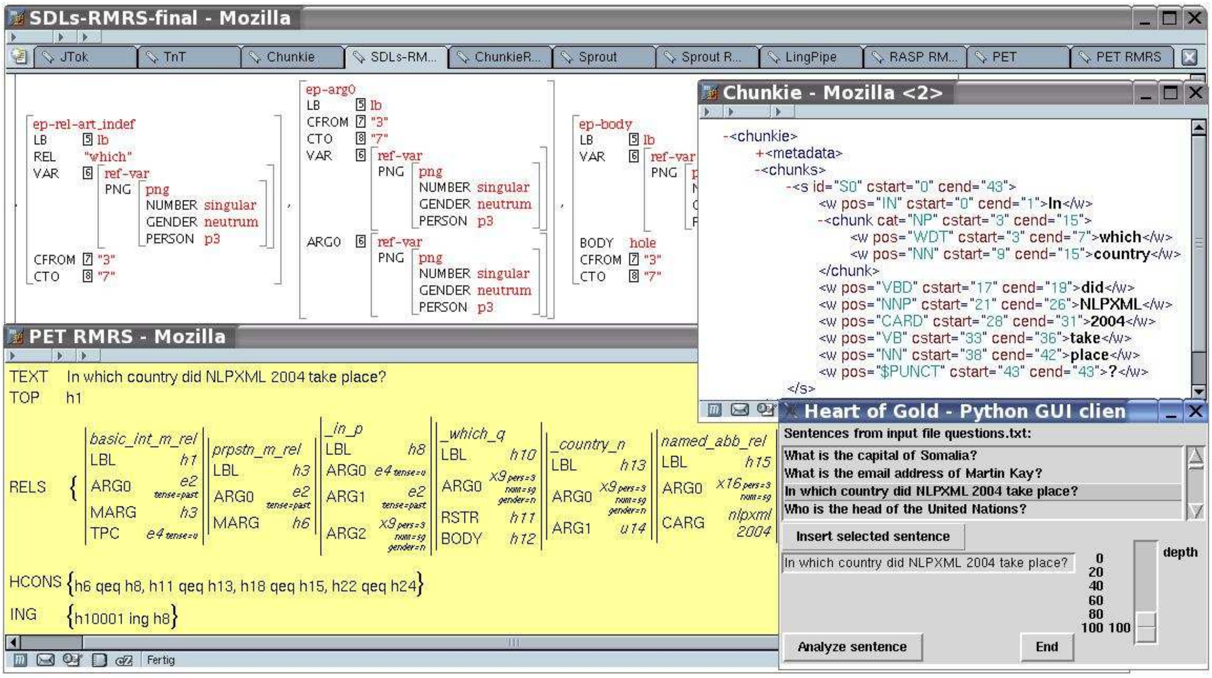
Finally, depending on the scenario, specific common, standardized markup formats are appropriate, an example is RMRS (Copestake, 2003) for deep-shallow integration in Section 3 or the XML-encoded typed feature structure markup generated by SProUT (Drożdżyński et al., 2004).

2.3 XSLT as 'glue' and query language

We propose and Heart of Gold heavily relies on the use of XSLT for combining and integrating multi-dimensional XML markup. The general idea has already been presented in (Schäfer, 2003), but the developments and experiences since then have encouraged us to proceed in that direction and Heart of Gold can be considered as a successful, more elaborated proof of concept. The idea is related to the open markup format framework presented above: XSLT can be used to transform XML to other XML formats, or to combine and query annotations. In particular, XSLT stylesheets may resolve conflicts resulting from multi-dimensional markup, choose among alternative readings, follow standoff links, or decide which markup source to give higher preference.

(Carletta et al., 2003), e.g. propose the NXT Search query language that extends XPath by adding query variables, regular expressions, quantification and special support for querying temporal and structural relations. Their main argument against standard XPath is that it is impossible to constrain both structural and temporal relations within a single XPath query. Our argument is that XSLT can complement XPath where XPath alone is not powerful enough, yet providing a standardized language. Further advantages we see in the XSLT approach are portability and efficiency (in contrast to 'proprietary' and slow XPath extensions like NXT), while it has a quite simple syntax in its (currently employed) 1.0 version. XSLT can be conceived as a declarative specification language as long as an XML tree structure

