AGILE

Automatic Generation of Instructions in Languages of Eastern Europe

---

Title ***Formal specification of full grammar models***

Implementation of tactical generation resources for all three languages in a Final Prototype

Authors Serge Sharoff,
Lena Sokolova,
Danail Dochev,
Ivana Kruijff-Korbayová,
Geert-Jan Kruijff
Jiří Hana
Kamenka Staykova
Elke Teich
John Bateman

Deliverables *WP6: SPEC3-Bg, SPEC3-Cz, SPEC3-Ru*;

WP7: IMPL3-Bg, IMPL3-Cz, IMPL3-Ru

Status *Final*

Availability *Public*

Date *2000-07-10*

INCO COPERNICUS PL961104

Abstract

This document comprises the deliverables SPEC3 and IMPL3 of Work Package 6, task 6.3 and Work Package 7, task 7.3 of the AGILE project, which were concerned with the specification and implementation of grammatical resources for the Final Prototype. The present deliverable thus presents both formal specifications of linguistic phenomena pertaining to the target texts and the implementation of these specifications in the KPML grammar development environment. The linguistic phenomena covered in the present deliverable include: sublanguage-specific government patterns (different linguistic realizations of identical conceptual structures), support verbs, circumstantial elements (an extended treatment of spatiotemporal prepositional phrases and means), modality (expression of possibility/enablement), aspect choice, subject dropping, textual conjunction (lexico-grammatical means of the expression of sequences of a user's actions), syntactic agreement, quantification (cardinal numbers and selection constructions), and clause complexity (linguistic realizations of conjunction, disjunction and of the RST relations of Means, Purpose, Logical Condition, and Sequence). Since this is the last deliverable in the work package, we briefly recapitulate the general approach taken in grammar development using the KPML (Komet-Penman MultiLingual) system and summarize the achievements made in this work package.

More information on AGILE is available on the project web page and from the project coordinators:

URL:	http://www.itri.brighton.ac.uk/projects/agile
email:	agile-coord@itri.bton.ac.uk
telephone:	+44-1273-642900
fax:	+44-1273-642908

Table of Content

1. Introduction 1

1.1 Goals of this deliverable 2

1.2 Overview of this deliverable 2

1.3 Notational conventions in this document 3

2. Theory, methods, techniques and organization of grammar development 5

2.1 Motivations for the approach of multilingual grammar development pursued in AGILE 5

2.2 Theory: Systemic Functional Linguistics 8

2.3 Tactical generation in KPML 10

2.4 Multilingual grammar development with KPML 13

2.4.1 Resource sharing: contrastive grammar 13

2.4.2 KPML as a development environment for multilingual grammars 14

2.5 Organization of work in Work Packages 6 & 7 16

3. Linguistic specification and implementation of grammatical resources for the Final Prototype 19

3.1 Transitivity 22

3.1.1 Theoretical background 22

3.1.2 Government patterns and circumstantials 23

3.1.3 Support verb constructions 31

3.2 Aspect 34

3.2.1 Theoretical background 34

3.2.2 Aspect choice in subordinated clauses 40

3.2.3 The Aspect region specification 41

3.2.4 The Aspect region implementation 42

3.3 Modality 45

3.3.1 Linguistic Specification 45

3.3.2 Formal Specifications 49

3.3.3 Implementation 53

3.4 Subject dropping 67

3.4.1 Description of the Phenomenon 67

3.4.2 Formal specifications 73

3.4.3 Implementation 75

3.5 Word order 84

3.5.1 Word Order Freedom in Czech, Russian and Bulgarian 85
3.5.2 Text Generation Overview 90
3.5.3 A Word Ordering Algorithm Using Information Structure 92
3.5.4 Structural Constraints on Word Order in the grammars 97
3.5.5 Concluding remarks 112
3.6 Textual conjunction 114
3.6.1 Description of the Phenomenon 114
3.6.2 Formal Specifications 119
3.6.3 Implementation 121
3.7 Agreement 132
3.7.1 Introduction 132
3.7.2 Subject-Predicate Agreement 133
3.7.3 Agreement within the nominal group 143
3.7.4 Agreement of subject and predicative adjective 147
3.7.5 Conclusion 150
3.8 Quantification 151
3.8.1 Cardinal numerals 151
3.8.2 Quantity selection 158
3.9 Clause complexity 167
3.9.1 Formal Specifications 170
3.9.2 Implementation 179
3.10 Summary and conclusions 209

Table of Figures

Figure 1: Notational conventions in Systemic Functional Grammar 3
Figure 2: Notation for system networks 3
Figure 3: Syntax for computational system network specifications..... 4
Figure 4: Organizing principles of representation in SFG..... 9
Figure 5: Input to generation with KPML 11
Figure 6: Output structures (clause level) for SPL-1 and SPL-2 12
Figure 7: Dimensions of multilingual description 14
Figure 8: Graph of a system network having used the INSPECT option in the development window..... 15
Figure 9: Government pattern realization 26
Figure 10: The TRAJECTORY-PROCESS-TYPE system and chooser..... 30
Figure 11: Verbal-group-type-chooser 33
Figure 12: The Aspect region of grammar. 43
Figure 13: Choosers for the Aspect region. 44
Figure 14: Classification of Verbal group complexes semanics (Halliday 94) 46
Figure 15: DEICTICITY system (Bg)..... 53
Figure 16: DEICTICITY-CHOOSER (Bg)..... 53
Figure 17: VOLITIONALITY system (Bg)..... 54
Figure 18: VOLITIONALITY-CHOOSER (Bg)..... 54
Figure 19: DEGREE-OF-MODALITY system (Bg) 54
Figure 20: DEGREE-OF-MODALITY-CHOOSER (Bg)..... 55
Figure 21: POSSIBILITY-TYPE system (Bg) 55
Figure 22: POSSIBILITY-TYPE-CHOOSER (Bg)..... 56
Figure 23: MODALITY-CONDITIONALITY system (Bg)..... 56
Figure 24: MODALITY-CONDITIONALITY-CHOOSER (Bg) 57
Figure 25: MODALITY-POLARITY system (Bg)..... 57
Figure 26: MODALITY-POLARITY-CHOOSER (Bg) 58
Figure 27: AUXSTEM-INSERTED system (Bg) 58
Figure 28: AUXSTEM-VOICE system (Bg)..... 59
Figure 29: SPL for the sentence in Example(11)..... 59
Figure 30: Grammatical structure generated from the SPL in Figure 29 by the Bulgarian grammar 60
Figure 31: CONATION system (Bg)..... 61

Figure 32: CONATION-CHOOSER (Bg) 61
Figure 33: CONATION-TYPE system (Bg) 62
Figure 34: CONATION-TYPE-CHOOSER (Bg) 62
Figure 35: CONATIONDEPENDENT-TYPE system (Bg) 63
Figure 36: CONATIONDEPENDENT-TYPE-CHOOSER (Bg) 63
Figure 37: CONATION-SECONDAGENT system (Bg) 64
Figure 38: SPL for the sentence in Example(11) 64
Figure 39: Grammatical structure generated from the SPL in Figure 38 by the Bulgarian grammar 65
Figure 40: SPL macros for conation in Bulgarian resources 65
Figure 41 : SPL for the sentence in Example (12) 66
Figure 42: Grammatical structure generated from the SPL in Figure 41 by the Bulgarian grammar 66
Figure 43: SUBJECT-DROPPING system 77
Figure 44: SUBJECT-DROPPING chooser 77
Figure 45: NOMINAL-GROUP-SIMPLEX system 78
Figure 46: NOMINAL-GROUP-SIMPLEX chooser 78
Figure 47: Generated grammatical structure for (60) 79
Figure 48: Generated grammatical structure for (61) 80
Figure 49: Generated grammatical structure with subject 81
Figure 50: Generated grammatical structure without subject 82
Figure 51: SUBJECT-DROPPING system (Bg) 82
Figure 52: SUBJECT-DROPPING chooser (Bg) 83
Figure 53: NOMINAL-GROUP-SIMPLEX system (Bg) 83
Figure 54: NOMINAL-GROUP-SIMPLEX chooser (Bg) 83
Figure 55: Flexible word ordering algorithm 93
Figure 56: Default order definitions in Bulgarian grammar 101
Figure 57: Generated output for (83) 102
Figure 58: Generated output for (84) 103
Figure 59: Generated output for (85) 104
Figure 60: Generated output for (86) 105
Figure 61: Generated output for (87) 106
Figure 62: A combination of sequence styles: using numbered list for the top-level GOALs, and explicit sequence discourse markers for the lower-level GOALs, along with aggregation. 114

Figure 63: A combination of sequence styles: using explicit sequence markers for the top-level GOALs within the list of steps, and a numbered list for the lower-level steps... 115
Figure 64: Different linguistic markers depending on the number of elements in a sequence116
Figure 65: Different linguistic markers depending on the number of elements in a sequence121
Figure 66: CONJUNCTION system (Cz, Bg, Ru) 122
Figure 67: CONJUNCTION chooser (Cz, Bg, Ru) 122
Figure 68: STRUCTURAL-CONJUNCTION system (Cz, Bg, Ru)..... 123
Figure 69: STRUCTURAL-CONJUNCTION chooser (Cz, Bg, Ru) 123
Figure 70: LEXIFIED-CONJUNCTION system (Cz, Bg, Ru) 124
Figure 71: LEXIFIED-CONJUNCTION chooser..... 124
Figure 72: CONJUNCTIVE-PROCESS-REGULATION system (Cz, Bg, Ru) 124
Figure 73: CONJUNCTIVE-PROCESS-REGULATION chooser (Cz, Bg, Ru)..... 125
Figure 74: PROCESS-REGULATED-TYPE system (Cz, Bg, Ru)..... 125
Figure 75: PROCESS-REGULATED-TYPE chooser (Cz, Bg, Ru) 125
Figure 76: SEQUENCE-CONJUNCTION system (Cz) 126
Figure 77: SEQUENCE-CONJUNCTION system (Bg) 126
Figure 78: SEQUENCE-CONJUNCTION system (Ru) 126
Figure 79: SEQUENCE-CONJUNCTION chooser (Cz, Bg, Ru)..... 127
Figure 80: ABSOLUTE-SEQUENCE-CONJUNCTION system (Cz) 127
Figure 81: ABSOLUTE-SEQUENCE-CONJUNCTION system (Bg)..... 128
Figure 82: ABSOLUTE-SEQUENCE-CONJUNCTION system (Ru)..... 128
Figure 83: ABSOLUTE-SEQUENCE-CONJUNCTION chooser (Cz, Bg, Ru)..... 129
Figure 84: SPL for the sentences in (102)..... 129
Figure 85: SPL macros for textual conjunctives..... 130
Figure 86: Grammatical structure generated from the SPL in Figure 84 by the Czech grammar 130
Figure 87: Grammatical structure generated from the SPL in Figure 84 by the Bulgarian grammar 131
Figure 88: Grammatical structure generated from the SPL in Figure 84 by the Russian grammar 131
Figure 89: Subject – predicate agreement (Cz)..... 141
Figure 90: Subject – predicate agreement (Ru)..... 142
Figure 91: Passive construction with zero auxiliary verb agreement (Ru)..... 142
Figure 92 - Agreement within nominal group 148
Figure 93: Agreement of subject and predicative adjective..... 150
Figure 94: Accusative with numeral above 4 155

Figure 95: Accusative with numeral not greater than 4 156
Figure 96: Fragment of a sentence with genitive with a numeral 157
Figure 97: The numeral “5” having Acc of the all nominal group (Ru) 158
Figure 98: Cardinal agreement (Ru) 158
Figure 99: The quantity selection construction as it is modelled for English and Bulgarian. 159
Figure 100: Structure of quantification construction realized for Czech and Russian grammar 160
Figure 101 SPL for the quantity selective construction (Ru) 160
Figure 102: Quantification system and chooser (Bg) 161
Figure 103: Quantifier-type and Quantifier-select systems (Bg) 162
Figure 104: The system minirange-type (Ru and Cz) 163
Figure 105 The inquiry-q-code of MINIRANGE-THING-TYPE system (Ru and Cz) 163
Figure 106: The NOMINAL-LIKE-GROUP-CLASS system (Ru and Cz) 164
Figure 107: NOMINAL-LIKE-GROUP-CLASS-CHOOSER chooser (Ru and Cz) 164
Figure 108 Quantification system 165
Figure 109: The MINOR-PROCESS-TYPE system. 165
Figure 110 - Clause nexus / tactic dimensions 171
Figure 111: Ordering of Dominant and Dependent in a hypotactic clause complex 171
Figure 112 - Types of enhancement 175
Figure 113 Type of interdependence: system (Cz, Ru, Bg) 180
Figure 114 Expansion taxis: system (Cz, Bg, Ru) 180
Figure 115 Type of interdependence: chooser (Cz, Ru, Bg) 181
Figure 116 Expansion taxis: chooser (Cz, Bg, Ru) 182
Figure 117 Expansion type: system (Cz, Bg, Ru) 182
Figure 118 Expansion type chooser (Cz, Bg, Ru) 183
Figure 119: Parataxis system (Cz, Bg, Ru) 183
Figure 120: Hypotaxis-alpha-complexity system 184
Figure 121: Hypotaxis system (Cz, Ru, Bg) 184
Figure 122: CONJUNCTION system (Cz, Ru, Bg) 184
Figure 123: Hypotaxis-alpha-complexity chooser (Cz, Ru, Bg) 185
Figure 124 Extending coordination type: system (Cz, Ru, Bg) 186
Figure 125 Extending coordination type: chooser (Cz, Ru, Bg) 187
Figure 126: Grammatical structure for (147), Czech 188
Figure 127: Grammatical structure for (148), Czech 189
Figure 128 Qualifying-coordination type: system (Cz, Ru, Bg) 189

Figure 129 Qualifying-coordination type: chooser (Cz, Ru, Bg) 190
Figure 130: Grammatical structure for (149), Czech..... 191
Figure 131: Qualifying condition type system (Cz, Ru, Bg) 192
Figure 132 Qualifying-condition-purpose-dependent: system (Cz)..... 192
Figure 133: Qualifying-condition-purpose-dependent chooser (Cz) 193
Figure 134 Qualifying-condition-means-dependent: system (Cz)..... 193
Figure 135: Qualifying-condition-conditional-dependent: system (Cz) 193
Figure 136: Generated structure for (150), Russian..... 194
Figure 137: Generated structure for (151), Bulgarian..... 195
Figure 138: Grammatical structure for (152), Czech..... 196
Figure 139: Grammatical structure for (153), Czech..... 197
Figure 140: Grammatical structure for (154), Czech..... 199
Figure 141: Grammatical structure for (154), Bulgarian 200
Figure 142: Grammatical structure for (154), Russian 201
Figure 143: Grammatical structure for (155), Czech..... 203
Figure 144: Grammatical structure for (155), Russian 204
Figure 145: Grammatical structure for (155), Bulgarian 205
Figure 146: Grammatical structure for (156), Czech..... 207
Figure 147: Grammatical structure for (156), Russian 208
Figure 148 Coverage of lexico-grammatical resources..... 211

1. Introduction

This document comprises the deliverables SPEC3 and IMPL3 of Work Packages 6 and 7 (henceforth: WP6, WP7), tasks 6.3 and 7.3, of the AGILE project, which have been concerned with the specification and implementation of grammatical resources for the Final Prototype. The work naturally builds on the previous deliverables SPEC1, SPEC2 and IMPL1, IMPL2, which presented linguistic specifications and their implementations for the Initial Demonstrator and the Intermediate Prototype.

The primary goal of AGILE consists in developing a suite of software tools to assist technical writers in the production of user manuals in the CAD/CAM domain in selected languages of Eastern Europe (Bulgarian, Czech and Russian). This problem is approached by means of multilingual generation from a common semantic representation of the procedural aspects of the task of using such software tools. Multilingual documentation is thus generated directly from the user interface and domain task model, in contrast to the current practice where the initial documentation is produced in one language only and subsequently translated.

The objective of WP6 has been to provide linguistic specifications of the phenomena considered relevant for modeling instructional texts in the CAD/CAM domain according to an initial corpus analysis of instructional texts (cf. deliverable CORP). WP6 has been divided up into three tasks in which the grammar models for Bulgarian, Czech and Russian have been incrementally built up, ensuring that each stage was self-contained and linguistically interesting and that the final description was robust.

The objective of WP7 has been to develop computational grammars based on the linguistic specifications provided in WP6 and adequate for the generation of texts in the CAD/CAM domain, but not restricted to the sublanguage. Like WP6, WP7 has been divided up into three tasks in which the computational grammatical resources of Bulgarian, Czech and Russian have been successively built up from an Initial Demonstrator to an Intermediate Prototype to the now due Final Prototype.

The Final Prototype to be delivered at this stage of the project is capable of generating text in more than one style, both for expressing complex procedures and for describing the functionality of the application being documented (cf. TEXM3). While style control options are set at the level of macroplanning and discourse models, style expression mechanisms are realised through lexico-grammatical means. Thus, the tasks 6.3 and 7.3 of WPs 6 and 7 comprised:

- accounting for the relations between stylistic features of discourse and lexico-grammatical means (wherever applying);
- describing this in terms of Systemic Functional Grammar;
- revising and extending the grammar models for Bulgarian, Czech and Russian accordingly;
- extending and detailing out the grammar implementations for Bulgarian, Czech and Russian

1.1 Goals of this deliverable

The primary goal of the present report is to present a linguistic description, specification and implementation of the set of grammatical phenomena that are relevant for the Final Prototype. This description is a natural extension of the SPEC2 and IMPL2 deliverables in the following two senses: We describe some additional grammar phenomena that need to be treated in order to be able to generate the target texts of the Final Prototype, and we detail out the description of some grammar phenomena already discussed in SPEC2, taking account of the language-specific idiosyncrasies of the grammars of Bulgarian, Czech and Russian. For example, we implemented a contrastive account of spatial locating circumstantials for all three languages and revised the aspect description and implementation given in SPEC2 and IMPL2. Doing so, we continued applying the strategy of sharing development efforts among the three sites by distributing tasks of grammar description and implementation among the partners according to linguistic phenomena rather than individual languages. This is reflected in the organization of the main part of the report (Chapter 3) in that the individual sections are organized according to phenomena rather than individual languages.

The second goal of this report is to provide the rationale behind choosing the particular approach to grammar development taken in AGILE, its underlying theory and the methodology adopted. Since this is the final report in WPs 6 and 7 and in order to be able to fully appreciate the results achieved in these two work packages, it appears necessary to recapitulate the starting situation at the beginning of the project that led to the choice of our approach and to the adoption of our particular work strategy.

1.2 Overview of this deliverable

We start our report with recapitulating the motivations for choosing the approach of grammar development pursued in AGILE (Section 2.1), its theoretical underpinnings (Section 2.2) and the methodology pursued in implementation, known as *resource sharing* (Sections 2.3 and 2.4). Sections 2.2 to 2.4 are slightly condensed versions of Sections 2 and 3 of the first deliverable in WP7 (IMPL1). Section 2.5 briefly describes the three phases of work in the WPs 6 and 7 and the distribution of responsibilities according to partners. The technical part of this report contains the linguistic description, specification and implementation of the linguistic phenomena we dealt with in tasks 6.3 and 7.3 (Chapter 3). The linguistic phenomena treated in this report are the following (the partners responsible for each are given in brackets):

- Transitivity (Section 3.1; RRIAI);
- Aspect (Section 3.2; RRIAI);
- Mood and Modality (Section 3.3; BAS);
- Word order (Section 3.4; CU)
- Subject dropping (Section 3.5; CU)
- Textual conjunctions (Section 3.6; CU)
- Clause complexity (Section 3.7; CU)
- Quantification (Section 3.8; RRIAI)

- Agreement (Section 3.9; CU)

We conclude with a summary of the achievements made in WPs 6 and 7, presenting the current coverage of the grammars of Bulgarian, Czech and Russian implemented in AGILE and pointing out their limits as well as assessing their re-usability for other domains and applications.

1.3 Notational conventions in this document

Figure 1 presents the general notational conventions used in Systemic Functional Grammar and Figure 2 shows the notation we adopted for the specification of system networks. Figure 3 provides the syntax of computational system network specifications in KPML. In running text, feature names are given in bold face, system names are given in capitals, and also the names of regions are given in capitals. Within the deliverable, languages are abbreviated by the following codes: Bg for Bulgarian, Cz for Czech, and Ru for Russian.

functional elements	Actor, Subject, etc
system names	MOOD
grammatical features	feature
selection expressions:	
delicacy	[feature-x : feature-y,...]
simultaneity	[feature-x & feature-y,...]
realization statements:	
insert	+Subject
conflate	Subject/Actor
expand	Mood(Finite)
order	Subject ^ Finite
preselect	Subject:nominal-group
lexical constraints:	
classify	Process::doing-verb
inflectify	Noun:::singular
lexify	Noun ! LEXEME
syntactic structures	trees (presented as screen dumps)

Figure 1: Notational conventions in Systemic Functional Grammar

```
System-name →
   [feature-a] (Function1 : feature-x)
   [feature-b] (Function2 : feature-y, Function3 ! feature-z)
   [feature-c] (Function4 : feature-f,feature-g)
```

Figure 2: Notation for system networks

```
(system
 :name SYSTEM-NAME
 :inputs (OR feature-x
          (AND feature-y feature-z))
 :outputs ((0.5 feature-a
                (insert Function-1))
           (0.5 feature-b
                (conflate Function-2 Function-3)
                (preselect Function-2 feature-c)))
 :chooser SYSTEM-NAME-CHOOSER
 :region REGION-NAME
 :metafunction METAFUNCTION)
```

`:name` gives the name of the system; `:inputs` specifies the features that act as entry conditions to the system; `:outputs` specifies the features of the system, where features may have realization statements attached (`insert, conflate, preselect` etc.); `:region` specifies the functional region the system belongs to---this can be thought of as a finer grained subclassification of metafunction given in the `:metafunction` slot and is used for the organization of resources.

Figure 3: Syntax for computational system network specifications

Furthermore, the following abbreviations are used:

- DM for domain model;
- A-box for Assertion Box;
- POS for Part Of Speech;
- so for someone;
- sth for something.

Throughout this report, in the English glosses of the examples given, we use abbreviations of morphological categories. Categories present in the abbreviations are in the following order: POS[1], Gender, Number, Case, Person and Determinacy. A category is omitted if it is not relevant in a given context. The possible values for each category are the following (not all are present in all three languages):

- POS: Adj (adjective), PastPart (past participle), Nominalization (nominalized verb), Imper (imperative mood), Indic (indicative mood), Inf (infinitive), Gerund (Russian verbal adverb);
- Gender: M (masculine, in Czech masculine animate), I (masculine inanimate), F (feminine), N (neuter);
- Number: Sg (singular), Pl (plural);
- Case: Nom (nominative), Gen (genitive), Dat (dative), Acc (accusative), Voc (vocative), Loc (locale) and Ins (instrumental);
- Person: 1, 2, 3;
- Determinacy: Det (definitive article for Bulgarian);

[1] Sometimes more detailed than classical divison to 9 or 10 POS categories, e.g. PastPart (past participle). This category is also omitted if it is the same for the English word.

- Voice: Refl (reflexive);
- Aspect: Imperf (imperfective), Perf (perfective).

2. Theory, methods, techniques and organization of grammar development

2.1 Motivations for the approach of multilingual grammar development pursued in AGILE

The platform for implementation of the grammatical resources of Bulgarian, Czech and Russian chosen in AGILE is the Komet-Penman MultiLingual (KPML) system (Bateman, 1997), which is based on the Penman system for generating English (Mann, 1993; Mann and Matthiessen, 1983; Penman, 1989). Below, we briefly explain the motivations for choosing KPML as tactical generator and grammar development workbench. To make the present document self-contained, we also provide a brief description of the theoretical underpinnings of the KPML system, Systemic Functional Linguistics (SFL; Halliday, 1973, Halliday, 1985, Halliday & Matthiessen, 1999) (Section 2.2) and describe KPML's multilingual design and use as a workbench for developing grammatical resources (Section 2.3).[2]

To be able to fully appreciate the achievements made in grammar development in AGILE, it is important to note what the particular starting situation was at the beginning of the project. At the very outset of the project, we decided that we wanted to avoid a situation in which we would have three different generators, using three different approaches to grammar and three different formalisms. Given the size of the task, we agreed that we needed a common approach, so that development efforts could be shared in a maximally efficient and effective way among the partners.

First of all, and not dissimilar to other multilingual projects, the starting situation in AGILE was that computational resources for more than one language had to be developed. However, unlike in other multilingual projects like DRAFTER (Paris et al., 1995) and many others, the languages we have dealt with in AGILE had not been treated computationally for the purpose of natural language generation at all. While there exist computational components for all of Bulgarian, Czech and Russian, in particular morphological components and lexicons,[3] there had not been computational accounts of their grammars that could be readily re-used for the tasks at hand in AGILE. Neither had any kind of generation algorithm been designed for any of the three languages. This meant that for all three languages *new grammars* of Bulgarian, Czech and Russian had to be developed.

Second, the expertise in computational linguistics for Bulgarian, Czech and Russian is rather variable. For example, while in the Czech Republic and in Russia, there is a large body

---

[2] A slightly more detailed description of the major principles of Systemic Functional Linguistics, its use in Natural Language Generation and its implementation of multilinguality is given in the IMPL1 deliverable.

[3] Existing morphological components of Bulgarian, Czech and Russian have been re-used in AGILE (cf. e.g., deliverable IMPL2).

of work in computational linguistics — in particular there is a long-standing tradition in machine translation in Russia — there is less expertise in this field concerning Bulgarian. Thus, any framework for natural language generation and grammar development would have been new to all the partners involved and it therefore seemed doubly significant to adopt an approach was easily accessible for all partners, i.e., a framework that was sufficiently documented and well tested.

A third factor that had to be taken into account in deciding for an approach to grammar development and tactical generation was the linguistic tradition in Eastern Europe. Even though formal, structurally oriented approaches exist, the primary linguistic tradition in Eastern Europe is a functional one. In particular, the Prague School had a major influence on continental European linguistics in general; and also in Russia and Bulgaria, there is a strong tradition in "communicative" approaches to language. For all three languages there exist large bodies of linguistic-descriptive work in the functional tradition, which we wanted to be able to make use of in our descriptions. Given this, the approach to be taken in AGILE needed to be as compatible as possible with the standing tradition in Eastern Europe.

Finally, concerning the linguistic properties of the three languages we had to deal with in the project, even though all of them are Slavonic languages, they belong to different subtypes — Czech is a Western-Slavonic language, Russian an Eastern-Slavonic language and Bulgarian a Southern-Slavonic language — with different typological features. In particular, comparing the three, while Czech and Russian are highly inflecting languages, Bulgarian lost most of its inflections, and thus among the Slavonic languages it is like English among the Germanic languages. Given that we wanted to use a common general linguistic and computational approach for all three languages, the approach chosen needed to be flexible enough to accommodate the particularities of Slavonic languages, their commonalities as well as their differences.

In summary, the constraints that had to be taken into account in the decision for a linguistic and computational approach to grammar development and tactical generation in AGILE at the outset of the project can be summarized under the following keywords:

- **Complexity & size of the task**. Computational resources for three languages that had not been described before for the purpose of natural language generation had to be developed.
- **Existing expertise**. Different starting points in terms of existing resources and general computational expertise relating to the three languages had to be accommodated.
- **Existing linguistic traditions**. It appeared desirable to be able to draw on the extensive bodies of descriptive work in the Eastern European functional linguistic tradition.
- **Typological considerations**. Typological variation among the three languages had to be accommodated.

Considering these constraints carefully, we chose the Komet-Penman MultiLingual (KPML) system (Bateman, 1997b) as a platform for grammar development and as a tactical generation system. Note that it is not common to find both functionalities developed at equal levels of sophistication in *one* system: KPML is *designed* to be a tactical generation system as well as a workbench for implementing computational grammars for natural language generation. KPML supports the desiderata we established above in the following ways:

- **Complexity & size of the task**. KPML's implementation of multilinguality is based on the notion of *resource sharing*, which says that the specification of a grammar of a new language can be based on existing specifications, even if there is a considerably typological distance between the new language and the one for which an implementation exists. In fact, throughout the project we have made extensive use of the existing grammar of English implemented in KPML, the Nigel grammar (Mann and Matthiessen, 1983), even though English is considered typologically rather different from Slavonic languages. More details about the notion of resource sharing are given in Section 2.4 below.
- **Existing expertise**. KPML is based on one of the oldest, most extensively tested and most widely used generation system for English, the Penman system (Penman, 1989). Extensive and detailed documentation of KPML is available (Bateman, 1997a), so that there was sufficient material to work from for the partners for acquiring expertise in using the system. Also, in the first phase of the project, based on several tutorials that were held by John Bateman and Elke Teich for the partners, training materials were developed.
- **Existing linguistic traditions**. KPML's underlying linguistic theory is Systemic Functional Linguistics (SFL; Halliday, 1973; Halliday, 1985; Halliday & Matthiessen, 1999). Thus, the approach implemented in the KPML generator is a functional one and can potentially be easily reconciled with other kinds of functional approaches. In fact, we have integrated in KPML an approach emanating from the Prague School for dealing with word order, which was previously not handled flexibly enough for the requirements of Slavonic languages (cf. Chapter 4).
- **Typological considerations**. KPML has a rather sophisticated notion of multilinguality when compared with other multilingual systems, e.g. machine translation systems. It acknowledges the simple insight that no matter from which angle you compare any two languages, there will always be commonalities *and* differences. While in machine translation, either commonality is adopted as the main principle (interlingua systems) or diversity is assumed to be prevalent (transfer systems), KPML allows both at all levels of representation. This is especially desirable and actually a must, if one wants to re-use existing computational resources of one language for the description of another, typologically rather different language. The model of multilinguality underlying KPML's multilingual functionality is briefly described in Section 2.4 below.

Using KPML as the common approach to tactical generation and as the common grammar development platform made it possible not only to build upon the existing Nigel grammar of English, but also to share development efforts among the partners in such a way that work was divided up not so much in terms if individual languages, but more in terms of linguistic phenomena. For example, the principles governing word order in Bulgarian, Czech and Russian are basically the same for the three languages, even if their surface realizations differ slightly. Similarly, the basic choices of aspect are the same in Bulgarian, Czech and Russian, again with slight variation in textual instantiations. Thus, after identifying the principal areas of commonality, it was possible to distribute the linguistic and implementation work according to linguistic phenomena (see Section 2.5).

To conclude, KPML has been chosen as development platform because it has been explicitly designed for the purpose of offering generation projects large-scale resources that

are well tested and appropriate for practical applications as well as supporting to work in a contrastive-linguistic fashion. In our view, resource development for new languages has thus been considerably facilitated compared to developing grammars from scratch.

In the following sections, we give a brief introduction to the linguistic theory underlying KPML, Systemic Functional Linguistics (SFL), and to generation based on SFL, as well as to the model of multilinguality that is supported by SFL's basic representational categories. Readers familiar with SFL and the KPML framework may choose to proceed to Section 2.5, which gives a synopsis of the three phases of work in WPs 6 and 7, or move to the technical description of work in tasks 6.3 and 7.3 directly (Chapter 3).

2.2 Theory: Systemic Functional Linguistics

The linguistic-theoretical basis of the KPML system is Systemic Functional Linguistics (SFL). SFL is a British school of linguistics, belonging to the tradition of functional approaches to language (Hjelmslev, 1943; Dik, 1978; Halliday, 1973, Halliday, 1985) and showing affinities with the Continental-European Prague School (Firbas, 1966; Daneš, 1974; Sgall *et al*., 1986).

SFL is characterized by the notions of *function* and *system*. SFL is functional in that it acknowledges three broad functions languages have: the ideational, the interpersonal and the textual (see further below for details). SFL is systemic in that the main focus in description is on the grammatical system, i.e., on the grammatical paradigm. The kernel of SFL is Systemic Functional Grammar (SFG). The grammar of a language is represented as a *system network*, which can be read as a declarative statement of grammatical features and the co-occurrence constraints holding between them. Systemic Functional Grammar is thus a classification-based approach to grammar, rather than a rule-based one, in which an inheritance hierarchy of grammatical types, here called *features*, which are organized in increasing *delicacy*, constitutes the grammatical description — very similar to other models of grammar currently used in computational linguistics, such as Head-Driven Phrase Structure Grammar (Pollard and Sag, 1987; Pollard and Sag, 1992.

What makes SFG stand out from other approaches using classification is the functional motivation of grammatical types. Grammatical types are functionally motivated in the following ways. The notion of function subscribed to in SFL is predominantly manifested in the concept of *metafunctions*, a set of generalized functions that language is said to fulfil. The *ideational metafunction* encodes a language's propositional content. At the level of the clause, its grammatical aspect is notably reflected in the system of transitivity, which gives rise to configurations of processes and the participants and circumstances therein, such as Actor, Goal, Spatial Circumstantial, Temporal Circumstantial etc.[4] The *interpersonal metafunction* encodes speakers' roles in an interaction, their attitudes and evaluations. One of its major grammatical reflexes is the clause system of MOOD, which distinguishes between declarative, interrogative and imperative and accounts for the differences in syntactic structure that these different moods come along with. The *textual metafunction* encodes properties of textual organization, such as global text structure, coherence and cohesion. In the grammar, this is reflected in systems of theme-rheme patterning and

[4] Actor, Goal, Cirumstantial etc. are called functional elements and are given with initial capitals as a convention.

information structuring at clause level. The functionally motivated systems and their features are associated with *realization statements*. Realization statements are the attributes of a feature and specify the syntagmatic, surface-syntactic constraints that the functional class denoted by the feature exhibits. For instance, a surface-syntactic constraint associated with the feature declarative of finite clauses in English is that in syntactic structure the Subject is ordered before the Finite Verb. The distinction between paradigmatic, functionally motivated classes and syntagmatic structure is referred to as *axiality*: a linguistic description in SFG always has these two aspects that are linked by the relation of realization. Apart from metafunction and axiality and delicacy, a linguistic description is also organized according to *rank*. Rank is implemented as the top system in the grammatical classification and distinguishes between clauses, nominal groups, prepositional phrases, adjectival and adverbial groups, words and morphemes. This rank scale gives the basic grammatical classes for which particular sets of systems and their features hold, where the description of any two ranks is pairwise disjoint. In the overall model of SFL, grammar constitutes the lexico-grammatical *stratum*, where the other, more abstract strata are semantics and context. The strata are in a relation of inter-stratal realization, where categories of the context stratum are said to be realized by semantic categories and these, in turn, are realized by lexico-grammatical categories. See Figure 1 for a graphical summary of the five organizing principles of metafunctional diversification, stratification, ranking, axiality and delicacy.[5]

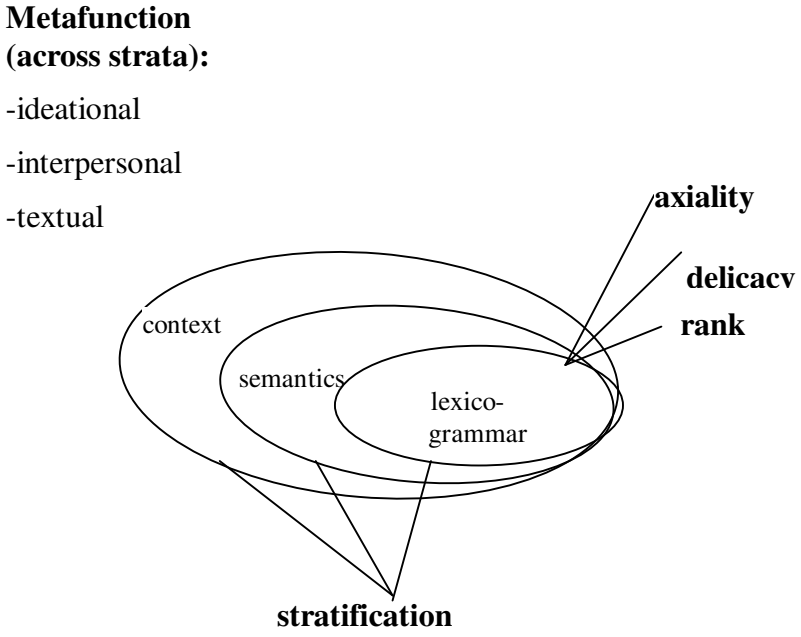


Figure 4: Organizing principles of representation in SFG

[5] For a fairly concise introduction to Systemic Functional Grammar see (Bateman, 1992). More comprehensive accounts of the theory can be found in (Halliday, 1978; Halliday and Matthiessen, 1999).

In the following section we relate the properties of SFG to the tasks involved in tactical generation and describe the process of generation with an SFG as implemented in the Penman-based generator KPML.

2.3 Tactical generation in KPML

Adopting a systemic functional model as a linguistic basis for generation generally supports the modelling of tasks involved in generation because many of its characteristics are suggestive of the kinds of resources needed in Natural Language Generation. This is evidenced by the fact that many generation systems are based on or have been inspired by SFL (e.g., Penman (Penman, 1989), KPML (Bateman, 1997), COMMUNAL (Fawcett and Tucker, 1989), Multex (Matthiessen *et al.*, 1995; Matthiessen *et al.*, 1998), Comet (McKeown *et al.*, 1990), Surge (Elhadad, 1990; Elhadad and Robin, 1996)) and many projects have used the SFG-based tactical generators Penman and KPML, e.g., the German KOMET project (Bateman *et al.*, 1991a; Bateman and Teich, 1995; Teich *et al.*, 1996; Bateman *et al.*, 1998), the German TECHDOC project (Roesner and Stede, 1992; Roesner and Stede, 1994), the British DRAFTER project (Paris *et al.*, 1995; Hartley and Paris, 1997) and the Canadian Healthdoc project (DiMarco *et al.*, 1995; Wanner and Hovy, 1996).

For describing how SFG-based generation works in Penman-style generators, let us briefly recall the tasks involved in tactical generation and then see how these tasks are handled in an SFG-based generator. The tasks involved in tactical generation are to do with breaking up a text plan into units that are grammatically realizable, such as clauses, nominal groups etc., and taking care of their grammatical and lexical realization. Thus, there are two subtasks (cf. Teich, 1999):

- task 1: to interpret a semantic input expression in terms of the grammar available;
- task 2: to spell out the syntactic constraints of the language in which an utterance is to be generated.

(Yang et *al.*, 1991) characterize task 1 as involving ”deciphering the goals given by the speaker program and determining how they can be realized in language” (Yang *et al.*, 1991, p.207) and task 2 as ensuring ”that the end product adheres to the syntactic rules of the language” (Yang *et al.*, 1991, p.207). The first is the functional aspect of tactical generation; the latter is the syntactic aspect.

For an SFG-based generator like KPML, the ‘generation question’ can then be formulated as follows: When to choose which feature from the grammatical system network (together with its associated syntactic constraints in the form of realization statements) according to a semantic input expression?

The generation process starts with an input expression in the form of SPL (Sentence Planning Language) (Kasper, 1989). See Figure 2 showing two such SPL expressions (henceforth: SPLs) for the sample sentences

(1) *The user chooses the PLINE command.*

and

(2) *Choose the PLINE command.*

The SPLs are instances of the Upper Model (Bateman *et al.*, 1990; Henschel and Bateman, 1994), which is the resort of ideational (i.e., propositional) meaning in KPML. Besides the ideational information expressed in an SPL, also interpersonal information (e.g., `:speechact command`) and textual information (e.g., `:identifiability-q identifiable`, `:theme o1`) is contained.[6]

(SPL-1) *The user chooses the PLINE command.*

```
(p / choose
     :speechact assertion
     :theme o1
     :actor (o1 / user
                :identifiability-q identifiable)
     :actee (o2 / command
                :identifiability-q identifiable
                :class-ascription (c / software-command
                                       :name PLINE)))
```

(SPL-2) *Choose the PLINE command.*

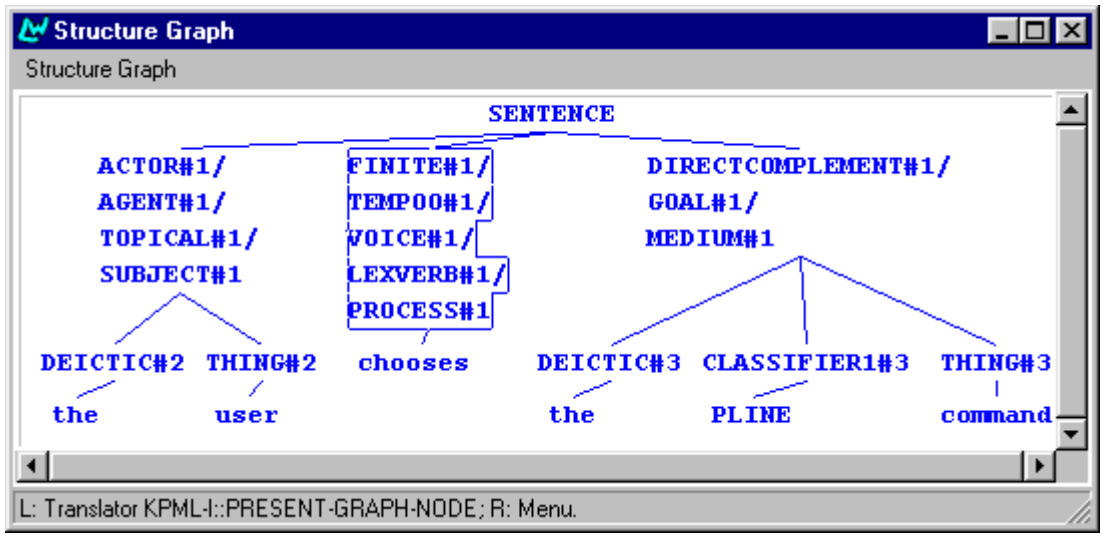
```
(p / choose
     :speechact command
     :actee (o2 / command
                :identifiability-q identifiable
                :class-ascription (c / software-command
                                       :name PLINE)))
```

Figure 5: Input to generation with KPML

Given an input such as (SPL-1) or (SPL-2) (Figure 2), the traversal of the grammatical system network is started. For English, this is the Nigel grammar (Mann and Matthiessen, 1983). At each choice point (each system), a *chooser* is invoked that checks the input representation for the semantic grounds to make a choice. A chooser is a decision procedure that is associated with a system (Matthiessen, 1988). Its task is to mediate between semantic

[6] Three kinds of keywords are defined for SPL: Upper Model or domain concepts (e.g., `choose`, `actor`), inquiries (e.g., `:identifiability-q identifiable`) and macros, which consist of a set of inquiries (e.g., `:speechact command`, `:theme o1`).

and grammatical information. A chooser is organized as a tree, the nodes of which are *inquiries*, which are the actual interpreters of semantic knowledge for the grammar. The process of choosing is applied throughout the traversal of the system network, invoking the chooser of each system, and realization statements successively build up syntagmatic structure at all ranks. The results of generation of (SPL-1) and (SPL-2) are shown in Figure 6.



