Creating a machine-readable version of Bulgarian valence dictionary (A case study of CLaRK system application)[1]

Elisaveta Balabanova[2] and Krasimira Ivanova
The BulTreeBank Project
http://www.BulTreeBank.org
Linguistic Modelling Laboratory - CLPPI, Bulgarian Academy of Sciences
Acad. G.Bonchev Str. 25A, 1113 Sofia, Bulgaria
Tel: (+3592) 979 28 25, Fax: (+3592) 70 72 73
elisavetabal@yahoo.com, krassy_v@abv.bg

1 Introduction

This paper describes the creation of a machine-readable version of the Bulgarian valence dictionary (Popova 1987).

The valence dictionary consists of about 1000 Bulgarian verbs with their valence frame, meaning, the morphological and semantic characteristics of their arguments and examples of their usage.

The wide application of the valence dictionary motivated us to create an electronic version of it and also to make further transformations over this machine-readable form. The electronic form will be used further for various linguistic applications (building an HPSG lexicon of Bulgarian, support of the syntactic annotation of Bulgarian sentences, syntactic generalizations over classes of verbs, etc. - see Section 4). Additionaly, the electronic form of the dictionary will provide information for top-down filtering during partial parsing of Bulgarian texts. The building of the electronic form was supported by the CLaRK system (Simov et al 2001).

To point out the advantages of the electronic form of the dictionary over the hard copy version, we need to compare the two. The machine-readable version is not in exact correspondence to the book and this is where its benefits come from. The reasons for the difference are:

i) the book is intended for human use, i.e. some linguistic knowledge about its usage is left implicit;

ii) in the paper version the linguistic information is presented in four separate levels - the frame (argument structure) of the verb, morphological values of the verb's arguments, semantic values of the arguments, examples of the verb's usage. In the electronic version all these values are represented as in an HPSG lexical entry in the form of a single constraint over the verb called *argument structure list* (or *subcategorisation list* in HPSG 1994). The last allows for checking over the all possible combinations of arguments;

iii) in the book the frame of the verb is presented as a complex structure - in a lexical entry several disjunctive specific frames are represented in one complex frame. In fact, in the actual use of the

[1] The work reported here is done within the BulTreeBank Project. The project is funded by the Volkswagen Stiftung, Federal Republic of Germany under the Programme *"Cooperation with Natural and Engineering Scientists in Central and Eastern Europe"* contract I/76 887.
[2] Also a PhD student at the Institute of Bulgarian Language, Bulgarian Academy of Sciences

verb only one of these frames is realized. The creation of the electronic version aims at such detailed frame representation.

As can be seen, a machine-readable version requires a totally different approach to the information representation (i.e. maximal explicitness) from the book form. Hence the creation of the electronic version of the valence dictionary is not just typing in the lexical entries into a file but also linguistic restructuring of the entries. Such a task entails linguistic competence and intensive human work. Therefore it is important to minimize the human labour and to develop a strategy for quality control of the result. The last two became possible with the use of the CLaRK system. The created XML electronic version of the valence dictionary allows for different restructurings of the information in it (according to the particular linguistic needs). We have made one such reordering over the electronic valence dictionary- an XSLTransformation with regard to the valence frame.

This paper has the following structure:
2. Valence dictionary paper version
3. What enabled the electronic version creation: The CLaRK system and the XML
4. The electronic version
5. XSLTransformation over the electronic form
6. Applications of the electronic valence dictionary
7. Conclusion

2 Valence dictionary paper version

The existing valence dictionary (Popova 1987) consists of about 1000 Bulgarian verbs with information about their valence frame, meaning, the morphological and semantic values of their arguments and examples of their usage. These verbs were chosen by the author as the most frequent ones. The dictionary has in its preface a list of all used abbreviations (the abbreviated names of the arguments and other used symbols).

Each entry in the dictionary is structured in the following way (see Appendix 2):

Verb, its transitivity and aspect.
Meaning
I. Frame (the arguments that the verb requires)
II. Morphology of the verb's arguments
III. Semantics of the arguments
IV. Examples of the verb's usage

The frame represents the verb's valence potential. It is written as a formula where the first element is the subject, followed by the predicate and then by the direct, indirect object, subordinate clause or adjunct of the predicate.

A sample of a frame is the following:

S(ubject)+P(redicate)+O2(indirect object) | C(lause)

Here the vertical bar symbol | is used for disjunction.

The morphological and semantic levels show the respective values that the arguments can take (e.g. on the morphological level the subject of certain verb can be a noun or a pronoun, on the semantic level the noun can take the values of a person, animal, etc.)