[2] Our experience is that a common tokenization is not realistic—too many existing NLP components have differing concepts of what constitutes a token.



```
ep-rel-art_indef
LB      [5] lb
REL     "which"
VAR     [6] ref-var
            PNG  png
                 NUMBER singular
                 GENDER neutrum
                 PERSON p3
CFROM   [7] "3"
CTO     [8] "7"
,
ep-arg0
LB      [5] lb
CFROM   [7] "3"
CTO     [8] "7"
VAR     [6] ref-var
            PNG  png
                 NUMBER singular
                 GENDER neutrum
                 PERSON p3
ARG0    [6] ref-var
            PNG  png
                 NUMBER singular
                 GENDER neutrum
                 PERSON p3
,
ep-body
LB      [5] lb
VAR     [6] ref-var
            PNG
BODY    hole
CFROM   [7] "3"
CTO     [8] "7"
```

```
-<chunkie>
   +<metadata>
   -<chunks>
      -<s id="S0" cstart="0" cend="43">
         <w pos="IN" cstart="0" cend="1">In</w>
         -<chunk cat="NP" cstart="3" cend="15">
            <w pos="WDT" cstart="3" cend="7">which</w>
            <w pos="NN" cstart="9" cend="15">country</w>
         </chunk>
         <w pos="VBD" cstart="17" cend="19">did</w>
         <w pos="NNP" cstart="21" cend="26">NLPXML</w>
         <w pos="CARD" cstart="28" cend="31">2004</w>
         <w pos="VB" cstart="33" cend="36">take</w>
         <w pos="NN" cstart="38" cend="42">place</w>
         <w pos="$PUNCT" cstart="43" cend="43">?</w>
      </s>
```

```
TEXT    In which country did NLPXML 2004 take place?
TOP     h1

RELS  { basic_int_m_rel        | prpstn_m_rel          | _in_p                 | _which_q              | _country_n            | named_abb_rel
        LBL   h1               | LBL   h3              | LBL   h8              | LBL   h10             | LBL   h13             | LBL   h15
        ARG0  e2 tense=past    | ARG0  e2 tense=past   | ARG0  e4 tense=u      | ARG0  x9 pers=3       | ARG0  x9 pers=3       | ARG0  x16 pers=3
        MARG  h3               | MARG  h6              | ARG1  e2 tense=past   |       num=sg          |       num=sg          |       num=sg
        TPC   e4 tense=u       |                       | ARG2  x9 pers=3       |       gender=n        |       gender=n        | CARG  nlpxml
                               |                       |       num=sg          | RSTR  h11             | ARG1  u14             |       2004
                               |                       |       gender=n        | BODY  h12             |                       |

HCONS {h6 qeq h8, h11 qeq h13, h18 qeq h15, h22 qeq h24}
ING   {h10001 ing h8}
```

Sentences from input file questions.txt:

What is the capital of Somalia?
What is the email address of Martin Kay?
In which country did NLPXML 2004 take place?
Who is the head of the United Nations?

Insert selected sentence

In which country did NLPXML 2004 take place?

depth: 0, 20, 40, 60, 80, 100 — 100

Analyze sentence   End

Figure 3: Heart of Gold analysis results in GUI with specialized XML visualizations

is preserved (not necessarily fully isomorphic to the input structure). However, XSLT is Turing-capable and therefore suited to solve in principle any markup integration or query problem. Finally, extensions like the upcoming XSLT/XPath 2.0 version or efficiency gains through XSLTC (translet compilation) can be taken on-the-fly and for free without giving up compatibility. Technically, the built-in Heart of Gold XSLT processor could easily replaced or complemented by an XQuery processor. However, for the combination and transformation of NLP markup, we see no advantage of XQuery over XSLT.

Heart of Gold comes with a built-in XSL transformation service, and module adapters (Section 2.4) can easily implement transformation support by including a few lines of code. Stylesheets can also be generated automatically in Heart of Gold, provided a systematic description of the transformation input format is available. An example is mapping from named entity grammar output type definitions in scenario 1 below. Stylesheets are also employed to visualize the linguistic markup, e.g. by transforming RMRS to HTML (Fig. 3) or LaTeX.

2.4 Integrated NLP components

NLP components are integrated through adapters called modules (either Java-based, subprocesses or via XML-RPC) that are also responsible for generating XML standoff output if this is not supported natively by the components (e.g., TnT, Chunkie). Various shallow and deep NLP components have already been integrated, cf. Fig. 4.

Component	Type	Depth	Languages
JTok	tokenizer	10	de, en, it,. . .
ChaSen	Jap. tagger	10	ja
TnT	stat. tagger	20	de, en
Chunkie	stat. chunker	30	de, en
ChunkieRmrs	chunk RMRS	35	de, en
LingPipe	stat. NER	40	en, es,. . .
SDL	subarchitect.		
Sleepy	shallow parser	40	de
SProUT	shallow NLP	40	de, el, en, ja,. . .
RASP	shallow NLP	50	en
PET	HPSG parser	100	de, el, en, ja,. . .

Figure 4: Integrated components. References for components and resources not cited are available on http://heartofgold.dfki.de/Publications_Components.html

3 Scenario 1: Deep-Shallow Integration

The idea of hybrid deep-shallow integration is to provide robust linguistic analyses through multi-dimensional NLP markup created by shallow and deep components, e.g. those listed in Fig. 4. Robustness is achieved in two ways: (1) various shallow components perform preprocessing and partial statistical disambiguation (e.g. PoS tagging of unknown words, named entity recognition) that can be used by a deep parser by means of a so-called XML input chart (multi-dimensional markup combined through XSLT in a single XML document in a format convenient for the parser). (2) shallow component's output is transformed through XSLT to partial semantic representations in RMRS syntax (Copestake, 2003) that is potentially more fine-grained and structured than what is digestible by the deep parser as preprocessing input (mainly PoS/NE type and span information via the XML input chart). This allows for (a) a fallback to the shallow representation in case deep parsing fails (e.g. due to ungrammatical input), (b) combination with the RMRS generated by deep parsing or fragments of it in case deep parsing fails.