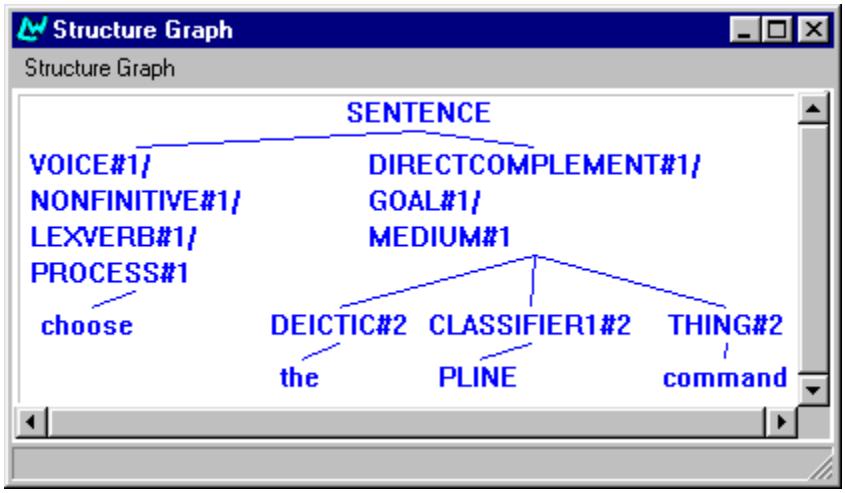


Figure 6: Output structures (clause level) for SPL-1 and SPL-2

For a systemically-based approach to tactical generation such as the one implemented in KPML, task 1 of tactical generation — the grammatical interpretation of the information contained in an input semantic representation — means choice of a set of paradigmatic features from the grammatical system network; and task 2 of tactical generation — building

up a syntactic structure — means spelling out the realization statements that are associated with the paradigmatic features.[7]

2.4 Multilingual grammar development with KPML

In the present section we describe the model of multilinguality underlying the KPML system. We present a set of dimensions of multilingual descriptions that reflect in which respects languages can differ or be the same (Section 2.4.1) and then briefly sketch the function of KPML as environment for the development of grammatical resources (Section 2.4.2).

2.4.1 Resource sharing: contrastive grammar

The representation of multilinguality in KPML is based on the observation that languages always exhibit differences *and* commonalities. Depending on the level of linguistic abstraction at which one carries out a contrastive analysis of two or more languages, one will find more or less commonality or more or less differences. However, no matter how abstract the level is one chooses for cross-linguistic comparison, there will most likely always be differences. These are hard to capture, if one assumes an interlingua at one particular level (semantic, conceptual), at which everything is supposed to be identical.[8] The method of describing multilinguality in KPML circumvents this problem by not enforcing one level of representation at which things have to be the same. Rather, several dimensions of cross-linguistic commonality *and* contrast are acknowledged and the contrastive-linguistic description can thus be just as flexible as it needs to be.

When we investigate the contrastive-linguistic properties of two or more languages using the representational categories of Systemic Functional Linguistics, we can make the observation that the grammars of different languages may be identical or different along the following dimensions:

- At the level of grammar, languages tend to be similar in terms of *systems* (functional paradigms) and different in terms of syntagmatic, surface-syntactic realization.
- Grammatical systems of low *delicacy* (grammatical types located high in the classification hierarchy) tend to be similar across language, and systems of higher delicacy tend to be dissimilar.
- There may be different preferences in different languages concerning the *rank* (clause, nominal group, prepositional phrase etc.) at which a certain phenomenon is expressed (e.g., nominally vs. verbally).
- Languages tend to exhibit more similarity on the higher *strata* of description and more differences on lower strata (e.g., more similarity on the semantic plane than on the lexico-grammatical plane).
- Languages may differ according to which *metafunction* a particular formal means serves.

---

[7] For more details on the tactical generation process in KPML see (Bateman, 1997a).

[8] This is a major problem with interlingua-based machine translation.

The representational constructs of stratification, metafunctional diversification, ranking, axiality and delicacy thus set up a space of dimensions of multilingual variation along which languages can be described as being similar or different. For a graphical overview of these dimensions see Figure 5 (adapted from (Bateman, 1995)). This is what is made use of in contrastive-linguistic representation in KPML and underlies the possibility of what is called *resource sharing*.

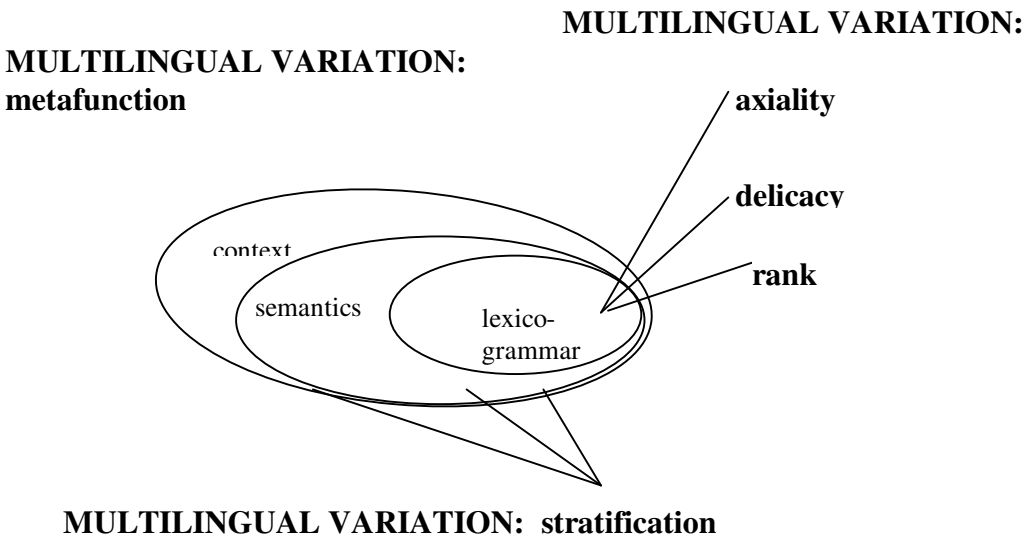


Figure 7: Dimensions of multilingual description

Given a computational specification of the grammar of a language according to rank, metafunction, axis and delicacy, and given that languages can exhibit commonality along these dimensions, one can attempt to re-use an existing grammatical description for the description of another language. Re-using an existing description will work to a large extent, especially of functional paradigms, but changes to the description taken as a basis must also be allowed for to account for language-specific features. For the languages we deal with here these are notably to do with the fact that Bulgarian, Czech and Russian count as free word order languages, and that Czech and Russian are highly inflecting.

The method of re-using an existing description of some language for the description of another language including the possibility of adapting that description is what is called resource sharing in KPML (cf. Bateman, 1995; Bateman et al., 1991b; Teich, 1995). More generally, the strategy of building up grammatical descriptions on the basis of existing ones has been referred to as *transfer comparison* in the literature (Halliday *et al.*, 1964).

2.4.2 KPML as a development environment for multilingual grammars

The KPML development environment implements a series of tools that support resource sharing in the sense described above. Multilingual resource development is particularly supported in that multilingual descriptions are constructed around the paradigmatic, functional part of the grammar rather than the syntagmatic, surface-syntactic one. Thus, commonalities among languages can be made use of, even between typologically rather different languages, while at the same time allowing the accommodation of cross-linguistic differences. Thus, potentially the effort of writing grammars for new languages is reduced.

As mentioned above, the KPML environment is an extension of the Penman system (Penman, 1989). It goes beyond that system notably in terms of multilinguality, but also in terms of easier handling and better development support. Among the added functionalities for enhanced development support are the following features (cf. Bateman, 1997a):

- test suites are interlinked with resource definitions, thus providing the possibility of looking at the resources either from the instance perspective (a string) or from the grammar perspective (the system network);
- debugging is generally graphically driven;
- grammatical resources are highly modularised (monolingually and multilingually);
- extensive graphical and textual inspection of all aspects of the grammatical resources and their use are provided (for an example of one of the graphing facilities see Figure 8).

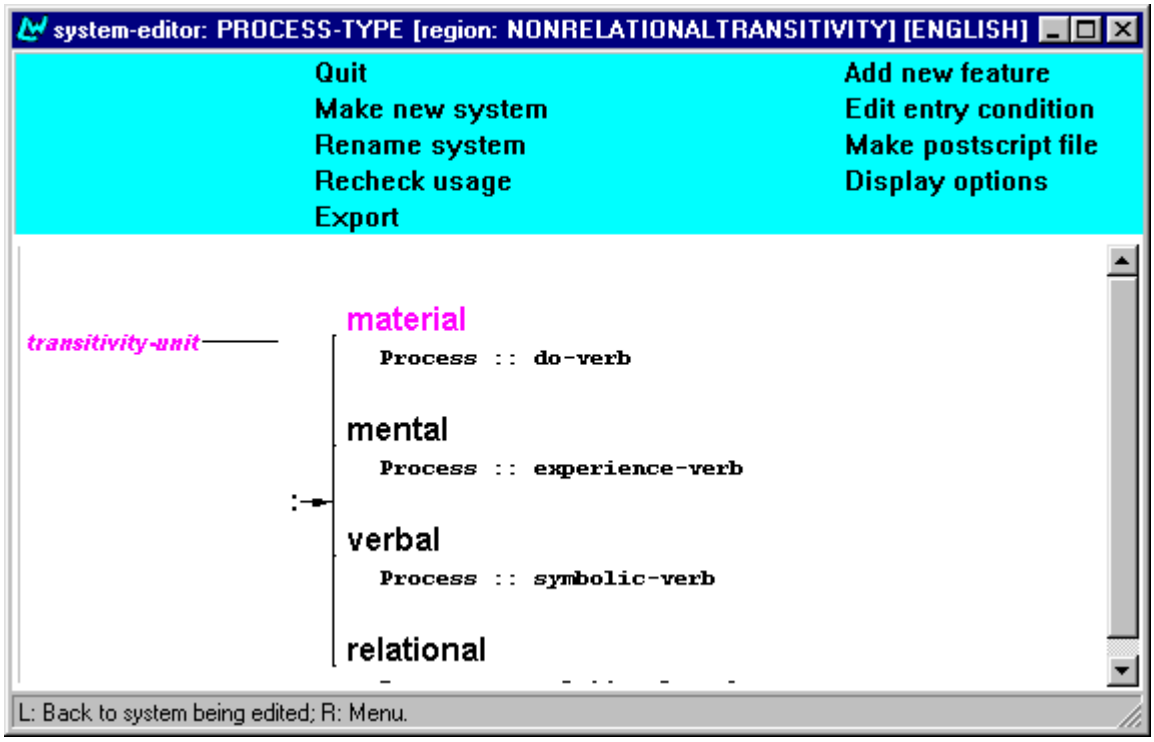


Figure 8: Graph of a system network having used the INSPECT option in the development window

KPML is currently being used by a number of researchers in natural language generation and is continuously refined according to the requirements of its users.

Before proceeding to the description of the linguistic specification and implementation of the phenomena that have been under focus in tasks 6.3 and 7.3, some comments about the organization of work in Work Packages 6 and 7 are in place. The particular organization of work adopted for WPs 6 and 7 throughout their duration was only made possible because of the method of resource sharing made available through KPML, which allowed not only to build upon its grammar of English, Nigel, but also to distribute development efforts

according to linguistic phenomena rather than according to individual languages. A truly contrastive-linguistic method of work was thus made possible.

2.5 Organization of work in Work Packages 6 & 7

Work packages 6 and 7 were divided into three phases of linguistic specification and implementation, Wp6 feeding into WP7. Phase I resulted in the Initial Demonstrator (month 6), phase II contributed to the Intermediate Prototype (month 18) and phase III contributed to the Final Prototype. Each phase was accompanied by at least one working meeting of a week at one partner site, which provided the opportunity to discuss problem areas of linguistic specifications and complementing implementations.

Phase I: tasks 6.1 and 7.1

The practical goal of the first phase of linguistic specification and implementation was to cover the linguistic phenomena of the target texts of the Initial Demonstrator. The strategy of resource sharing based on the English grammar Nigel was followed rather closely in the first phase of resource implementation. In effectively three months[9], sentences of the complexity occurring in the first set of sample texts in Bulgarian, Russian and Czech could be generated (see Appendix II of IMPL1). This was only possible because we could make use of the strategy of resource sharing as adopted in KPML and employ the support tools offered by KPML for applying such a strategy. Minimal changes were made to the English grammar, e.g., adopting particular system networks, such as e.g., the MOOD system, and only changing realization statements. In this first phase, work was distributed according to language, i.e., the Bulgarian partners worked on Bulgarian, the Czech partners on Czech, the Russian partners on Russian.

Phase II: tasks 6.2 and 7.2

After an analysis of a corpus of Bulgarian, Czech and Russian instructional texts (reported on in the CORP deliverable), we had a clear idea of what kinds of grammatical phenomena had to be dealt with in the project in order to be able to generate instructional texts in different styles. In order to avoid a sublanguage-bias, we merged this list of phenomena with the functional regions that are established for systemic functional grammars in KPML (see Table 1 given below). We thus had a basis for a method of linguistic description and resource implementation that was *instance-oriented* (corpus analysis of texts of the given register) and *system-oriented* (keeping in view the whole grammatical system) at the same time.

[9] Work on task 7.1 started only in the second week of February 1998 with the kick-off workshop in Prague. For another month the participants in WP7 only had a stand-alone image of KPML and various grammar exploration tools at their disposal. During that time, KPML was ported to Harlequin Lisp; the Harlequin version was distributed in mid-March 1998.

		Ideational			Interpersonal	Textual
		Logical		Experiential		
Clause		complexity		Transitivity Circumstance	Mood, Polarity, Attitude, Modality, Tense	Theme, Culmination, Conjunction, Voice
groups/phrases	prep			Minor transitivity		
	nom		Meta-actant	Nominal-type, Epithet, Qualification, Selection	Person, Attitude	Determinat-ion
	adj.		Modi-fier	Quality-type		
	quant		Modi-fier	Quantity-type		
	adv.		Modi-fier	Circumstantial	Comment	Conjunctive
		complex	simplex			

Table 1: Functional regions in KPML grammars

Also, at the end of the first phase the limits of resource sharing with English started to become apparent. In particular, in Slavonic languages we encounter a number of rather complex agreement phenomena and a rather flexible word order, which is determined pragmatically in the first place. Also, the concept of aspect is rather specific to Slavonic languages and had not been dealt with explicitly in the Nigel grammar. Given that according to standard language typology Bulgarian, Czech and Russian belong to the same group of languages, it appeared desirable to try to effect resource sharing among the three languages under investigation in the very process of grammar development. Therefore, in the second phase of WP 6 and 7, we divided up the work involved in tasks 6.2 and 7.2 across the partners according to linguistic phenomena and functional regions (transitivity, circumstance, qualification, classification etc.), paying special attention to linguistic features specific to Slavonic languages, such as word order and agreement. For example, the Czech partners would deal with word order, using Czech as the primary language to test their specifications and implementations on and then use the partners as informants about Bulgarian and Russian and adapt their account according to the requirements of the other two languages. Subsequently, the computational specifications were exchanged among the partners and integrated into their grammars at their sites.

Table 2 below reproduces the table of functional regions given above, this time highlighting the regions under focus in grammar specification and implementation in Work

Packages 6 and 7. Responsibilities of the partners are given in brackets. The regions not highlighted have also been treated, however only with respect to the requirements of the target texts not with a view to entire grammatical systems.

The practical goal of this second phase was to describe the linguistic phenomena occurring in the target texts for the Intermediate Prototype. The syntax in these texts was more varied than previously, which meant that other, previously not treated linguistic phenomena had to be dealt with (cf. TEXS2 for the kinds of target texts for the Intermediate Prototype). Again, rather than just being driven by the requirement of being able to generate the target texts and treating the phenomena that occurred in these texts only, we proceeded in such a way that a linguistic coverage would be achieved that was system-based, considering the three language systems as a whole rather than just the sublanguage.

<table>
<tr><td colspan="3"></td><td colspan="3">Ideational</td><td>Interpersonal</td><td>Textual</td></tr>
<tr><td colspan="3"></td><td colspan="2">Logical</td><td>Experiential</td><td></td><td></td></tr>
<tr><td colspan="2">Clause</td><td rowspan="7">Complexity (clause and nom; CU)</td><td></td><td>Transitivity, Circumstance,
Aspect
(RRIAI)</td><td>Mood, Polarity, Attitude, Modality, Tense
(BAS)</td><td>Theme, Culmination, Conjunction,
Determinat-ion
(CU)
Voice
(RRIAI)</td></tr>
<tr><td rowspan="5">g r oup s/ph r a s e s</td><td>prep</td><td></td><td>Minor transitivity
(RRIAI)</td><td></td><td></td></tr>
<tr><td>nom</td><td>Meta-actant</td><td>Nominal-type, Epithet, Qualification, Selection
(CU)</td><td>Person, Attitude
(CU)</td><td>Determinat-ion
(CU)</td></tr>
<tr><td>adj</td><td>Modi-fier</td><td>Quality-type</td><td></td><td></td></tr>
<tr><td>quant</td><td>Modi-fier</td><td>Quantity-type</td><td></td><td></td></tr>
<tr><td>adv</td><td>Modi-fier</td><td>Circumstantial</td><td>Comment</td><td>Conjunctive</td></tr>
<tr><td colspan="2"></td><td>complex</td><td colspan="4">simplex</td></tr>
</table>

PLUS: **word order** and **agreement** (both: CU); interfacing external **morphological components** (RRIAI). Note that determination has been placed at clause level as well because in Russian and Czech it is also clause phenomenon. Aspect has been placed in the experiential metafunction at clause level.

Table 2: Functional regions under focus in AGILE

Phase III: tasks 6.3 and 7.3

In the first two phases of the project, tasks 6.1, 6.2, 7.1 and 7.2 were dealt with consecutively, 6.1 and 6.2 serving as input to 7.1 and 7.2, respectively. Given that after the second phase reasonable-sized grammar components for all of Bulgarian, Czech and Russian already existed and in order to carry out tasks 6.3 and 7.3 as efficiently as possible, we decided to carry them out as one combined task. The way of work distribution adopted in phase II was continued for the combined task 6.3/7.3, partners being responsible for the description and implementation of particular phenomena rather than for their own language.

The practical goal of this phase was to refine the existing implementations in terms of language-specific requirements and extend the resources according to the need for generating the target texts for the Final Prototype. Special attention was paid to the refinement of linguistic specifications and to the phenomena specific to Slavonic languages, such as aspect, word order and agreement. For the latter two, the capabilities of KPML had to be extended.

In summary, the adopted strategy of work distribution according to linguistic phenomena rather than individual languages stimulated some productive phases of genuine contrastive-linguistic analysis and shared implementations — even if this kind of work distribution was harder to carry through than a simple distribution according to language because it required setting up interaction structures that supported frequent exchanges and continuous information flow between partners, including, for instance, weekly reporting to work package coordination, and the discipline to adhering to these structures.

3. Linguistic specification and implementation of grammatical resources for the Final Prototype

This part of the deliverables describes the extensions to the grammars of Bulgarian, Czech and Russian made during phase III of linguistic specification and implementation in tasks 6.3 and 7.3 of WPs 6 and 7.

Below we provide a brief synopsis of each phenomenon dealt with. Each section deals with one phenomenon or region and starts with some relevant theoretical considerations and then moves to linguistic specification and implementation.

Transitivity (Section 3.1). In the grammar of English, the participants of the process and its circumstantial elements are distinguished by different realization constraints. Participants are realized as nominal groups, while circumstantial elements are realized as prepositional phrases. In this deliverable, we consider the border situations where a concept that is a semantically a participant is realized by a prepositional phrase. We describe some changes in the domain model and apply more delicate systems to realize the domain model concepts adequately for the three lexico-grammars. We also consider the case where functional elements are treated as Circumstantials, but can be realized by means of either nominal groups or prepositional phrase.

Support verbs (Section 3.1). The phenomenon of support verbs is related to the lexical means available in a language, cultural traditions and the style of speech. The register of software manuals does not contain a significant proportion of support verbs, but they occur in the Russian corpus. We consider just one type of support constructions where the verb hardly contributes anything to the process semantics, but merely serves as a placeholder for

some syntactic features such as tense, number, etc. (e.g., Russian: *производить автоматическую вставку* — English: *to carry out an automatic insertion*).

Circumstantials (Section 3.1). We present a more elaborate account of spatio-temporal prepositional phrases in Slavic languages. Two phenomena distinguishing Slavic languages from English are considered. The first is that instead of distinguishing three types of locations according to the number of dimensions implied, Slavic languages distinguish only two. Second, this distinctions come with different realization patterns. In Bulgarian realization is by choice of preposition, but in Czech and Russian also case is used, e.g., for expressing the difference between motion and rest processes and for expressing the additional, newly added opposition of contact vs. vicinity

Aspect (Section 3.2). For the treatment of aspect we consider two oppositions. One concerns the process as repeated/not repeated. The other consider the process as a sequence of phases including the state preceding the process and the resulting state after the process is finished. Also, we discuss the choice of aspect in relation to styles of discourse (e.g., Russian and Bulgarian use imperfective for the impersonal style. whereas Czech prefers perfective).

Modality (Section 3.3). Modality belongs to the interpersonal resources of grammar, expressing a speaker's assessment of a process in terms of its definiteness. Modality is closely related to polarity. While each Finite verbal operator has positive and negative forms, modality allows us to refer to intermediate degrees between the positive and negative poles, thus introducing various levels of indeterminacy. Instructional texts by nature do not exhibit the full spectrum of modality options: The most frequent modal expressions occur in explanations about the possibilities available to the user in a particular software tool or reflect necessity (e.g., in preconditions or warnings). For the Final Prototype we focus on the generation of modal clauses expressing possibility, since this is the prevalent type of modality in the target texts.

Subject dropping (Section 3.4). Czech and Bulgarian are characterized as so-called pro-drop languages. This characteristic reflects the fact that the Subject in a declarative clause (indicative or interrogative mood) can be unexpressed/unrealized on the surface. This can be treated as the Subject either being left implicit or realized by a zero string. For the sake of comparison, we present a formalization of both views. In the implementation we have developed so far, we adopt the surface deletion approach.

Word order (Section 3.5). Slavic languages are known as free word order languages. Word order is primarily pragmatically determined. We present a word ordering algorithm that is much more flexible than the one previously used in the Nigel grammar, taking into account some notions established by the Prague School (communicative dynamism, contextual boundness/nonboundness) and show how this harmonises with the grammatical categories implemented in the Nigel grammar, notably the category of Theme.

Textual conjunction (Section 3.6). The textual conjunctions under focus here concern the marking of sequence (conjuncts such as 'firstly', 'secondly', …and so on, as far as 'now', 'then', 'finally'). We consider different styles of sequence presentation: unmarked vs. linguistically marked (with textual conjunctions) vs. numerical lists. For the Final Prototype, we focus on one type of discourse markers representing sequence called temporal sequence regulation.

Agreement (Section 3.7). In Slavic languages, agreement phenomena can be found at various syntactic levels. In this report we consider agreement at clause level (Subject-Finite and Subject-Predicator agreement) and at nominal group level (Deictic-Quality-Thing agreement).

Quantification (Section 3.8). The Quantification constructions we consider here are of two types: cardinals, for example, *one, two, three point*(*s*), and quantity selection, for example, *one of the following methods*. The latter requires a different treatment of Bulgarian from Russian and Czech, which again shows that Bulgarian is more similar to English than to Czech or Russian in many respects.

Clause complexity (Section 3.9). We provide detailed formal specifications concerning those types of clause complexity which occur in the target texts of the Final Prototype. This involves a brief recapitulation of the general specification of clause complexity adopted from SFG, and a detailed specification concerning the types of clause complexity covered in our grammars. The grammars handle: paratactic extension (addition and variation type), paratactic enhancement (Result and Temporal Sequence Circumstantials), hypotactic enhancement (Manner, Purpose and Conditional circumstantials). We pay special attention to the additional coverage of clause complexity that is offered in the Final Prototype grammars in comparison to the grammars for the Intermediate Prototype. Secondly, we describe the implementations of the clause complexity region in the Final Prototype grammars, and we present examples of the generated sentences.

3.1 Transitivity

In this section we detail out the description and implementation of transitivity features for Slavic languages we started in deliverables SPEC1 and SPEC2. Transitivity features as types of processes and circumstances were described in the SPEC2 and IMPL2 deliverables. In this chapter we address the phenomena relevant for the Final Prototype, including the realization of process participants by prepositional phrases. We describe the cases where prepositional phrases are treated as participants in DM and present a systemic description for the three Slavic languages for spatial locations realized by prepositional phrases (PPs) which are the most important type of circumstantial elements of the clause structure for our texts (Section 3.1.2). We also describe support verb constructions, i.e., constructions in which the main verb primarily serves as carrier of syntactic information (such as tense, person, number, gender) and the argument of that verb is the main carrier of the process meaning (often realized as a nominalization) (Section 3.1.3). The use of support verbs constructions is related to socio cultural traditions and relevant more for Russian and less for Bulgarian and Czech on our text type.

3.1.1 Theoretical background

The Nigel grammar of English separates participants of the process from circumstantial elements by assigning them different realization constraints. All participants are realized as nominal groups (NGs), while circumstantial elements are realized as PPs. Halliday (1985) treats PPs as contracted forms of clauses, while NGs are extended words (cf. Ch. 6.1 and 6.5). A typical nominal group contains elements that add more information to the description of an object denoted by its Thing element (Head), while a PP functioning at clause rank adds information about the circumstances in which a process takes place. For this reason, the nominal group that belongs to a prepositional phrase is called the *Minirange* element, while the preposition is called the *Minorprocess* element. Halliday gives several reasons why prepositions in English are similar to processes. His argumentation seems to be valid in Slavic languages as well:

- the possibility of mutual substitution between some verbs and prepositions:

  En: *about the trial* *concerning the trial*

  Ru: *о процессе* *касается процесса*

  Bg: *за процеса* *що се отнася до процеса*

  Cz: *o procesu* *co se týče procesu*

- the derivation of some prepositions from verbal forms:

  En: *include* <-> *including* *regard* <->*regarding*

  Ru: *включать* <->*включая* *благодарить* <-> *благодаря*

  Bg: *включвам*<->*включвайки* *благодаря* <-> *благодарение на*

  Cz: *pomoci* (help, aid) <-> *pomocí*+gen (with the help of)

  ohlédnout se -> noun: *ohled* (regard)<-> *ohledně* (regarding)

- the possibility of negation and other modifications of prepositions:

  En: *not without some misgivings* *right behind the door*

Ru: *не без опасений*	*сразу за дверью*
Bg: *не без опасения*	*непосредствено зад вратата*
Cz: *ne bez nehody*	*rovnou/hned za dveřmi*

However, the data of Slavic languages suggest that while proper circumstantial elements like spatial locations are realized as prepositional phrases (similar to English), potential participants can be expressed either as nominal groups or prepositional phrases (see examples (1) and (2) below). We account for this possibility as a choice in the realization of the addressee as a nominal group or a prepositional phrase at a level of delicacy above the choice between participants and circumstances.

(1) Ru: *обратиться к пользователям*

turn to users-Dat

approach the users (with a suggestion)

(2) Cz: *klepnout na tlačítko*

click on button-Acc
click a button

Alternatively, the Addressee can be realized as a nominal group using the dative case as in (3) or accusative case as in (4):

(3) Ru: *справка подсказывает пользователю*

hint-Nom suggest-Sg-3 user-Dat
a hint tells the user

(4) Ru: *сообщение информирует пользователя о…*

message-Nom inform-Pl-3 user-Acc about…
a message tells the user when…

The choice of the realization of the Addressee depends on the verb type. Also, examples (1)-(4) suggest that the difference between participants and circumstances in many cases constitutes a cline: (1) is rather a circumstance, because it can be replaced by another destination specification, for example, "*обратиться в общество пользователей комьютеров*" (to apply to the computer users society), while (3) and (4) are rather participants, because they are realized by Addressee and Directcomplement functions.

The Nigel grammar deals with similar cases in English by adding markers to NGs: *They were created by us; I told a story to her* (*by* and *to* are treated as markers, which are put in front of NGs). In the Intermediate Prototype, we inherited this representation, so empty prepositions that govern dative or instrumental case were used. We follow the same technique in the Final Prototype too, since the delicate realization of Addressee is not required.

3.1.2 Government patterns and circumstantials

To cover some more of the relevant realisations of process participants in the Final Prototype, we need some extensions of the domain model (DM) concerning process type. In the Intermediate Prototype processes were classified as user-action which mapped onto the Upper Model (UM) concept of dispositive-material-action. For the Final Prototype we need the DM to distinguish two subclasses of user-action: one which is directed and another one

which is nondirected, the USER-ACTION-DIRECTED corresponding to the old user-action and USER-ACTION-NONDIRECTED representing processes that are similar to UM motion processes. Their main characteristic is the lack of the Actee slot and the implication of a source or destination from which or towards which the process extends. These changes are needed both to facilitate the text planning of overviews and functional descriptions, as well as to enable us to generate the correct realizations of participants and circumstances in our languages.

Prepositional phrases with no clear circumstantial meanings can be regarded as circumstances or participants depending on the semantics of the predicate and on the context. For example, the preposition "*c*" in Russian or "*s*" in Czech, normally signaling a circumstantial meaning, can also express a relation between a process and an inherent participant in that process, for example,

(5) Ru: *соединить начальную точку с конечной точкой*
connect start-Adj point-Acc with end-Adj point-Ins

to connect the start point with the end point

(6) Cz: *spojit počáteční bod s koncovým bodem*
connect initial point-Acc with final point-Ins

to connect the start point with the end point

Here, *start point* (Ru: *начальную точку*) and *end point* (Ru: *конечной точкой*) are participants of the process *connect* (Ru: *соединить*). The participants can also be presented as conflated in one functional element, a Goal, as in 1 below.

(7) Ru: *соединить (эти) точки*

connect (these) points-Pl-Acc

to connect the points

Considering destination as one of features realizing spatial-locating, we also note the fact that destination**,** signaled by the prepositions Ru: *к,* Bg: *към,* Cz*: k*+Dat*, na*+Acc*, do*+Gen*,* realizes also participant roles of some concepts of the DM. This concerns for example the following two concepts, which have an Actee and a Destination slot: DM::SNAP and DM::APPLY-PROPERTY, see examples (8) and (9) below. We added a DESTINATION slot in the concepts SNAP (snap something to something) and APPLY-PROPERTY (apply something to something).

(8) …*so the arc snaps to the end point of the line.*

(a) Ru: … *чтобы привязать дугу к конечной точке линии,*
in order to attach arc-Acc to end point-Dat line-Gen

(b) Bg*: за да прикачите дъгана към крайната точка на линията*
in order to attach arc to end point of line

(c) Cz: *pro přichycení koncového bodu oblouku k čáře,*
for snapping end-Adj point-Gen arc-Gen to line-Dat

(8) expresses a relation similar to that of connection (note that no vicinity is meant here, an attachment implies a link). Also, the Russian preposition " *к* " can express the target of an action**:**

(9) *to apply a style to an item*

(a) Ru: *применить стиль к элементу*
apply style-Acc to item-Dat

(b) Bg: *прилагам стила към елемента*
apply style to item

(c) Cz: *aplikovat styl na element*
apply style-Acc to element-Acc

The DESTINATION slot is used with two other domain concepts: DM::RETURN-TO and DM::SWITCH-MODE. In Slavic languages these concepts are expressed by verbs having no Actee role, so we made RETURN-TO and SWITCH-MODE subconcepts of USER-ACTION-NONDIRECTED:

(10) *Switch to the X mode*

(a) Ru: *перейти в режим X*
go into mode-Acc X

(b) Bg: *Превключвам на режим X*
switch into mode X

(c) Cz: *Přepnout do režimu X*
switch into mode-Gen X

(11) *Return to the X mode*

(a) Ru: *вернуться в режим X*
return-Refl into mode-Acc X

(b) Bg: *Връщам се в режим X*
Return-Refl into mode X

(c) Cz: *Vrátit se do režimu X*
return-Refl into mode-Gen X

As examples (8)-(11) illustrate, in Russian the concepts having Actee realise the role by the **destination** and **vicinity features** (see Section 3.1.2.1 below) and the concepts having no Actee role realise it by **destination** and **contact**. The choice between vicinity/contact is controlled in the following inquiry code:

```
(defun REACHING-PROCESS-Q-CODE (locativerelation place)
  (let ((parent (term-graph-parent (SYMBOL-VALUE place))))
    (if (or
         (term-type-p parent 'dm::snap)
         (term-type-p parent 'dm::apply-property)
         (fetch-feature 'reaching-process-q place))
       'nonreaching
       'reaching)))
```

The structure produced by the SPL given below is shown in Figure 9:

```
(x / DM::SNAP :speechact imperative
       :actee (y / DM::ARC)
       :destination (z / DM::POINT)))
```

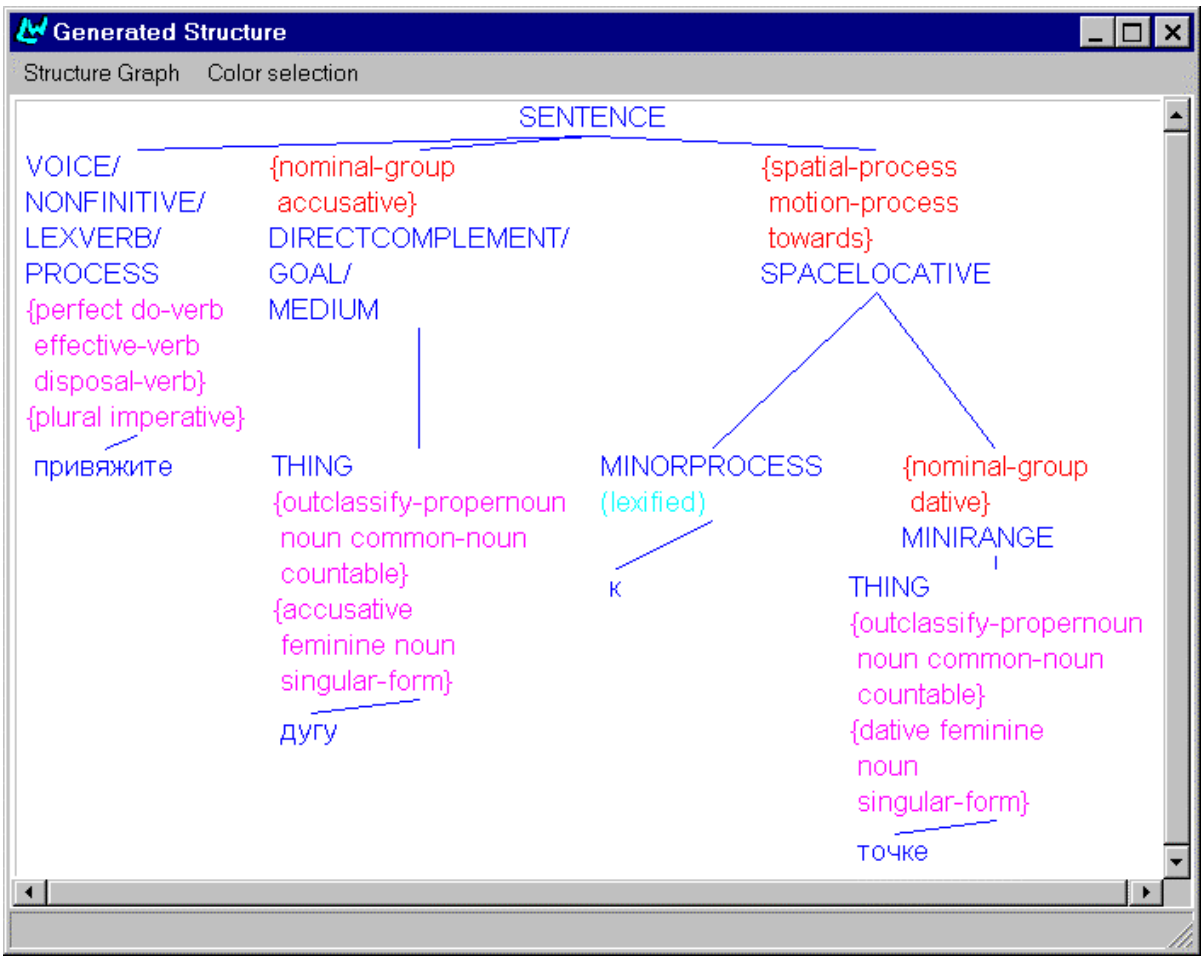


Figure 9: Government pattern realization

The realisation of DM::SAVE is quite interesting in the three Slavic languages, because in addition to the Actee role, there is an additional spatial-locating role, which, however, differs in the three languages. While Russian and Bulgarian uses a nonorienting location (corresponding to the English "save in a file"), Czech uses a destination (corresponding to the English "save into a file"). We handle this discrepancy among the languages as follows: We model the concept SAVE as having two slots in the DM, namely an ACTEE and a TARGET slot. The TARGET slot is then appropriately translated into either nonorienting location (Ru and Bg) or destination (Cz) in the slot-mapper within the splizer.

(12) *Save the file X in/into the directory Y*

(a) Ru: *сохранить файл X в директории Y*
Save file X in directory-Loc Y

(b) Bg: *Запазете файла X в директория Y*
save file X in directory Y

(c) Cz: *Uložte soubor X do adresáře Y*
Save file-Acc X into directory-Gen Y

3.1.2.1 Realisation as Prepositional phrase vs. Nominal group

This is a variation that is presented in the Final Prototype for Means circumstances, which can be expressed in Czech and Russian by a nominal group in the instrumental case:

(13) *By/with the OFFSET command you can create copies.*

(a) Ru*: Командой OFFSET Вы можете создать копии*
Command-Ins OFFSET you-Nom can create copies-Acc

(b) Cz*: Příkazem OFFSET můžete vytvářet kopie*
Command-Ins OFFSET can-Pl-2 create copies-Acc

(c) Bg*: С командата OFFSET можете да създадете копие*
With command OFFSET can-Pl-2 create copies

or by a prepositional phrase with the preposition *с помощью-(by means of)*:

(14) *By means of the OFFSET command you can create copies.*

(a) Ru:*С помощью команды OFFSET Вы можете создать копии*
With-help-of command-Gen OFFSET you-Nom can create copies-Acc

(b) Cz: *Pomocí příkazu OFFSET můžete vytvářet kopie*
With-help-of command-Gen OFFSET can-Pl-2 create copies-Acc

(c) Bg: *С помоща на командата OFFSET можете да създадете копие*
With-help-of command OFFSET can-Pl-2 create copies

The only context where we generate MEANS realized by a nominal group in Instrumental case in the instructional texts generated in the Final Prototype is the context where the MEANS refers to methods in an abstract way (like in "using one of the following methods"). In all other cases of concrete instruments (functional objects– button or similar) we generate the realisation using a prepositional phrase (like in "using a hardware tool", "using a command"). The distinction is based on domain concept types.

In the area of spatial PPs, the choice of a particular preposition in English underlies a distinction in the dimensionality of the object that realizes the range of the relation expressed by the PP. For both PPs expressing a location and PPs expressing motion, English distinguishes between three-dimensional objects (*in, into*), one-or-two-dimensional objects (*on, onto*) and zero-dimensional objects (*at, to*). This is formally expressed by the following (simplified) system network:

```
LOCATION-DIMENSION
(spatial-process) →
   [zero-dimension]
   [one-two-dimensions]
   [three-dimensions]

LOCATION-STATE
(nonphoric-place) →
   [rest] (spacelocative:rest-process)
   [motion] (spacelocative:motion-process)

MOTION-DIRECTION
(motion) →
   [towards-motion] (spacelocative : towards)
   [away-from-motion] (spacelocative : away-from)

AT
(zero-dimension, rest-process) →
   [at] (minorprocess ! at)

FROM
(away-from) →
```

```
   [from] (minorprocess ! from)

IN
(containment-implicit) →
   [in] (minorprocess ! in)

INTO
(towards, three-dimensions) →
   [into] (minorprocess ! into)

ON
(one-two-dimensions, rest-process) →
   [on] (minorprocess ! on)

ONTO
(one-two-dimensions, towards) →
   [onto] (minorprocess ! onto)

TO
(zero-dimension, towards) →
   [to] (minorprocess ! to)
```

Basically the same system is valid for Slavic languages as well. The systemic choice network above works for Bulgarian without alterations, except for lexicalization of the minor process:

```
AT
(zero-dimension, rest-process) →
   [at]     En:   (minorprocess ! at)
            Bg:   (minorprocess ! pri)

FROM
(away-from) →
   [from]   En:   (minorprocess ! from)
            Bg:   (minorprocess ! ot)

IN
(containment-implicit) →
   [in]     En:   (minorprocess ! in)
            Bg:   (minorprocess ! v)

INTO
(towards, three-dimensions) →
   [into]   En:   (minorprocess ! into)
            Bg:   (minorprocess ! vutre_v)

ON
(one-two-dimensions, rest-process) →
   [on]     En:   (minorprocess ! on)
            Bg:   (minorprocess ! na)

ONTO
(one-two-dimensions, towards) →
   [onto]   En:   (minorprocess ! onto)
            Bg:   (minorprocess ! vurhu)

TO
(zero-dimension, towards) →
   [to]     En:   (minorprocess ! to)
            Bg:   (minorprocess ! kum)
```

However, Czech and Russian use morphological case marker for expressing the difference between motion and rest processes in realization of NGs in addition to prepositions. Thus, zero-or-three dimensional objects (realized with the prepositions *в* in Russian and *v*+loc in Czech) are opposed to one-or-two-dimensional ones (*на* Ru, *na*+*loc Cz*); *в* or *на*+prepositional case (*v*+loc *na*+loc Cz) is selected for locations, and *в* or *на*+accusative case (*do*+gen *na*+acc Cz) for a movement towards an object (entering or reaching the object, which is the ultimate destination of the movement), and *к*+dative case (*k*+dat Cz) is selected for movement towards an object (not necessarily reaching the object). In the converse relation, i.e. motion away from an object, the same distinction shows in the preposition: (*из*+gen Ru) (*z*+gen Cz) for movement from a zero-or-three dimensional object, (*с*+gen Ru) (*z*+gen Cz) for movement from a one-or-two-dimensional object, (*от*+gen Ru) (*od*+gen Cz) for movement away from the vicinity of an object. Here, all prepositions (*из, с, от*) in Russian and Czech govern the genitive case. In the case of Bulgarian, there is no difference between prepositions expressing the motion away from object, *от* is used in these circumstances.