To determine the specific frame for each particular verb's usage one has to combine the formula with the morphological and semantic restrictions on the arguments. So, in fact, every frame splits

into several detailed frames (for example, the predicate described by the above formula can take an indirect object or a subordinate clause, therefore the formula results in two frames).

As obvious, the valence dictionary provides information not only for the verbs but also for the syntax and semantics of their arguments, so the dictionary is a valuable source for various linguistic research (see section 6).

Besides its undoubted linguistic value, the valence dictionary has one important feature making it human-limited in terms of use - the information representation in it relies on the human intuition and is not formal enough for machine usage. The conversion of the dictionary into machine-readable form gives much more possibilities for its application.

3 What enabled the electronic version creation: The CLaRK system and the XML

The CLaRK System is an XML-based software system for corpora development. It incorporates several technologies: (1) XML technology; (2) Unicode; (3) Regular Cascade Grammars; (4) Constraints over XML Documents.

For document management, storing and querying XML technology was chosen because of its popularity and ease of understanding. The core of CLaRK System is an XML editor which is the main interface to the system.

XML defines the structure of a document as a sequence of elements. The whole document is an element containing the rest of the elements. The structure elements of a document are marked-up by means of tags. Tags can surround the content of an element or they can mark some points in it. The use of the XML language enabled the logical structuring of the information in the electronic valence dictionary.

For optimization of the work on XML documents the CLaRK system was supplied with a module for imposing constraints over XML documents. The constraints imposed by the module are of different type (none of which can be defined in the standard XML technology). These are: (1) finite-state constraints - additional constraints over the content of given elements on the base of the context in a document; (2) number constraints - cardinality constraints over the content of a document; (3) value constraints - restriction (based on a context) of the possible content or parent of element in a document.

From these the value-constraints module was used in the creation of the machine-readable form of the valence dictionary. This module served two purposes then. On the one hand, it enabled the creation of constraints for validation of the document's structure (according to pre-defined requirements). On the other, it supported the filling in of the elements values (restricting the linguist to choose from lists of values). In this way, the use of the constraints' engine helped for the minimization of human labour in the conversion of the valence dictionary into electronic format.

4 The electronic version

As already mentioned, the paper version relies on the linguistic intuition of the person who uses it (to connect, for example, the morphologic values of the arguments with the appropriate example of the verb's usage). For instance, the morphologic level says the subject of the verb *translate* can be a noun or a pronoun but on the level of the verb's usage nothing points to the right example of the subject being a noun or a pronoun (see Appendix 2). Besides, in some cases not all combinations of

morphological and semantic constraints are possible. For the exclusion of the impossible ones the book relies on the user's intuition for the language. In a computer-oriented version such cases have to be explicitly excluded in order to avoid overgeneration.

It is obvious that all the above problems are typical for the human usage of the dictionary.

A totally new approach to the information representation and the control of the document's quality has been adopted in the electronic form. The building of a machine-readable version requires a detailed and structured information representation and is error prone besides, since it demands a lot of human work. To avoid the possible inconsistencies caused by the human approach, the CLaRK system capabilities for document structure representation and for imposing constraints have been used. The document's structure has been defined into a DTD (Document Type Definition), according to the XML requirements.The check of the constitency of the result, as well as the optimization of the work, has been enabled by the value constraints tool.

The constraints have been used to realize dependency in two ways:

i) locally - within lexical entries, and

ii) globally - to connect the information from the entries with the dictionary's metalanguage (header in the XML document)

i) Local imposing of constraints

The lexical entries have been restructured in such a way that all the linguistic information became explicit. As suitable mode of restructuring has been chosen the HPSG approach to the verb where all the constraints over an argument are given on one and the same place. This helps to check whether all the combinations of arguments are possible and thus verify the formula. The book's four levels of information representation has been compressed into two main levels (defined as elements in the XML structure of the document) - *headword* and *meaning* (see Appendix 1.1). The level *headword* contains information about the transitivity and aspect of the verb. The level *meaning* has the sublevel *frame*. The *frame* is given as a regular expression which always contains one predicate and at least one argument, preceding or following the predicate. The *arguments* and the *predicate* are sub-levels of the frame. They are ordered in the way they follow from left to right in the valence formula (beginning with the subject). The *syntax* (defined as element *function* in the XML structure) and *semantics* of each argument are given as its sub-levels. The *conjunction*, introducing a particular argument is also defined as a separate level. The complex representation of information for a given argument ends with a level of an *example* of the verb's usage with each particular combination of argument values.