First application scenarios have been investigated successfully in the DEEPTHOUGHT project (Uszkoreit et al., 2004). A further application (hybrid question analysis) is presented in (Frank et al., 2006). Recently, linking to ontology instances and concepts has been added (Schäfer, 2006).

4 Scenario 2: Shallow Cascades

The second scenario is described in (Frank et al., 2004) in detail. A robust, partial semantics representation is generated from a shallow chunker's output and morphological analysis (English and German) by means of a processing cascade consisting of four SProUT grammar instances with four interleaved XSLT transformations. The cascade is defined using the declarative system description language SDL (Krieger, 2003). An SDL architecture description is compiled into a Java class which is integrated in Heart of Gold as a sub-architecture module (Fig. 5). The scenario is equally a good example for XSLT-based annotation integration. Chunker analysis results are included in the RMRS to be built through an XSLT stylesheet using the XPath expression

```
document($uri)/chunkie/chunks/chunk[
    @cstart=$beginspan and @cend=$endspan]
```

where `$uri` is a variable containing an annotation identifier of the form `hog://sid/acid/aid` as explained in Section 2.1.

5 Scenario 3: Corpus Annotation

Given the powerful online middleware architecture described above, automatic, multi-dimensional corpus annotation can then be regarded as a simple by-product. Heart of Gold supports persistent storage of XML markup either on the file system or to XML databases through the built-in XML:DB interface. Through XSLT, it is possible to combine multi-dimensional markup (that would straightforwardly be stored in multiple XML documents) into a single XML document.

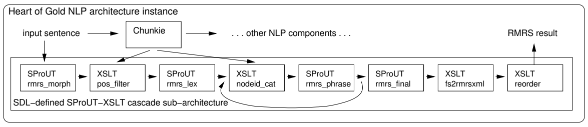


Figure 5: SProUT XSLT cascade in a Heart of Gold architecture instance.

References

U. Callmeier, A. Eisele, U. Schäfer, and M. Siegel. 2004. The DeepThought core architecture framework. In *Proc. of LREC-2004*, pages 1205–1208, Lisbon, Portugal.

J. Carletta, S. Evert, U. Heid, J. Kilgour, J. Robertson, and H. Voormann. 2003. The NITE XML toolkit: flexible annotation for multi-modal language data. *Behavior Research Methods, Instruments, and Computers, special issue on Measuring Behavior*, pages 353–363.

A. Copestake. 2003. Report on the design of RMRS. Technical Report D1.1b, University of Cambridge, UK.

H. Cunningham, D. Maynard, K. Bontcheva, and V. Tablan. 2002. GATE: A framework and graphical development environment for robust NLP tools and applications. In *Proceedings of ACL-2002*.

W. Drożdżyński, H.-U. Krieger, J. Piskorski, U. Schäfer, and F. Xu. 2004. Shallow processing with unification and typed feature structures — foundations and applications. *Künstliche Intelligenz*, 2004(1):17–23.

A. Frank, K. Spreyer, W. Drożdżyński, H.-U. Krieger, and U. Schäfer. 2004. Constraint-based RMRS construction from shallow grammars. In *Proceedings of HPSG-2004*, pages 393–413. CSLI Publications, Stanford.

A. Frank, H.-U. Krieger, F. Xu, H. Uszkoreit, B. Crysmann, and U. Schäfer. 2006. Question answering from structured knowledge sources. *Journal of Applied Logics, Special Issue on Questions and Answers*. To appear.

H.-U. Krieger. 2003. SDL—a description language for building NLP systems. In *Proceedings of the HLT-NAACL Workshop on the Software Engineering and Architecture of Language Technology Systems*, pages 84–91.

U. Schäfer. 2003. WHAT: An XSLT-based infrastructure for the integration of natural language processing components. In *Proceedings of the HLT-NAACL Workshop on the Software Engineering and Architecture of Language Technology Systems*, pages 9–16, Edmonton, Canada.

U. Schäfer. 2006. OntoNERdIE—mapping and linking ontologies to named entity recognition and information extraction resources. In *Proc. of LREC-2006*, Genoa, Italy.

H. S. Thompson and D. McKelvie. 1997. Hyperlink semantics for standoff markup of read-only documents. In *Proceedings of SGML-EU-1997*.

H. Uszkoreit, U. Callmeier, A. Eisele, U. Schäfer, M. Siegel, and J. Uszkoreit. 2004. Hybrid robust deep and shallow semantic processing for creativity support in document production. In *Proc. of KONVENS-2004*, pages 209–216.