Thus, the system network for Russian is as follows:

```
LOCATION-DIMENSION
(spatial-process) →
   [zero-three-dimensions]
   [one-two-dimensions]

LOCATION-STATE
(nonphoric-place) →
   [rest] (spacelocative:rest-process)
   [motion] (spacelocative:motion-process)

MOTION-DIRECTION
(motion) →
   [towards-motion] (spacelocative : towards)
   [away-from-motion] (spacelocative : away-from)

V-STAT
(zero-three-dimensions, rest-process) →
   [at] Ru: (minorprocess ! v) (Minirange : prep)
        Cz: (minorprocess ! v) (Minirange : prep)

NA-STAT
{one-two-dimensions, rest-process)→
   [on] Ru:(Minorprocess ! na) (Minirange : prep)
        Cz:(Minorprocess ! na) (Minirange : prep)

TRAJECTORY-PROCESS-TYPE
(motion-process) →
   [contact]
   [vicinity]

V-MOTION
(zero-three-dimensions, towards, contact)→
   [into] Ru:(Minorprocess ! v) (Minirange : acc)
          Cz:(Minorprocess ! do) (Minirange : gen)

NA-MOTION
(one-two-dimensions, towards, contact) →
   [onto] Ru:(Minorprocess ! na) (Minirange : acc)
          Cz:(Minorprocess ! na) (Minirange : acc)
```

```
IZ
(away-from, zero-three-dimensions, contact) →
   [from] Ru:(Minorprocess ! iz) (Minirange : gen)
          Cz:(Minorprocess ! z) (Minirange : gen)


S
(away-from, one-two-dimensions, contact) →
   [from-2d] Ru:(Minorprocess ! s) (Minirange : gen)
             Cz:(Minorprocess ! z) (Minirange : gen)

K-PREPOSITION
(towards, vicinity)→
   [to-vicinity] Ru:(Minorprocess ! k) (Minirange : dat)
   [to-vicinity] Cz:(Minorprocess ! k) (Minirange : dat)

OT-PREPOSITION
(away-from, vicinity) →
   [from-vicinity] Ru:(Minorprocess ! ot) (Minirange : gen)
   [from-vicinity] Cz:(Minorprocess ! od) (Minirange : gen)
```

The TRAJECTORY-PROCESS-TYPE is shown in Figure 10 which accounts for the options in the range of the trajectory by taking information from the semantic specification of features of the process: whether it implies contacting the destination or not. The systems and gates realizing the spatial prepositional phrases simply include the appropriate feature: **contact** or **vicinity** and realize the appropriate lexicalisation and case inflection.

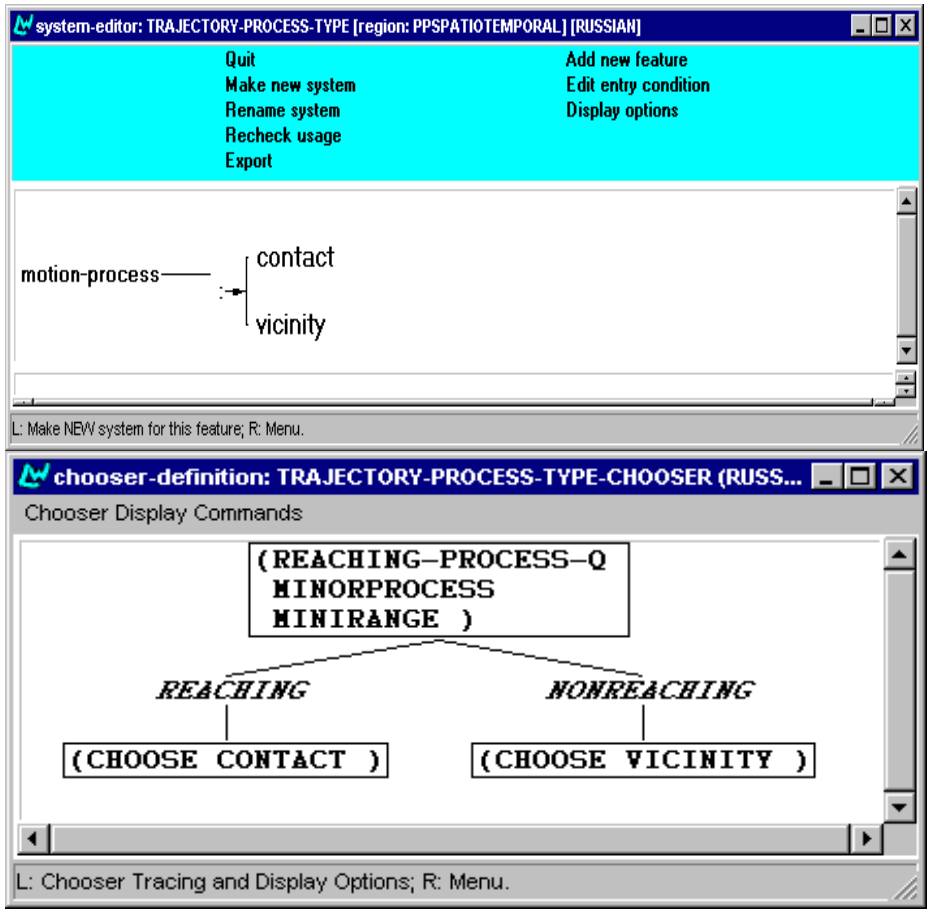


Figure 10: The TRAJECTORY-PROCESS-TYPE system and chooser.

3.1.3 Support verb constructions

3.1.3.1 Theoretical background

The phenomenon of support verbs occurs in many languages, including English and the three Slavic languages dealt with here. In traditional terms, this type of syntactic constructions consists of a support verb, which designates a generic action, and its direct object, which is the main carrier of the process meaning. Some examples of support verb constructions are given in (15).

(15)

(a) En: *He took a shower*

(b) Bg*: Той си взе душ* - (He took a shower)

(c) Cz: *provést / udělat analýzu* - (carry out / do an analysis)

(d) Ru*: Он принял душ* - (He took a shower)

Note that in this section we sometimes use different examples for different languages, when the identical ones are impossible by reason of the difference in language conceptualisations reflected in possible word combinations.

The phenomenon of support verbs is related to the lexical means available in a language, cultural traditions and the style of discourse. The register of software manuals does not contain a significant proportion of support verbs, but they do occur. There are two possibilities for treating this phenomenon. The first one is to consider support verbs as idiomatic ways to express lexical functions in the tradition of Mel'čuk's Meaning-Text model (Mel'čuk 1974). The second is to look at it under the systemic perspective considering them as quite delicate lexico-grammatical choices. The choices are aimed at expressing some situation by means of a Range participant, which further specifies or restricts the meaning of the Process rather than being a genuine participant in its own right, while the verb situates the expression within a clause structure, cf. (Halliday, 1984: 135). The Range can be more or less specific, like in English *they played games; they played a good game; they played tennis.*

We assume that there is a continuum where the verb can contribute more or less semantics to the description of an event. In this section we consider just one end of this continuum where the verb really hardly contributes any semantics at all, but merely serves as a placeholder for tense, number, etc.

Support verbs are typically used in the following situations:

- Lack of an appropriate verbal lexical item, e.g.,

(16)

(a) En: *to result*;

(b) Ru: *давать результат* – (to give a result);

(c) Cz: *dávat výsledek* (to give a result);

(d) Bg: *давам резултат* (to give a result).

- Preference of an adjective modifier for a nominal group instead of an adverbial modifier for a verbal group, e.g.,

(17)

(a) En: *to insert automatically*;

(b) Ru: *производить автоматическую вставку* –(to make an automatic insertion);

(c) Cz: *provést / udělat autmatické uložení* (carry out / do automatic saving)

- Style considerations. Support verbs constructions are preferably used in Slavic in special registers or types of texts and text elements. For example, in Russian or in Czech they often used in official or scientific texts. Because of the more formal character of these constructions, they are better used in the non-personal style in the register of software manuals than in the personal style. In Bulgarian only constructions with no nominalization are used (see example (15)). Sometimes the constructions with nominalization of the semantically full verb are used in headings in order to highlight that the text describes a sequence of actions to achieve the goal expressed in the heading.

Two possibilities occur: The nominalization fills the Subject function or it fills the Directcomplement function. Consequently, it is either in nominative case (for Ru and Cz) or in the accusative case (for Ru and Cz). Examples (15-17) illustrate the realisation with Directcomplement. Examples with Subject function for Czech and Russian are given below.

(18) *If you quit and then restart Word, the template or add-in is not automatically reloaded.*

(a) Ru: *При следующем запуске Word не происходит автоматической загрузки общих шаблонов и надстроек.*
At next-Adj startup-Loc Word-Gen doesn't perform-Sg-3 automatic-Adj loading-Nom general-Adj-Pl-Gen template-Pl-Gen and add-in-Pl-Gen

(b) Cz: *Při následujícím spuštění Word neprovádí automatické natažení šablon a add-in procedur.* (literal translation of the Russian sentence)

(19) *The automatic saving of the file is in progress.*

(a) Cz: *Probíhá automatické ukladání souboru*
progress-Sg-3 automatic-Adj-Nom saving-Nom file-Gen

The Subject function of the semantically full verb is possible in Bulgarian in reflexive construction, but rather nominalization of the support verb is used for titles:

(20) *to insert automatically*

(a) Bg: *извършва се автоматично вмъкване*
performing-Refl automatic insertion-Nom

(b) Bg: *Извършване на автоматично вмъкване*
Performing-Nom of automatic insertion-Nom

In this deliverable, we consider implementation of only one type of support verb construction - the construction with a direct complement that is realized by a nominalized verb.

3.1.3.2 Formal implementation

When a transitivity unit is realized, the first choice is between simple and complex realization of the process:

```
VERBAL-GROUP-TYPE
(transitivity-unit) →
   [simplex] (Process/Finite)
   [complex] +Support (Process : nominal-group) (Process : accusative)
```

The chooser takes into account several parameters to express a process using a complex verbal group structure, such as the process meaning, e.g., whether the process denotes a result, or the process quality, or the style. The chooser is shown in Figure 11.

There are many Russian verbs that are used as support verbs in such constructions, but only several of them are important for our purposes:

Ru: *иметь, давать, производить, происходить (to have, to give, to perform, to occur)*

Cz: *mít, dávat, provádět, probíhat (to have, to give, to perform, to occur)*

The choice may depend on characteristics of the process, as well as on lexical constraints on its realization.

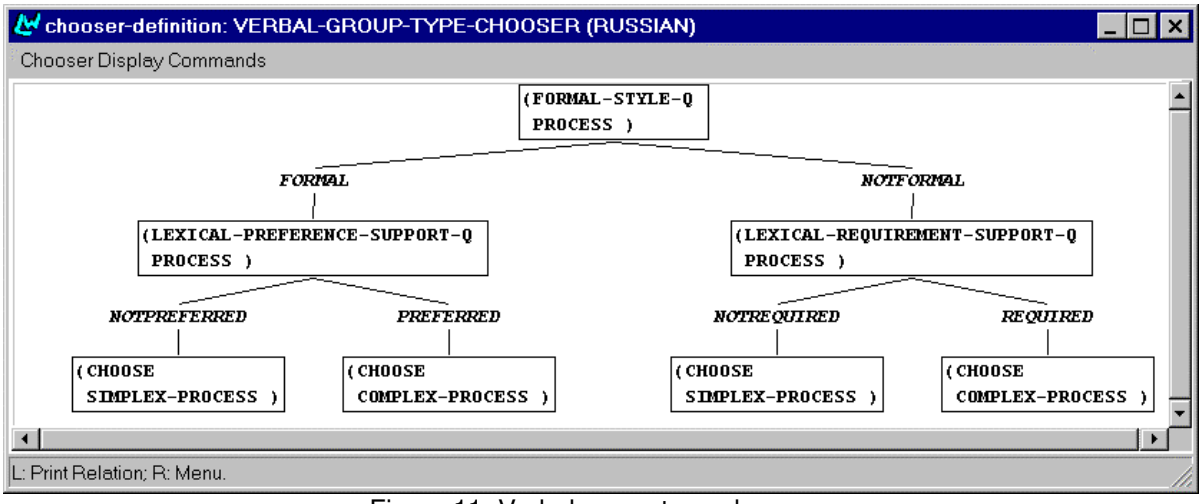


Figure 11: Verbal-group-type-chooser

```
SUPPORT-VERB-TYPE
 (complex, lexical-verb-term-resolution) →
   [static-support-verb] (Finite :: possesion-verb)
   [dynamic-support-verb] (Finite :: effective-verb)
```

```
STATIC-SUPPORT-VERB-TYPE
 (static-support-verb) →
   [reception] (Finite ! imetj)
   [giving] (Finite ! davatj)

DYNAMIC-SUPPORT-VERB-TYPE
 (dynamic-support-verb) →
   [action] (Finite ! proishoditj)
   [happening] (Finite ! proizvoditj)
```

3.2 Aspect

Existing treatments of aspect in Slavic languages in descriptive or computational linguistics typically propose a classification based on the syntagmatic contexts of a verb carrying aspect. From the perspective of natural language generation, however, we have to start from the differential *meanings* that are expressed by the category of aspect. Starting from the perspective of meaning rather than form, the present section describes the theoretical background and the specification and implementation of the ASPECT region for Bulgarian, Czech and Russian in the AGILE grammars.

The SPEC2 deliverable discussed the main linguistic properties of aspect in Bulgarian, Czech and Russian using a traditional descriptive viewpoint. Also, aspect properties were not implemented in relation to other grammatical categories existing in the computational version of our grammars. Aspect choice was more or less hard-wired with other selections in the grammar in that imperative clauses were realized with the perfective aspect in all three languages, and medio-passive clauses were realized with the imperfective aspect for Russian and Bulgarian and with perfective for Czech. This was motivated by the two styles of text dealt with in the intermediate prototype (cf. SPEC2).

In the present deliverable, we attempt to specify and implement a more general treatment of aspect starting from a meaning perspective. We then move towards a formal specification and present the implementation.

3.2.1 Theoretical background

3.2.1.1 The meanings of aspect

The aspect category in Slavic languages expresses “different ways of viewing the internal temporal constituency of an event or a situation” (Comrie, 1976), or “how an event occurs in time or how is it distributed in time” (Пешковский 1934). The aspect category in Slavic languages is a lexico-grammatical category. That is each verb belongs to one of the two classes, imperfective or perfective. Rarely the same lexical unit is used for both. The classification of lexemes in terms of aspect pair oppositions for our three languages is presented in (SPEC2, pp. 89-90). Semantically, aspect encodes a perspective on an event or process as on-going, repetitive or accomplished. According to (Падучева 1996), Vendler’s classification (Vendler, 1967), which distinguishes between states, activities, achievements and accomplishments, is relevant for the modeling of Slavic aspect. Падучева extends Vendler’s classification by the opposition of controlled vs. non-controlled processes, for example, non-controlled achievement is happening, non-controlled accomplishment is limited process - *растаять* (*to melt-perfective*), non-controlled activity is unlimited process – (*to boil*). States are subdivided into temporal states and constant states or features, as Vendler also points out, for example, *to see* (now) and *to see* (=able to see, for

example after operation returned this ability). Even though Vendler's classification is based on lexical items, and we are concerned with a classification of semantic contexts and ultimately grammatical classes, we take his classification into account. Падучева considers a process as a sequence of phases including the state preceding the process and the resulting state after the process is finished. Aspect highlights one phase of the process: the imperfective aspect highlights some middle phase of a process, while the perfective aspect highlights the state after the process was accomplished. For example, the imperfective imperative form refers the beginning phase: *chitayte* (*read*) means *nachnite chitatj* (*start reading*); the imperfective indicative form refers to the middle phase of an action in progress: *Misha chitaet* (*Mike is reading*). Apart from these general considerations, a number of specific constraints have to be taken into account:

- Since in Slavic languages the perfective aspect highlights the state after the process was accomplished it is not possible with phase verbs – *begin, continue, finish*, which highlight only a stage of the process proper, e.g.,

(21) *Begin writing*

(a) Ru: *(начинать / начать) писать / *написать*
(begin-Imperf / begin-Perf) write-Imperf / *write-Perf

(b) Bg: *(започвам / започна) пиша/*да напиша*
(begin-Imperf / begin-Perf) write-Imperf / *write-Perf

(c) Cz: *začínat / začít psát / *napsát*
(begin-Imperf / begin-Perf) write-Imperf / *write-Perf

- States (in terms of Vendler's classification) or relational processes (in terms of the Upper Model) can have no states before or after the process is finished. Thus, in Slavic languages, they can only be realized by imperfective aspect.
- An achieved state can be expressed by either a relational verb or by a verb denoting an achievement/accomplishment. For example, the fact of a button residing in a location can be expressed as a relation as in (22) or as a result of its insertion into a location as a material process in passive voice as in (23):

(22) *The X button {can be found / lies } on the Y toolbar.*

(a) Ru: *Кнопка X находится (располагается) на панели Y*
Button X finds-Refl (situates) on toolbar Y

(b) Bg: *Бутонът X се намира на функционалния ред Y.*

Button X finds-Refl on toolbar Y

(c) Cz: *Tlačítko X {se nachází \ leží } na panelu Y.*
Button X finds-Refl (lies) on toolbar Y

(23) *The X button is situated on the Y toolbar.*

(a) Ru: *Кнопка X расположена на панели Y*
Button X is-situated on toolbar Y

(b) Bg: *Бутонът X е разположен на функционалния ред Y.*
The X button is situated on the Y toolbar.

(c) Cz: *Tlačítko X { je situováno \ je umístěno} na panelu Y.*
Button X {is situated is placed} on toolbar Y

The verbs *расположить* (Ru), *разполага* (Bg:), *situovat* (Cz) (to place) in active voice express an action. However, in 23 they highlight the state that is the result of the accomplishment of the action, i.e., the fact that the button's location is on the toolbar.

In terms of systemic-choice network, this is expressed by the opposition of the features activity-highlighted vs. result-highlighted, which lead to choosing the imperfective vs. perfective aspects, respectively.

Another meaning expressed by aspect concerns the repetition or iteration of a process. In this case, the imperfective aspect expresses either one of the two meanings: habituality (real or potential) or multiple instantiation of the same process. An example of imperfective aspect use for multiple instantiations from our sample texts is given in (23).

(24) *Press r every time*

(a) Ru: *Нажимайте клавишу r каждый раз...*
press-Imperf key r every time …

(b) Bg: *Всеки път натискайте клавиша r.*
Every time press-Imperf key r

(c) Cz: *Pokaždé tiskněte klávesu r*
Every time press-Imperf key r

A repeated process is not a simple iteration of actions. A construction denoting several successive events does not mean a "repeated quality", which is required for selection of the imperfective form. For example, the imperfective aspect is hardly relevant to an explicit quantity of steps in a sequence of commands:

(25) *Double-click the mouse button*

(a) Ru: *Дважды нажмите мышь.*
click-Perf mouse twice

(b) Bg: *Щракнете два пъти с мишката.*
click-Perf twice the mouse.

(c) Cz: *Dvakrát klikněte| klikejte myší.*
Twice ClickPerf/ Click-Imperf the mouse.

We see that the style consideration (instruction-step) is here the main for the aspect choice since in another discourse, the imperfect with quantification is quite possible.

The repeated process feature depends on the lexical, syntagmatic and pragmatic context of the clause (Падучева 1996). The choice of this feature can depend on the lexical meaning of a verb (*Он ходит в магазин – He goes shopping*), or on syntagmatic constraints (*каждый раз*, *at every time*), or on styles of realization of events (*Нарисуйте*

отрезок прямой vs. *Рисуется отрезок прямой*; see the discussion below). These constraints tend to be language-specific rather than the same across Czech, Bulgarian and Russian. For example, in example (25) above, perfective is possible for Czech, while it is not possible for Bulgarian or Russian. Based on these observations, we suggest a second distinction that is relevant for the aspect choice: repeated/nonrepeated.

Generally speaking, the motivations for aspect choice are not always clearly detectable. For example, the choice of the imperfective aspect in the impersonal style in Bulgarian and Russian can be described either as a consequence of habituality or repetition or as a consequence of highlighting the middle stage of a process. In Czech, the impersonal style preferably uses the perfective form, thus highlighting the result stage. Also, there is an influence on aspect choice by differences in lexical meanings of verbs in the perfective/imperfective pair, as well as the important interaction between aspectual and tense meanings. For example, for Bulgarian and Russian the present form + perfective aspect is the form for future tense. Even though this is the form selected in the impersonal style in Czech, here, the form is not primarily chosen to express future: the motivation lies in the text style and the future meaning of the form is pushed into the background, as it were. There are also interrelations between aspectual meaning and the meanings of certain circumstantial elements. These co-occurrence constraints are often used in descriptive linguistics as tests. For example, for Bulgarian and Russian the adverb *често* (Bg) and *часто* (Ru) can be used only with the imperfective aspect form:

The motivations for a particular aspect choice are not always clearly detectable. For example, the choice of the imperfective aspect in the impersonal style in Bulgarian and Russian can be described either as a consequence of its habitual repetition or as a consequence of highlighting the middle stage of a process. In Czech, the impersonal style preferably uses the perfective form, thus (implicitly at least) highlighting the result stage.

Aspect choice in the AGILE system is controlled by constraints inserted into SPL by the Text Structuring Module, which does not take into account some delicate considerations of the Slavic aspectual system. We also do not consider the influence the aspect choice caused by differences in lexical meanings of verbs in the perfective/imperfective pair, as well as the important interaction between aspectual and tense meanings. For example, for Bulgarian and Russian the present form perfective definitely means future tense that contradicts to impersonal style semantics, but in Czech it seems that the future time semantics does not affect the situation. There are also interrelations between the aspectual meaning and the meanings of certain circumstantial elements. These co-occurrence constraints are often used in descriptive linguistics as tests for the contexts classification. For example, for Bulgarian and Russian the Adverb Ru: *часто,* Bg: *често* can be used only with the imperfective aspect verbs:

(26) *He often wrote letters*

(a) Ru: *Он часто писал (*написал) письма*
He often wrote-Imperf letters

(b) Bg: *Той често пишеше (*писа) писма.*
He often wrote-Imperf letters

(c) Cz: *Často psal dopisy (dopis).*
Often wrote-Imperf-Sg-3 letters (a letter).
He often was writing letters. (no assumption about whether a single or multiple letter(s) on each occasion)

(d) Cz: *Často napsal dopisy (dopis).*
Often wrote-Perf-Sg-3 letters (a letter).
He often wrote letters. (multiple letters on every occasion)

This is not true for Czech. In Czech, perfective aspect is used much more often. In contrast to Bulgarian and Russian it is also possible to use perfective with adjuncts denoting habiltuality or iteration, such as *často* (*often*):

In SFG, syntagmatic co-occurrence constraints like the one just mentioned above is referred as "harmony". In the present implementation, we do not model these relations, since they need extensive discourse as well as contextual models.

3.2.1.2 Motivation for the aspect choice from discourse

In this section we briefly discuss some of the discourse parameters which control aspect choice for the main predicate of a sentence. However, none of them are implemented at the moment because more sophisticated discourse representations than currently available would be needed.

The two main styles of procedural texts – personal and impersonal – require the selection of specific aspects for the delivery of the same A-box content from the domain model. In Bulgarian and in Russian, actions are realized with the perfective aspect and imperative mood in the personal style and with the indicative mood, imperfective aspect and medio-passive voice in the impersonal style. For Czech, the perfective is preferred both for the personal and impersonal styles (though the imperfective aspect is also possible in Czech in impersonal style):

(27) *Draw a line segment* vs. *A line-segment is drawn.*

(a) Ru: *Нарисуйте отрезок прямой* vs. *Рисуется отрезок прямой*
draw-Perf-Imper segment-Acc line-Gen vs. draws-Indicative-Imperf-Refl segment-Nom line-Gen

(b) Cz: *Nakreslete rovný segment* vs. *Nakreslí se rovný segment.*
draw-Perf-Imper straight-Adj segment-Acc vs. Draw-Indicative-Perf-Refl straight-Adj segment-Nom

(c) Cz: *Kreslete rovný segment* vs. *Kreslí se rovný segment.*
draw-Imperf-Imper straight-Adj segment-Acc vs. Draw-Indicative-Imperf-Refl straight-Adj segment-Nom

(d) Bg*: Начертайте отсечка.* vs. *Чертае се отсечка.*
draw-Perf-Imper line-segment vs. draw-Indicative-Imperf-Refl line-segment

Below we outline a difference between the structure of the communicative situation and the text type structure. The main communicative goal of instructions is formulated as follows: to make the user able to deal with a machine, increase his/her knowledge about it (Hartley & Paris, 1996). In this, "dealing and knowing" is considered as a generalized event to be accomplished as the result of the communicative situation. Thus, the structure of the communicative situation of instructional texts consists of the three elements:

1. Generalized-speaker;
2. Addressee (which in the case of the personal instructional texts is also the Generalised Actor presented by potentially infinite amount of Users);
3. Event of "dealing and knowing".

The Generalised speaker's goal is to enable the user to achieve the event. This structure defines conditions for interaction between the Speaker and the Addressee, thus, it pertains to the interpersonal metafunction. It also has no speech time reference, thus, it has no relation to tense and aspect. In contrast to this, the text structure is defined for particular predicates, representing some action in the A-box:

1. Author-of-the-text;
2. Addressee-of-the-text = User;
3. Observer;
4. An action.

The text structure is based on the realization of the communicative situation, according to means available in the styles and lexico-grammatical resources of a target language. In the text structure the role of Observer is included which controls the consistency of the content deployment. Style discourse characteristics are the result of the interrelation between the two structures.

In the personal style, the Author-of-the-text coincides with the Generalized-speaker role in the communicative structure model. The imperative mood of the text type just copies the structure of the communicative situation: the user is commanded to accomplish an action. The Observer role is conflated with the Generalised-Speaker role that is competent to organise the sequence of actions.

The imperfective aspect in the imperative mood highlights the beginning stage of the process representing the action. The choice of the imperfective verb for "click the button" expresses the meaning of "begin to clicking the button". Only the perfective aspect implementing the **result-highlighted** feature is appropriate here (with the exception of the rare situations where the **repeated** feature selected for the process leads to a realization in the imperfective aspect). This motivation is the same for the three Slavic languages.

In the impersonal style, the Author-of-the-text role coincides rather with the communicative role of Addressee. In this case, the intention is still that the User = Author = Observer executes a process. Or, in another perspective, the software system executes a process. These two possible perspectives sometimes cause the ambiguity (who carries out the action – the user or the system?), which makes the impersonal style not quite appropriate for instructional texts in Russian. For Czech the impersonal style is very common for instructional texts in general. So impersonal style is used for the whole text of instruction, if no ambiguity arise. Another set of A-boxes is required to be suitable for this style: Their

goal, in this case, is to provide information about the software, rather than a plan for achieving user's goals. This was the reason, why Agile evaluators objected the use of the impersonal style (EVAL1). The Slavic languages of the project differ in their ways for presenting the impersonal style. The Southern and Eastern Slavic languages (Bulgarian and Russian) use the imperfective aspect and differ from Czech, which prefers the perfective aspect (though the imperfective is also possible). Our hypothesis is that Bulgarian and Russian reflect the introspection perspective on a process in which the user acts as the Observer, whereas Czech prefers an external perspective on the process where the user and Observer roles are disjointed.

3.2.2 Aspect choice in subordinated clauses

Motivation for aspect choice in subordinated clauses is different from the motivation for main clauses discussed above. In subordinated clauses, aspect choice depends on the aspect choice in the main clause. This is achieved in the Text Structuring Module upon considering the aggregation of user actions. We have one domain model construction which is aggregated: Goal – Method. There are two possibilities

1. The Goal element of the text plan is presented as a subordinate Purpose clause:

(28) *Click Add to add this element*

(a) Ru: *Нажмите кнопку Add, чтобы добавить этот элемент.*
Press button Add in-order-to add– Inf-Perf this element

(b) Bg: *Натиснете бутона Add, за да добавите този елемент.*
Press button Add in-order-to add –Indic-Perf-Pl-2 this element

(c) Cz: *Stiskněte tlačítko Add, abyste tento element přidali.*
Press button Add sothat-you this element added-Indic-Perf-Pl-2

2. The Method is presented as a subordinate Means clause:

(29) *Add the element by clicking Add.*

(a) Ru: *Добавте этот элемент, нажав кнопку X*
Add-Perf this element by-pressing-Gerund button X

(b) Bg: *Добавете този елемент, като натиснете бутона X.*
Add-Perf this element by-conj press-Indic-Perf-Pl-2 button X.

(c) Cz: *Stiskněte tlačítko Add pro přidání elementu.*
Press button Add for adding-Acc element-Gen.

Example (29) shows that the Means clause in Slavic languages can be realized in three ways: as gerund (Ru), as finite clause (Bg) and as nominalization (Cz). In Bulgarian, all three options are possible, but the finite clause is preferred; in Russian, the gerund and the nominalization are possible, and the gerund is preferred; and in Czech, the subordinate finite clause and the nominalization are possible, and the nominalization is preferred.

Irrespective of the type of realization, the result of the subordinated process is highlighted. So, we choose the result-highlighted feature for the subordinate clause. As for the nominalization, Bulgarian does not allow nominalizations in perfective aspect, Russian allows them, but they occure rarely. In Czech, nominalization aspect pairs are rather regular. Below we present some Czech aspectual pairs for verbs and their nominalizations and the corresponding Russian nominalizations, whenever possible:

(30)

Verbs: Cz: *kreslit - nakreslit,* Ru: *risovatj – narisovatj* (draw)

Nouns: Cz: *kreslení – nakreslení,* Ru: *risovanije - *narisovanije* (drawing)

Verbs: Cz: *otevírat – otevřít,* Ru: *otkryvatj – otkrytj* (open)

Nouns: Cz: *otevírání – otevření*, Ru: *otkryvanije – otkrytije* (opening)

The aspect characteristics of a noun are used in the Czech lexico-grammatical resources, for example, for the titles the imperfective nominalizations are chosen. For Russian and Bulgarian the aspect for nominalized process is fixed by the lexical item.

3.2.3 The Aspect region specification

The realization of the aspect takes into consideration the process types. For the Slavic aspect modelling we needed to expand Vendler's classification of processes. In Nigel we only have a transitivity-oriented classification that allows choosing the configuration of participants, but this is not directly appropriate for aspect choice. But we can use it in our restricted area since we have two main types of processes in our texts – mostly material and some relational processes. So we can consider the opposition relational ("states" in Vendler's classification) vs. others ("activities", "accomplishments", "achievements") as relevant to the project purposes.

The choice of aspect is represented in a new grammar region – Aspect. The topmost choice in the region is between notrepeated and repeated. The latter directly leads to the realization of the verb as imperfective (feature: imperfect):

```
(TRANSITIVITY-UNIT) →
            [notrepeated],
            [repeated].
```

The default choice is notrepeated, while in the case of repeated processes the SPL should have: repeated-q repeated.

The second choice is based on the type of process. Relational processes are states, so they lack stages of development to be highlighted using the aspect perspective. Lexical units which function only as relational processes have no aspect oppositions and are always realized as imperfective, for example, Ru: *находиться*, En: *to be*, Bg: *намира се,* Cz: *být (be), nacházet se (be sitated).* In the formal specification, we do not consider the complex case, when an action verb can be used to denoted a relation (cf. the discussion of *расположить* above). Thus, the choice is as follows:

```
(notrepeated) →
   [aspect-relational],
   [aspect-others].
```

The third opposition takes into account whether a process is oriented itself (Vendler's activities) or is aimed towards a result (accomplishments or achievements):

```
(aspect-others) →
   [activity-highlighted],
   [result-highlighted].
```

The default choice is result-highlighted, while in the impersonal style the Text Structuring Module must add the following to the SPL:

```
:NONSTATIC-PROCESS-ASPECT-Q ACTIVITY-HIGHLIGHTED
```

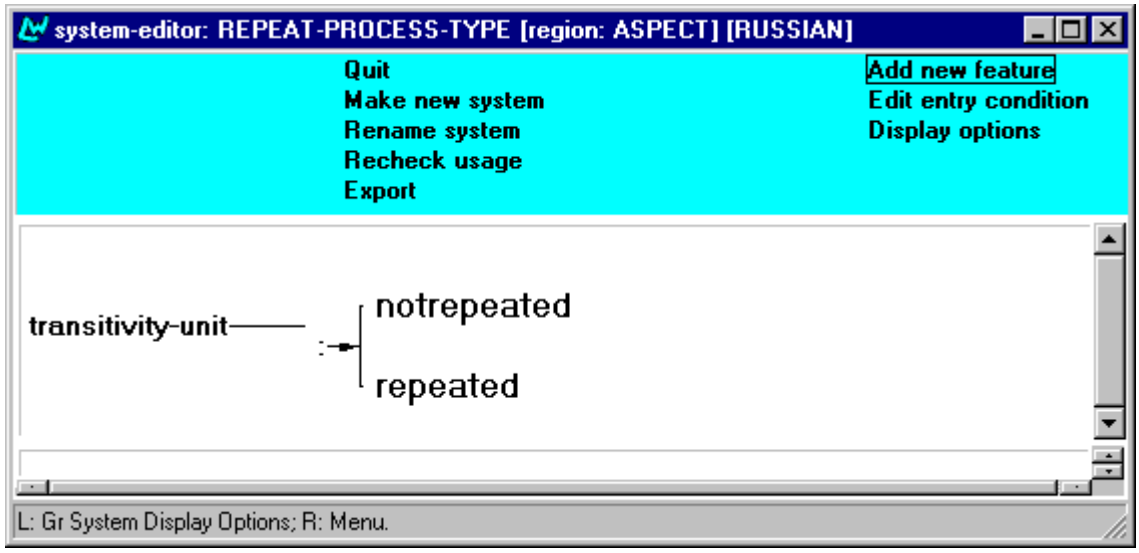
Finally, we include two gates for lexicalization and to ensure that lexical term resolution happens after the aspect meaning is selected:

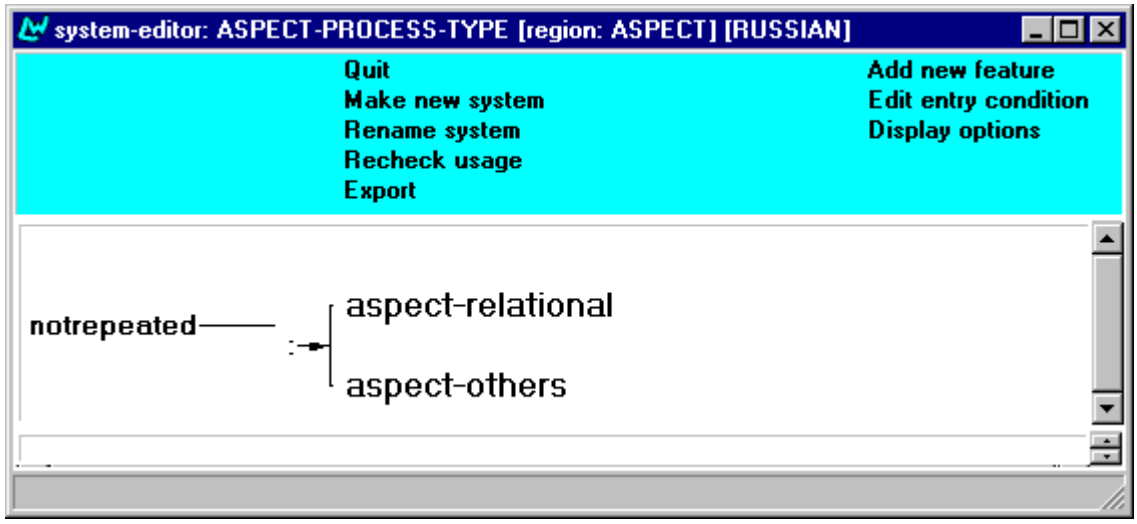
```
IMPERFECTIVE-ASPECT-SELECTION
(or repeated activity-highlighted aspect-relational) →
            [lexverb-imperfective] (classify process imperfect).

PERFECTIVE-ASPECT-SELECTION
 (result-highlighted) →
            [lexverb-perfective] (classify process perfect).
```

3.2.4 The Aspect region implementation

For the purposes of a proper aspectual description, we introduced a new region into the grammar: ASPECT region. This region is shown in Figure 12.





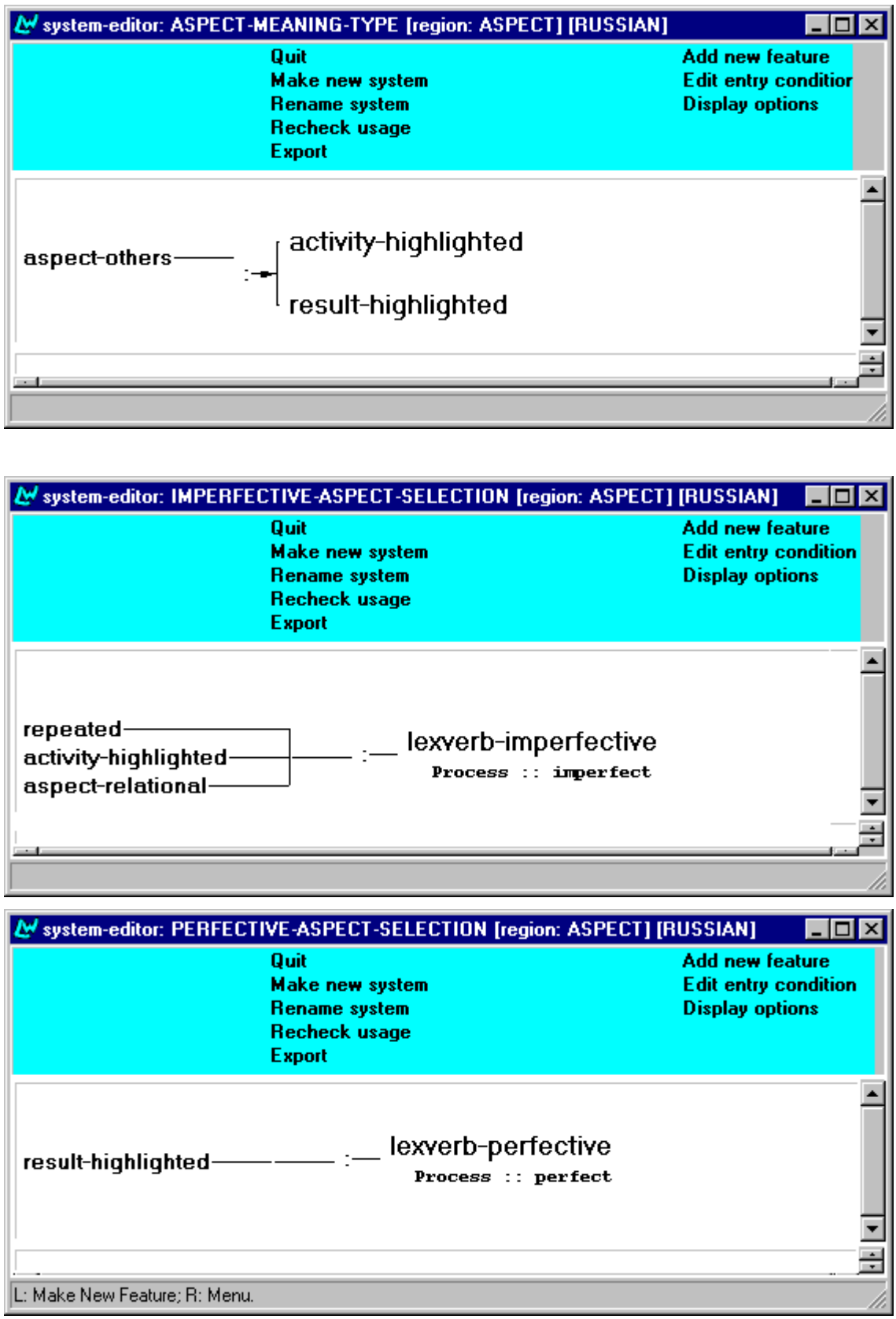
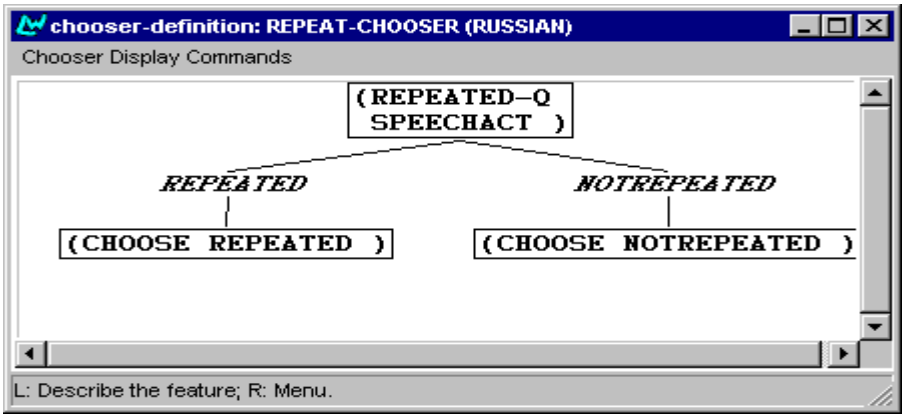
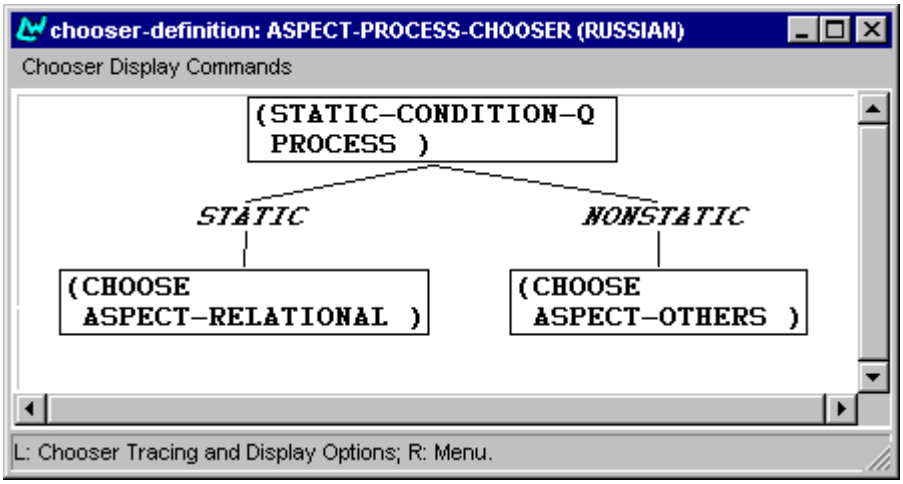


Figure 12: The Aspect region of grammar.

Choosers of the Aspect region are shown in Figure 13: Choosers for the Aspect region.





The ASPECT-PROCESS-CHOOSER is based on the static-condition-q inquiry for Nigel module.