As can be seen, in the electronic version the levels of morphology, semantics and the verb's usage examples are compressed into one - the level of the argument. In this way, for each morphological value of a given argument the corresponding semantic value is presented together with an example of the verb's usage in the particular case. This makes the information maximally explicit and hence machine-readable.

ii) Global imposing of constraints

The information from the lexical entries is connected by use of the constraints engine with the metalanguage (the header) of the document. The header comprises three lists of abbreviations (in comparison to the book version which has only one list), each of them with different function. These lists are:

- full names of the arguments used as abbreviations in the document;
- all the morphological values of the arguments
- the arguments' semantic values

The lists in the header serve globally as constraints over the whole XML document. They perform this in two ways. On the one hand, the lists support the creation of the whole XML structure of the dictionary via the value-constraints module, and on the other, they facilitate the building of the lexical entries. So, the process of optimization is in both directions - the element values from the lexical entries are automatically stored into the lists of the header and also any further entering of semantic and morphological constraints is simply a choice from these lists.

5 XSLTransformation over the electronic form

The so created electronic version allows for various designs of its structure due to its XML structure. Such restructurings can be made with the XSL (Extensible Stylesheet Language) by means of transformations (see Appendix 1.3). We have made one such transformation over the machine-readable form of the valence dictionary. This operation was done in order to achieve particular linguistic aims. The ClaRK system with its XSLT tool enabled the transformation. Before describing how this was done, we will briefly present the XSLT.

A transformation in the XSLanguage (see XSLT) is expressed as a well-formed XML document conforming to the Namespaces in XML Recommendation which may include both elements which are defined by XSLT and such which are not. XSLT-defined elements are distinguished by belonging to a specific XML namespace. A prefix of `xsl:` is used to refer to the elements in the XSLT namespace.

A transformation in XSLT describes rules for transforming a source tree into a result tree. The transformation is achieved by associating patterns with templates. A pattern is matched against elements in the source tree. A template is instantiated to create part of the result tree. The result tree is separate from the source tree. The structure of the result tree can be completely different from the structure of the source tree. In constructing the result tree, elements from the source tree can be filtered and reordered, and arbitrary structure can be added.

A transformation in XSLT contains a set of template rules. A template rule has two parts: a pattern which is matched against nodes in the source tree and a template which can be instantiated to form part of the result tree. This allows the transformation to be applicable to a wide class of documents with similar source tree structures. (see XSLT)

XSLT makes use of the expression language defined by XPath (XPath) for selecting elements for processing, for conditional processing and for generating text. The data model used by XSLT is the same as the one used by XPath with some additions. Any two XML documents that have the same tree will be treated the same by XSLT.

The transformation over the electronic valence dictionary reorders child elements of the *entry*. The goal is a valence dictionary structure consisting of frames. From *entry – head,meaning – frames – frame* (see Appendix 1.1), the child elements of the *entry* become *frame- structure,headword, meaning* (see Appendix 1.2). For example, if in the source document a verb has two meanings with three frames each, the result document will have six frames for the verb. Each frame contains information about the verb and its current meaning. The result document of the transformation over the valence dictionary is a list of all the frames from the dictionary. The frames are sorted according to their structure. The achieved result are classes of frames – each class containing all the verbs that have this particular frame.

6 Applications of the electronic valence dictionary

The electronic form of the valence dictionary with its two XML structures has diverse applications.
1) The structure of the valence dictionary as an XML document helps:

-to check the validity and completeness of the dictionary itself. This is done by matching the information about the verbs in the corpus to the verb frames in the dictionary

-to build HPSG lexicon

-to make the syntactic annotation of the sentences in the corpus

The most frequent verbs are taken out of the corpus. The corpus we use is tokenized, morphologically annotated, manually disambiguated, and syntactically annotated on the level of phrases. Examples of the usage of these verbs are extracted from the corpus. The examples are then matched to their corresponding frames from the valence dictionary. Thus some verb frames, missing in the dictionary, could be added to the respective verb's entry. In this way the completeness of the valence dictionary is checked. In the matching of examples to frames there may be frames from the dictionary that will prove unproductive. They will be excluded. Actually, this verifies the validity of the dictionary. Performing the above procedures, important linguistic results for the language resources in the corpus will be achieved.

First, the most frequent verbs, together with their frames, will become part of the HPSG lexicon we are aiming to build in our project. Then, the most productive valence frames of the Bulgarian verbs will be distinguished. These frames will be assigned to verbs in sentences from the corpus. In the syntactic annotation of these sentences the valence frames will be used to provide information for the verb's arguments.

