Multipurpose Linguistic Web Service for Polish

Maciej Ogrodniczuk, Michał Lenart

Introduction

Numerous actions taken throughout Europe to fulfil the CLARIN mission of *creating, coordinating and making language resources and technology available and readily useable for scholars in the humanities and social sciences* resulted in making various language processing tools available online, most often in the form of Web services. Following this direction, the Linguistic Engineering Group at the Institute of Computer Science, Polish Academy of Sciences (ICS PAS) provided common interface to several offline tools for linguistic processing of Polish in the form of a Web service offering multi-level stand-off annotation (hence the service is further referred to as *the Multiservice*).

The Multiservice represents linguistic properties in a newly developed TEI P5-based format which offers high flexibility of TEI and the power of the embedded feature structure formalism. The scope of operation is determined by the underlying offline tools, which currently offer segmentation (paragraph and sentence splitting, tokenization, identification of compound word forms), lemmatization, morphological analysis (all with Morfeusz SGJP morphological analyser), morphosyntactic disambiguation (TaKIPI and Pantera taggers), named entity recognition (statistical NER tool), shallow parsing (Spejd) and deep parsing of Polish (Świgra) – all available as annotation layers in the target output format.

Architecture of the Online Service

The offline tools are bound together in a common infrastructure, capable of chaining them and presenting results in the unified format. Language processing is triggered by requests sent to the Web service, enqueued and handled in asynchronous manner to allow processing large amounts of text. Invoking one of the available methods results in returning the request token (identifier) which can be used to check the request status and retrieve the result when processing completes.

Each part of the processing chain is defined by operation type (i.e. linguistic function such as tagging or shallow parsing), requested tool name (e.g. “TaKIPI” or “Pantera” for tagging, since there can be many variant tools of the same type configured) and a map of properties specific to the provided tool. By using chains, one request can trigger several (interrelated or independent) operations at once, e.g. “segment, lemmatize, morphologically analyze and tag text with Pantera, then perform shallow parsing with Spejd, possibly using disambiguation information provided by the tagger”).

Linguistic Representation

A common representation binding the tools together is based on the format implemented for the National Corpus of Polish (see http://nkjp.pl) – a stand-off, TEI P5-encoded annotation storing different levels of description in separate, interlinked files similarly to PAULA or MAF. For interchange reasons the Multiservice uses its “packaged” version, with all annotations saved in a single file under a `<teiCorpus>` element (forming “a temporary collection of annotation layers”).

Below we present a sample partial description from the morphosyntactic layer (see http://nlp.ipipan.waw.pl/TEI4NKJP/ for more detailed and complete examples):

```
<seg xml:id="m-seg1" corresp="s-seg1">
  <fs type="morph">
   <f name="interps">
    <vAlt>
     <fs type="lex" xml:id="m-seg1-lex">
      <f name="base"><string>lato</string></f>
      <f name="ctag"><symbol value="subst"/></f>
```

```
     <f name="msd"><symbol value="pl:gen:n" xml:id="m-seg_1-msd"/>
    </fs>
    <fs type="lex" xml:id="m-seg2-lex">
     <f name="base"><string>rok</string></f>
     <f name="ctag"><symbol value="subst"/></f>
     <f name="msd"><symbol value="pl:gen:m3" xml:id="m-seg_2-msd"/>
    </fs>
   </vAlt>
  </f>
  <f name="disamb">
   <fs feats="#pantera" type="tool_report">
    <f fVal="#m-seg_2-msd" name="choice"/>
    <f name="interpretation">
     <string>rok:subst:pl:gen:m3</string>
     ...
```

The results of processing can be also output in a popular WebLicht Text Corpus Format (TCF).

Interface and Usage

The Multiservice is intended to be used via a dedicated API. Additionally, a simple Web interface has been prepared to facilitate online tests.

The interaction with the service consists of 4 steps:

(1) user sends a processing request with the linguistic function name and its parameters;

(2) the service generates a token for the request (used for further operation), stores the request in the queue and returns the token to the user;

(3) the user keeps querying the service about the status of execution of a request identified with a given token until the status shows that the execution stopped – because of error or ended successfully;

(4) if execution failed, the user retrieves an error message from the service; if execution succeeded, the user retrieves the result from the service.

The pull execution method gives potential interfaces far more flexibility and allows for better control over annotation processes as compared to callback-based implementations which is important with respect to using Web services for processing larger collection of texts which may result in long processing times.

Dedicated Web interface (http://chopin.ipipan.waw.pl:8083/WSWebClient/) can be also used to test execution of the service. After constructing the chain and starting the analysis, the Web application checks periodically the status of the request. When it ended execution, the result is retrieved and displayed to the user. In case of a failure, an appropriate error message is presented.

Further Work

Further technical work will concentrate on integration of a wider range of input and output formats with multiple encodings and integrated converters as well as plugging in additional annotation components, including coreference resolvers or word-sense disambiguators, currently under active development.

Research activities could further delve into aspects of maintaining semantic interoperability of the representation format (including issues related to Polish morphosyntax vs. e.g. ISOCat Data Category Registry) as well as Web service chaining issues such as algorithms for automated chain detection.