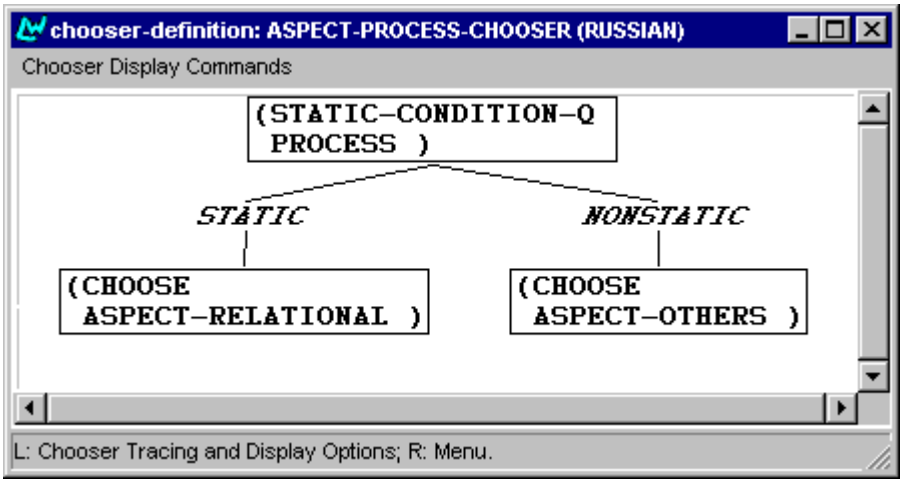


Figure 13: Choosers for the Aspect region.

3.3 Modality

In the deliverables SPEC2 and IMPL2 we discussed one of the major systems in the interpersonal domain, the system of MOOD, and its realization in the three Slavic languages for the purposes of the project and provided a computational account. The other major system at clause rank in the interpersonal metafunction to be accounted for is MODALITY. In this chapter we will present linguistic specifications for modality as relevant for the target text of the Final Prototype (Section 3.3.1), we give a formal description (Section 3.3.2) and describe the implementation (Section 3.3.3).

3.3.1 Linguistic Specification

Instructional texts by nature do not exhibit the full spectrum of modality options. Here, we therefore concentrate on the options that are instantiated in the target texts of the Final Prototype.

The most frequent modal expressions occur in explanations about the **possibilities** available to the user in a particular software tool (cf. SPEC2) or reflect **necessity** (e.g., in preconditions or warnings). For the Final Prototype we concentrate on the generation of modal clauses expressing possibility (cf. the new styles introduced in the Final Prototype: Functional description and Overview).

The examples given in example (31) below show typical expressions of possibility in Bulgarian, Czech and Russian as they occur in our corpus for the Final Prototype.

(31) *You can draw an arc.*

(a) Bg: *Вие можете да начертаете дъга.*

You can-Pl2 draw-Pl2 arc

(b) Cz: *Můžete kreslit oblouk.*

You can-Pl2 draw-Inf arc

(c) Ru: *Вы можете нарисовать дугу*

You can-Pl2 draw-infinitive arc-Acc

In all three languages possibility modality is expressed by a modal auxiliary - analogously to the English ”can”. The semantics of “general possibility” in the three languages (just like in English) is expressed by the same kind of structure: Modal Operator followed by Main Verb.

Another kind of modality expression we consider in the Final Prototype is illustrated in example (32) below. All of them have the meaning of enabling someone to do something. In the process are involved an Initiator and an Agent, the former being the provider of the possibility of doing something, and the latter being the real “doer” of the action.

(32) *The system enables (you) to create a multiline.*

(a) Bg:(i)*Системата позволява да създавате мултилиния.*
System-Det enable-Sg3 create-Pl2Imperf. multiline
(ii) *Системата позволява създаване на мултилиния.*
System-Det enable-Sg3 create-nominalization of multiline

(b) Cz:(i) *Systém umožňuje vytvořit multičáru.*
Systém-Sg enable-Sg3 create-Inf Perf. multiline.
(ii) *Systém umožňuje vytvoření multičáry.*
Systém-Sg enable-Sg3 create-nominalization multiline.

(c) Ru: *Система позволяет создавать мультилинии*
System-Sg3 enable-Sg3 create-Infinitive multiline

We account for the structure of such constructions in the verbal group complex.

All types of verbal group complexes in English are described in (Halliday 85). Figure 14 bellow shows the classification of verbal group complexes in English with examples.

Verbal group complexes
(i)Parataxis
(i)-I Elaborating
*"She **got killed, got run over** by a lorry."*
(i)-II Extending
*"I **neither like nor dislike** it."*
(i)-III Enhancing
*"He **tried, but failed**, to extract the poison.*
(ii) Hypotaxis: *"**tried to do**"*
Primary verbal group (*"**try**"*) carries the mood of the clause
Secondary verbal group (*"**do**"*) has a dependent status
(ii)-I Expansion
Ia Elaborating a process: phase
"stops doing", "turns out to do"
Ib Extending a process: conation
"avoid doing", "learn to do"
Ic Enhancing a process: modulation
"happen to do", "help to do"
(ii)-II Projection- mentalized or verbalized
"want to do", "pretend to do", "be scared to do"

Figure 14: Classification of Verbal group complexes semanics (Halliday 94)

This classification seems valid for our Slavic languages as well. In the following we concentrate on the class of hypotactic extending verbal group complexes, because this is the only type that occurs in our target texts. The shorter name given to hypotactic extending verbal group complexes is **conation**. There are four subtypes of this type of verbal group complex:

Conative

(33) *"try to do", "attempt to do", "avoid doing"*

(a) Bg: *опитвам да стремя се да избягвам да*
Try to attempt-Refl to avoid to

(b) Cz: *zkoušet pokusit se vyhnout se*
try to attempt to avoid doing

(c) Ru: *попробовать сделать постараться сделать*
Try do-Inf. attempt do-Inf.

Reussive

(34) *"succeed in doing", "menage/get to do", "fail doing/to do"…*

(a) Bg: *успявам да направя пропускам да направя*
succeed to do fail to do

(b) Cz: *uspět podařit se selhat*
Succeed manage to do fail

(c) Ru: *удаваться сделать*
succeed do-Inf.

Potential

(35) *"be (un)able to do", "know how to do"*

(a) Bg: *(не съм) в състояние съм да направя зная да правя*
(not be-Sg1) able be-Sg1 to do know-Sg1 to do

(b) Cz*: být (nebýt) schopen vědět, jak*
be (not to be) able know how to

(c) Ru: *быть в состоянии сделать знать, как сделать*
Be-Inf. able do-Inf. know how do-Inf.

Achieval

(36) *"learn to do", "practise doing"*

(a) Bg: *уча се да правя упражнявам се да правя*
learn-Refl. to do exercise-Refl. to do

(b) Cz: *učit se cvičit*
learn practise

(c) Ru: *научиться делать*
learn-Refl. do-Inf.

Conation in the three languages is typically realized in active voice, either with a reflexive form or as a causative structure. Passive constructions are atypical for Slavic languages.

In Bulgarian and Czech there are different constructions to express conation- In Bulgarian "da-construction" (see example (2a(i)) above) and nominalization (2a(ii)) after the primary verb, in Czech- infinitive (2b(i)) or nominalization (2b(ii)) after the primary verb. In Russian only infinitive could appear as secondary verb (2c(i)).

Conative and Reussive verbal group complexes tend to involve one and the same person in both processes (primary and secondary), so they are usually realized by active voice. More frequently, Potential and Achieval are expressed by a causative structure, which makes both the Initiator and the real "doer" explicit:

(37) *Enable someone to do something*

(a) Bg: *давам възможност/позволявам някому да прави нещо*

give possibility /enable so-Dat to do sth

(b) Cz: *Umožnit někomu něco udělat.*

Enable so-Dat sth to do

(c) Ru: *давать возможность кому-нибудь делать что-нибудь*

give possibility so-Dat do-Inf. sth

(38) *Teach someone to do something*

(a) Bg: *уча някого да прави нещо*

teach so to do sth.

(b) Cz: *Učit někoho něco dělat.*

to teach so sth do-inf

(c) Ru: *учить кого-нибудь делать что-нибудь*

teach so do-inf. sth

In example (32) above, the semantics reflected is "to give possibility in general", which makes the real doer (Agent of secondary verb) not explicit, but generic. This is realized by an infinitival form of the secondary verb in Czech and Russian and by a nominalization in Bulgarian. The same feature could be stressed by omitting the surface realization of the Agent of the secondary process. We prefer realization without pronoun (see examples (39) and (40) below).

(39) *Implicit Agent of secondary process*

(a) Bg: *Системата позволява да правите това.*

System-Det allow-Sg3 to do-Pl2 it

(b) Cz: *Systém umožňuje udělat to.*

Systém allow-Sg3 to do-Inf smth.

(c) Ru: *Система позволяет делать/сделать это*

System allow-Sg3 do-Inf.Perf/Inf.Imperf it.

(40) *Explicit Agent of secondary process*

(a) Bg: *Системата Ви позволява да правитетова.*
System-Det you-AccPl2 allow-Sg3 to do-Pl2 it

(b) Cz: *Systém vám umožňuje udělat to.*
Systém you-DatPl allow-sgš to do it.

(c) Ru: *Система позволяет Вам делать/сделать это.*
System allow-Sg3 you-Dat do-Inf.Perf/Inf.Imperf it.

The next section presents the formal grammar specifications for the three Slavic grammars to cover these types of modality, made on the base of English grammar Nigel.

3.3.2 Formal Specifications

3.3.2.1 Modal Operator (POSSIBILITY)

We adopted the main part of system network presented bellow from Nigel grammar.

```
System: DEICTICITY
FINITE-CLAUSE →
 TEMPORAL (+TEMPO0, TEMPO0/FINITE)
 MODAL    (+MODAL, MODAL::MODAL-AUX)
Chooser: DEICTICITY-CHOOSER
 If a modal operator is specified in the clause
    Then choose MODAL
    Else choose TEMPORAL
```

In Bulgarian, Czech and Russian modal verbs have tense forms, but for the purpose of generation of our particular examples and lack of different tenses in our register in general we keep the “temporal-modal” opposition here.

The next three systems deal with the semantics of modal verb.

```
System: VOLITIONALITY
MODAL →
 VOLITIONAL     (MODAL::VOLITION-AUX)
 NONVOLITIONAL
Chooser: VOLITIONALITY-CHOOSER
 If the process is ambient
    Then choose NONVOLITIONAL
    Elseif the process is existential
                Then choose NONVOLITIONAL
                Elseif the modal operator is explicitly specified as
                volitional
                   Then choose VOLITIONAL
                   Else choose NONVOLITIONAL
```

```
System: DEGREE-OF-MODALITY
NONVOLITIONAL →
 POSSIBILITY (MODAL::POSSIBILITY-AUX)
 NECESSITY   (MODAL::NESSECITY-AUX)
Chooser: DEGREE-OF-MODALITY-CHOOSER
 If a modal operator is explicitly specified as necessity
    Then choose NECESSITY
    Else choose POSSIBILITY
```

```
System: POSSIBILITY-TYPE
POSSIBILITY →
 ABILITY         (MODAL::ABILITY-AUX)
 GENERAL-POSSIBILITY  (MODAL::NOT-ABILITY-AUX)
Chooser: POSSIBILITY-TYPE-CHOOSER
 If the process is ambient
    Then choose GENERAL-POSSIBILITY
    Elseif the process is existential
                Then choose GENERAL-POSSIBILITY
                Elseif the modal operator is explicitly specified as
                ability
                   Then choose ABILITY
                   Else choose GENERAL-POSSIBILITY
```

The generation of sentences on example (1) from the previous section goes through the NONVOLITIONAL, POSSIBILITY and GENERAL-POSSIBILITY branches and the features collected for Modal are possibility-aux and not-ability-aux. The systems are identical for all three languages.

```
System: MODALITY-CONDITIONALITY
MODAL →
 MODALITY-CONDITIONAL (FINITE::CONDITIONALITY-AUX,
                              FINITE^MODAL)
 MODALITY-NONCONDITIONAL    (FINITE/MODAL)
Chooser: MODALITY-CONDITIONALITY-CHOOSER
 If the clause is explicitly specified as conditional
    Then choose MODALITY-CONDITIONAL
    Else choose MODALITY-NONCONDITIONAL


System: MODALITY-POLARITY
MODAL →
    MODALITY-POSITIVE
 MODALITY-NEGATIVE
Chooser: MODALITY-POLARITY-CHOOSER
 If the clause is explicitly specified as negative
    Then choose MODALITY-NEGATIVE
    Else choose MODALITY-POSITIVE
```

Modal verbal groups in Bulgarian are constructed by a modal verb and a so called “da-construction” consisting of the “da” particle followed by the main verb, which is the same person and number as the Finite. The Nigel function for this main verb is Auxstem. Finite-Auxstem agreement is discussed in detail in Agreement chapter (INSERT REFERENCE HERE LATER) of this deliverable.

In Czech and Russian (as in English) the main verb in modal construction is an infinitive, realized in the grammar network by inflectifying the AuxStem function as Stem.The system AUXSTEM-INSERTED presented below shows the difference between Bulgarian and the other two Slavic languages in this respect. In Bulgarian case the “da-construction” is built by inserting da-particle, lexifying it and specifying the order “da-particle before Auxstem”. Finite-Auxstem agreement in person and number is done here. In Czech and Russian Auxstem is inserted and inflectified as Stem.

```
Gate: AUXSTEM-INSERTED
MODAL →
    AUXSTEM-INSERTED  BG:   (+AUXSTEM, +DA-PARTICLE,
```

```
                DA-PARTICLE!DA, DA-PARTICLE^AUXSTEM,
                FINITE~AUXSTEM PERSON-FORM,
                FINITE~AUXSTEM NUMBER-FORM)
            CZ:   (+AUXSTEM, AUXSTEM:::IFINITIVE-FORM)
            RU:   (+AUXSTEM, AUXSTEM:::IFINITIVE-FORM)
```

In the three languages the main verb in the modal construction is the carrier of voice. This is realized by the conflation Auxstem/Voice in the system AUXSTEM-VOICE shown below.

```
Gate: AUXSTEM-VOICE
AUXSTEM-INSERTED →
    AUXSTEM-VOICE  (AUXSTEM/VOICE)
```

3.3.2.2 Verbal Group Complex

There is no implementation of verbal group complexes in the Nigel grammar. The only complex verbal group covered there is PHASE verbal group, so we used its principle idea in organizing the system network for generation of CONATION verbal group complexes. The solution is common for the three languages, differences are explicitly stressed in following presentation.

The system of CONATION introduces Conation and Conationdependent elements into the structure of the clause to be generated.

```
System: CONATION
TRANSITIVITY-UNIT →
    NOT-CONATION
 CONATION (+CONATION, +CONATIONDEPENDENT,
                CONATION/VOICE, CONATIONDEPENDENT/LEXVERB)
Chooser: CONATION-CHOOSER
 If the clause contains conation verbal group complex
    Then choose CONATION
    Else choose NOT-CONATION
```

Each subtype of CONATION leads to particular lexical items in each language. See the system CONATION-TYPE:

```
System: CONATION-TYPE
CONATION →
    CONATIVE leads to the choice of the verb "try"
 REUSSIVE leads to the choice of the verb "succeed"
 POTENTIAL   leads to the choice of the verb "allow"
 ACHIEVAL leads to the choice of the verb "learn"
Chooser: CONATION-TYPE-CHOOSER
 If the verbal group complex is explicitly specified as conative
    Then choose CONATIVE
    Elseif the verbal group complex is explicitly specified as
          Reussive
                Then choose REUSSIVE
                Elseif the verbal group complex is explicitly specified
                As expressing potential
                   Then choose POTENTIAL
                   Else choose ACHIEVAL
```

The system CONATIONDEPENDENT-TYPE of the Bulgarian grammar is shown below. When the branch da-conation is chosen Da-particle and Conationdependent are inserted to build the "da-construction". In the case of conation-nominalization the Conationdependent element is preselected to be a nominal group with feature processual for the head noun (Thing).

```
System: CONATIONDEPENDENT-TYPE
CONATION →
    DA-CONATION       (+DA-PARTICLE, DA-PARTICLE!DA,
                DA-PARTICLE^CONATIONDEPENDENT)
 CONATION-NOMINALIZATION (CONATIONDEPENDENT:NOMINAL-GROUP,
                          CONATIONDEPENDENT:PROCESSUAL-IN-CONATION)
Chooser: CONATIONDEPENDENT-TYPE-CHOOSER
    If the realization of the secondary process is explicitly chosen as
    nominalization realization
    Then choose CONATION-NOMINALIZATION
    Else choose DA-CONATION
```

The grammars of Czech and Russian use the following system:

```
Gate: CONATIONDEPENDENT-TYPE
CONATION →
    CONATION-INFINITIVE  (CONATIONDEPENDENT:::INFINITIVE-FORM)
```

The system CONATION-SECOND-AGENT deals with the Agent of Secondary process in the verbal group complex.

Conation-nominalization in Bulgarian clauses makes explicit surface realization of the Agent of Secondary process impossible. This feature gives different input conditions to the CONATION-SECOND-AGENT system for Bulgarian, on the one hand, and for Czech and Russian, on the other hand.

In all the instructional texts generated in the Final Prototype of the AGILE system the User plays substantial role. S/He is the Hearer and often the Actor in the sentence plans. Here S/He becomes the Agent of the secondary process in the verbal group complex and in Bulgarian "da construction" this is visible in Person and Number of the secondary verb (Conationdependent). Conationdependent element is inflectified here as secondperson-form and plural-form of the verb.

```
System: CONATION-SECONDAGENT
 BG:   DA-CONATION
 CZ:   CONATION
 RU:   CONATION
→
      GENERIC-AGENT   BG:   (CONATIONDEPENDENT:::SECONDPERSON-FORM,
                            CONATIONDEPENDENT:::PLURAL-FORM)
      SPECIFIED-AGENT
Chooser: CONATION-SECONDAGENT-CHOOSER
    If the target meaning verbal group complex is "generalization" or
    statement in general
          then choose generic agent
          else choose specified-agent
```

For the purposes of generation constrained to the texts of Final Prototype of AGILE project the choice is fixed to the choice of Generic-agent branch.

3.3.3 Implementation

3.3.3.1 Modal operator (POSSIBILITY)

We present bellow the systems and choosers from Bulgarian grammar resources needed to generate modal clauses in the Agile Final Prototype texts. We show an example of generated Bulgarian sentence with modal operator. Czech and Russian grammars use the same systems and choosers (adopted from Nigel grammar) with the minor differences discussed in previous part (Formal specifications).

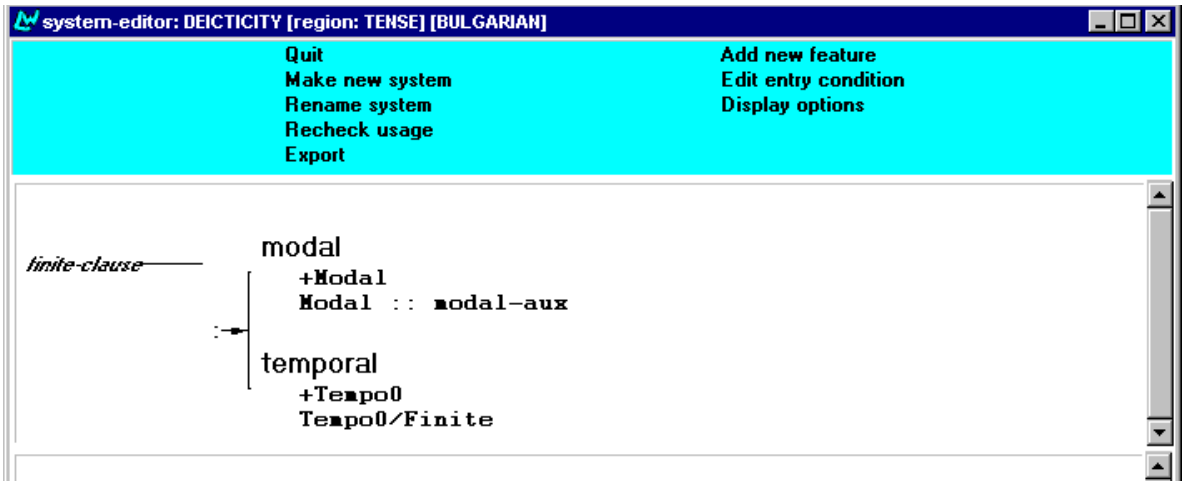


Figure 15: DEICTICITY system (Bg)

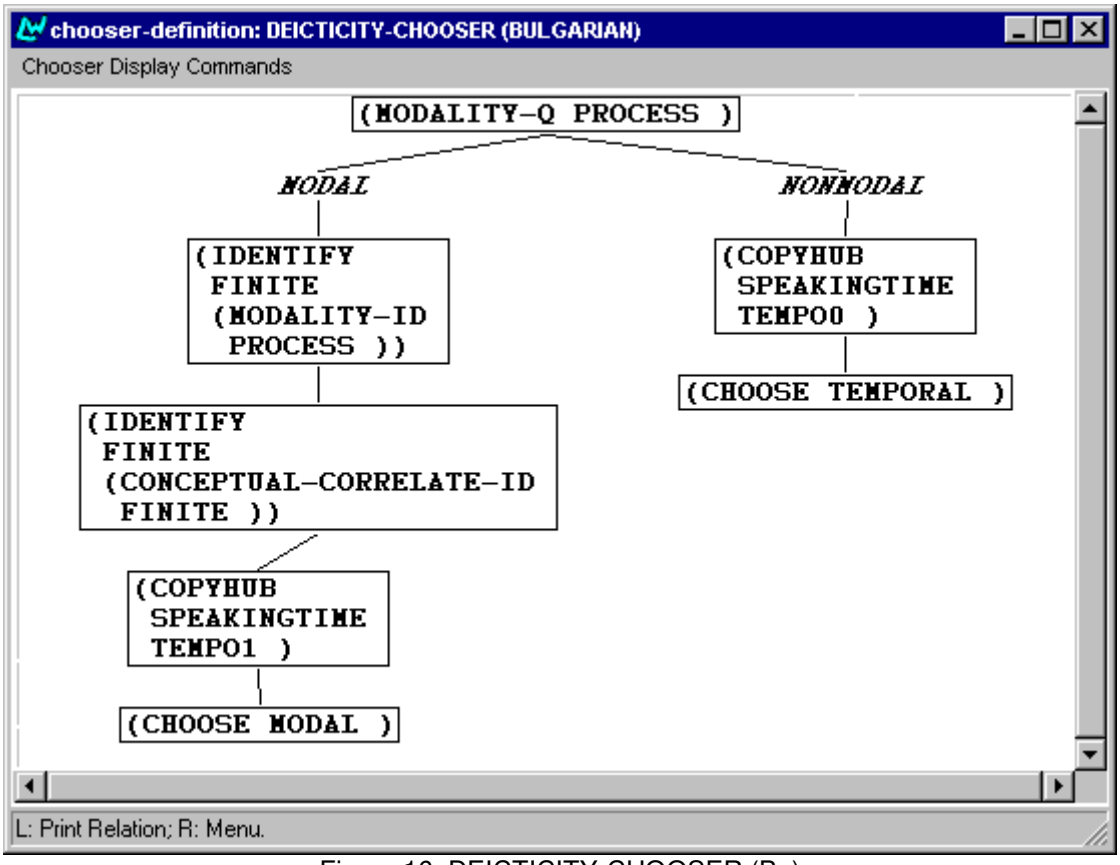


Figure 16: DEICTICITY-CHOOSER (Bg)

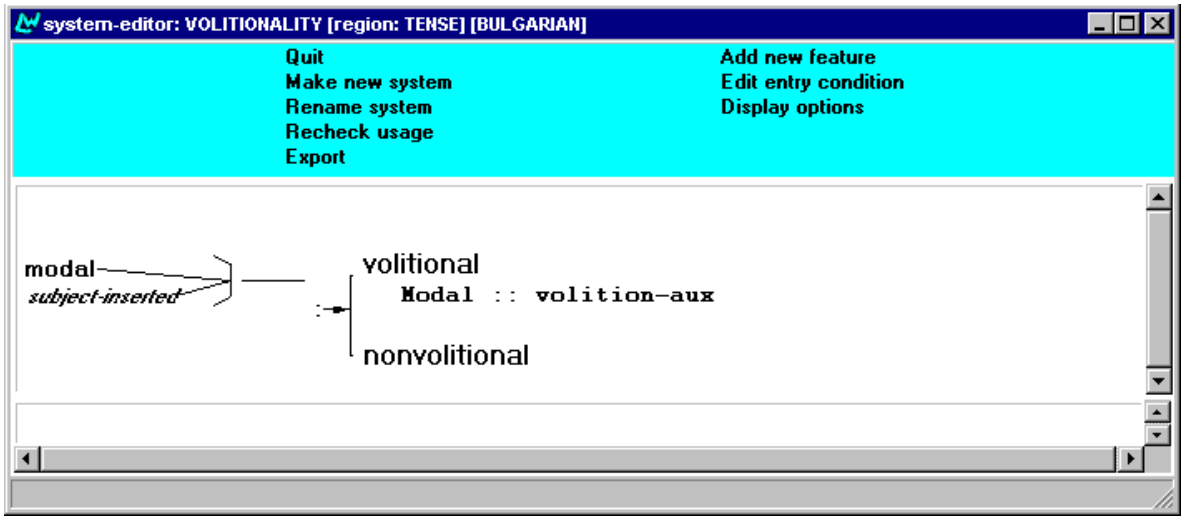


Figure 17: VOLITIONALITY system (Bg)

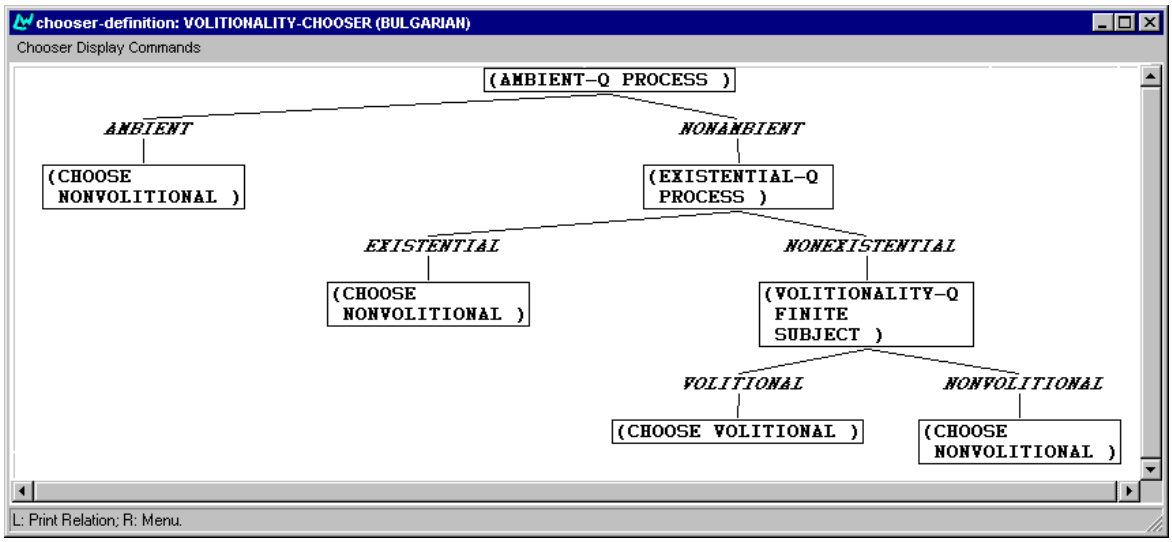


Figure 18: VOLITIONALITY-CHOOSER (Bg)

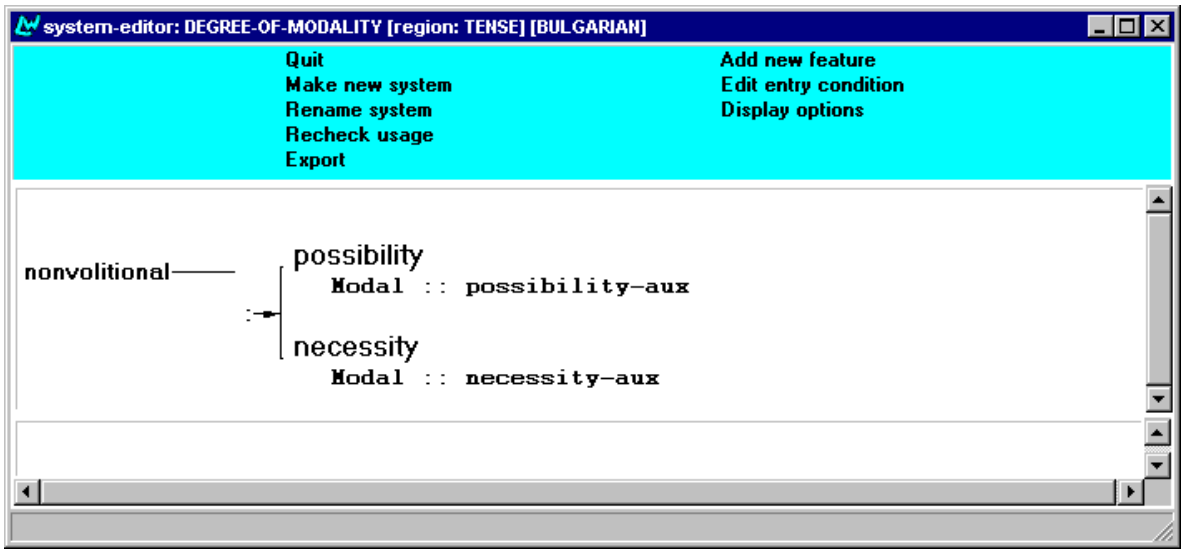


Figure 19: DEGREE-OF-MODALITY system (Bg)

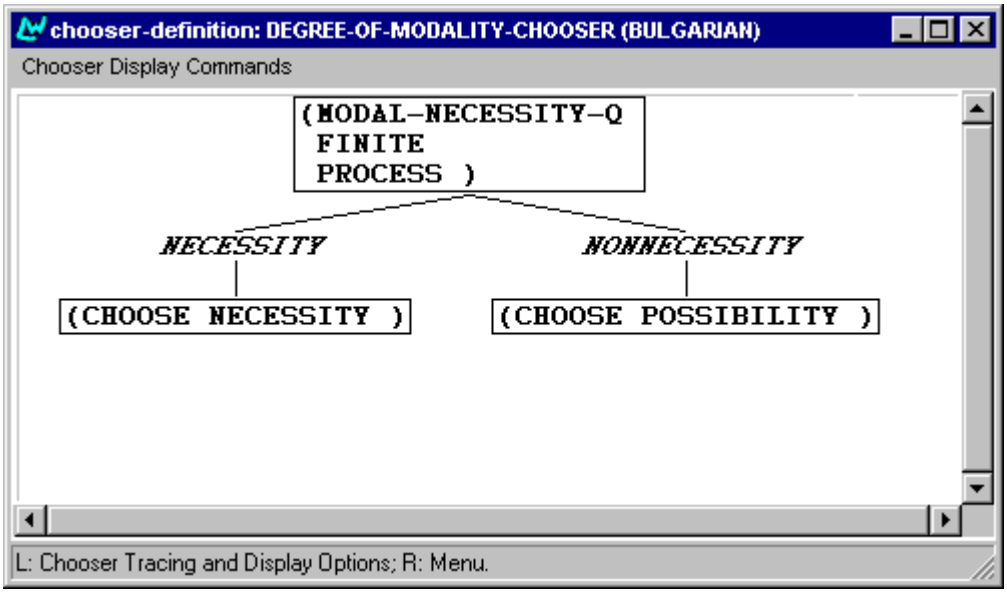


Figure 20: DEGREE-OF-MODALITY-CHOOSER (Bg)

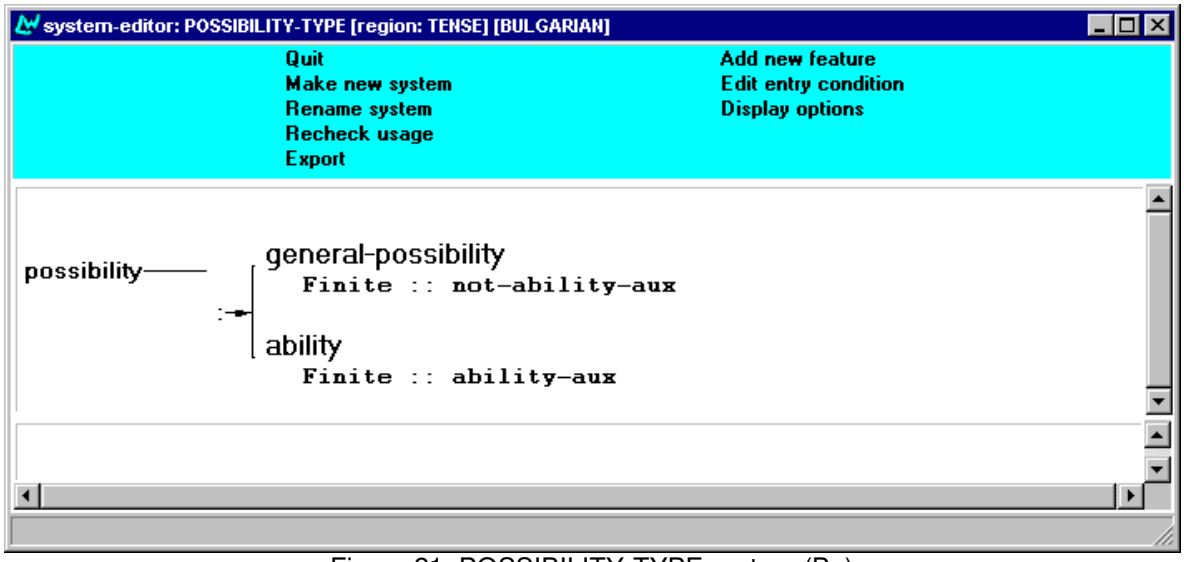


Figure 21: POSSIBILITY-TYPE system (Bg)

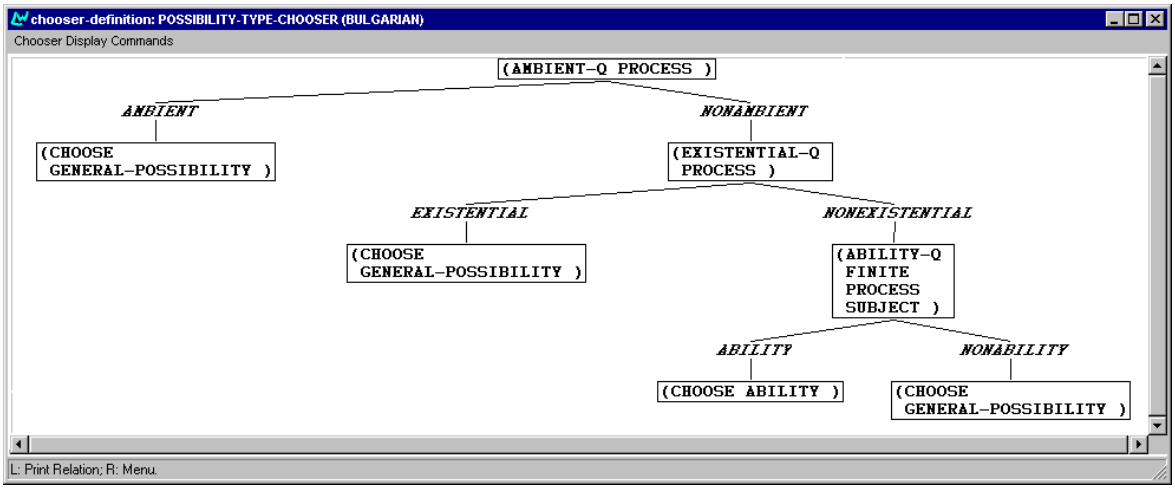


Figure 22: POSSIBILITY-TYPE-CHOOSER (Bg)

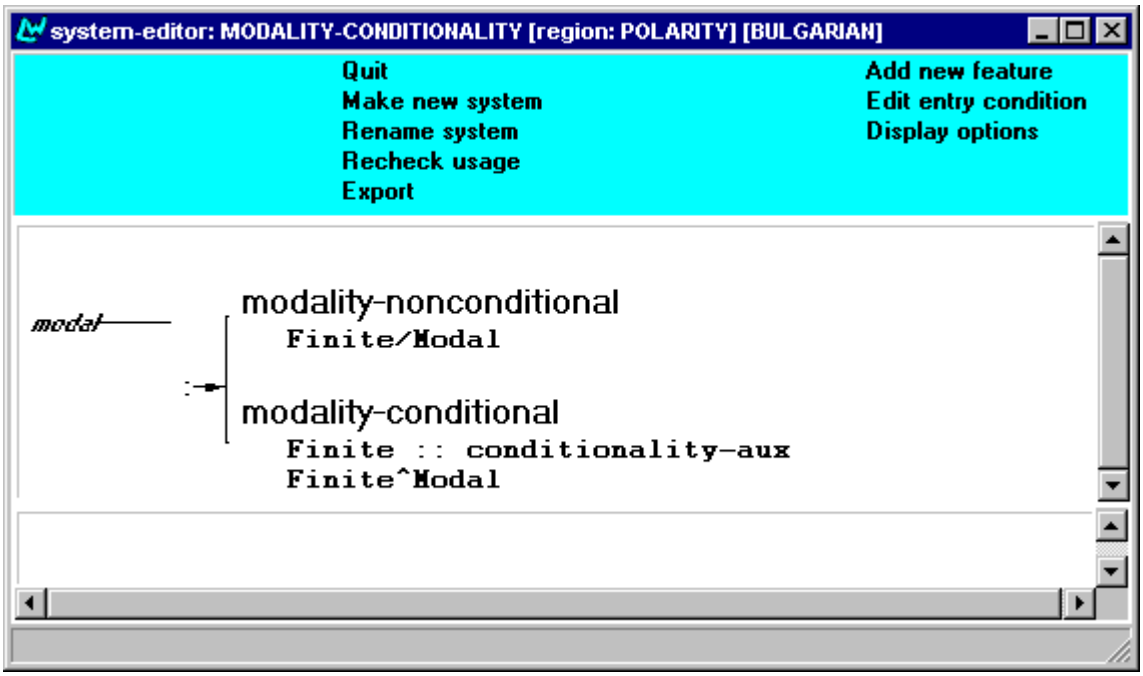


Figure 23: MODALITY-CONDITIONALITY system (Bg)

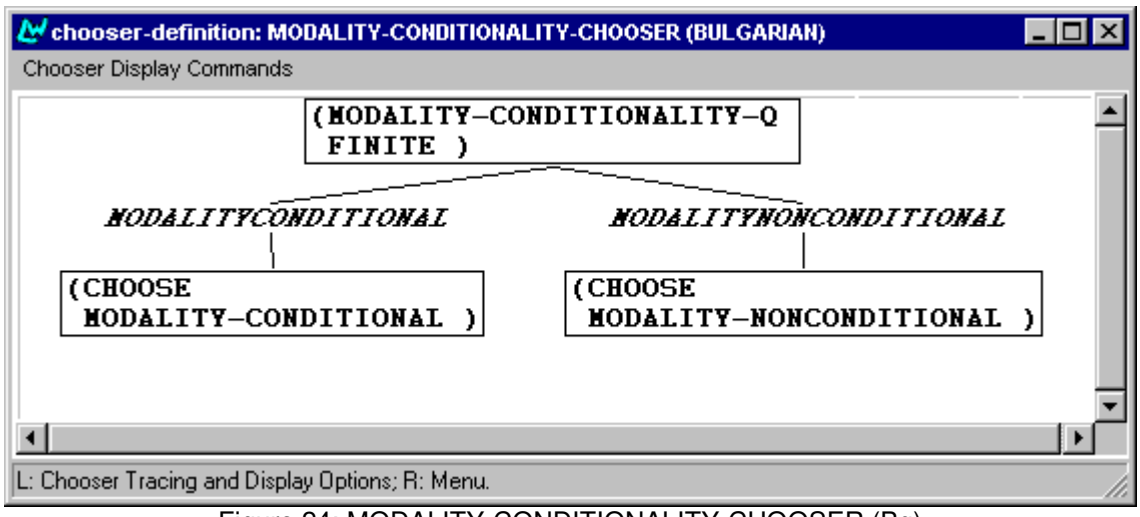


Figure 24: MODALITY-CONDITIONALITY-CHOOSER (Bg)

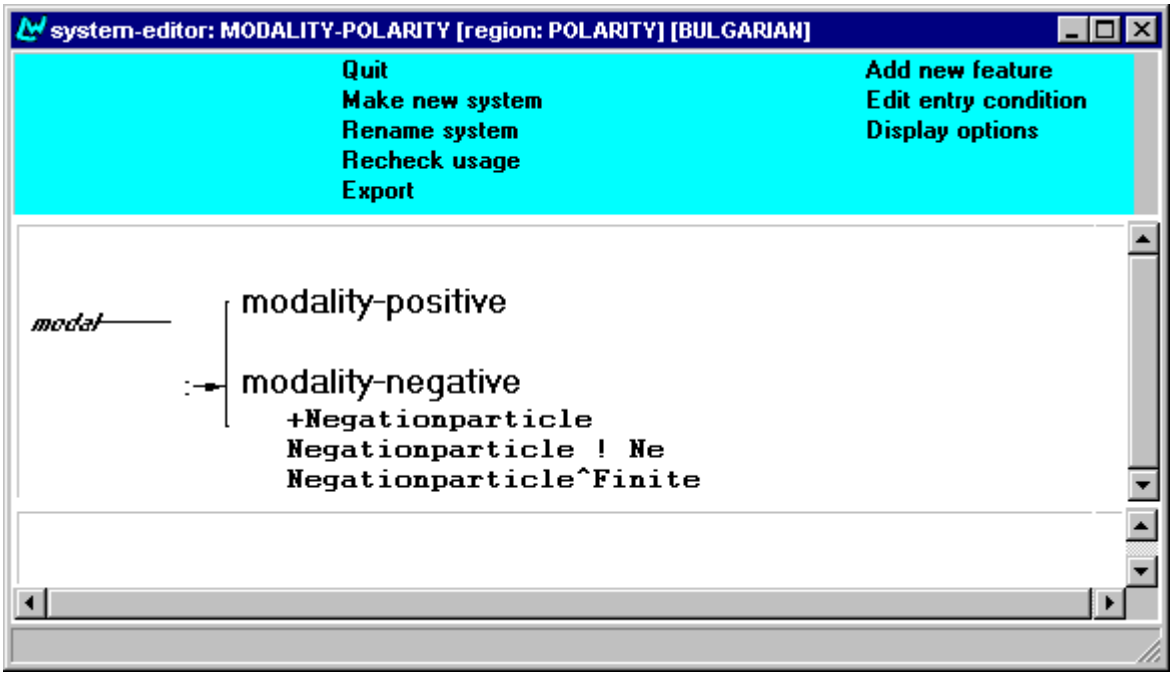


Figure 25: MODALITY-POLARITY system (Bg)