2) The structure of the valence dictionary as a transformed XML document

The restructured by XSLTransformation electronic valence dictionary will be used to:

-verify the consistency of the frames for groups of verbs - if a given verb has frame f1, and f1 is realized in the corpus as two separate frames (more specific ones), the XSLT result document facilitates to check if the last holds true for the other verbs from the same frame class f1.

-make generalizations over verbs - if two verbs have three identical frames but the first verb has also one frame more, the XSLT document helps to find out whether the fourth frame of this verb also belongs to the second verb.

7 Conclusion

The paper describes the creation of a machine-readable version of the valence dictionary of Bulgarian verbs. The electronic version has been built by restructuring the lexical entries into an HPSG style of information representation and by defining the document’s metalanguage. The aim has been maximal explicitness of the information in order to make it machine-readable. By use of the XSLT the created electronic valence dictionary was restructured with regard to the verb frame. The tasks of creation and transformation of the machine-readable version have been performed with the help of the CLaRK system (especially its facilities for document structure representation, its constraints engine and its XSLT tool). The use of the system minimized the human work and helped to check the consistency of the whole document. The result is the development of a valuable language electronic resource.

References

Maria Popova, *Kratuk valenten rechnik na glagolite v savremennia bulgarski knizoven ezik.* Sofia 1987, Bulgarian Academy of Sciences Publishing House.

Kiril Simov, Zdravko Peev, Milen Kouylekov, Alexander Simov, Marin Dimitrov, Atanas Kiryakov. *CLaRK - an XML-based System for Corpora Development*. In: Proc. of the Corpus Linguistics 2001 Conference, pages: 558-560.

Simov K, Popova G, Osenova P, *HPSG-based syntactic treebank of Bulgarian (BulTreeBank)*. In: "A Rainbow of Corpora: Corpus Linguistics and the Languages of the World", edited by Andrew Wilson, Paul Rayson, and Tony McEnery; Lincom-Europa, Munich (to appear), pages: 135-142.

XSLT *1999.XSL Transformations(XSLT) version 1.0. W3C Recommendation*. http://www.w3.org/TR/xslt

XPath. *1999. XML Path Language (XPath) version 1.0. W3C Recommendation*. http://www.w3.org/TR/xpath

Appendix 1.1- The valence dictionary as an XML document

The document type definition (DTD) of the electronic valence dictionary is the following:

<!DOCTYPE dictionary
[<!ELEMENT dictionary (header,entry+)>
<!ELEMENT **header** (grammar, function, semantics)>
<!ELEMENT grammar (abbr+)>
<!ELEMENT function (abbr+)> <!ELEMENT semantics (abbr+)>
<!ELEMENT abbr (entity, value+)>
<!ELEMENT entity (#PCDATA)>
<!ELEMENT value (#PCDATA)>
<!ELEMENT **entry** (headword, meaning*)>
<!ELEMENT *headword* (#PCDATA,type)>
<!ELEMENT type (sv?,nesv?, prh?, neprh?)>
<!ELEMENT sv EMPTY>
<!ELEMENT nesv EMPTY>
<!ELEMENT prh EMPTY>
<!ELEMENT neprh EMPTY>
<!ELEMENT *meaning* (#PCDATA, prh?, neprh?, frames)>
<!ELEMENT frames (frame+)>
<!ELEMENT frame (arg*,(arg,pred | pred, arg),arg*)>
<!ELEMENT arg (func, sem)>
<!ELEMENT func (#PCDATA)>
<!ELEMENT sem (semArg+)>
<!ELEMENT semArg (element,descrElem*)>
<!ELEMENT element (#PCDATA)>
<!ELEMENTdescrElem((linkW|qW)?,value,example)>
<!ELEMENT linkW (#PCDATA)>
<!ELEMENT qW (#PCDATA)> <!ELEMENT value (#PCDATA)>
<!ELEMENT example (#PCDATA)>
<!ELEMENT pred EMPTY>]

For convenience the main levels are bolded, their sub-levels - italicized, the levels of argument and the predicate - underlined.

Appendix 1.2 -DTD of the result document (after the transformation)

```
<!DOCTYPE frames [
<!ELEMENT frames (frame+)>
<!ELEMENT frame (structure?,headword,meaning?)>
<!ELEMENT structure (arg*,(arg,pred | pred, arg),arg*,examples?)>
<!ELEMENT headword (#PCDATA,type)>
<!ELEMENT type (sv?,nesv?, prh?, neprh?)>
<!ELEMENT sv EMPTY>
<!ELEMENT nesv EMPTY>
<!ELEMENT prh EMPTY>
<!ELEMENT neprh EMPTY>
<!ELEMENT meaning (#PCDATA, prh?, neprh?)>
<!ELEMENT pred EMPTY>
<!ELEMENT arg (func, sem)>
<!ELEMENT func (#PCDATA)>
<!ELEMENT sem (semArg+)>
<!ELEMENT semArg (element,descrElem*)>
<!ELEMENT element (#PCDATA)>
<!ELEMENT descrElem ((linkW|qW)?,value,example)>
<!ELEMENT linkW (#PCDATA)>
<!ELEMENT qW (#PCDATA)>
<!ELEMENT value (#PCDATA)>
<!ELEMENT example  (#PCDATA)>
]>
```