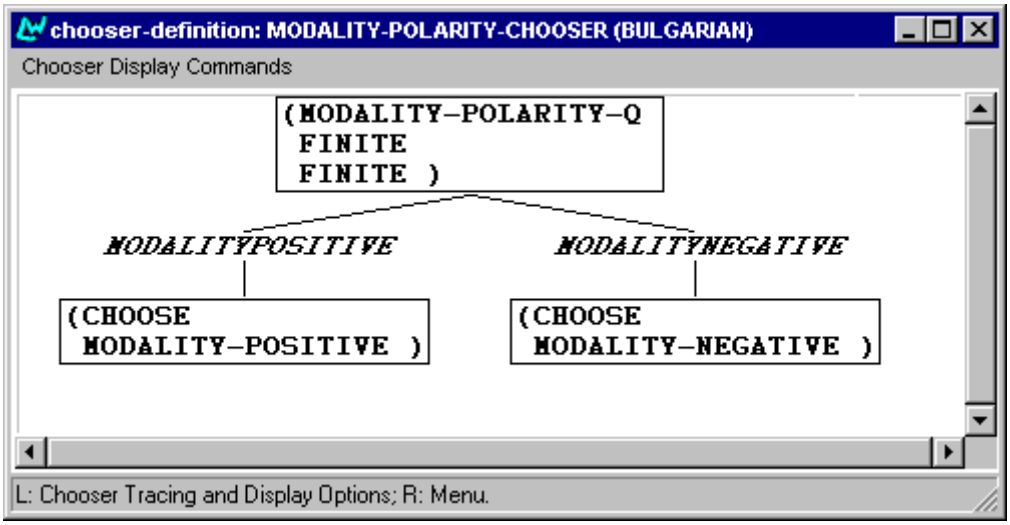


Figure 26: MODALITY-POLARITY-CHOOSER (Bg)

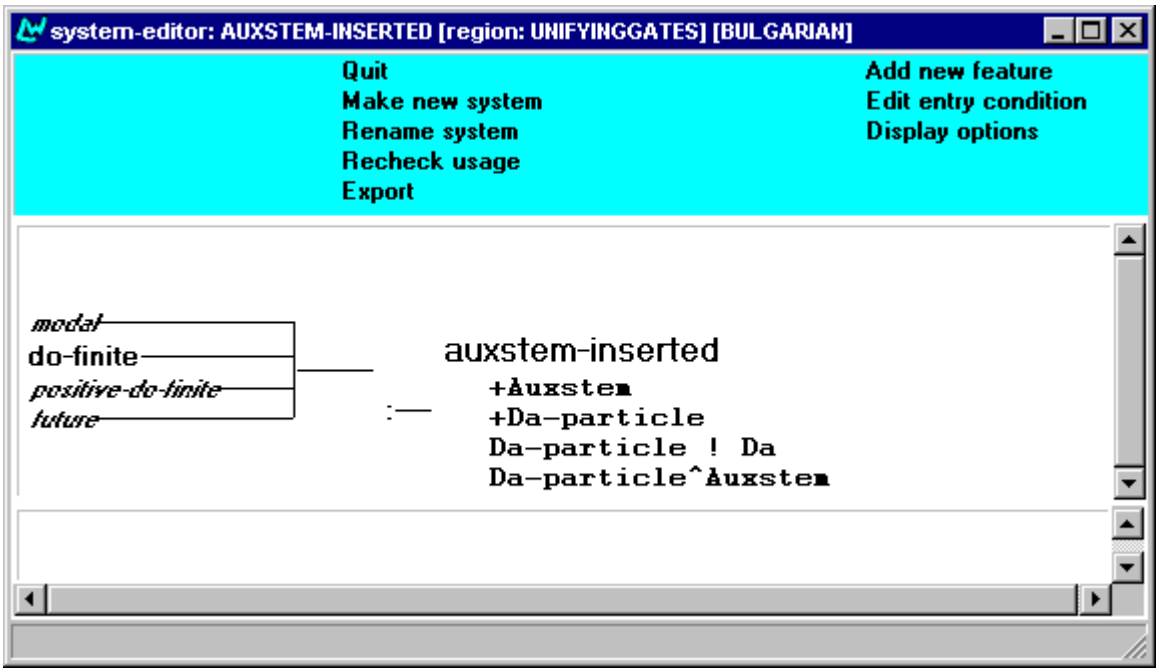


Figure 27: AUXSTEM-INSERTED system (Bg)

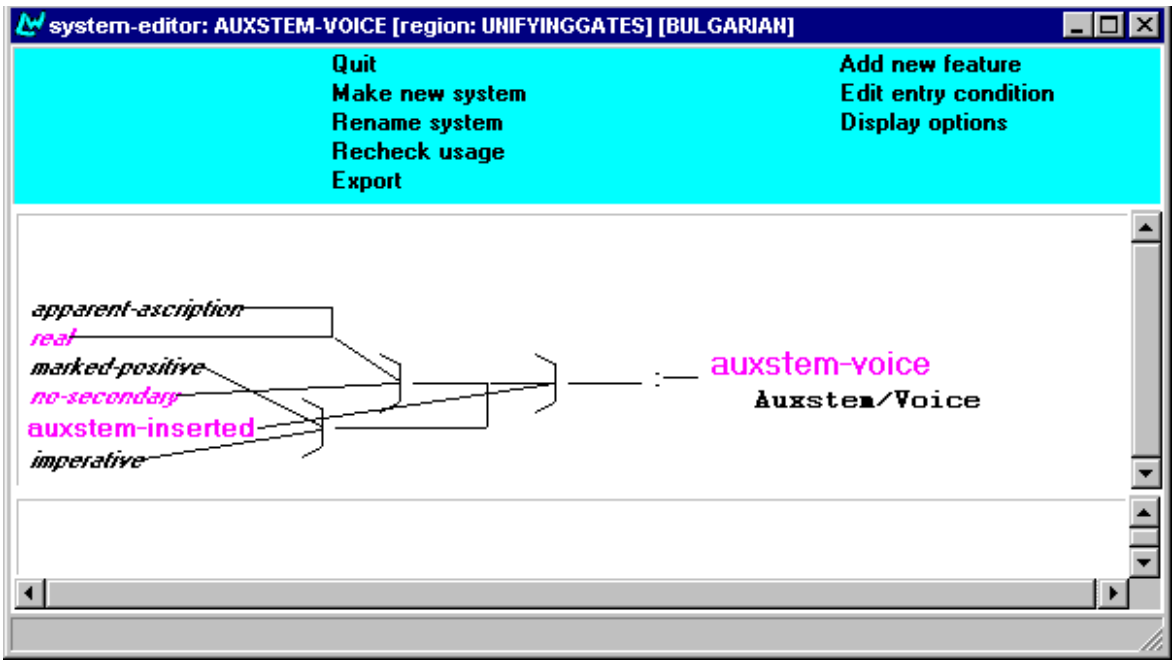


Figure 28: AUXSTEM-VOICE system (Bg)

The following example illustrates the generation of a Bulgarian sentence with modal operator for expressing possibility. In Figure 29 we present the corresponding SPL, and in Figure 30 the generated grammatical structure is shown.

(41) *You can draw an arc.*

(a) Bg: *Вие можете да начертаете дъга.*

You can-Pl2 draw-Pl2 arc.

```
(|a1| / DM::DRAW
      :MODAL-PROPERTY-ASCRIPTION     GENERAL-POSSIBILITY
      :ACTOR (HEARER / DM::USER  :IDENTIFIABILITY-Q IDENTIFIABLE)
      :ACTEE (|a2| / DM::ARC))
```

Figure 29: SPL for the sentence in Example(11)

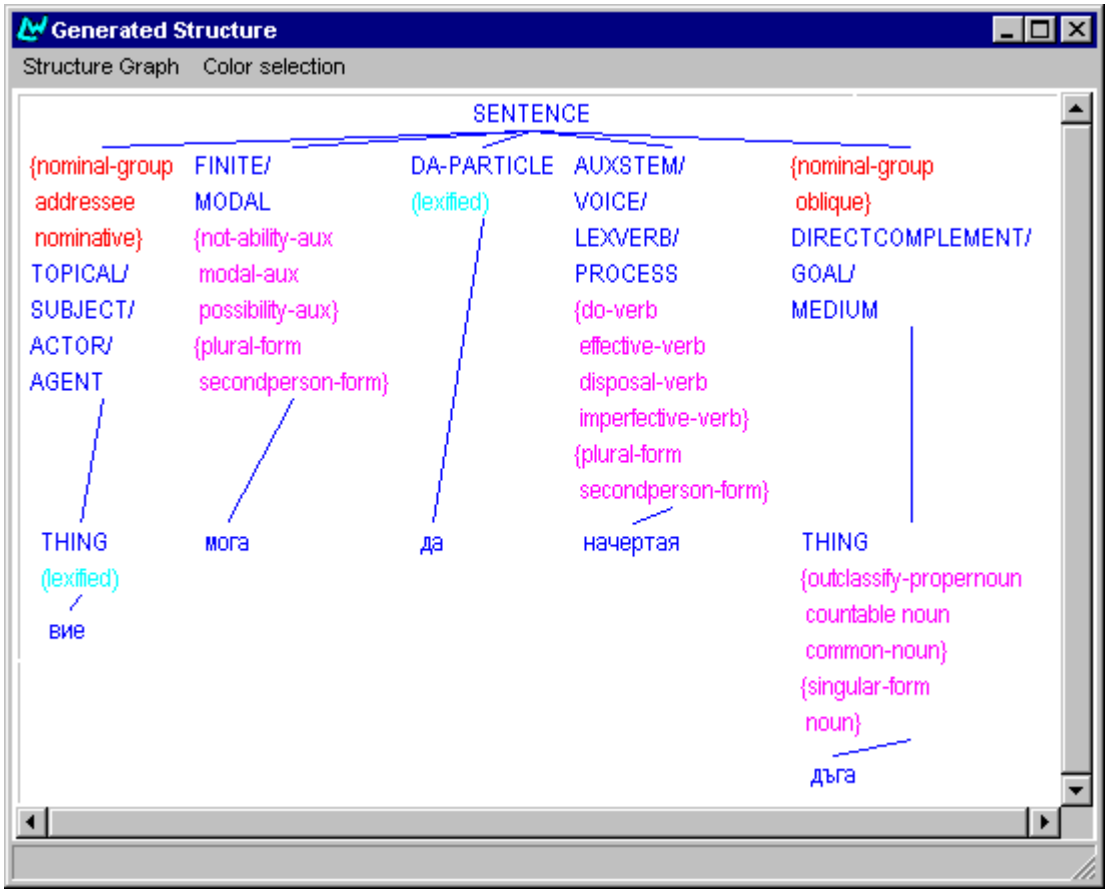


Figure 30: Grammatical structure generated from the SPL in Figure 29 by the Bulgarian grammar

The most important line in respect to Modality in the SPL above is

```
:MODAL-PROPERTY-ASCRIPTION      GENERAL-POSSIBILITY
```

This way we code in the SPL that the clause is modal and the type of Modality is GENERAL-POSSIBILITY.

The FEATURES defined for the modal verb CAN in the lexicons of the three languages are the same:

```
(LEXICAL-ITEM
  :NAME    CAN
  :SPELLING      Bg:   "мога"
        Ru:   "можете"
        Cz:   "moci"
  :FEATURES     (NOT-ABILITY-AUX POSSIBILITY-AUX
                 OUTCLASSIFY-MODALITYNEGATIVE
                 OUTCLASSIFY-CONDITIONALITY-AUX
                 MODAL-AUX VERB))
```

3.3.3.2 Verbal Group Complex (POTENTIAL)

We present bellow the systems and choosers from Bulgarian grammar resources needed to generate verbal group complexes with conation in the Agile Final Prototype. We show two examples of Bulgarian sentences with conation to illustrate generation of “da-construction”

and generation of nominalization in conation. Czech and Russian grammars use the same implementations with the minor differences discussed in previous part (Formal specifications).

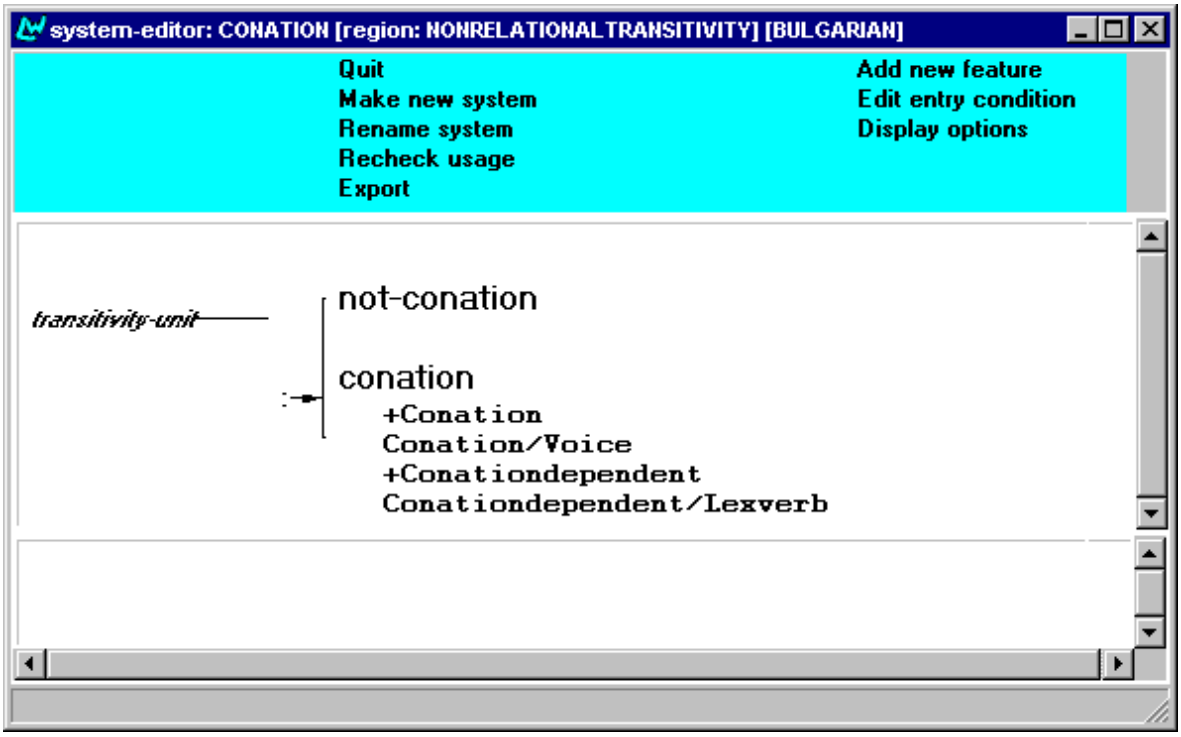


Figure 31: CONATION system (Bg)

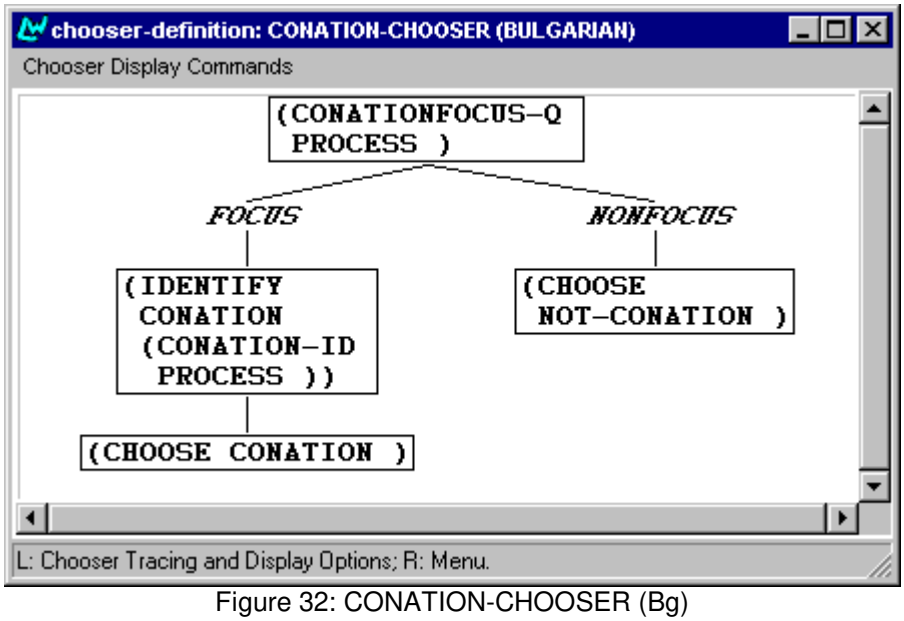


Figure 32: CONATION-CHOOSER (Bg)

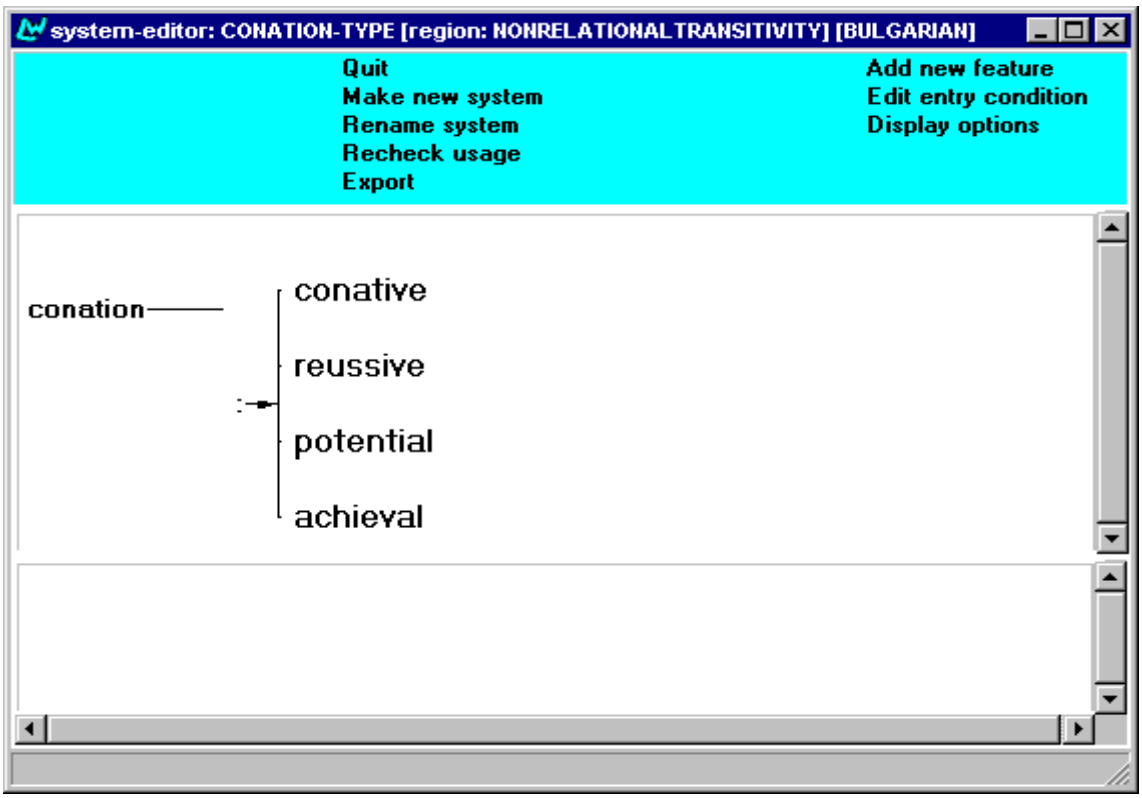


Figure 33: CONATION-TYPE system (Bg)

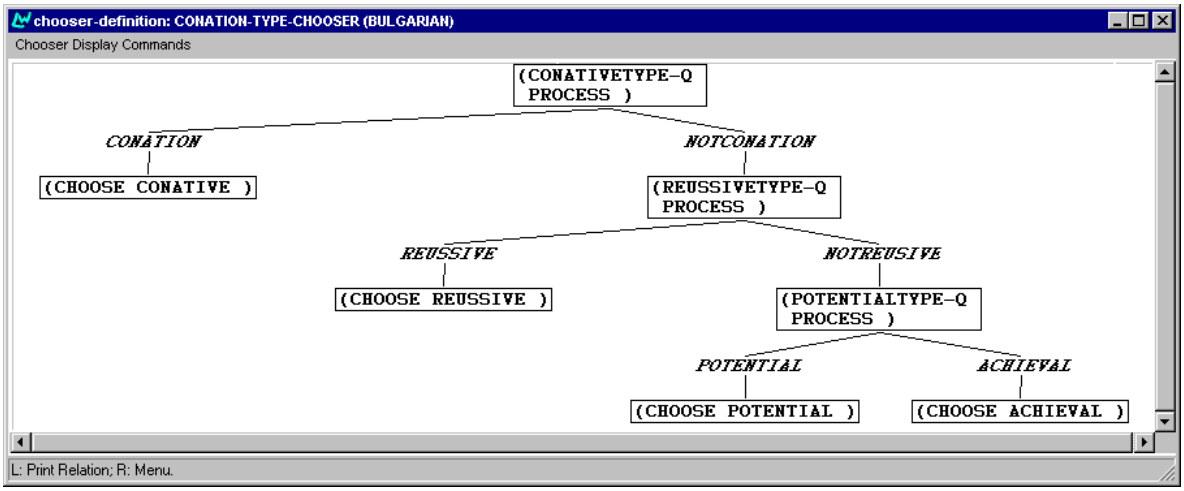


Figure 34: CONATION-TYPE-CHOOSER (Bg)

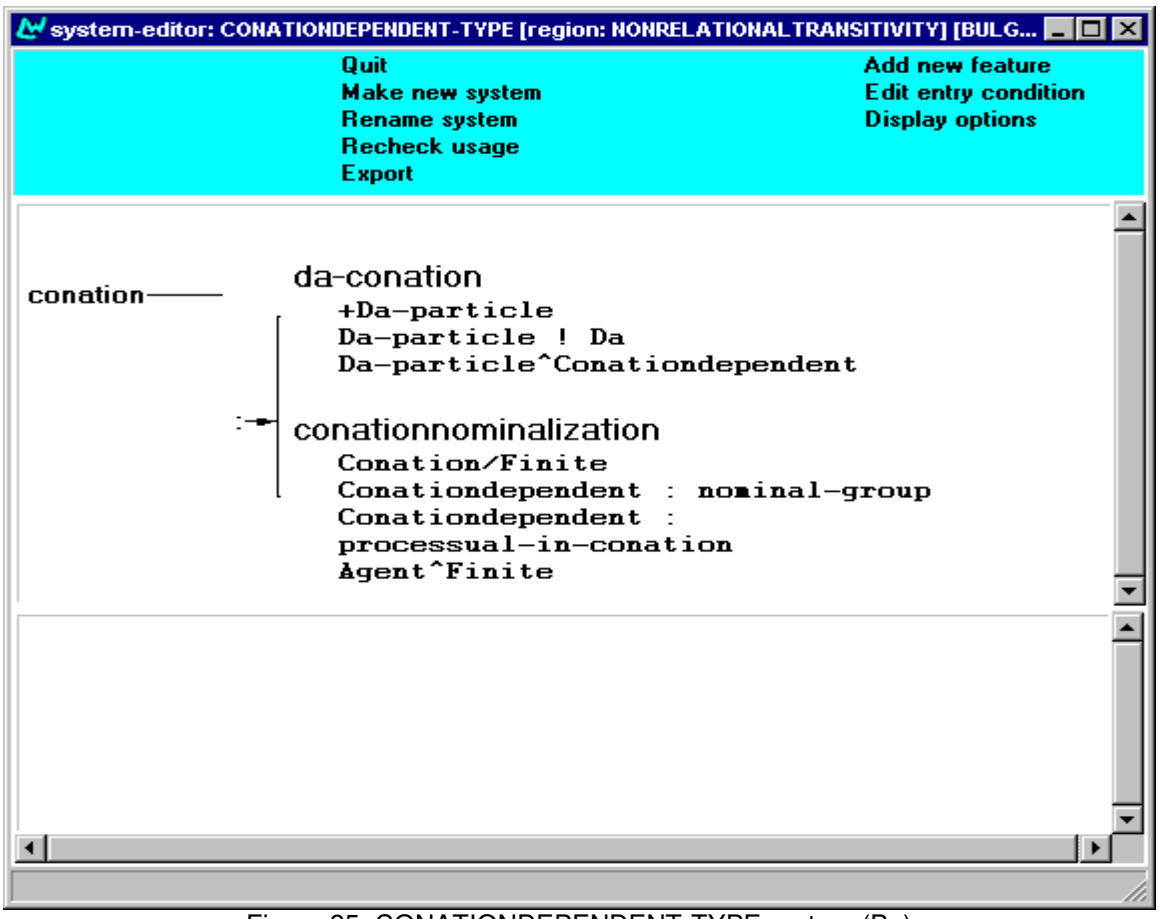


Figure 35: CONATIONDEPENDENT-TYPE system (Bg)

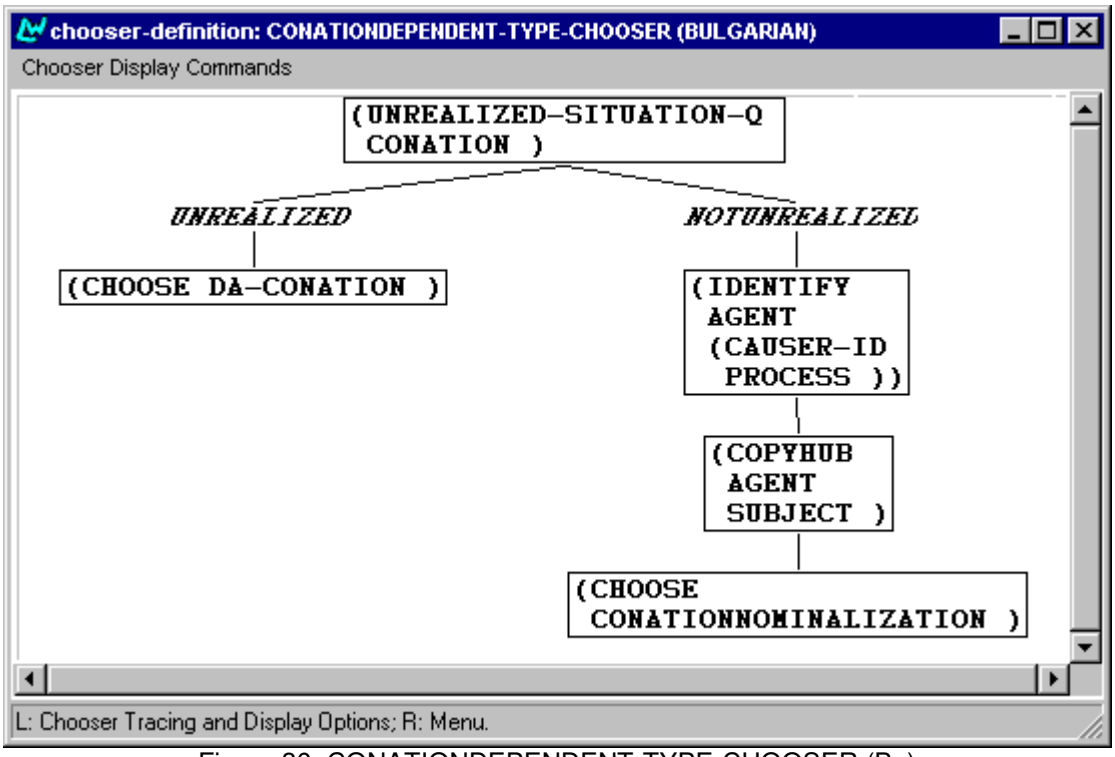


Figure 36: CONATIONDEPENDENT-TYPE-CHOOSER (Bg)

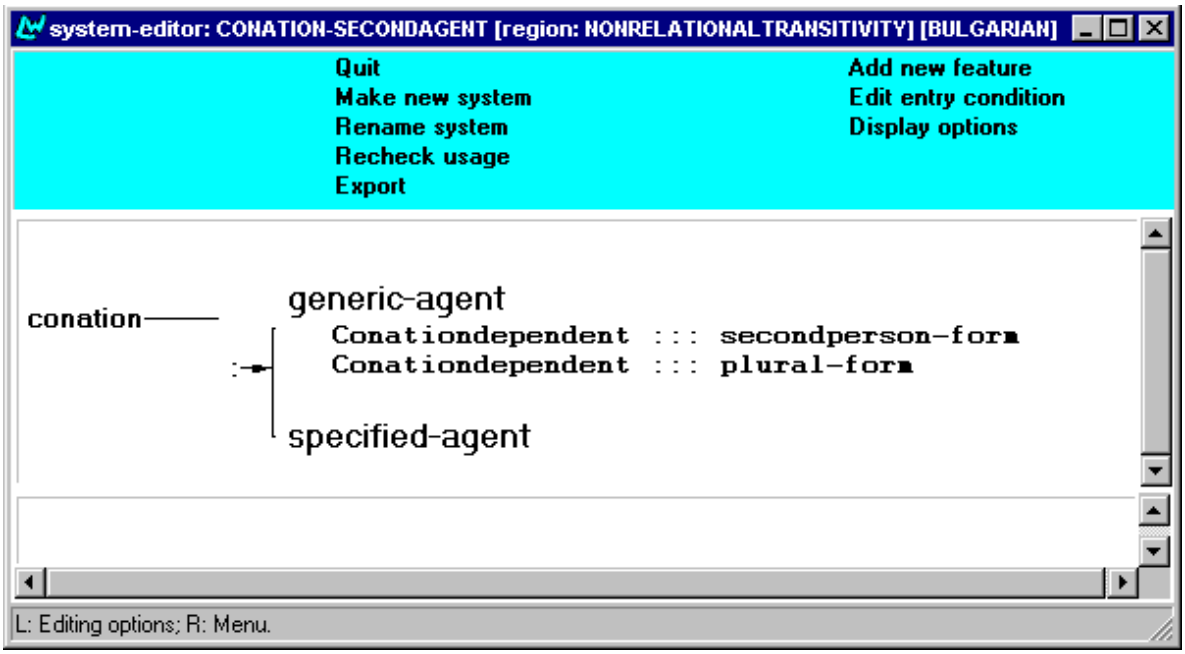


Figure 37: CONATION-SECONDAGENT system (Bg)

The following example illustrates the use of conation potential in a Bulgarian sentence with "da-construction".

(42) *SAVE button allows saving of the multiline.*

(a) Bg: *Бутонът SAVE позволява да запазвате мултилинията.*

Button-det. SAVE allow-3p,sing save-"da-construction"2p.,pl. multiline-det.

In Figure 38 we present the corresponding SPL, and in Figure 39 the generated grammatical structure is shown.

```
(S / DM::SAVE
    :CONATION-ASPECT POTENTIAL
    :ACTOR (A1 / DM::BUTTON
              :CLASS-ASCRIPTION (KPML::|a71| / DM::GUI-SAVE))
    :ACTEE (A2 / DM::MULTILINE :IDENTIFIABILITY-Q DM::IDENTIFIABLE)
 )
```

Figure 38: SPL for the sentence in Example(11)

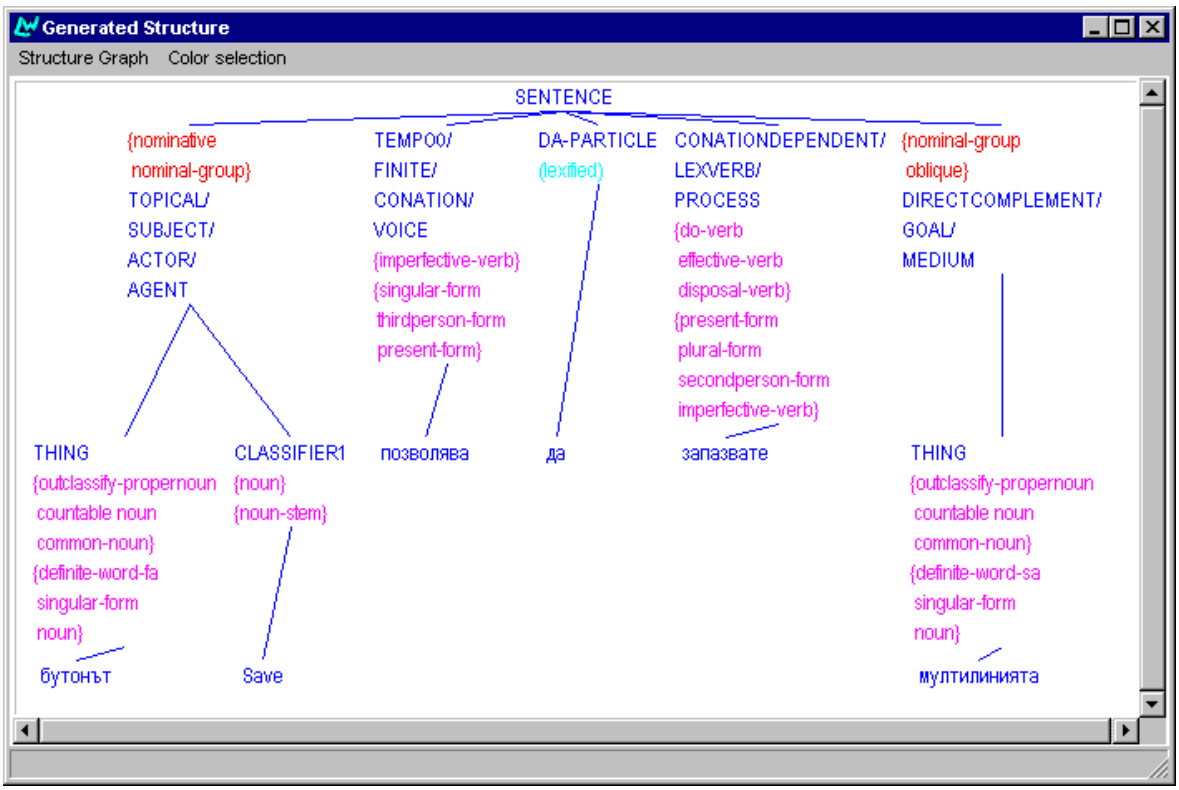


Figure 39: Grammatical structure generated from the SPL in Figure 38 by the Bulgarian grammar

The SPL statement `:conation-aspect potential` in the SPL in Figure 38 is an abbreviation using an SPL macro. The SPL macros for different conation types are as shown in Figure 40:

```
(defspl-macro :conation-aspect (
;try
(conative :conationfocus-q focus
    :conation-id (conative / conative :lex opitvam))
;succeed
(reussive  :conationfocus-q focus
     :conation-id (reussive / reussive :lex uspiavam))
;allow
(potential :conationfocus-q focus
     :conation-id (potential / potential :lex pozvoliavam))
;learn
(achieval  :conationfous-q focus
     :conation-id (achieval / achieval :lex ucha-se))
))
```

Figure 40: SPL macros for conation in Bulgarian resources

We show bellow an example to illustrate the use of conation potential in a Bulgarian sentence with nominalization.

(43) *SAVE button allows saving of the multiline.*

(a) Bg: *Бутонът SAVE позволява запазване на мултилинията.*

Button-Det. SAVE allow-Sg3 save-Nominalization of multiline-Det.

In Figure 41 we present the corresponding SPL, and in Figure 42 the generated grammatical structure is shown.

```
(S / DM::SAVE
:CONATION-ASPECT  POTENTIAL
:UNREALIZED-SITUATION-Q NOTUNREALIZED
:ACTOR (A1 / DM::BUTTON
           :CLASS-ASCRIPTION (KPML::|a71| / DM::GUI-SAVE))
:ACTEE (A2 / DM::MULTILINE
           :IDENTIFIABILITY-Q DM::IDENTIFIABLE))
```

Figure 41 : SPL for the sentence in Example (12)

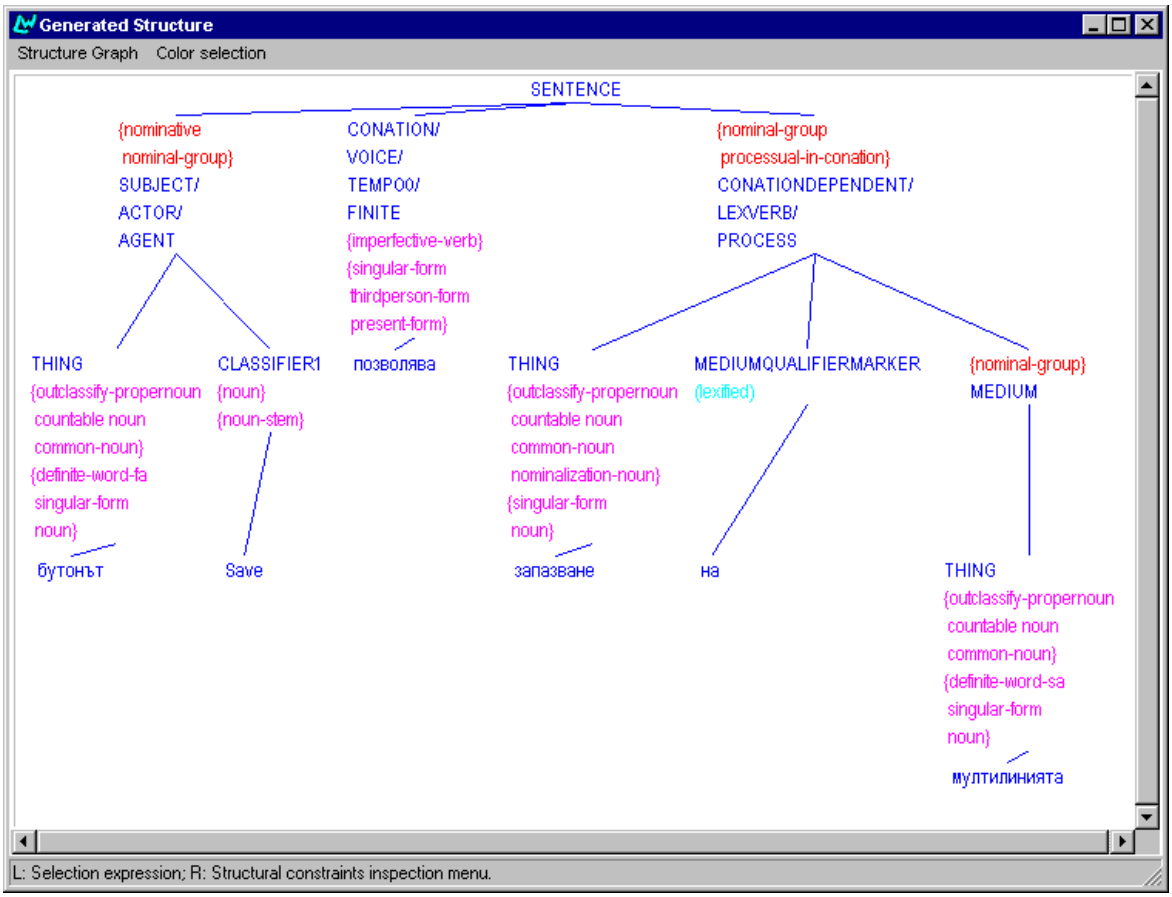


Figure 42: Grammatical structure generated from the SPL in Figure 41 by the Bulgarian grammar

3.4 Subject dropping

3.4.1 Description of the Phenomenon

Czech and Bulgarian are characterized as so-called *pro-drop* languages. This characteristic reflects the fact that the Subject in a declarative clause (indicative or interrogative mood) can be unexpressed/unrealized (on the surface).[10] It can also be said the Subject is left implicit, perhaps in a similar way as a Subject is left implicit in a sentence in imperative mood (also in English). Compare the examples below (we show also the Russian counterparts, which illustrate the fact that Russian patterns with English in this case as a language which does not drop Subjects):

(44) **Imperative mood**

(a) En: *Open the* Styles *dialog box.*

(b) Cz: *Otevřete dialogové okno* Styles.
Save-Pl2 dialogue window-Acc Styles.

(c) Bg: *Отворете диалоговия прозорец* Styles.
Open-Pl2 dialogue window Styles.

(d) Ru: *Откройте диалоговое окно* Styles.
Open-Pl2 dialogue window-Acc Styles.

(45) **Indicative: Declarative mood**

(a) En: *We open the* Styles *dialog box.*

(b) Cz: *Otevřete* | *Otevřeme dialogové okno* Styles.
Save-Pl2 | Save-Pl1Pl1 dialogue window-Acc Styles.

(c) Bg: *Отваряте* | *Отваряме диалоговия прозорец* Styles.
Open-Pl2 | Open-Pl1 dialogue window Styles.

---

[10] Even though the Subject is not expressed, the main finite verb in the sentence exhibits agreement with it in gender and number. This case of a sentence with an unexpressed Subject needs to be distinguished from the case of a sentence with no Subject at all. An example of the latter would be the following use of reflexive passive, where the Goal accompanying a non-transitive verb is realized by a prepositional phrase and therefore cannot become the Subject in passive voice, since only the Goals of transitive verbs (realised in accusative in active voice) can take the Subject function in passive voice.

Cz: Klikne se na tlačítko OK
Clicks refl on button OK
One clicks on the OK button. / The Ok button is clicked on.

In this case, in a clause with no Subject, the Finite element takes the third person singular form in Czech (Hana Skoumalová, p.c.).

John Bateman (p.c.) has suggested an alternative approach, starting from the view that the gender and number of the Finite element in a clause with a dropped Subject does not agree with any non-existent Subject, but instead serves exactly the same function as such markers in switch-reference languages; i.e., it indicates whether or not the Subject has changed or not. However, the dropped Subject can correspond also to entities from the preceding clause(s) that were not Subjects before, but which are just pronominalizeable. This adds an interesting extra twist to the switch-reference cases. Unfortunately, we have to defer the investigation of this alternative view to the furture.

(d) Ru: *Вы откроете | Мы откроем диалоговое окно* Styles.
Open-Pl2 | Open-Pl1 dialogue window -Acc Styles.

(46) **Indicative: Interrogative mood**

(a) En: *Do you open the* Styles *dialog box?*

(b) Cz: *Otevřete dialogové okno* Styles*?*
Save-Pl2 dialogue window-Acc Styles?

(c) Bg: *Отваряте ли диалоговия прозорец* Styles*?*
Open-Pl2 dialogue window Styles.

(d) Ru: *Вы откроете диалоговое окно* Styles*?*
You open-Pl2 dialogue window-Acc *Styles*?

However, the apparent parallel between imperative and indicative mood as far as Subject implicitness goes does not stretch very far, as it applies only to interactants: while in an ordinary imperative sentence, the Subject is one of the interactants, typically the hearer, in an indicative sentence, the Subject is as commonly an interactant as a non-interactant, and still it can be dropped. For example, consider the sentence in (47) below as a continuation after (45) above:

(47) **Indicative: Declarative mood** (cont. after (45))

(a) En: *It appears in the middle of the screen.*

(b) Cz: *Objeví se uprostřed obrazovky*.
Appears refl middle-Prep screen-Gen.

(c) Bg: *Появява се в център на екрана*
Appears refl in middle of screen.

(d) Ru: *Оно появится в центре экрана*
It appears-refl v middle screen-Gen.

The conditions for dropping a Subject appear to be very similar to the conditions for Subject-realization by a weak (reduced or unstressed) form of pronoun (in non-pro-drop languages like English). That is:

- The entity referred to by the Subject has to be *salient* in the context, either in the (immediately) preceding text or in the situation shared between the speaker and the hearer.
- The entity referred to by the Subject must be presented as *Given* (*contextually bound*, see the Section 3.5 for a detailed discussion of these notions) as far as information structure is concerned, and it must be *non-contrasted* with another entity.

Dropping of a Subject referring to a contextually salient entity in Czech and Bulgarian has been illustrated in the examples (45) and (47) above. In (45), the Subject refers to an *interactant*. The interactants in an exchange (in particular in an instructional text) can always be considered sufficiently salient. Therefore, the Subject referring to an interactant can always be dropped, unless it is presented as *New* or contrasted with another entity. Although it would be in general possible in the instructional texts to encounter contrast between an interactant and another entity (as illustrated in (48) and (49) below; bold depicts the Subjects), such cases do not occur in the instructional texts we are generating in Agile.

(48) Cz:
Systém *zobrazí obrys okna.* ***Vy*** *pak musíte určit jeho polohu.*
System displays frame window-Gen. You-nom then must-Pl2 specify its location-Acc
The system displays a window contour. You then have to specify its location.

(49) Cz:
Styl musíte definovat ***vy***.
Style-Acc must-Pl2 define you-Nom.
The style must be defined by you. / You must define the style yourself.

In (48), the interactant-Subject "vy" (you) as Given (contextually bound) as far as information structure is concerned, but it is contrasted with the non-interactant Subject of the preceding sentence, namely "systém" (system). In (49), on the other hand, the interactant-Subject is presented as New as far as information structure is concerned. Again, there is an element o contrast involved. However, in the Agile target texts, interactants are never presented as new and/or contrasted, and therefore Subjects referring to interactants always end up being dropped.

In Bulgarian, the situation is similar, as illustrated in the following parallel examples.

(50) Bg:
Системата изобразява рамката на прозореца.
System displays frame of window-det
The system displays the window contour.
Вие *трябва да окажете неговото място.*
You must-Pl2 specify its location-Acc
You have to specify its location.

(51) Bg:
Вие трябва да дефинирате стила.
you must define-Pl2 style-det
You must define the style yourself.

(52) Bg:
Стилът трябва да се дефинира от Вас.
style-det must-Pl2 refl define by you-Acc.
The style must be defined by you.

(53) *Стилът трябва да се дефинира от Вас.*
style-det must-Pl2 refl define-refl. by you-Acc
The style must be defined by you.

Unlike interactants who can be considered always salient, non-interactants *become salient* in a context. In general, non-interactants can either become salient *situationally* or they can be made salient *textually*. Even though the writer of an instructional text can presume that the reader can see particular screen-configurations or objects on the screen, the situational context is shared by the writer and the reader, and therefore, the situationally available entities cannot be considered sufficiently salient without explicitly introducing them into the

textual context. Therefore, situational salience is not enough to allow for Subject dropping in an instructional text,[11] as the next example illustrates:

(54) Cz:
Stiskněte Ctrl-O *pro otevření dokumentu. Vyberte soubor a klepněte na* OK.
Press Ctrl-O for opening document-Gen. Choose file-Acc and click on OK.
Press Ctrl-O to open a document Choose a file and click OK.

? *Zmizí.*
Disappears-Sg3
It disappears.

(55) Bg:
Натиснете Ctrl-O *за отваряне на документа.*
Press-Pl2 Ctrl-O for opening of document-det
Изберете файл и натиснете OK.
Choose file and click OK.
Press Ctrl-O to open a document Choose a file and click OK.

? *Изчезва.*
Disappears-Sg3
It disappears.

In both Czech and Bulgarian, it is impossible to drop the Subject of the last sentence on the basis of situational salience of the dialogue window. Due to the insufficiency of situational salience for Subject dropping in written instructions, only textual salience is considered in the texts generated in Agile.

Textual salience is usually thought of as being a matter of *degree* (see for example (Hajičová 1993), (Strube 1999)). Roughly speaking, entities referred to more recently in the text are considered more salient, while the salience of entities referred to earlier on fades away as the text progresses.

Hajičová (1993) also proposed to take information structure and the forms of referring expressions into account when determining the degree of salience of an entity once it is explicitly introduced into the context, or when it is subsequently co-referred. In addition, when a text is considered to consist of hierarchically related *segments*, and the segments are considered to constitute "focus spaces" (Grosz and Sidner 1986), the salience of entities referred to in an embedded segment may drop significantly when such an embedded segment is "closed off", while when a segment is "returned to", the salience of the entities referred to in it may raise again (recover).

However, Subject dropping is a rather *local* phenomenon[12]. It seems impossible to drop a non-interactant Subject unless it was explicitly referred to in an immediately preceding

[11] The insufficiency of situational salience for Subject dropping in written instructions constitutes an important difference between written and spoken instructions; in the latter, the speaker and the hearer share the same situational context, and therefore, situational salience may suffice to allow for Subject dropping.

sentence.[13] The following examples illustrate this: (56) provides the first sentence, and (a), (57)(a) and (57)(b) are alternative continuations.

(56) Cz: *Zadejte název souboru.*
Specify name file-Gen
Specify the name of the file.

(57) Cz:

(a) *Objeví se v horní liště okna editoru.*
Appears refl in top ledge window-Gen editor-Gen
It appears in the title bar of the editor window.

(a) ? *Stiskněte* Enter. *Objeví se v horní liště okna editoru.*
Press *Enter*. Appears refl in top ledge window-Gen editor-Gen
Press Enter. It appears in the title bar of the editor window.

(b) *Stiskněte* Enter. *Název se objeví v horní liště okna editoru.*
Press Enter. Name refl appears in top ledge window-Gen editor-Gen
Press Enter. The name appears in the title bar of the editor window.

While (56)+(a) is fine, (56)+(57)(a) does not appear very felicitous. Although it is still possible to figure out the referent of the dropped Subject in the last sentence on lexico-semantic grounds, our judgement is that Subject dropping is improper in this case. (56)+(57)(b), on the other hand, is a completely fluent and natural piece of continuous text.

In Bulgarian, the Subject cannot be dropped in any of continuations. Either a personal pronoun or a nominal group has tobe used, as illustrated beow:

(58) Bg: *Задайте име на файла.*
Specify name of file-det
Specify the name of the file.

(59) Bg:

(a) ***То*** *се появявя в заглавния ред на прозореца.*
it appears refl in top ledge of window-det
It appears in the title bar of the window.

(b) *Натиснете* Enter. ***То*** *се появявя в заглавния ред на прозореца.*
Press Enter. it appears refl in top ledge of window-det
Press *Enter*. It appears in the title bar of the window.

(c) *Натиснете* Enter. ***Името*** *се появявя в заглавния ред на прозореца.*
Press *Enter*. Name-def refl appears in top ledge of window-det
Press *Enter*. The name appears in the title bar of the window.

These examples illustrate that Subject dropping is more restricted in Bulgarian than in Czech. The generalization seems to be that Bulgarian allows Subject dropping only when refers to an entity which was referred to by the Subject of the immediately preceding clause.

---

[12] We are hereby leaving aside the cases of "permanently salient entities" in a text, entities which would constitute a *discourse topic*, and would therefore allow Subject dropping at any point in the text.

[13] This is another paralles with the so-called switch-languages (John Bateman, p.c.)

The observations made on these kind of textual continuations prompt to take the *Centering Theory* into consideration (Grosz et al. 1995). Centering Theory relates focus of attention, choice of referring expressions and the coherence of utterances within a discourse segment. In Centering Theory, an utterance has its *backward-looking center* (of attention), C_b, which links it to the preceding utterance, and a partially ordered *list of forward-looking centers*, C_f. The ordering of the C_f list reflects the relative prominence of the elements in an utterance. For English, it was argued to depend on grammatical roles, with SUBJECT>OBJECT>OTHER (having in mind, however, that other factors are also important, such as word order, clausal subordination, and lexical semantics). The ranking of the forward-looking centers affects the choice of the backward-looking center of the subsequent utterance. Centering theory expresses constraints on the realization of centers by linguistic expressions and on the transitions from one backward-looking center to another across utterances within a discourse segment. The connection between the C_b of utterance U_{n+1} and the C_f list of utterance U_n may be of several types:

1. CENTER CONTINUATION: $C_b(U_{n+1}) = C_b(U_n)$, and this entity is the most highly ranked element of $C_f(U_{n+1})$. In this case, $C_b(U_{n+1})$ is the most likely candidate for $C_b(U_{n+2})$; it continues to be C_b in U_{n+1}, and continues to be likely to be C_b in U_{n+2}.
2. CENTER RETAINING: $C_b(U_{n+1}) = C_b(U_n)$, but this entity is not the most highly ranked element of $C_f(U_{n+1})$. In this case, $C_b(U_{n+1})$ is not the most likely candidate for $C_b(U_{n+2})$; although it is retained as C_b in U_{n+1}, it is not likely to be C_b in U_{n+2}.
3. CENTER SHIFTING: $C_b(U_{n+1}) \neq C_b(U_n)$.

The coherence of a discourse segment is affected by the kinds of *centering transitions* engendered by a speaker's choices of linguistic realizations in the utterances constituting that segment. Sequences of continuation are preferred over sequences of retaining; and sequences of retaining are preferred over sequences of shifting.

Our speculative generalization concerning Subject dropping is as follows:

Subject dropping is licensed in U_{n+1} if and only if the Subject of U_{n+1} is a C_b and the transition from U_n to U_{n+1} is a center continuation.

However, there has not been any research that would investigate the applicability and/or adaptability of Centering Theory to Czech or any other Slavonic language so far, and carrying out such a more general research ourselves has turned out to be beyond our capacity within the Agile project. Besides, such more general investigation also did not appear necessary in order to describe the cases of Subject dropping occurring in the texts we are generating. Therefore, we cannot offer more than the above speculation at this moment. However, this is an interesting topic which we would like to pursue in the future.

Before we proceed to present the formal description of Subject dropping for the sake of the Agile Final Prototype, let us now briefly summarize the observations presented so far:

- **Interactants** are always salient. In our texts, interactant are always presented as Given (contextually bound) and are also never contrasted with another entity. A Subject realizing an interactant can therefore be always dropped.
- **Non-interactants** can have varying degree of salience as text progresses. From the point of view of Subject-dropping, the salience of a non-interactant entity decreases very quickly. Therefore, a Subject referring to such an entity can only

be dropped in the immediately succeeding sentence, and only if it is presented as Given and not contrasted with another entity.

In the next section, we present a formalization which reflects these conclusions.

3.4.2 Formal specifications

There are two possible alternative views of what Subject dropping means for the grammatical structure of a sentence:

- Every sentence is considered to have an explicit Subject constituent. In case of Subject dropping, this constituent either does not have any surface realisation or is realised by an empty string. Because of the presence of a Subject constituent in the grammatical structure, but the absence of a surface realisation for, Subject dropping is thus conceived of as a case of **surface deletion** under this view.
- The assumption that every indicative sentence has an explicit Subject constituent can be given up. In case of Subject dropping, the sentence is conceived of as having an **implicit subject**. As a result, there is no Subject constituent inserted into into the grammatical structure. Subject dropping in indicative mood is thus handled in the same fashion as the (typically) implicit subject in imperative mood.

Both views are subject of ongoing linguistic debate. Which of the two views one adopts depends on whether one's grammar theory allows for empty constituents of any kind or not.. The implicit subject view would be consistent with a theory like Steedman's Combinatory Categorial Grammar (Steedman 1996), where no traces or other empty categories are assumed. Although we are not aware of any treatment of subject dropping in SFG, it seems to us that the implicit subject view would also be consistent with SFG.

The surface deletion view is, for example, adopted in the Functional Generative Description (FGD, Sgall et al. 1986, Hajičová et al. 1999). However, FGD is a multi-level approach; a node corresponding to the Subject is present in the underlying tectogrammatical representation (deep structure), but not necessarily in any of the representations at the levels above tectogrammatics, towards the surface realization.

For the sake of a comparison, we present here a formalization of both views. In the implementation we have developed so far and we describe in Section 3.4.3, we currently adopt the surface deletion view.

3.4.2.1 Formalization of the Implicit Subject Approach

In this approach, we it not assumed that every indicative clause has an explicit Subject constituent. Subject is inserted in the system SUBJECT-INSERTED:

```
System: SUBJECT-INSERTED →
          [SUBJECT-INSERTED] (+SUBJECT)
```

The input features of this system cover the cases where a subject is obligatory, and the cases where subject is not dropped. The cases of obligatory subject are reflected in the SUBJECT-INSERTION system:

```
System: SUBJECT-INSERTION → [DO-INSERT-SUBJECT]
```

In the cases where subject dropping is possible, the SUBJECT-DROPPING system below decides whether the subject can be dropped. The **subject-notdrop** feature corresponds to an overt subject, the **subject-drop** feature corresponds to the subject being implicit. In this case, no Subject is inserted.

```
System: SUBJECT-DROPPING →
            [SUBJECT-DROP]
            [SUBJECT-NOTDROP]
```

The decision choice between **subject-drop** and **subject-notdrop** is based on criteria discussed above, referring to information structure (contextual boundness or nonboundness), the presence or absence of contrast and the degree of salience.

```
Chooser: SUBJECT-DROPING-CHOOSER
    If the entity is contextually nonbound
      Then choose subject-notdrop
        Else
          If the entity is contrasted
            Then choose subject-notdrop
            Else
              If the entity is an interactant
                Then choose subject-drop
                Else
                  If the entity is salient
                     Then choose subject-drop
                     Else choose subject-notdrop
```

In the **subject-drop** case, there is no Subject constituent inserted into the grammatical structure. Subject dropping in indicative mood is thus handled in the same fashion as the (typically) implicit subject in imperative mood.

3.4.2.2 Formalization of the Surface Deletion Approach

Following this approach, a Subject is inserted in any case, so we do not need the system SUBJECT-INSERTION. Its input features are included in the input condition of the system SUBJECT-INSERTED:

```
System: SUBJECT-INSERTED → [SUBJECT-INSERTED]
```

Next, it has to be decided whether the subject can be dropped. This decision concerns only **nominative** Subjects. Non-nominative Subjects, e.g. propositions, cannot be dropped. When they are presented as Given (contextually bound) and non-contrasted, they may be pronominalized (for example, a proposition can be referred to by the personal pronoun "to" or a demostrative pronoun, e.g. "toto" in Czech, in the same way as in English propositions can be referred to anaphorically by "it" or "this"). Therefore, the decision whether or not to drop a Subject in the SUBJECT-DROPPING system is made for nominative Subjects only.

```
System: SUBJECT-DROPING →
        [SUBJECT-DROP] (Subject:nominal-group-drop)
        [SUBJECT- NOTDROP]
Chooser: SUBJECT-DROPING-CHOOSER
    If the entity is contextually nonbound
      Then choose subject-notdrop
```

```
Else
  If the entity is contrasted
    Then choose subject-notdrop
    Else
      If the entity is an interactant
        Then choose subject-drop
        Else
          If the entity is salient
             Then choose subject-drop
             Else choose subject-notdrop
```

Again, the **subject-notdrop** feature corresponds to an overt subject, the **subject-drop** feature corresponds to the subject being implicit. In this case, a preselection is made, which will lead to the corresponding decision at a lower rank.

The lower rank is that of a nominal group. We have the following two possibilities:

- to insert Thing in any case, and subsequently lexify it as an empty string (the preselection under the feature **nominal-group-drop** serves the purpose of making the right choice at the next lower rank)

```
System: NOMINAL-GROUP-SIMPLEX →
     [NOMINAL-GROUP-SIMPLEX] (+THING)
System: NOMINAL-GROUP-SIMPLEX-DROP →
     [NOMINAL-GROUP-NOTDROP]
     [NOMINAL-GROUP-DROP] (Thing:drop-pronoun)
```

- not to insert a Thing at all

```
System: NOMINAL-GROUP-SIMPLEX →
     [NOMINAL-GROUP-SIMPLEX] (+THING)
     [NOMINAL-GROUP-DROP]
```

The decision whether to choose **nominal-group-drop** is driven by a preselection from the higher rank (clause) in both cases. If no preselection has been made, the default choice is for the other feature, that is for **nominal-group-simplex** in the first case, or **nominal-group-notdrop** in the second case.

```
Chooser: NOMINAL-GROUP-SIMPLEX-CHOOSER
    If there are preselections
        Then choose according to the preselections
        Else choose nominal-group-simplex by default
Chooser: NOMINAL-GROUP- DROP-CHOOSER
    If there are preselections
        Then choose according to the preselections
        Else choose nominal-group-notdrop by default
```

In the following section we describe the implementation we have adopted.

3.4.3 Implementation

3.4.3.1 Implementation in the Czech Grammar

We described three possible solutions to handle Subject dropping in the preceding section:

- Implicit subject approach

- Explicit subject approach either with a Thing inserted and realized by an empty string, or with no Thing inserted at all

Once again, we are thus facing a decision which requires balancing out alternative solutions each of which introduces some additional difficulties.

The implicit subject has the following technical complication: the implicit subject approach requires extensions in the region handling agreement between Subject and Finite. This region currently relies on a Subject being inserted in all indicative clauses, and it then ensures that the values of the relevant congruence features pertaining to the Subject (i.e., number and gender) are handed over to the Finite element of the clause. Therefore, if no Subject is inserted, the values of these features if the Finite element cannot be determined. This is not an obstacle that cannot be overcome, if one decides for the implicit subject approach. However, at the moment, we do not (yet) have these additional implementations.

Once these additional implementations regarding number and gender determination of the Finite element would be available, they could be employed to handle number and gender determination for the Finite element in both indicative and imperative clauses with implicit subject. This solution would then have the nice property that it would reflect the parallelism between explicitness vs. implicitness of Subject in indicative and imperative mood we noted in Section 3.4.1.

Because we currently do not have the additional implementations concerning gender and number of the Finite element, which would enable us to adopt the former solution, we opt for a solution involving empty elements for the time being. Either of the two described empty-element approaches has the following disadvantage: It complicates word ordering. In particular, empty elements complicate the ordering of such elements which are fixed to or preferred at a particular position in a clause. This is the case of clitics in Czech, as they have to occur in the so-called Wackernagel position, roughly characterized as the second position in a clause (see the section on word order in this deliverable and in LSPEC2). Another such case is the verb-secondness preference. The complication arising from empty elements is that any empty element still counts as an element when in fact it should not, because it is empty. More complex handling of ordering is therefore required, which takes emptiness of an element occupying the first position into account: the Wackernagel position is only after the first non-empty element.

Let us now present the implementation of one of the empty-element solutions, namely the one which involves inserting a Subject and not inserting any Thing element, as implemented in the Czech grammar. As we said above, the decision whether to drop or not drop the Subject is relevant for nominative subjects, and is made in the system SUBJECT-DROPPING, see Figure 43. The corresponding chooser is shown in Figure 44.

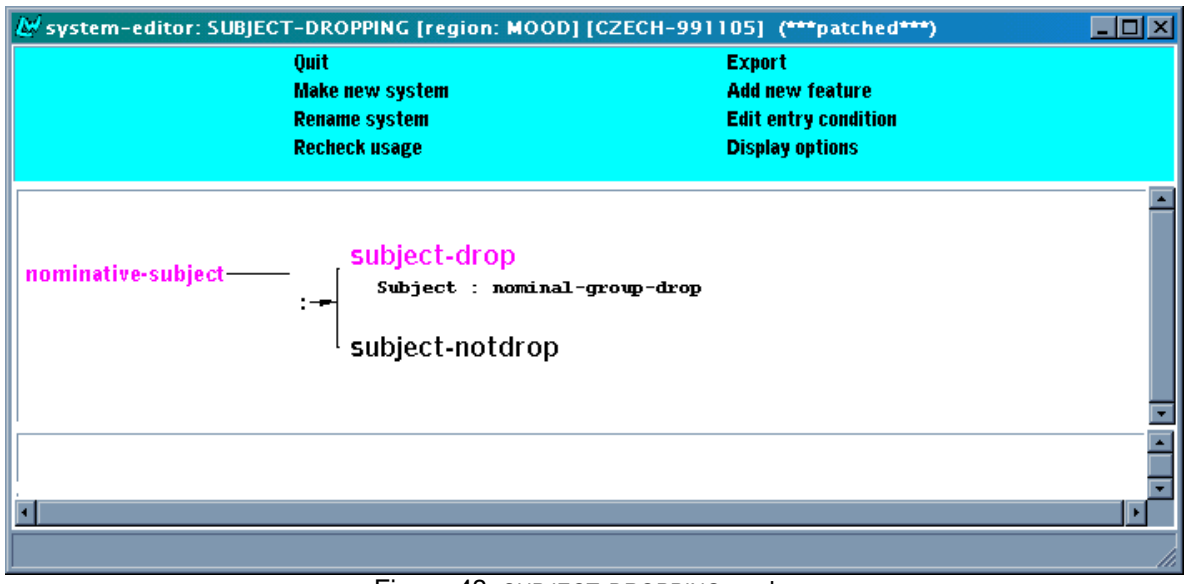


Figure 43: SUBJECT-DROPPING system

As we said above, the preselection which is made when the feature **subject-drop** is chosen determines the decision whether or not to insert Thing, which is made at the nominal-group rank, in the NOMINAL-GROUP-SIMPLEX system, see . The corresponding chooser (Figure 44) just needs to specify what is the default choice in case there is no preselection coming from the higher rank.

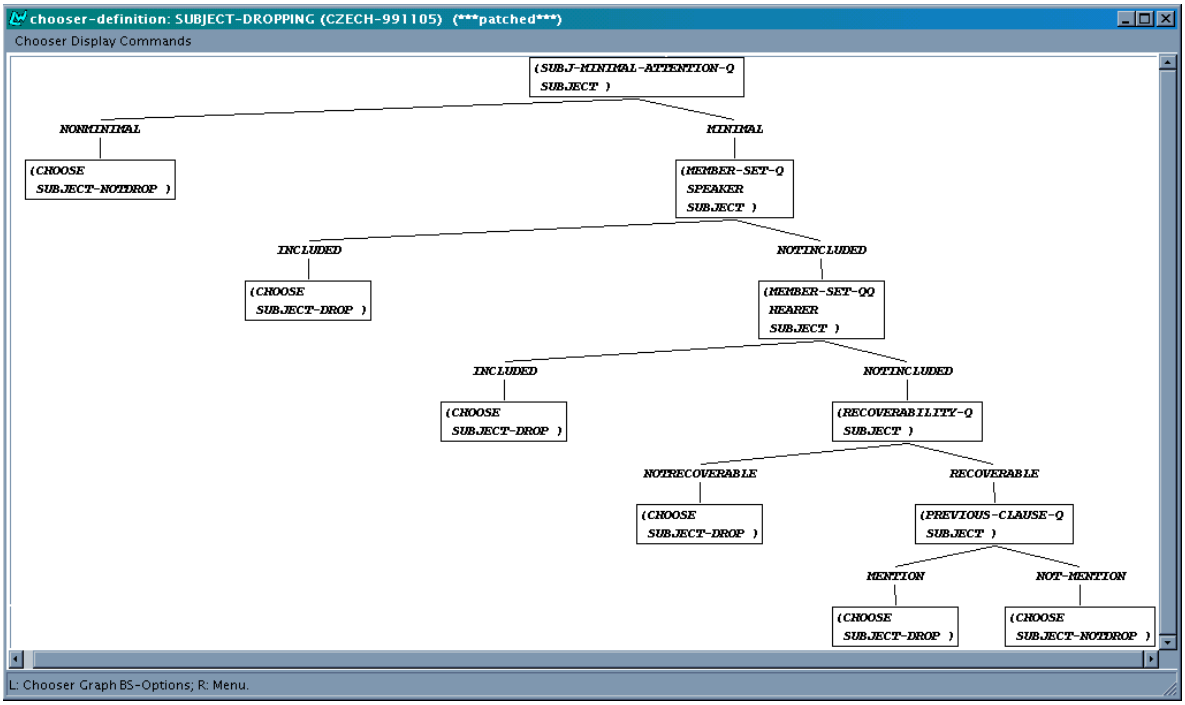


Figure 44: SUBJECT-DROPPING chooser

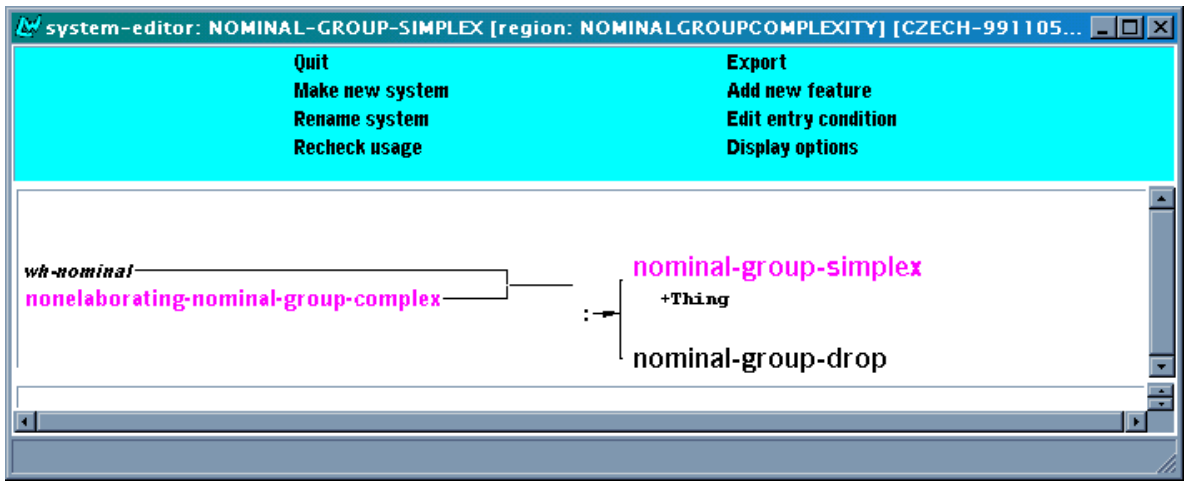


Figure 45: NOMINAL-GROUP-SIMPLEX system

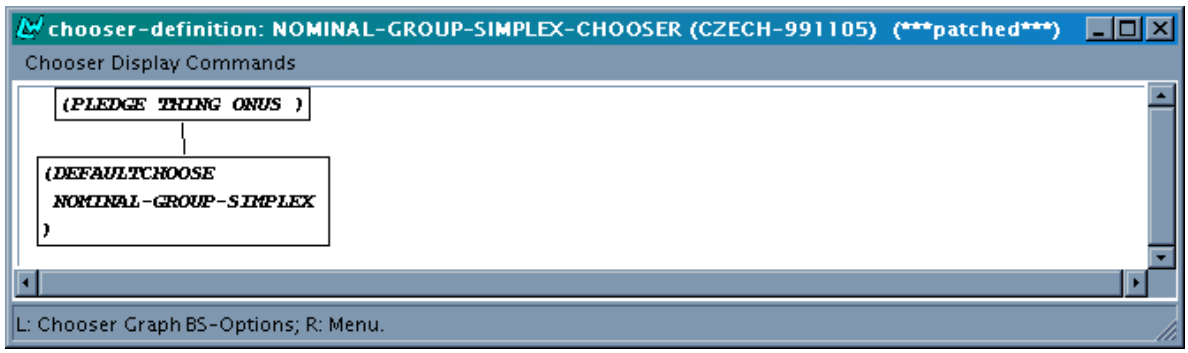


Figure 46: NOMINAL-GROUP-SIMPLEX chooser

Let us now illustrate Subject droping vs. not-dropping by providing several examples of generated sentences and the corresponding input SPLs. Figure 47 and Figure 48 show the generated grammatical structures with dropped Subject in (60) and (61), respectively.

(60) Cz:

(a) *Přidáte* *obrázek.*
Add-perfective.Pl2-present drawing-Acc.NB
You add a style.

(b) SPL:

```
(|a121| / DM::ADD
 :ACTEE (|a105| / DM::DRAWING )
 :ACTOR (HEARER / DM::USER
                  :IDENTIFIABILITY-Q DM::IDENTIFIABLE ) )
```

(61) Cz:

(a) *Vy přidáte* *obrázek.*
You add-perfective.Pl2-present drawing-Acc.NB
You add a style.

(b) SPL:

```
(|a121| / DM::ADD
 :ACTEE (|a105| / DM::DRAWING )
 :ACTOR (HEARER / DM::USER
          :SUBJ-MINIMAL-ATTENTION-Q NONMINIMAL
          :IDENTIFIABILITY-Q DM::IDENTIFIABLE
          :CONTEXTUAL-BOUNDNESS yes ) )
```

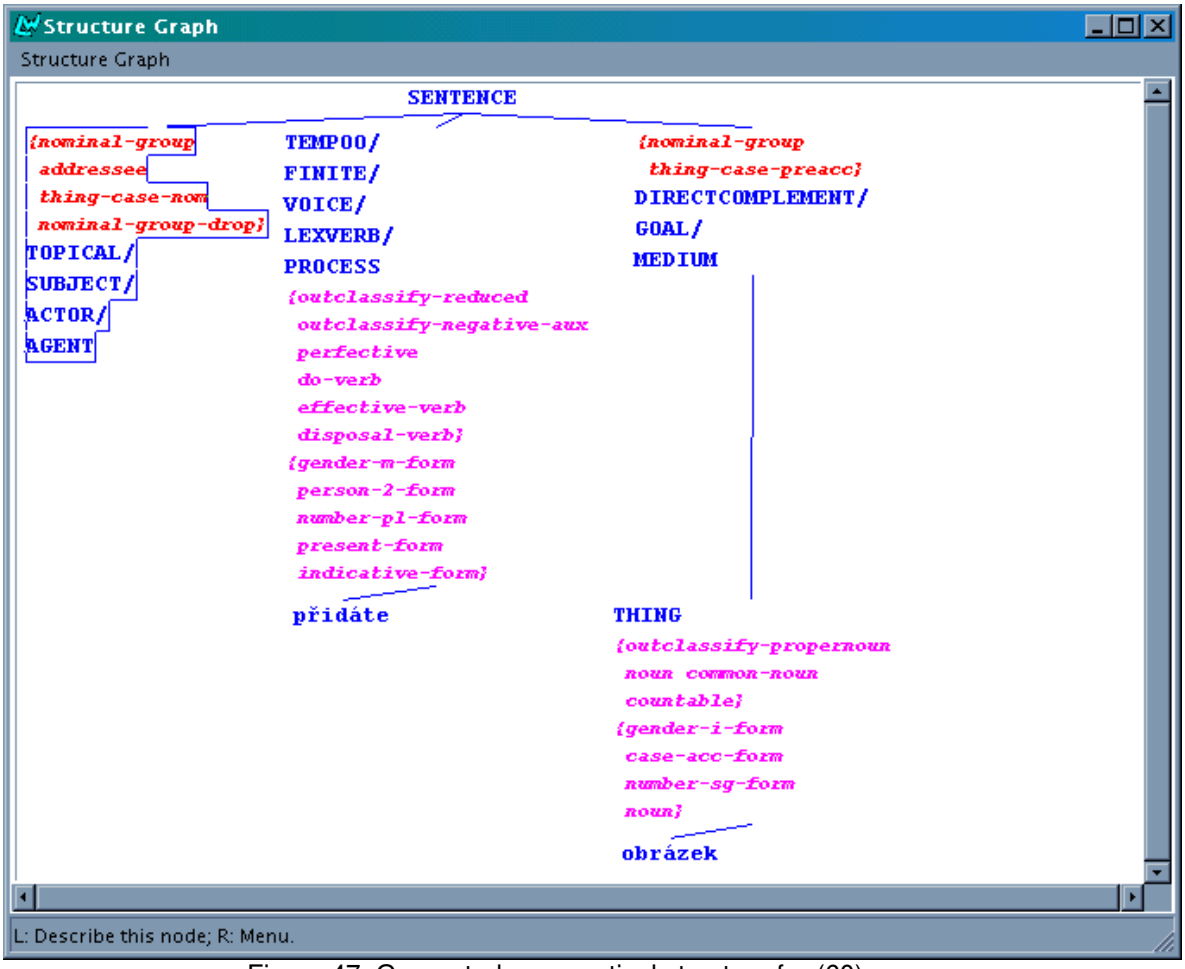


Figure 47: Generated grammatical structure for (60)

(62) Cz:

(a) *Přidá* *obrázek.*
Add-perfective.Sg3-present drawing-Acc.NB
It adds a style.

(b) SPL:

```
(|a121| / DM::ADD
 :ACTEE (|a105| / DM::DRAWING )
 :ACTOR (SYSTEM / DM::OPERATING-SYSTEM :LEX SYSTE2M
          :RECOVERABILITY-Q DM::RECOVERABLE
          :PREVIOUS-CLAUSE-Q DM::MENTION
          :CONTEXTUAL-BOUNDNESS yes) )
```

(63) Cz:

(a) *Systém přidá* *obrázek.*
System adds Add-perfective.Sg3-present drawing-Acc.NB
System adds a style.

(b) SPL:

```
(|a121| / DM::ADD
 :ACTEE (|a105| / DM::DRAWING )
 :ACTOR (SYSTEM / DM::OPERATING-SYSTEM :LEX SYSTE2M
          :RECOVERABILITY-Q DM::RECOVERABLE
          :PREVIOUS-CLAUSE-Q DM::NOT-MENTION
          :CONTEXTUAL-BOUNDNESS yes) )
```

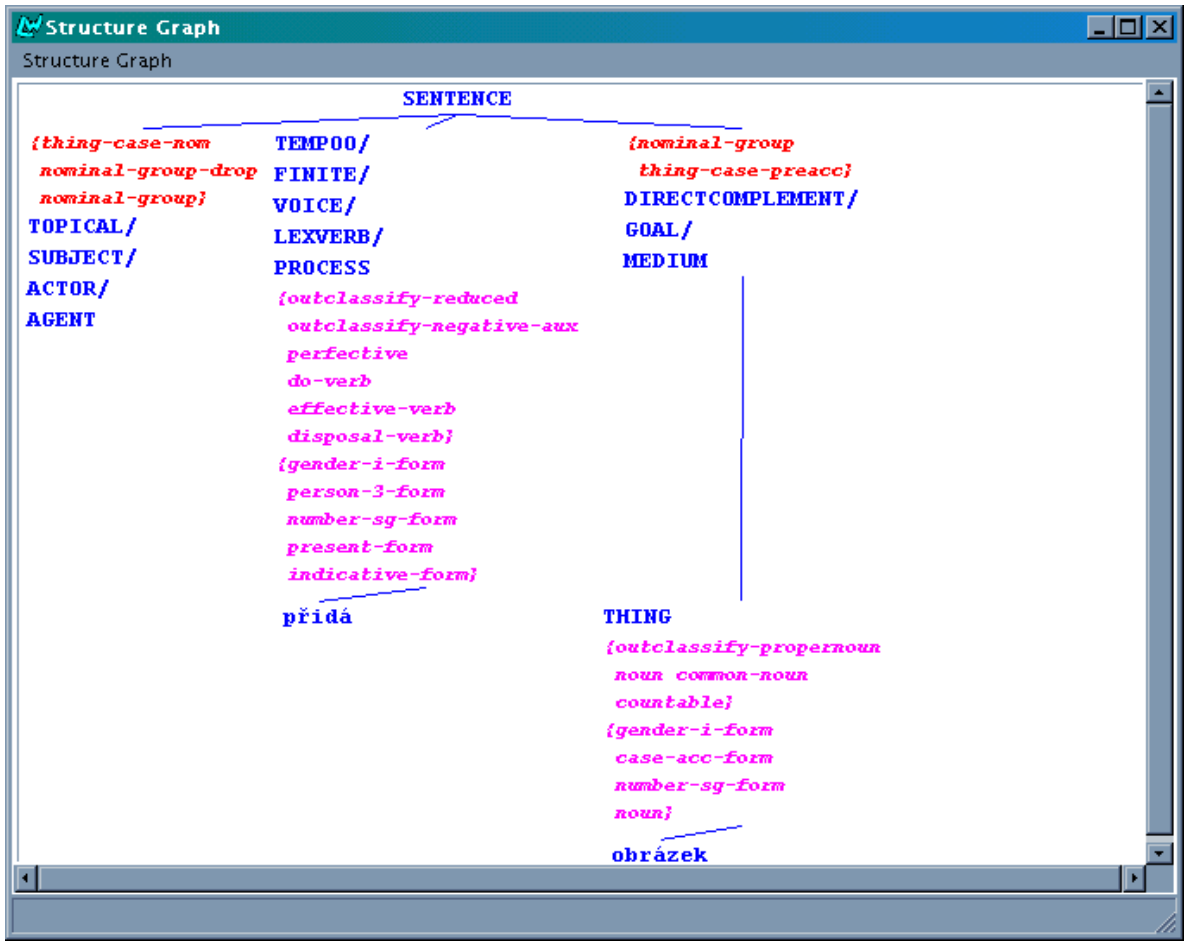


Figure 48: Generated grammatical structure for (61)

3.4.3.2 Implementation in the Bulgarian Grammar

For the purposes of generation constrained to the texts variations of AGILE Final Prototype, we used in Bulgarian grammar the same simple way as shown for Czech grammar to achieve generation of indicative clauses with and without Subject in the surface realization.

We show bellow two examples to demonstrate both cases.