Appendix 1.3 XSLTransformation

```
<xsl:transform>
<xsl:template match="self::dictionary">
<frames>
 <xsl:apply-templates select = "//entry"/>
</frames>
</xsl:template>

<xsl:template match="self::entry">
<xsl:for-each select = "descendant::frame">
 <frame>
   <structure>
    <xsl:apply-templates select = "*" mode = "1"/>
   </structure>
   <xsl:copy-of select = "parent::frames/parent::meaning/preceding-sibling::headword"/>
   <xsl:apply-templates select = "parent::frames/parent::meaning"/>
 </frame>
 </xsl:for-each>
</xsl:template>
```

```
<xsl:template match="self::meaning">
 <xsl:copy>
  <xsl:apply-templates select = "*[not(self::frames)]" mode = "1"/>
 </xsl:copy>
</xsl:template>

<xsl:template match="self::*" mode = "1">
 <xsl:copy-of select = "."/>
</xsl:template>
</xsl:transform>
```

Appendix 2 - Sample lexical entries

A sample lexical entry from the dictionary's paper version:

Translate - imperfective, intransitive
Transfer a message, text from one language into another.
I. Subject+ translate+Direct obj.1
(Indirect obj.2)
II. Subject=Noun, Pronoun
Direct obj.1=Noun1, Pronoun demonstr, Pronoun accusative form
Indirect obj.2=to+Noun2, from+ Noun2
III. Noun=person
Noun1=text - spoken or written
Noun2=language
IV. He translated the article into Russian.
She translates poems from German.
One of the Slovak poets has already translated all the poems from Botev.
Translate it now into Turkish.

The same entry in the electronic version of the dictionary:

<headword>**translate**<type><nesv/><prh/></type>
</headword>
<meaning>**Transfer a message, text from one language into another**<frames><frame><arg>
<func>**Subject**</func><sem><semArg>
<element>**Noun**</element>
<descrElem><value>**Person**</value><example>**Let Misha translate the article.**</example></descrElem></semArg>
<semArg><element>**Pronoun**</element>
<descrElem><value></value><example>**She translates poems.**</example></descrElem></semArg></sem></arg><pred/>
<arg><func>**Direct object**</func><sem><semArg>
<element>**Noun**</element>
<descrElem><value>text</value><example>**He translated the article**.</example></descrElem>
<descrElem><value>**spoken and written text**</value><example>He **translated their words.**</example></descrElem></semArg><semArg>
<element>**Pronoun full form**</element>
<descrElem><value></value><example></example></descrElem></semArg><semAr>
<element>**Personal pronoun accusative form**</element>
<descrElem><value></value><example>**Translate it now into Turkish.**</example></descrElem></semArg></sem></arg></frame>
<frame><arg>
<func>**Subject**</func><sem><semArg>
<element>**Noun**</element>
<descrElem><value>**Person**</value><example></example></descrElem></semArg><semArg>
<element>**Pronoun**</element>
<descrElem><value></value><example>**He translated the document.**</example></descrElem></semArg></sem></arg><pred/><arg>
<func>**Direct object**</func><sem><semArg>
<element>**Noun**</element>

<descrElem><value>text</value><example>**She translated the letter.**</example></descrElem>
<descrElem><value>**spoken and written text**</value><example>**He translated their words**.</example></descrElem></semArg><semArg>
<element>**Pronoun full form**</element>
<descrElem><value></value><example></example></descrElem></semArg><semArg>
<element>**Personal pronoun accusative form**</element>
<descrElem><value></value><example>**Ivan translated it.**</example></descrElem></semArg></sem></arg><arg>
<func>**Indirect object**</func><sem><semArg>
<element>**Noun2**</element>
<descrElem><linkW>**to**</linkW><value>person</value><example>**He was translating to the president at the reception.**</example></descrElem></semArg><semArg>
<element>**Noun2**</element>
<descrElem><linkW>**from**</linkW><value> person</value><example>**She translates poems from Goethe.**</example></descrElem></semArg></sem></arg></frame></frames></meaning></entry>
<entry>