(64) Bg: (Figure 49)

```
(|a121| / DM::ADD
 :ACTEE (|a105| / DM::STYLE )
 :ACTOR (HEARER / DM::USER
              :IDENTIFIABILITY-Q DM::IDENTIFIABLE ))
```

(65) Bg: (Figure 50)

```
(|a121| / DM::ADD
 :ACTEE (|a105| / DM::STYLE )
 :ACTOR (HEARER / DM::USER
         :SUBJECT-MINIMAL-ATTENTION-Q MINIMAL
         :IDENTIFIABILITY-Q DM::IDENTIFIABLE ) )
```

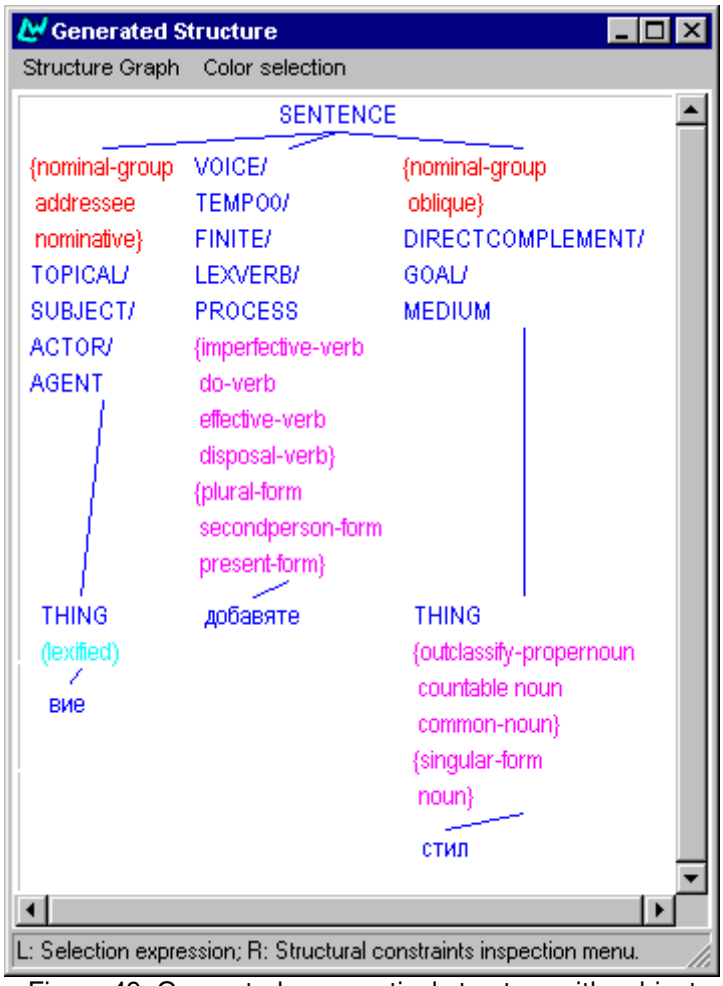


Figure 49: Generated grammatical structure with subject

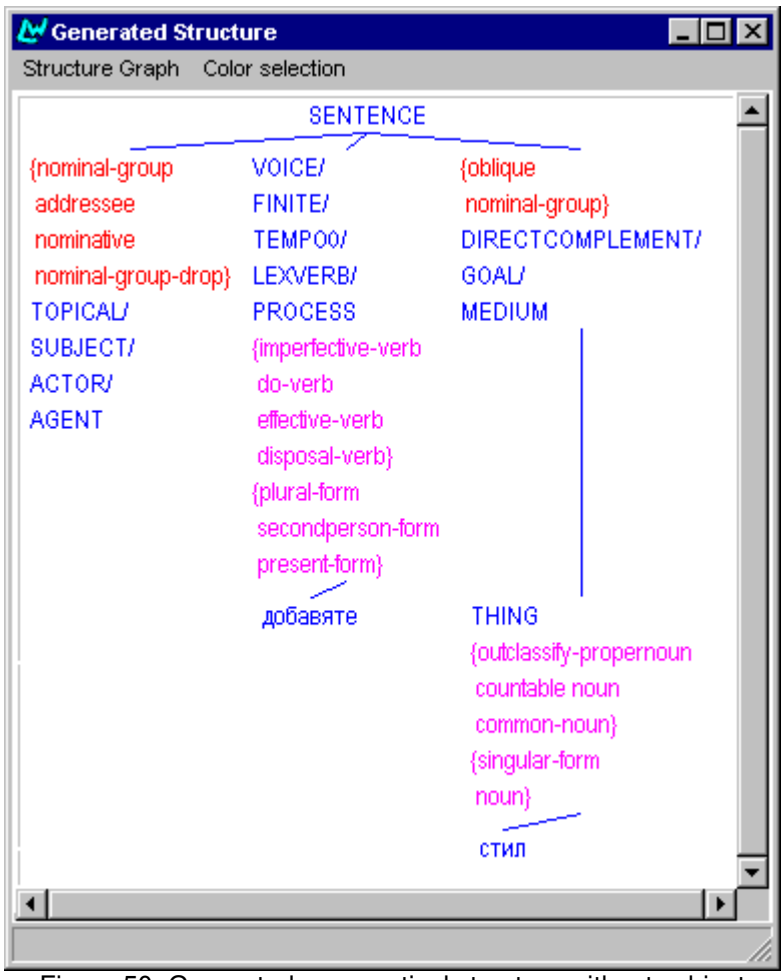


Figure 50: Generated grammatical structure without subject

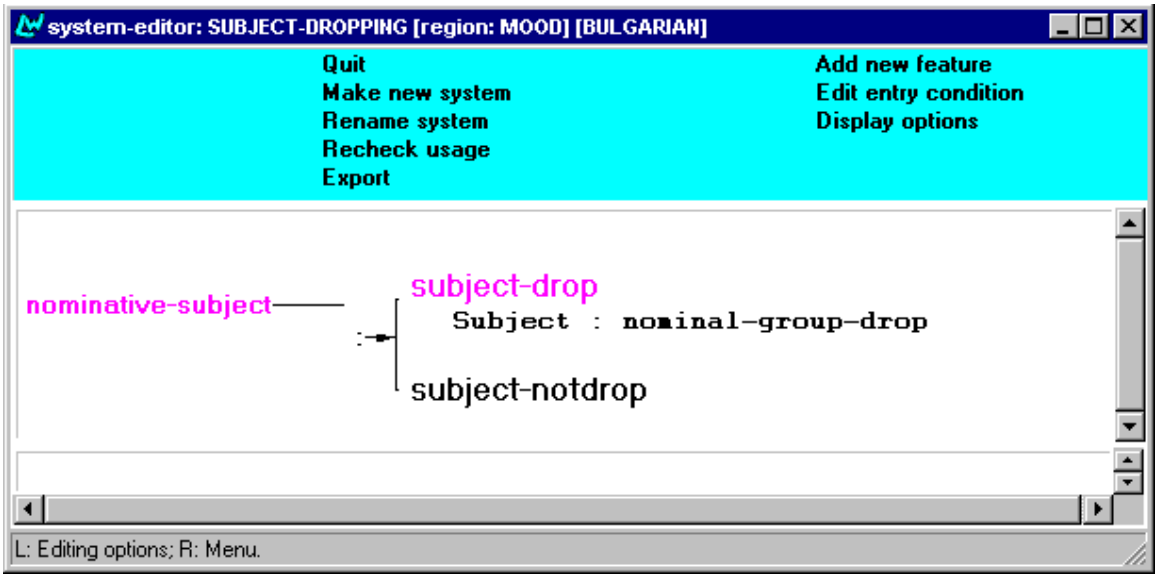


Figure 51: SUBJECT-DROPPING system (Bg)

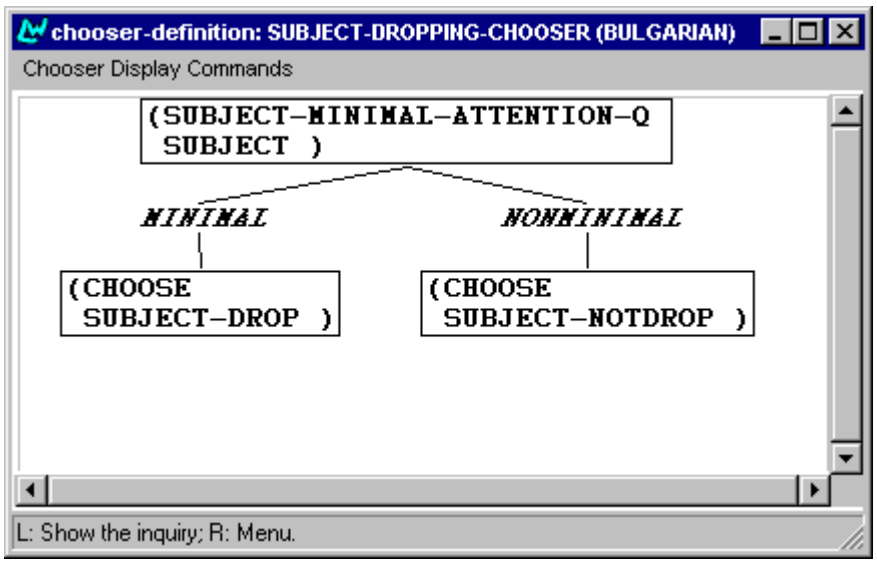


Figure 52: SUBJECT-DROPPING chooser (Bg)

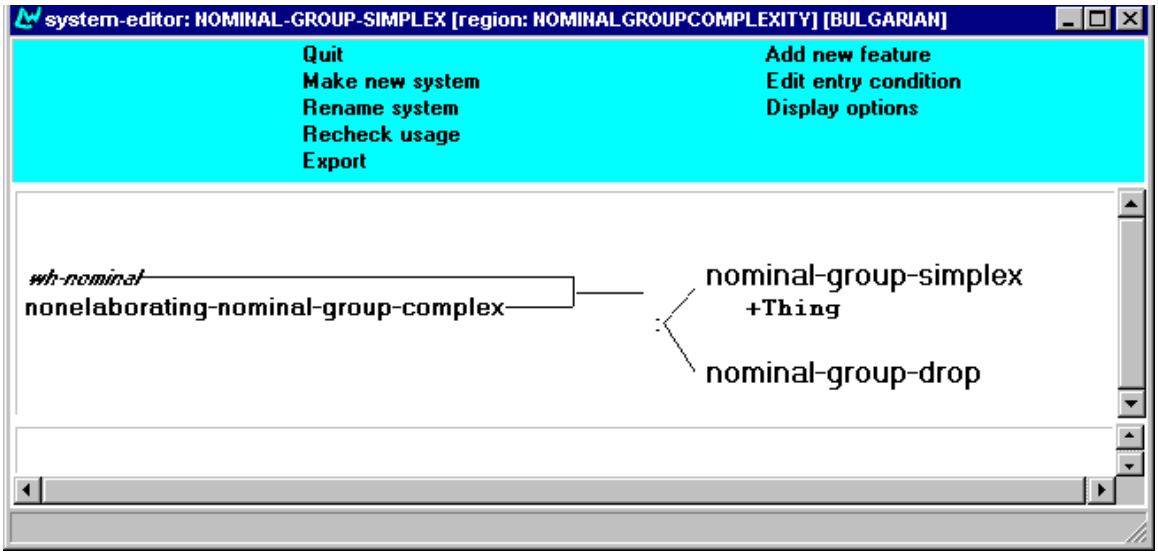


Figure 53: NOMINAL-GROUP-SIMPLEX system (Bg)

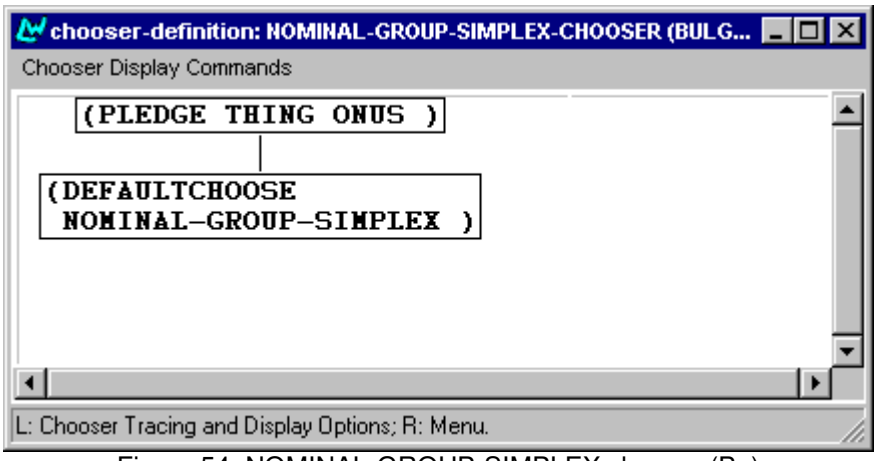


Figure 54: NOMINAL-GROUP-SIMPLEX chooser (Bg)

3.5 Word order

The problem of generating correct ordering of clause elements is a very important and challenging one in NLG, especially when the generation of connected, natural text (spoken or written) is concerned. Without a proper handling of word order any generated text is much less fluent than it should be, and wrong word ordering can even lead to misinterpretation. Well known examples for English include interpretations that assign differing preferred quantifier scopes depending on the order in which the linguistic expressions introducing those quantifiers appear in the surface string; others include controlling preferred sites of prepositional phrase or relative clause attachment by careful ordering. These problems, significant though they are for English, become substantially more important when we consider languages with higher degrees of **word order freedom**, like the Slavic languages. The issue of controlling word order selection for such languages must be addressed in NLG because the various word order variants of a sentence, although *grammatically well-formed*, do not necessarily have the same meaning and are generally not interchangeable in a given context. Finding the ordering appropriate for a given context is then an essential NLG task for such languages.

A detailed discussion of the phenomena concerning ordering elements at the clause level has been presented in LSPEC2,[14] and we rehearse one example in Section 3.5.1 below.

Although Slavic languages are in general characteristic by allowing a relative high degree of word order freedom, this does not mean that word order is arbitrary. On the contrary, it reflects the current discourse context and the communicative intention(s). Differences in word order very often correspond to differences in **information status** of the entities and processes about which the text is, in particular whether they are already familiar or not, and whether they are assumed to be salient for the addressee. But also, like in other languages, word order reflects text organization which have to do with the way how some information is to be presented to the addressee.

Considerations of this kind are reflected in the SFG framework In particular, SFG recognizes the **thematic structure** and the **information structure**, and they both have to do with ordering of constituents with respect to the larger context. In the LSPEC2 and IMPL3 deliverables, we proposed elaborate the Systemic Functional Grammar (SFG, Halliday 1985) treatment concerning information structure by incorporating insights from the Praguian framework of Functional Generative Desription (FGD, Sgall et al. 1986), and we proposed an ordering algorithm based on the combined insights.

However, the approach proposed in the LSPEC2 and IMPL3 deliverables was not implemented in the Intermediate Prototype, because KPML2.0 did not include the implementation of handling the **:contextual-boundness** parameter in SPLs. This is supported in KPML2.1 and higher, so we are now in the position that we can apply our flexible ordering algorithm.

In this chapter, we describe the particular approach to handling word order we have developed within the AGILE project. We first briefly illustrate the word ordering phenomena in our three languages as they were discussed in LSPEC2, in order to facilitate

[14] See also (Kruijff-Korbayová et al. in prep.).

the readers' understanding of the notions we use in the ordering algorithm and the problems the algorithm is aimed to solve (Section 3.5.1). Then we briefly recapitulate our text planning strategy and the interface between the text planner and the sentence generator, in order to make clear the interaction between the modules involved in text generation (Section3.5.2).

In Section 3.5.3 we present the flexible ordering algorithm, which operates on the elements in a clause and takes both grammatical constraints on ordering and information structure into account. This algorithm has been applied to Czech (Section 3.5.3.2) and Russian (Section 3.5.3.3). The main characteristics of the approach to word ordering we adopt in the Czech and Russian grammars in Agile can be summarized as follows: We preserve the SFG notion of Theme as appearing at the beginning of a clause and being determined by text organization. For ordering of non-thematic constituents within a clause we use notions adopted from FGD, namely the so-called systemic ordering in combination with the distinction between contextually bound and non-bound elements. Contextually bound elements are ordered separately from the non-bound ones. The ordering of the latter follows systemic ordering. The ordering of the former does not have to follow systemic ordering, but we use systemic ordering as a default.

Besides the constraints on word order imposed by information-structure, each language imposes certain amount of structural constraints. As we said earlier, individual languages differ in the degree to which word order is constrained by the grammar or "free" (i.e., determined by information structure). We address the issue of structural constraints in Section 3.5.4. In Czech, the need for structural constraints arises with respect to the placement of clitics, as they occupy the so-called Wackernagel's position, i.e. second position in a clause. We describe the handling of clitics in the Czech grammar in Section 3.5.4.1. For Bulgarian, the word ordering algorithm using information structure has not been employed in Agile, but rather, all ordering is done by structural constraints imposed by the grammar, essentially in the same way as in the Nigel grammar for English. The implementation of word ordering in the Bulgarian grammar is described in Section 3.5.4.2.

3.5.1 Word Order Freedom in Czech, Russian and Bulgarian

Sentences which differ only in word order (and not in syntactic realizations of constituents) are not freely interchangeable in a given context in Czech, Russian and Bulgarian. At the same time, the degree of word order freedom is not the same in across the three languages under consideration. In order to recall the differences in meaning that are often reflect by differences in word order, let us now consider another example in all three languages. These examples illustrate how word order reflects information structure in our languages.

(66) Cz:

(a) *Otevřete soubor příkazem* Open.
Open-imp file-Acc command-Ins Open
Open a|the file by the Open command.

(b) *Soubor otevřete příkazem* Open.
File-Acc open-imp command-Ins Open.
Open the file by the Open command.

(c) *Příkazem* Open *otevřete soubor.*
command-Ins Open open-imp file-Acc
By the Open command open a file.

(d) *Příkazem* Open *soubor otevřete.*
command-Ins Open file-Acc open-imp
By the Open command open the file.

(e) *Otevřete příkazem* Open *soubor.*
open-imp command-Ins Open-imp file-Acc
Open a file by the Open command.

(f) *Soubor příkazem* Open *otevřete.*
File-Acc command-Ins Open open-imp.
Open the file by the Open command.

The sentence in (66)(a) is neutral. It can be used "out of the blue", or in a context which can be approximated by the question *What should we do?* There are no implicatures concerning the presence of a file, or the identity of the file to be opened. So, both file and command are presented as contextually non-bound.

The sentence in (66)(b) is appropriate when some file is salient, for instance when the user is working with a file, because it means opening of a specific file. That is why we put the definite article into the English translation. The action of opening can, but does not have to, be salient, too. The contexts in which (9b) can be appropriately used can be characterized by the questions *What should we do with the file?* or *How should we open the file?* This means that the file is presented as contextually bound in this sentence, while the command is presented as non-bound.

The verb form in (66) (b) is homonymous in imperative and declarative mode. The declarative sentence could also by used in a generic meaning, equivalent to *A(ny) file can be opened by the* Open *command,* but in this case also the verb would have to be interpreted not as bound to the particular situation and one event of opening. We concentrate on the meanings concerning one particular action of opening throughout the example in (66).

Also (66) (f) presumes salience of a file, and also of the Open command. So, they are both presented as contextually bound. The contexts in which (66) (f) can be appropriately used can be characterized by the questions *What should we do to the file with the* Open *command?*

(66) (c,d,e) all presume the salience of the Open command. (66) (d) presumes also a salient file, that is why we used the definite article in English. The contexts in which (66) (c) can be used are characterized by the question *What should we do by the* Open *command?*. It is also possible to use (66) (c) in a context characterized by the question *What should we*

do? if it is somehow presumed that we are talking about using various commands (or various means or instruments) to do various things. In the latter type of context, the Open command in particular does not have to be salient. (66) (c) does not indicate specificity of the file. So, the file is presented as non-bound, while the command is presented as contextually bound.

For (66) (d), the appropriate contexts are characterized by the question *What should we do with the file by the* Open *command?* In this case both file and command are presented as contextually bound.. This is why this version is the most restricted.

(66) (e) presumes the salience of the opening action and of the *Open* command. The contexts for its use can be characterized by the question *What should we open by the* Open *command?* So, this sentence presents the command as contextually bound and the file as non-bound.

(67) Ru:

(a) *Откройте файл командой Open.*
Open-imp file-Acc command-Ins Open
Open a|the file by the Open command.

(b) *Файл откройте командой Open.*
File-Acc open-imp command-Ins Open.
Open the file by the Open command.

(c) *Командой Open откройте файл.*
command-Ins Open open-imp file-Acc
By the Open command open a|the file.

(d) ?Командой Open файл откройте.
command-Ins Open file-Acc open-imp
By the Open command open the file.

(e) ?*Откройте командой Open файл.*
open-imp command-Ins Open file-Acc
Open a file by the Open command.

(f) **Файл командой Open откройте.*
File-Acc command-Ins Open open-imp.
Open the file by the Open command.

(g) *Файл открывается командой Open.*
File-Acc open-3-pers-refl command-Ins Open.
A|the file is opened by the Open command.

(h) *Команда Open открывает файлы.*
command-nom Open open-3-pers files-PlAcc
The Open command opens **a** file.

As well as in Czech, the word order of the Russian (67) (a) is neutral, while (67) (d-f) are impossible in a written text. (67) (b) and (67) (c) are possible with approximately the same acceptability conditions as in Czech (note the difference that (67) (c) does *not* indicate specificity of the file, this example rather discusses possibilities of the Open command). However, wide use of impersonal constructions in Russian texts in this register suggests expressions of meanings corresponding to (66) (b) and (66) (c) as (67) (g) and (67) (h) (cf.

the sections on diathesis in this report). Generally the non-specific reading of the noun *file* can be emphasised in these examples by use of the plural number, while the use of specific determiners like *этот* (*this*) suggests definite reading, for example, in (67) (b) *Этот файл откройте командой* Open (cf. the discussion of determination in this report).

(68) Bg

(a)

(i) *Отворете файл с командата Open.*
Open-imp file-indef with command-def Open
Open a file by the Open command.

(ii) *Отворете файла с командата Open.*
Open-*imp* file-def with command-*def-Ins Open*
Open the file by the Open command.

(b) *Файла отворете с командата Open.*
File open-imp with command-def Open.
Open the file by the Open command.

(c)

(i) *С командата Open отворете файл.*
With command-def Open open-imp file-indef
By the *Open* command open a file.

(ii) *С командата Open отворете файла.*
With command-def- Open open-imp file-def
By the *Open* command open the file.

(d) * С командата Open файла отворете.
With command Open file-def open-imp
By the Open command open the file.

(e)

(i) *Отворете с командата Open файла.*
Open-*imp* with command *Open-imp* file-*def*
Open the file by the *Open* command.

(ii) Отворете с командата *Open* файл.
Open-imp with command Open-imp file-indef
Open a file by the Open command.

(f) * *Файла с командата Open отворете.*
File with command Open open-imp.
Open the file by the Open command.

(g) *Файлът се отваря с командата Open.*
File refl opens with command-def Open.
The file is opened by the Open command.

(h) *С командата Open се отваря файл.*
With command-def Open refl open-refl file
By the Open command is opened a file.

The sentences (68)(d) and (68)(f) are ungrammatical in Bulgarian. The word order in sentence (68)(a) shows the default word order in Bulgarian: Lexverb-Directcomplement-Means. It is important whether Directcomplement element (Goal) has determination marker.

Marker's presence indicates that the element "file" is already specified, contextually bound (68)(a(ii)). In the sentence of example (68)(a(i)) the element "file" is not determined and contextually non-bound. In Bulgarian CB and NB elements appear in the same position after the verb.

In the sentence shown on (68)(b) the variant with non-determined Directcomplement element (Goal) in the first position is not possible, because the first position is reserved for Theme in a sentence. "File" is contextually bound element, while "the command" is contextually non-bound element. In written Bulgarian reflexive-passive verb form is preferred (68)(g) to express the meaning of "usuality": "The file is usually opened by Open command".

Two clause variants with one and the same word order are possible in example (68)(c). The first one (68)(c(i)) has the meaning of "*A(ny) file can be opened by the* Open *command"*. "File" is non-bound element there, while the command is contextually bound element. The semantics of this clause is often expressed by reflexive-passive verb form (68)(h) if the imperative is not obligatory fixed or needed. The second clause (68)(c(ii)) contains two contextually bound elements "command" and "file". "Command" takes the first position in the clause being the Theme. "File" is ordered among the other clause elements according to the default Lexverb-Directcomplement order.

Two clause variants containing determined or not-determined "file" element are possible with word order shown in example (68)(e). In the sentence (68) (e(i)) "file" element is contextually non-bound. In the second variant (68)(e(ii)) "file" element is already specified i.e. contextually bound. When is put in the last position in the clause, the element "file" has to be interpreted as the "new" information and carries intonational marker. These two clauses shows, that the default order Lexverb-Directcomplement-Means could be changed to Lexverb-Means-Directcomplement order, but the context and communicative purposes of such usage should be strongly defined. Large analysis of occurrences, formal grammar specifications of functions New and Given, and implementations in the register of AGILE project were not our aim within the project frame.

This set of examples illustrates not only that the three Slavic languages exhibit high degree of word order freedom, allowing word order to reflect information structure, but also that word order is more free in Czech and Russian than in Bulgarian. Czech and Russian allow to preserve the grammatical structure and permute the constituents according the communicative purpose in a given context. The contexts corresponding to some of the permutations may of course be less likely to occur in the AGILE texts. Bulgarian, on the other hand, seems to be close to English.

In comparison to English, (68)(e) is interesting, because in this case Bulgarian allows a word order which in not possible in English, i.e. *Open with command* Open *a file.* So the generalization appears to be that the order of elements after the verb is free, or at least more free, in Bulgarian than in English. We work with the assumption that this order is determined by the communicative purpose, in the same way as in Czech and Russian. -> In Bulgarian contextually boundness or non-boundness of a particular clause element is signalled not only by its position among the other clause elements, but also by determination, which is manifested by (presence or absence of) a morphological marker carried by the word.

In general, the ordering at every level of grammatical structure can reflect information structure at least to some degree. Thus, what is being ordered are (i) clauses in a complex

sentence, (ii) elements of individual simplex clauses, i.e. prepositional phrases, nominal groups, adverbs, (iii) elements within each nominal group, etc. At each of these levels, also information structure is of relevance. We illustrate this with the following Czech examples:[15]

(69) **Clause ordering within a complex sentence with a hypotactic relation**

(a) Cz:
Když program neraguje, stiskněte Ctrl+Alt+Del.
When program-Nom not-reacts press Ctrl+Alt+Del
When the program does not respond, press Ctrl+Alt+Del.

(b) Cz:
Ctrl+Alt+Del stiskněte, když program nereaguje,
Ctrl+Alt+Del press when program-nom not-reacts
When the program does not respond, press Ctrl+Alt+Del.

(70) **Ordering of constituents at the clause level**

(a) Z menu Data vyberte Styl Multičáry.
From menu-Gen Data choose Style-Acc Multiline-Gen
From the Data menu choose Multiline Style.

(b) *Vyberte Styl Multičáry z menu Data.*
choose *Style-Acc Multiline* from menu-*Gen Data*
Choose *Multiline Style* from the *Data* menu.

(71) **Ordering within a nominal group**
Nejdříve určete počáteční bod oblouku a potom určete bod koncový.
First specify starting point-Acc arc-Gen and then specify bod-Acc ending
First specify the startings point of the arc and then the end point.

However, it is the ordering of elements at the clause level that is most strongly influenced by information structure, and that cannot be modelled adequately without taking information structure into account. It is also the level of ordering that is most widely manifested in the texts generated in Agile. Since the influence of information structure- at the clause level constitutes the most important and most frequent problem in the texts generated in Agile, it is this aspect of ordering that we concentrate on.

3.5.2 Text Generation Overview

The entire text generation process takes as an input a knowledge representation of the content to be conveyed by the text. In this section, we explain how our ordering algorithm fits into the overall architecture of text generation in AGILE. Within AGILE, we aim at so-called "end-to-end" generation: A user provides a content specification in the form of an A-box using an authoring interface, and obtains as output a text realizing that A-box, in a particular text style.

The A-box serves as input to the Text Structuring Module (TSM) which yields a set of formulas in a Sentence Planning Language (SPL) for the sentences to be generated to

[15] Obviously, the word order-alternatives presented in (69), (70) and (71) are generally not mutually substitutable in a given context.

convey the given content, in a style that is appropriate given the text style. To guide the text planning done by the TSM, we employ text structure elements that correspond to identifiable parts in an A-box configuration, and text templates that specify particular text styles.

The TSM is described in more detail in (Kruijff&Kruijff-Korbayová 1999). The essential ideas can be summarized as follows. The major component is formed by the *text planner* which is a systemic networks for text structuring. In the spirit of (Bateman&Teich 1994), we have constructed a region that defines an additional level of linguistic resources for the level of *genre*. The region enables the composition of text structures in a way that is very similar to the way a lexico-grammar builds up grammatical structures. In fact, by using KPML to implement the means for text structuring, a smooth interaction between global level text generation (strategic generation) and lexico-grammatical expression (tactical generation) is facilitated.[16].

The main organizing principle employed in the region is that text templates and text structure elements are essentially orthogonal ideas. Therefore it consists of two parts. One part deals with interpreting the A-box in terms of text structure elements. By traversing the network that the systems of this part make up, we obtain a text structure for the A-box conforming to the way the A-box structures the content. The other part of the region imposes constraints on the realisation of the text structure elements that are being introduced by traversing the other part of the region. Naturally it will depend on our choice of a particular text template (style) which constraints will be imposed. These choices are made through interaction between the user and the system.

After a text structure is built by traversing the systemic networks, we divide the A-box into smaller pieces of content that can be associated with the text structure elements from which the text structure is composed. In the setting of KPML this can easily be done using so-called ID-inquiries.

Finally, the so-called *sentence planner* translates the text plan in to a set of SPLs, specifying how these individuated pieces of content should be realized, in conjunction with the realization constraints imposed by text templates. The SPLs then serve as input to language-specific tactical generators that actually generate these sentences.

It is in this setting, in which individual sentences are specified and generated that ultimately make up a text, that the issue of generating contextually appropriate sentences arises. For only in that way a smooth flow of the text can be assured. To this end, the sentence planner introduces indication of information structure into the SPLs it produces.

Information structure is indicated in terms of marking individual semantic units in the SPLs as either *contextually bound* or *contextually non-bound* (see LSPEC2 for a detailed discussion of the terminology which is adopted from Sgall et al. 1986). Naturally, the question may be raised, what items can be considered as eligible for being labelled either as contextually bound, or contextually nonbound. In FGD (Sgall et al. 1986), any *semanteme*

---

[16] In this way we try to overcome the notorious problem known as the *generation gap* in which a text planning module lacks control over the fine-grained distinctions that are available in the grammar. In our case, both text planning and sentence planning are integrated into one and the same system, organized in a stratificational manner.

(or autosemantic word) can be either in contextually bound or nonbound --function words normally cannot. Relating that to the AGILE context, we can conceive of every instantiation of a *content* concept (i.e. as opposed to a *configurational* concept like PROCEDURE) as constituting a semantic unit, and being marked as either contextually bound or non-bound.

The sentence planner decides on the contextual boundness of a semantic unit as follows. We build a *discourse model* in parallel to building the text plan, whereby the discourse model keeps track of what concept identifiers (which can double as *discourse referents*) have been used. Subsequently, the sentence planner uses this discourse model to decide whether or not a particular concept should be realized as contextually bound: If the identifier of the concept was already introduced in the preceding context, then the concept will be realized as contextually bound. The TSM thus models the idea of contextual boundness as re-use of identical content.

3.5.3 A Word Ordering Algorithm Using Information Structure

The important conclusion drawn from the comparisons of the degree of word order freedom in Czech, Russian, Bulgarian and English is that in developing a uniform mechanism for handling word order in the AGILE multilingual generation system, we need to allow for a sufficient degree of flexibility such that each grammar can set whatever grammatical constraints are needed in the individual language, while at the same time, leaving grammatically unconstrained ordering to be decided by other appropriate strategies.

The approach to word ordering developed in Agile employs the following principles:

- The grammar imposes structural constraints, such as the ordering of connectives, placement of sentential clitics and possibly verb-second position.
- The text planner can determine the element to become the Theme in order to reflect a particular textual organization. If no particular element is chosen as the Theme, the grammar chooses the least communicatively dynamic element as the Theme. The Theme is then placed at the beginning of the clause, although not necessarily at the very first position, as this might be occupied, e.g., by a connective. The placement of the Theme is resolved by the grammar.
- The text and sentence planner determine information structure, in particular, contextual boundness of each item. Contextual boundness is then used to constrain word order at the clause level as follows:
    - The contextually bound (CB) elements (if there are any) precede the contextually non-bound (NB) elements in the prototypical cases.[17]

[17] While CB items precede NB items at the level of a clause in the prototypical case, the order can be reversed in non-prototypical cases. Such reversed order when NB items precede CB items at the sentene level is called *subjective ordering* (Mathesius 1939, Firbas 1992). For example, the question (Cz) *Co zmizelo?* (What disappeared?) can be answered either by a sentence with the prototypical order: (Cz) *Zmizelo okno*, or by a sentence with subjective ordering (Cz) *Okno zmizelo*. Not only word order, but also intonation play an important role in such cases. Subjective ordering is possible, but rather scarce in written texts in a formal register, like the texts generated in Agile. Therefore we leave subjective ordering aside.

- The mutual ordering of multiple CB items in a clause corresponds to communicative dynamism, and the mutual ordering of multiple NB items in a clause follows systemic ordering (SO) --with the exceptions required by grammatically constrained ordering as described below. The default for communicative dynamism is systemic ordering.
- The main verb of a clause is ordered at the boundary between the CB elements and the NB elements, unless the grammar specifies otherwise (e.g. verb-second position in Czech or Russian).

To show our approach in a nutshell, we present an abstract word ordering algorithm in Figure 55. Using "<" for linear precedence, the ordering it produces (for Czech) can be schematized as follows:

Theme < Clitics < Rest-CB < Verb < Rest-NB

```
Given:
    a set GC of ordering constraints imposed by the grammar
    a list L1 of constituents that are to be ordered,
    a list D giving ordering of CB constituents (default is SO)

Create two lists LC and LN of default orderings:

    Create empty lists LC and LN        % LC for CB items, LN for NB
items
    Repeat for each element E in L1
            if E is CB,
                then add E into LC,
                else add E into LN.
    Order all elements in LC according to D   % default for D is SO
    Order all elements in LN according to SO
    if the Verb is yet unordered then
        Order the Verb at the beginning of LN

Order the elements of L1
    if GC is not empty then
         use the contraints in GC, and
         if the contraints in GC are insufficient,
            apply first the default orders in LC and then those in LN
```

Figure 55: Flexible word ordering algorithm

According to this algorithm, the ordering constraints posed by the grammar have the highest priority. Note that this includes the ordering of the textually determined Theme. Then, any elements which are not ordered by the grammar, are subject to the ordering according to information structure, i.e. systemic ordering in combination with the CB/NB distinction. The NB elements are ordered by SO. The ordering of the CB elements can be (i) specified on the basis of the context, (ii) restricted by the grammatical structure, (iii) follow SO. The verb is placed between the last CB and the first NB element, unless it is itself the Theme.

The ordering algorithm as such is the same for all the three languages under consideration. What may differ is first of all the systemic ordering. Further differences between the languages are encountered in the constraints on which elements can be ordered rather freely and which elements are subject to ordering requirements posed by the syntactic structure.

3.5.3.1 Implementation Details

This section describes the technical details of the implementation and thus serves as a part of the documentation accompanying the word ordering algorithm. The new ordering algorithm[18] works with the **:contextual-boundness** statements in SPLs, which can have the value **yes** or **no**. It is defined in the file **cb-orderings.lisp** as a method specialization for the particular language used, e.g., **:CZECH-990015**.

The main new feature as far as implementation is concerned is that actually we need to do two kinds of ordering: the default (systemic) ordering is used for the list of contextually bound (LCB in the algorithm) as well as for the list of contextually nonbound elements (LNB), but separately for each of the lists. The algorithm also needs to recognize the main verb is, so there a global variable which is set as follows

```
(defvar *verb-functions* '(lexverb process))
```

In addition, the following changes are required in the grammar resources:

Systemic ordering needs to be defined as one of the default ordering-constraints, for example, the initial version of the SO in Czigel-FINP is:

```
(agent timelocative spacelocative means goal)
```

The entire list of default orders thus becomes as follows:

```
(define-default-orders
 :language :CZECH-990015
 :orders '((agent timelocative spacelocative means goal)[19]
;;the rest are the Nominal group, etc. orderings of the old constraints
   (REPRESENTATIVE RSELECTOR PART PSELECTOR QUANTIFIER QSELECTOR
ORDINATOR OSELECTOR SUPERLATIVE SSELECTOR TYPIC TSELECTOR DEICTIC POST-
DEICTIC NUMERATIVE SIZE STATUS AGE COLOUR PERIOD PROVENANCE
LOCATIONCLASSIFIER USE PRE-EVENT MATERIAL CLASSIFIER4 CLASSIFIER3
CLASSIFIER2 CLASSIFIER1 THING COMP-STANDARD-QUALIFIER QUALIFIER
MEDIUMQUALIFIERMARKER AGENTQUALIFIERMARKER PHENOMENON PORTION
ATTRIBUENDQUALIFIER ACCOMPANIMENT LOCATIVEQUAL BENEFICIARYQUALIFIER
TEMPORAL EVENT)
   (DEICTIC COMPARATOR QUALITY STANDARDINDICATOR STANDARD)
   (PRE-CARDINAL MILLION THOUSAND HUNDRED SUPRATEN SUBTEN)
   (PRE-CARDINAL TEMPERER APEX)
   (PRE-CARDINAL THOUSANDDIGIT HUNDREDDIGIT TENDIGIT UNITSDIGIT DIGIT)))
```

- (see file **ordering-constraints.gram**)

The revised KPML ordering algorithm thus enables us to handle the combination of ordering constraints imposed by (i) the grammar and (ii) information structure. The structural ordering constraints originating from the grammar are taken into account first, and information structure is used to compute an ordering of any elements not ordered by the grammar constraints.

---

[18] The new ordering algorithm has been implemented by John Bateman.

[19] Note that the algorithm leaves anything that is not in the SO unplaced, so unless the grammar does it, there will be many possible orderings for sentences with other constituents than those mention in the SO list (`agent timelocative spacelocative means goal`).

In this way, the ordering algorithm is general, and the grammar of each language can determine to what extent the word order in that language is "fixed" (i.e., structurally constrained) or "free" (i.e., constrained by information structure). In the next two section we describe the application of the ordering algorithm in the Czech and Russian grammars.

3.5.3.2 Application of the algorithm in the Czech Grammar

We show below several input SPLs which reflect contextual boundness,[20] and the corresponding generated output in Czech using the current version of Czigel-FINP. The grammatical constraints remain the same in all the examples: they determine the ordering of the Theme at the front and of the reflexive particle (clitic) after the Theme. Bracketing in the English glosses of the examples indicates semantic units. The glosses also contain subscripts indicating the functions of the individual, i.e. Goal, Means, and an indication of contextual boundness (CB) and non-boundness (NB).

(72) (opens) (refl) (command-ins *Open*)$^{Means\text{-}NB}$ (file)$^{Goal\text{-}NB}$:

(a) Input SPL:

```
(S1 / directed-action :lex otevr3i2t
  :speechact assertion :prefer-mention-agent-q withhold
  :ACTOR (hearer / Person )
  :ACTEE (file / object :lex soubor )
  :Instrumental (command / object :lex pr3i2kaz
    :class-ascription (label / object :name GUI-OPEN)))
```

(b) Generated output:

Cz: *Otevírá se příkazem* Open *soubor*.

In (72)(a), no element is specified as CB, so the generated ordering in (72)(b) is based only on the structural constraints imposed by the grammar, and the SO defaults. The grammar chooses the verb as the default Theme.

(73) (file)$^{Goal\text{-}CB}$ (refl) (opens) (command-ins *Open*)$^{Means\text{-}NB}$:

(a) Input SPL:

```
(S1 / directed-action :lex otevr3i2t
  :speechact assertion :prefer-mention-agent-q withhold
  :ACTOR (hearer / Person )
  :ACTEE (file / object :lex soubor :contextual-boundness yes)
  :Instrumental (command / object :lex pr3i2kaz
    :class-ascription (label / object :name GUI-OPEN)))
```

(b) Generated output:

Cz: *Soubor se otevírá se příkazem* Open.

In (73), 'soubor' (fie) is specified as CB, and it is the only CB element. No Theme has been specified. Thus, 'soubor' is chosen as the default Theme and ordered at front, and the remaining elements (except the clitic) are ordered by SO.

(74) (command-ins *Open*)$^{Means\text{-}CB}$ (refl) (opens) (file)$^{Goal\text{-}NB}$:

(a) Input SPL:

```
(S1 / directed-action :lex otevr3i2t
```

[20] When contextual boundness equals the default, i.e. contextualy non-bound, it is not explicitly mentioned in the SPL.

```
:speechact assertion :prefer-mention-agent-q withhold
:ACTOR (hearer / Person )
:ACTEE (file / object :lex soubor )
:Instrumental (command / object :lex pr3i2kaz
  :contextual-boundness yes
  :class-ascription (label / object :name GUI-OPEN)))
```

(b) Generated output:
Cz: *Příkazem* `Open` *se otevírá soubor.*

In (74), 'příkaz' (command) is specified as CB, and it is the only CB element. No Theme has been specified. Thus, 'příkaz' is chosen as the default Theme and ordered at front, and the remaining elements (except the clitic) are ordered by SO.

(75) (command-ins *Open*)$^{Means\text{-}CB}$ (refl) (file)$^{Goal\text{-}CB}$ (opens):

(a) Input SPL:

```
(S1 / directed-action :lex otevr3i2t
  :speechact assertion :prefer-mention-agent-q withhold
  :ACTOR (hearer / Person )
  :ACTEE (file / object :lex soubor :contextual-boundness yes)
  :Instrumental (command / object :lex pr3i2kaz
    :contextual-boundness yes
    :class-ascription (label / object :name GUI-OPEN)))
```

(b) Generated output:
Cz: *Příkazem* `Open` *se soubor otevírá.*

In (75), both 'soubor' (file) and 'příkaz' (command) are specified as CB. No Theme has been specified. Thus, 'příkaz' is chosen as the default Theme, because Means has a lower CD than Goal.

(76) (file)$^{Goal\text{-}CB}$ (refl) (command-ins *Open*)$^{Means\text{-}CB}$ (opens):

(a) Input SPL:

```
(S1 / directed-action :lex otevr3i2t
  :speechact assertion :prefer-mention-agent-q withhold
  :ACTOR (hearer / Person )
  :ACTEE (file / object :lex soubor :contextual-boundness yes)
  :Instrumental (command / object :lex pr3i2kaz
    :contextual-boundness yes
    :class-ascription (label / object :name GUI-OPEN)
    :theme file))
```

(b) Generated output:
Cz: *Soubor se příkazem* `Open` *otevírá.*

In (76), again both 'soubor' (file) and 'příkaz'(command) are specified as CB, but this time 'soubor' has been specified as the Theme, and therefore it is ordered at front.

Our current approach as implemented so far in the Czigel-FINP grammar has the following shortcoming. It handles ordering after the determination of syntactic structure and after the completion of syntactic constituents at a given level, i.e. the clause. Therefore, word order choice is currently the only grammatical reflection of information structure in the Czech sentences we generate.

In an improved version of the Czigel-FinP grammar, it would be desirable to take information structure into account also when making particular choices of syntactic structures, for example, when deciding whether to use passive or active voice, or whether to

use a subordinate clause or a nominalization. However challenging this issue is, pursuing it in-depth is outside the scope of the Agile project.

3.5.3.3 Application of the algorithm in the Russian Grammar

The algorithm has been also applied with the Russian grammar.

3.5.4 Structural Constraints on Word Order in the grammars

As mentioned above, information structure is only one source of ordering constraints. Another important source of ordering constraints is the grammar as such. Even in languages with a high degree of word order freedom, like Czech or Russian, information structure can be seen as providing a default ordering which is used unless other restrictions apply. While in Czech and Russian, there are relatively few structural (grammatical) restrictions, Bulgarian is more similar to English with respect to word order, in that its word order is more fixed than in other Slavonic languages.

In this section, we discuss the structural constraints on word order the Agile languages. In Section 3.5.4.1, we discuss a particular issue of structurally constrained ordering in the Czech grammar, namely the placement of clitics.

Section 3.5.4.2 addresses the implementation of word ordering in the Bulgarian grammar. Since the flexible word ordering algorithm proposed in Agile (LSPEC2 and IMPL2) and described in Section 3.5.3 has not been applied in the Bulgarian grammar, the Bulgarian grammar adopts essentially the handling of word order present in the Nigel grammar for English.

3.5.4.1 Clitic Placement in Czech

The instructional texts generated in AGILE contain constructions which bring about the problem of **clitic placement**. In Czech, examples of the use of clitics are sentences containing the medio passive and lexically reflexive verbs, conditional mood or weak forms of pronouns, as illustrated below.

(77) **Medio-passive**
Cz:*Vybere se* Color. **Vybere* Color *se*. ** *Se vybere* Color.
Selects refl Color
One selects the Color command.

(78) **Reflexive verb(s)**

Cz: Objeví se okno. *Objeví okno se. *Se objeví okno.
Appears refl box
A box appears.

(79) **Conditional mood**

Cz: Program by skončil. *Program skončil by. *By program skončil.
Program would end
A/The program would end.

(80) **Weak pronouns**

(a) Cz: Označte čáru. Smažte ji stisknutím `Del`.
Select line Delete it pressing -Ins `Del`.
Select a line. Delete it by pressing the `Del` key.

(b) Cz: Stisknutím `Del` ji smažte.

(c) Cz: *Ji stisknutím `Del` smažte.

(d) Cz: Stisknutím `Del` smažte ji.[21]

The correct placement of clitics in Czech is a difficult issue in general. In the Intermediate Prototype, we have only addressed the placement of reflexive particles (used in the reflexive passive form or in the so-called "reflexive tantum" verb forms), and we only addressed the cases when a single clitic occurred in a sentence.

The following example illustrates the placement of multiple clitics within a sentence.

(81) **Multiple clitics**

(a) Cz: *Program by se ji pokusil smazat.*
Program would refl it tried delete

(b) Cz: *Program by se pokusil ji smazat.*
Program would refl tried it delete
The program would try to delete it.

In general, multiple clitics form a **clitic cluster** within a sentence in Czech. This means, all sentential clitics are grouped together. The placement of the clitic cluster is subject to the same ordering restrictions as the placement of a single clitic. In example (81), we demostrate all the possible correct placements of the clitics in the given sentence.

The mutual ordering of clitics within the clitic cluster is as follows:

- Cz: *aux-be-form ^ reflexive-clitic ^ dative-pronominal-clitic(s) ^ accusative-pronominal-clitic*

[21] The sentence *Stisknutím `Del` smažte ji* is grammatical. However, the pronoun is not a weak form, at least not under neutral sentence intonation pattern with the intonation center at the end. Such sentence would receive a contrastive interpretation, where the referent of the pronoun is contrasted with some other element in the context, an interpretation which is not supported by the context provided.

If a sentence contains a finite verb modified by an infinitive verb form, as in example (81), then there are two possibilities for the placement of those clitics which originate from the subordinated verb:

- The clitics belonging to the infinitival modifier can stay and be placed within the "clause-domain" of the infinitive verb form.
- The clitics belonging to the infinitival modifier can "climb" to the "main clause-domain" and be placed together with other clitics (if any) belonging to the finite verb form There are two restrictions on this climbing in Czech:
  - Two identical clitic forms are prohibited within one clitic cluster.
  - Two reflexive clitics are prohibited within one clitic cluster.

The example in (82) illustrates the possibilities of clitic placement in case of a verb plus an infinitival modifier where both verbs are accompanied by a clitic in Czech. In (82)(a), the reflexive particle *se* stays with the infinitive it belongs to, in (82)(b) it is raised to become part of the clitic cluster at the main clause level, as a result of which there are two identical clitic forms next to one another, which situation is prohibited in Czech, and therefore, the two occurrences of *se* are "reduced" (collapsed) to just one as shown in (82)(c).

(82) **Two occurrences of the same clitic**

(a) Cz: *Program by se pokusil se restartovat.*
Program would refl tried refl restart

(b) Cz: **Program by se se pokusil restartovat.*
Program would refl refl tried restart.

(c) Cz: *Program by se pokusil restartovat.*
Program would refl tried restart.
The program would try to restart itself.

In the following sections, we present the solutions of clitic placement adopted in the Final Prototype.

In Czech, clitics have to be placed in the so-called **Wackernagel position**, characterized roughly as the position between the first and the second element in a clause.[22] If there are multiple clitics in a single clause, then they have to be grouped in a cluster. Their mutual ordering depends on their functions and is constrained by the grammar.

Various Czech grammarians have discussed the clitic placement problem, and proposed some solutions (cf. Bauer&Grepl 1972, Daneš et al. 1987, Kopečný 1958, Trávníček 1949, Šmilauer 1966 cited in Oliva&Avgustinova 1995). We acknowledge that this issue is a very complex one. The target texts for the Final Prototype do not contain the full range of difficult cases involving many different kinds of clitics within a single sentence and complicated combinations of multiple spatial or temporal expressions, such as discussed in

[22] Naturally, this leaves to be defined what ``element" means. It is easy to show that ``first element" does not equate to ``first constituent" in any simple sense, since the element can be of arbitrary complexity. It is also possible to find examples where multiple elements precede the clitic cluster; however, these cases seem to be restricted to cases where all those pre-clitic elements are strongly backgrounded (Oliva&Avgustinova 1995).

(Oliva&Avgustinova 1995). Therefore, we can restrict the treatment to the cases involving combinations of *by* (conditional) and *se* (reflexive).

Our approach to handling the clitic placement problem in the Final Prototype is formulated in general terms and takes the findings of the above cited works into account. However, we have only tested it insofar the texts generated in the Final Prototype are concerned, and not against the full range of examples presented in the works cited above. Our approach can be characterized as follows:

- Following (Halliday 1985), we assume that each sentence has a **Theme**. The Theme can consist of a **Textual, Interpersonal and Experiential** element. The Theme contains exactly one experiential element, the so-called **Topical Theme**. The ordering within the Theme is *Textual^Interpersonal^Experiential*.
- The Topical Theme can be determined by the text planner. If no particular element is chosen as the Theme, the grammar chooses the least communicatively dynamic experiential element as the Theme.
- The grammar constrains the Theme to be ordered at the beginning of the clause.
- The placement of the clitic cluster within a clause is decided in the grammar: the clitic cluster is placed after the first thematic element. There are thus the following possibilities of ordering of the Clitic Cluster with respect to the thematic elements, depending on what thematic elements occur:
    - *Topical-Theme^Clitic-Cluster* (ex. …)
    - *Interpersonal-Theme^Clitic-Cluster* (ex. …) *or*
      Interpersonal-Theme^Clitic-Cluster ^Topical-Theme (ex. …)
    - *Textual-Theme^Clitic-Cluster* (ex. …) or
      Textual-Theme^Clitic-Cluster^Interpersonal (ex. …) *or*
      Textual-Theme^Clitic-Cluster^Interpersonal^Topical-Theme (ex. …)
- The mutual ordering of clitics within the clitic cluster is as mentioned earlier:
    - Cz: *aux-be-form ^ reflexive-clitic ^ dative-pronominal-clitic(s) ^ accusative-pronominal-clitic* (ex. …)

3.5.4.2 Implementation of Word Ordering in the Bulgarian Grammar

3.5.4.2.1 Reflection of Contextual Boundness in the Bulgarian Grammar

In Bulgarian sentences both contextually bound and contextually non-bound elements can appear after the verb (see (68)(a)(68)(c)(68)(e)). So, we do not apply the algorithm described above to Bulgarian, because it makes the verb a kind of border between CB and NB elements. Communicative relationships among the clause elements in Bulgarian coherent text are expressed by combining two means: determination and word order. Concerning word order our approach to get the clause varieties generated by Final Prototype was to adapt further the Theme region inherited from Nigel grammar. Concerning determination, instead of contextual-boundness marker provided by the presented Word Ordering Algorithm we give in our SPLs direct answer to the inquiry IDENTIFIABILITY-Q. In fact, Text Structuring Module produces in these cases SPLs to generate Bulgarian examples, which are different from Czech and Russian ones generating the corresponding examples.

Word order constraints are given by the grammar (default orders or logic of default order changes): ordering of the Theme at the front position, clitic ordering, Subject-Finite or Finite –Subject order. The default order definitions of Bulgarian grammar are shown on the Figure 56 below.

```
(define-default-orders :language :BULGARIAN
      :orders '((SUBJECT NON-FINITIVE)
 (SUBJECT NEGATOR LEXVERB QUALITY EXISTENT ATTRIBUTE DIRECTCOMPLEMENT
       RANGE REPORT ADDRESSEE BENEFICIARY AGENTMARKER MEDIUMMARKER
       INTERNALMATTER SPACEEXTENT TIMEEXTENT SPACELOCATIVE
       TIMELOCATIVE CAUSE MEANS ACCOMPANIMENT MATTER ROLE ATTITUDE
       CONJUNCTIVE DEPENDENT)
 (REPRESENTATIVE RSELECTOR PART PSELECTOR QUANTIFIER QSELECTOR
            ORDINATOR OSELECTOR SUPERLATIVE SSELECTOR TYPIC
            TSELECTOR DEICTIC POST-DEICTIC NUMERATIVE SIZE STATUS
            AGE COLOUR PERIOD PROVENANCE LOCATIONCLASSIFIER
            MATERIAL USE CLASSIFIER4 CLASSIFIER3 CLASSIFIER2
            CLASSIFIER1 THING COMP-STANDARD-QUALIFIER QUALIFIER
            MEDIUMQUALIFIERMARKER AGENTQUALIFIERMARKER PHENOMENON
            PORTION ATTRIBUENDQUALIFIER ACCOMPANIMENT LOCATIVEQUAL
            BENEFICIARYQUALIFIER TEMPORAL EVENT)
 (DEICTIC COMPARATOR QUALITY STANDARDINDICATOR STANDARD)
 (PRE-CARDINAL MILLION THOUSAND HUNDRED SUPRATEN SUBTEN)
 (PRE-CARDINAL TEMPERER APEX)
 (PRE-CARDINAL THOUSANDDIGIT HUNDREDDIGIT TENDIGIT UNITSDIGIT DIGIT)
 (SUBJECT DA-CONSTRUCTOR TEMPO0 FINITE REALITY TEMPO1 TEMPO2 TEMPO3
       VOICE LEXVERB)
 (STRUCTURAL TEXTUAL INTERPERSONAL TOPICAL)
 (IDENTIFIED PROCESS IDENTIFIER)
 (AGENT FINITE)))
```

Figure 56: Default order definitions in Bulgarian grammar

We show bellow the SPLs reflecting the logic of contextual boundness of sentence elements from Bulgarian clauses on example (68) (see the given analysis).

(83) (open-imp) (file)Goal-CB (command Open)Means-NB

```
 (S / DM::OPEN
     :SPEECHACT IMPERATIVE
     :ACTOR (HEARER / USER :IDENTIFIABILITY-Q IDENTIFIABLE)
     :ACTEE (A / DM::FAIL
 :IDENTIFIABILITY-Q  IDENTIFIABLE )
     :INSTRUMENTAL (I / DM::KOMMAND
     :CLASS-ASCRIPTION (LABEL / OBJECT :NAME GUI-OPEN)))
```

This is the implementation of example (68)(a)(i). The Goal element “file” is determined and contextually bound. This is signaled by the line :IDENTIFIABILITY-Q IDENTIFIABLE in Actee description in the SPL. Means element “command” is non-bound.

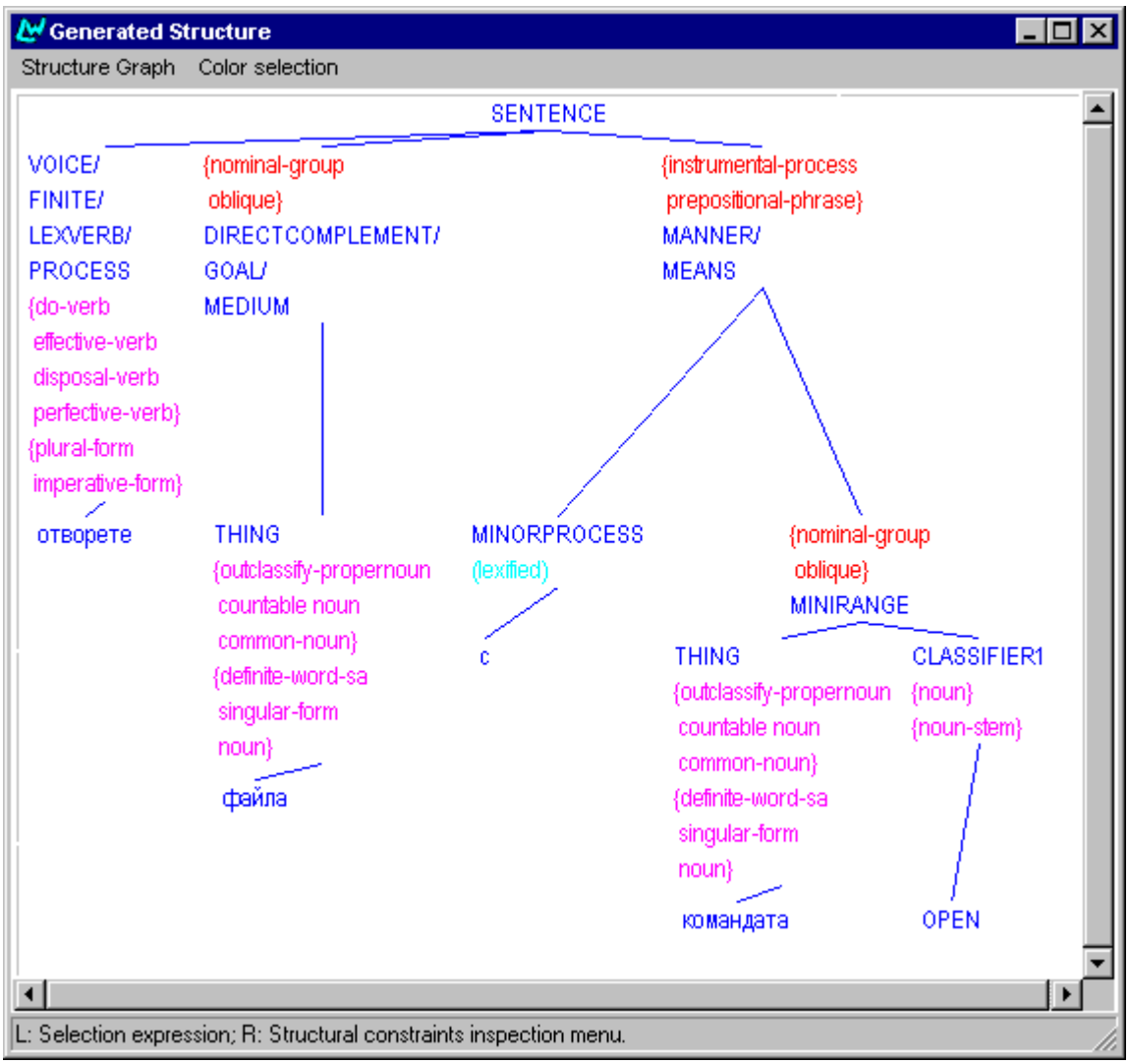


Figure 57: Generated output for (83)

(84) (open-imp) (file)Goal-NB (command Open)Means-NB

```
(S / DM::OPEN
     :SPEECHACT IMPERATIVE
     :ACTOR (HEARER / USER :IDENTIFIABILITY-Q IDENTIFIABLE)
     :ACTEE (A / DM::FAIL
:IDENTIFIABILITY-Q  NOTIDENTIFIABLE)
     :INSTRUMENTAL (I / DM::KOMMAND
     :CLASS-ASCRIPTION (LABEL / OBJECT :NAME GUI-OPEN)))
```

The target clause is shown on example (11a(ii)). The Goal element “file” is not determined and is contextually non-bound. This is explicitly given by the line `:IDENTIFIABILITY-Q NOTIDENTIFIABLE` in Actee description in the following SPL. Means element “command” is contextually non-bound.

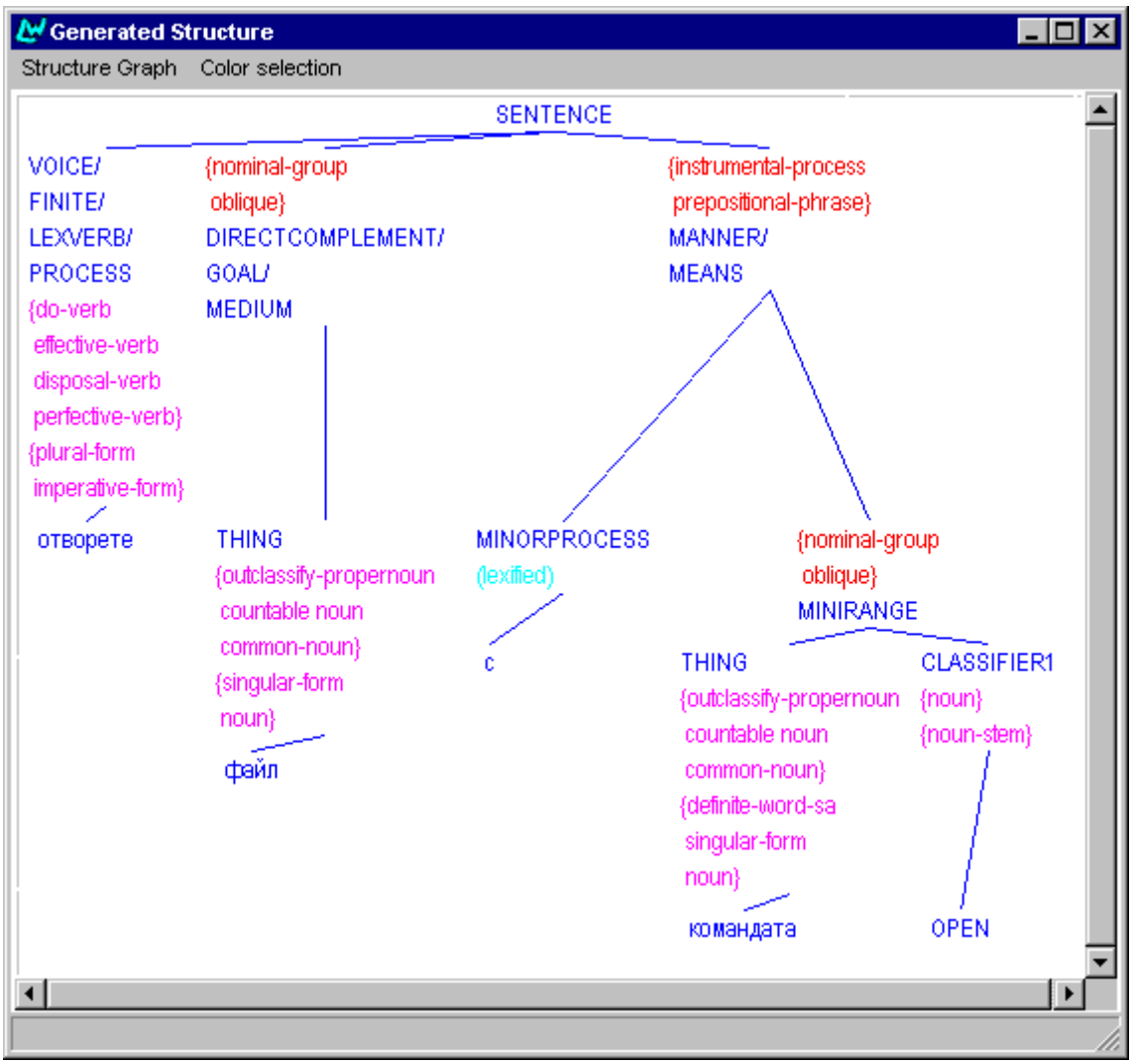


Figure 58: Generated output for (84)

(85) (file)[Goal-CB] (open-refl) (command Open)[Means-NB]

```
(S / DM::OPEN
     :THEME A
     :PREFER-MENTION-AGENT-Q WITHHOLD
     :ACTOR (HEARER / USER :IDENTIFIABILITY-Q IDENTIFIABLE)
     :ACTEE ( A / DM::FAIL
:IDENTIFIABILITY-Q  IDENTIFIABLE)
     :INSTRUMENTAL (T / DM::KOMMAND
     :CLASS-ASCRIPTION (LABEL / OBJECT :NAME GUI-OPEN))
)
```

Preferred surface realization of the semantics given by the (68)(b) example requires in written Bulgarian medio-passive verb form. Target clause is shown on example (68)(f). Medio-passive construction of the clause is reached by the line `:PREFER-MENTION-AGENT-Q WITHHOLD` in the SPL. The Goal element "file" is determined, contextually bound and takes the roles of Theme (first position) and Subject. Determination is signaled by the line `:IDENTIFIABILITY-Q IDENTIFIABLE` in Actee description. THEME slot points to the value of ACTEE slot (Goal element). Means element (INSTRUMENTAL slot) "command" is contextually non-bound.

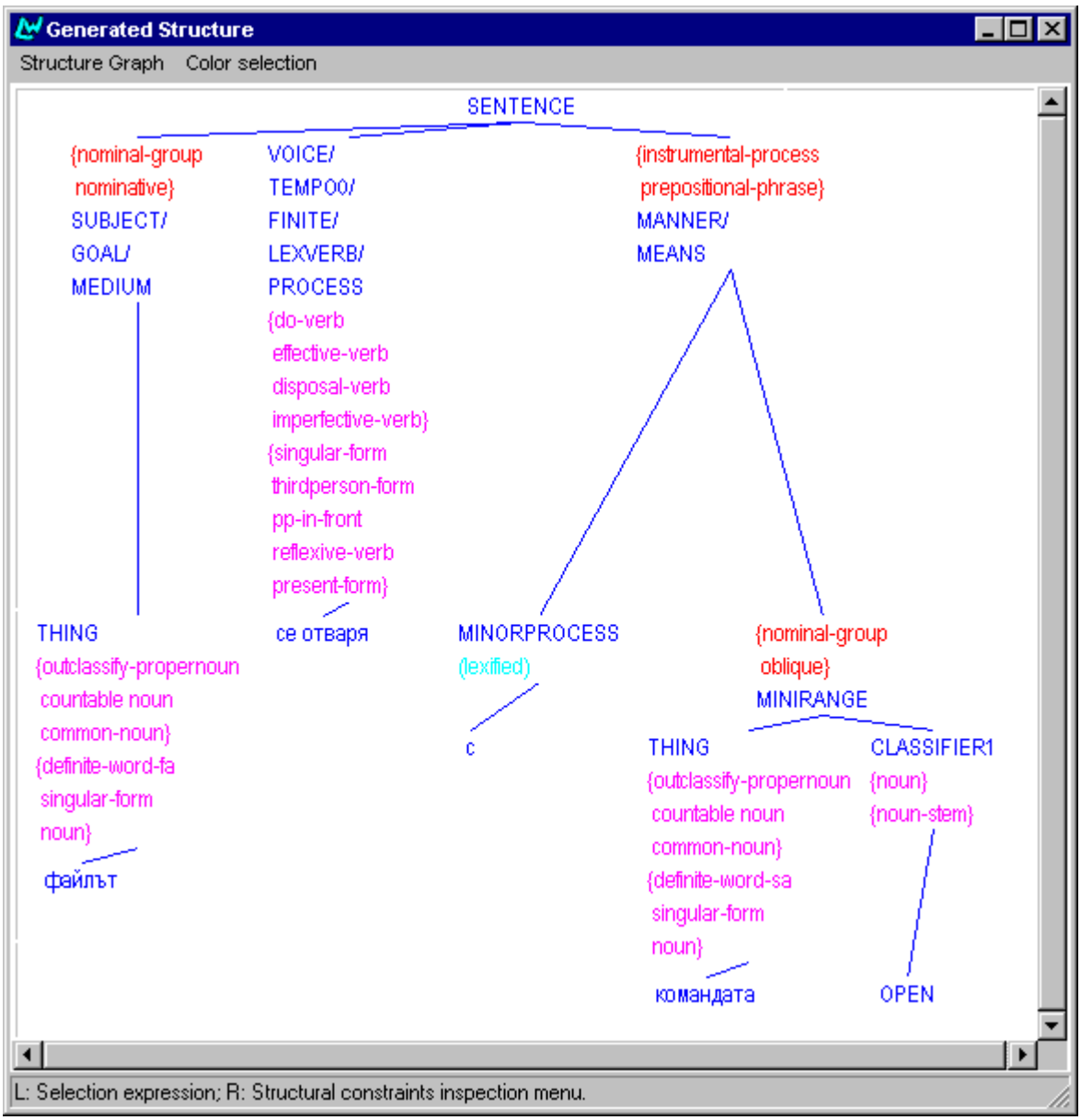


Figure 59: Generated output for (85)

(86) (command Open)$^{\text{Means-CB}}$ (open-imp) (file)$^{\text{Goal-CB}}$

```
(S / DM::OPEN
     :THEME T
     :SPEECHACT IMPERATIVE
     :ACTOR (HEARER / USER :IDENTIFIABILITY-Q IDENTIFIABLE)
     :ACTEE ( A / DM::FAIL
:IDENTIFIABILITY-Q  IDENTIFIABLE)
     :INSTRUMENTAL (T / DM::KOMMAND
    :CLASS-ASCRIPTION (LABEL / OBJECT :NAME GUI-OPEN)))
```

This is the implementation of the example (68)(c)(i). Means element “command” is contextually bound and placed in first position in the clause. This is directed by the THEME slot, which points the Means element described in INSTRUMENTAL slot of the SPL. Goal element is contextually bound, which is determined by the line `:IDENTIFIABILITY-Q IDENTIFIABLE` in its description.

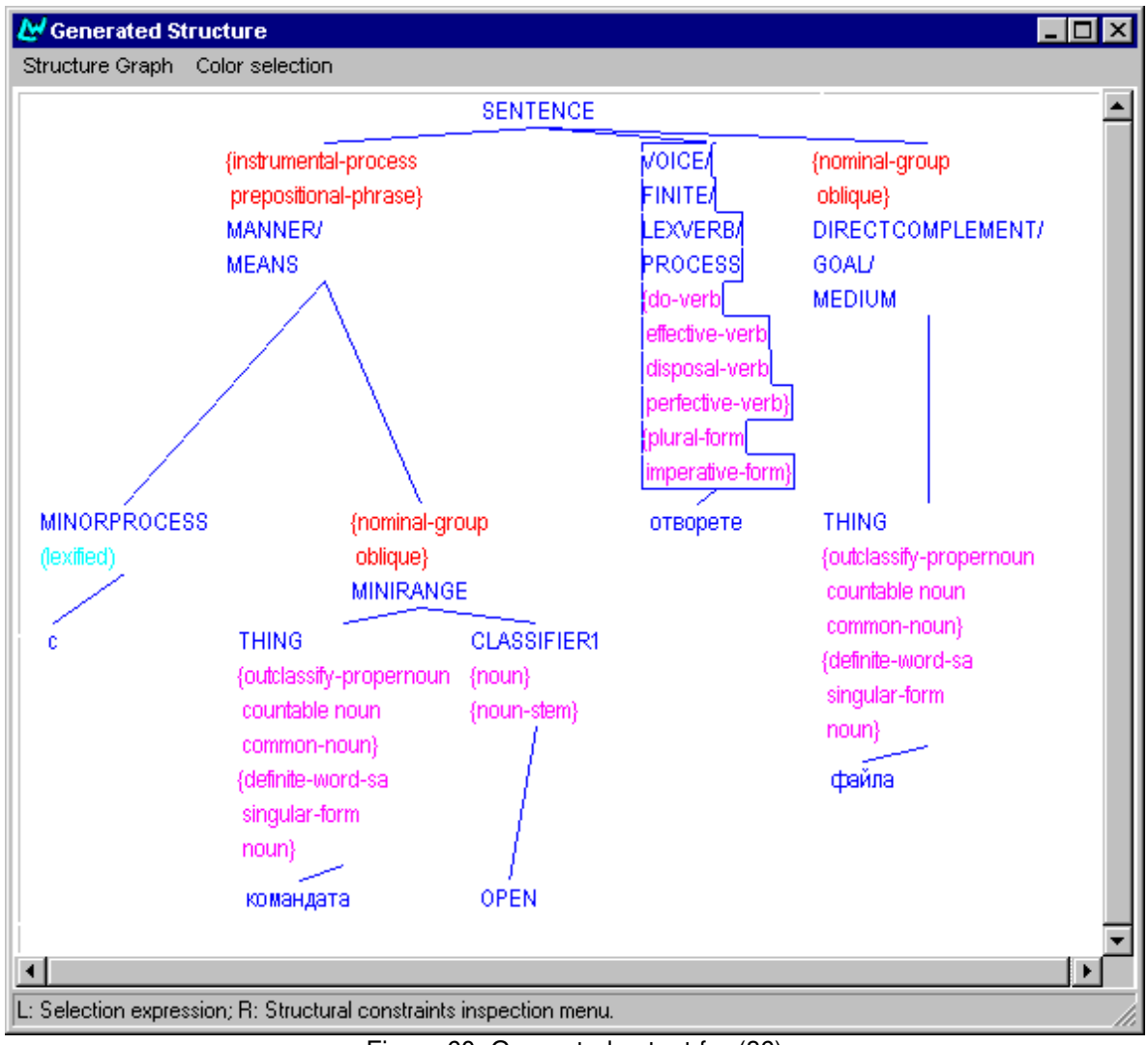


Figure 60: Generated output for (86)

(87) (command Open)$^{\text{Means-CB}}$ (open-imp) (file)$^{\text{Goal-NB}}$

```
(S / DM::OPEN
     :THEME T
     :SPEECHACT IMPERATIVE
     :ACTOR (HEARER / USER :IDENTIFIABILITY-Q IDENTIFIABLE)
     :ACTEE ( A / DM::FAIL
:IDENTIFIABILITY-Q  NOTIDENTIFIABLE)
     :INSTRUMENTAL (T / DM::KOMMAND
     :CLASS-ASCRIPTION (LABEL / OBJECT :NAME GUI-OPEN)))
```

This is the implementation of the example (68)(c)(ii). The only difference between this example and previous one is contextual non-boundness of the Goal element given explicitly by the line :IDENTIFIABILITY-Q IDENTIFIABLE Means element “command” is contextually bound and placed in first position in the clause. This is directed by the THEME slot, which points the Means element fully described in INSTRUMENTAL slot of the following SPL.

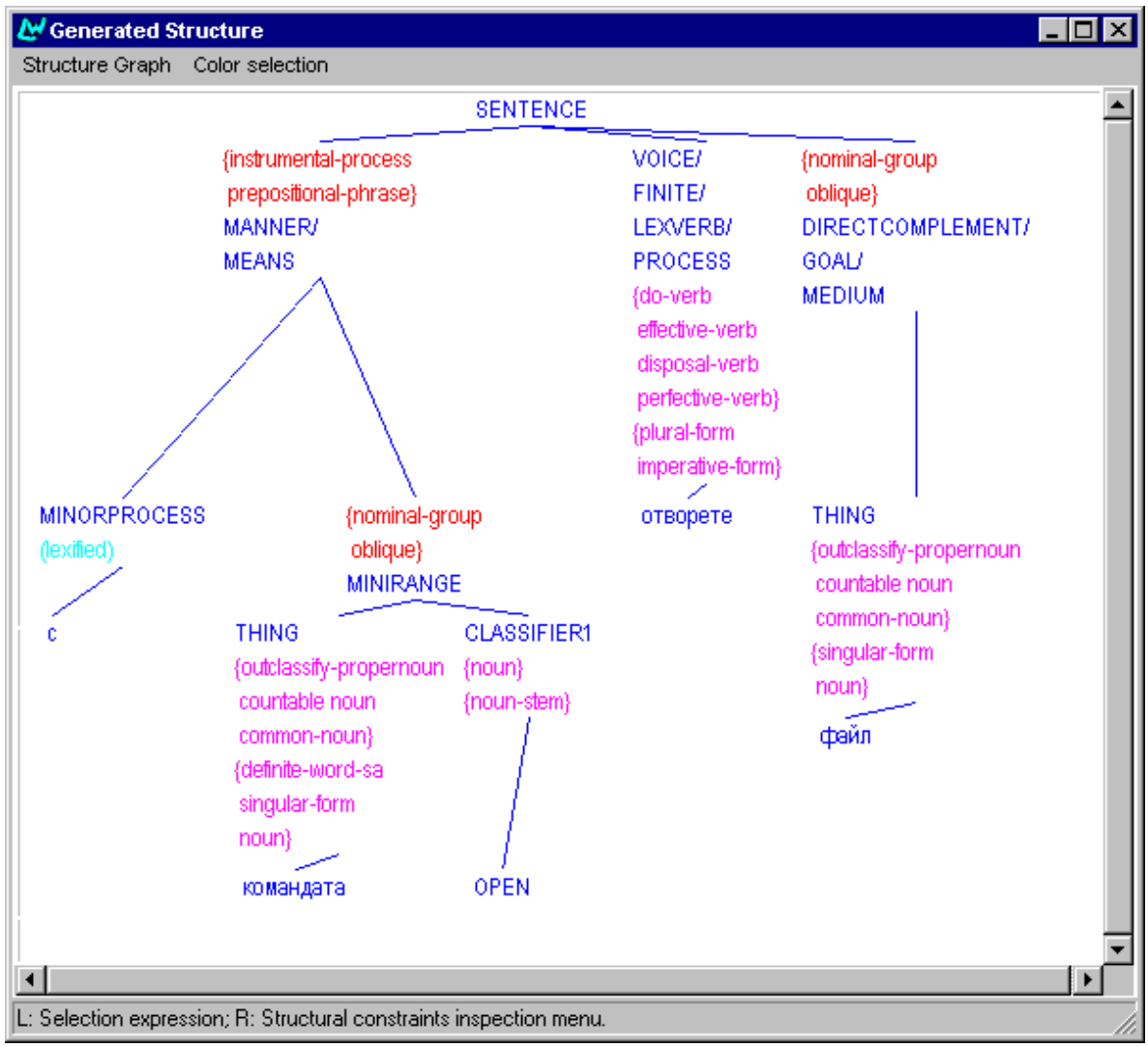


Figure 61: Generated output for (87)

3.5.4.2.2 Clitic Placement in Bulgarian

In Bulgarian, clitics are used in sentences containing medio-passive verbs, reflexive verbs or weak forms of pronouns, as illustrated below.

(88) **Medio-passive**

*Избира се Color. *Избира Color се. *Се избира Color.*
Selects refl Color
One selects the Color command.

(89) **Reflexive verb(s)**

(b) *Появява се диалогов прозорец.*

(c) **Появявя диалогов прозорец се. *Се появява диалогов прозорец.*
Appears refl dialog box
A dialog box appears.

(90) **Weak pronouns**

(a) *Маркирайте линията. Изтрийте я с натискане на Del.*
Select-impPl2 line Delete it by press-nom of Del.
Select a line. Delete it by pressing the Del key.

(b) *С натискане на Del я изтрийте.*

(c) **Я с натискане на Del изтрийте.*

(d) **С натискане на Del изтрийте я.*

When multiple clitics are used they form a "cluster" in Bulgarian sentences. The following example illustrates the forming of clitic clusters and the use of multiple clitics in a sentence.

(91) **Forming clitic clusters containing accusative pronominal clitic**

(d) *Вие сте въвели име на файла.*
you be-aux.Pl2 fill-PastParticiplePl2 name of file-det.
You have filled the file name.

(e) *Вие сте му въвели име.*
you be-aux.Pl2 to it/it-DatMasc fill-PastParticiplePl2 name
You have filled its name.

(f) *Вие сте му го въвели.*
you be-aux.Pl2 to it/it-DatMasc it-AccMasc fill-PastParticiplePl2
You have filled it.

(92) **Forming clitic clusters containing reflexive clitic**

(g) *Вие се представяте.*
you yourself-refl present-Pl2
You are presenting yourself.

(a) *Вие сте се представили.*
you be-aux.Pl2 yourself-refl present-PastParticiple-Pl2
You have presented yourself.

(b) *Вие сте му се представили.*
you be-aux.Pl2 to him/he-Dat yourself-refl present-PastParticiple-Pl2
You have presented yourself to him.

The elements inside Clitic cluster have to be adjacent in the given order:

aux-be-form ^ dative-pronominal-clitic(s) ^ accusative-pronominal-clitic

or

aux-be-form ^ dative-pronominal-clitic(s) ^ reflexive-clitic.

It is interesting, that in third person singular the auxiliary form (*aux-be-form*) "е" follows the pronominal clitics.

(93) **Clitic cluster with aux-be-form in third person, singular**

(h) Програмата е изтрила името на файла.
program-det be-aux.Sg3 delete-PastParticipleSg3Fem name of file
The program has deleted the file name.

(i) Програмата го е изтрила.
program-det it-Acc be-aux.Sg3 delete-PastParticipleSg3Fem
The program has deleted it.

(j) Програмата му го е изтрила.
Program to him/he-Dat it-Acc be-aux.Sg3 delete-astParticipleSg3Fem
The program has deleted it (for him).

For third person singular the rules of ordering elements inside clitic cluster are transformed to the following two rules respectively:

dative-pronominal-clitic(s) ^ accusative-pronominal-clitic ^ aux-be-form (e)

or

dative-pronominal-clitic(s) ^ reflexive-clitic ^ aux-be-form (e)

The placement of the clitic cluster is subject to the same ordering restrictions as the placement of a single clitic. Clitics (and clusters) in Bulgarian have their reserved place among the other sentence elements. They can appear immediately before or immediately after the element presenting the Process in the clause. On the other hand, clitics (or clitic cluster) take the position immediately after the last Thematic element. In Bulgarian the sentences beginning by clitic (or clitic cluster) are not possible, because the first position in the sentence is under stress, it is Thematic position.

Specificity of register chosen for the purposes of AGILE project- instructional texts simplified our tasks and aims concerning generation of sentences with clitics. Clitic clusters are hardly used in written Bulgarian in general and in instructional texts in particular. Typically instructional texts tend to be maximally clear in naming the Things and Processes in described situation and pronouns are not frequently used.

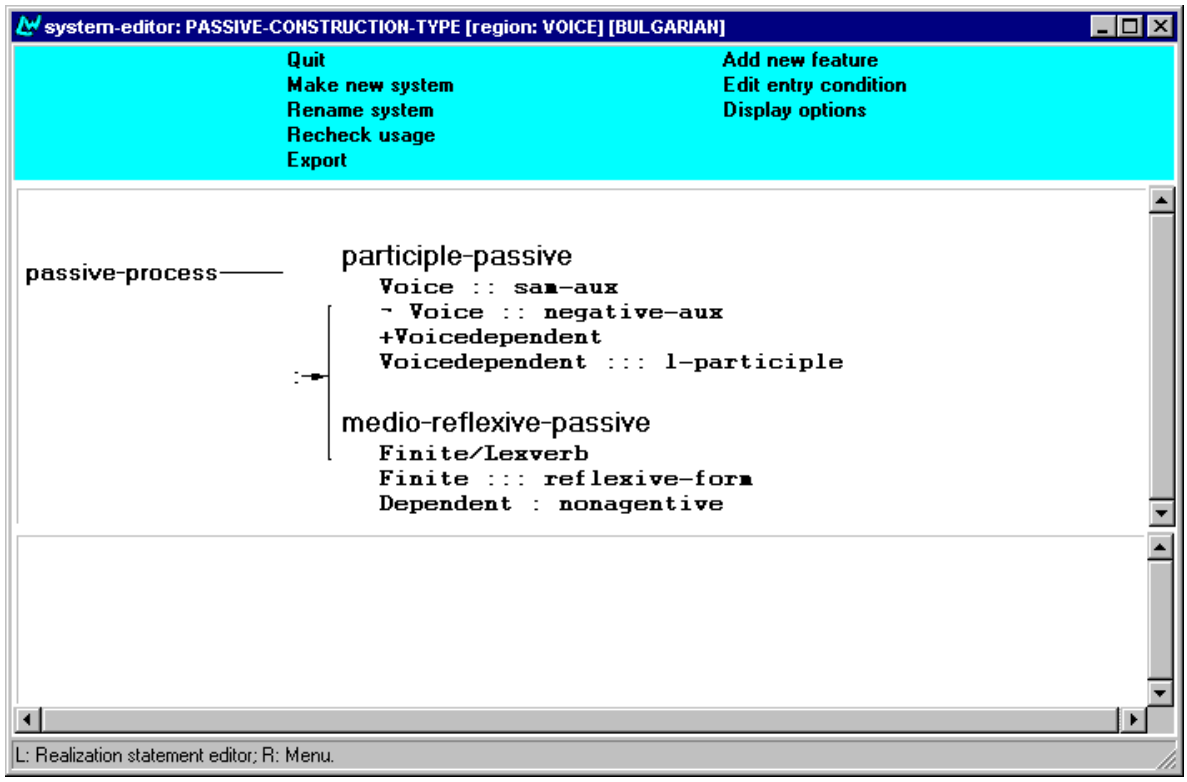
Presenting different styles of our target texts, Personal and Impersonal, we faced the problem of placing the clitic “se” in its right position in the sentence, because each clause in Impersonal style contains medio-passive verb form(s). In Bulgarian, when we disregard the placement of other clitics, the reflexive particle “ce” is placed either after or before the main finite verb form in a clause, depending on whether the verb is clause-initial or not. The sentences in the following examples show Clitic^Finite and Finite^Clitic order.

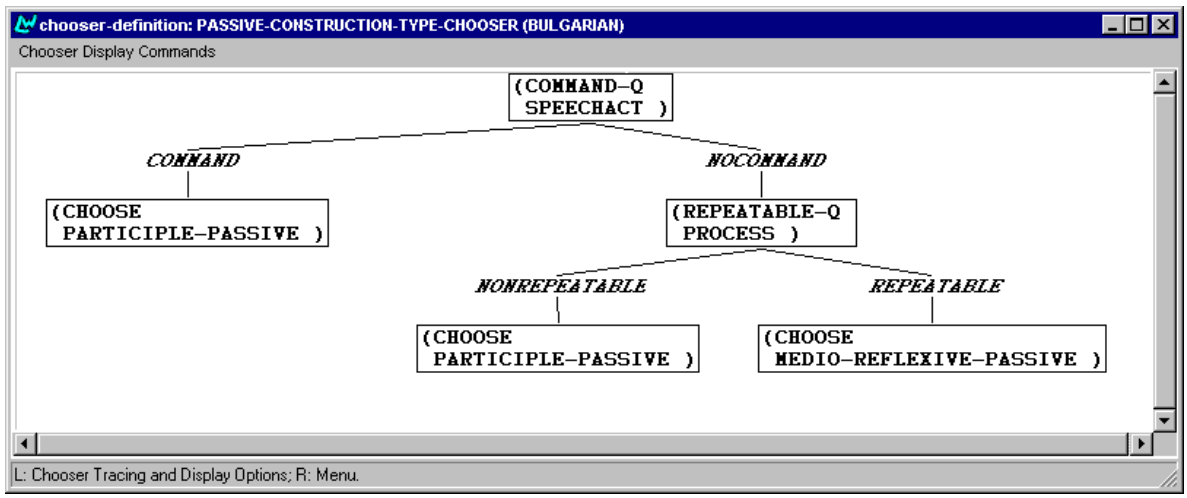
(94) **Finite-Clitic order**
Натиска се клавишът RETURN.
press-medio.passive key-det RETURN
You are pressing RETURN key./RETTURN key is pressed by you.

(95) **Clitic-Finite order**
Добавя се елемент като се натиска клавишът RETURN.
add-medio.passive element by press-refl.passive key-det RETURN
A element is added by pressing RETURN key.

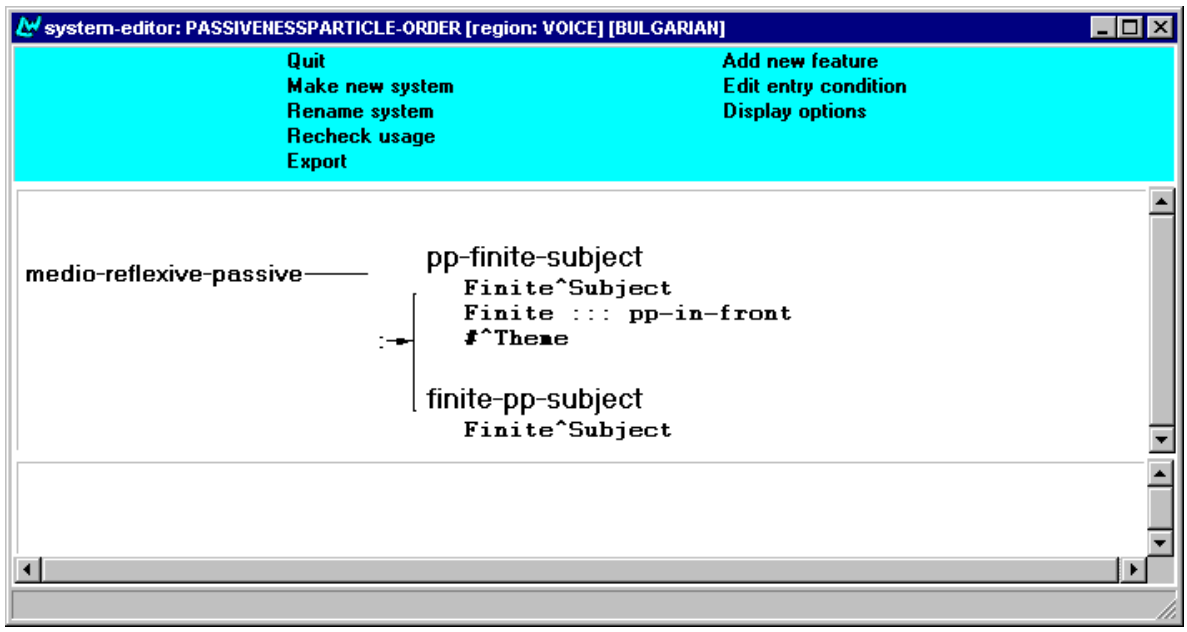
We show bellow the systems and choosers of Bulgarian grammar, which deal with the right place of (medio-passive) reflexive clitic.

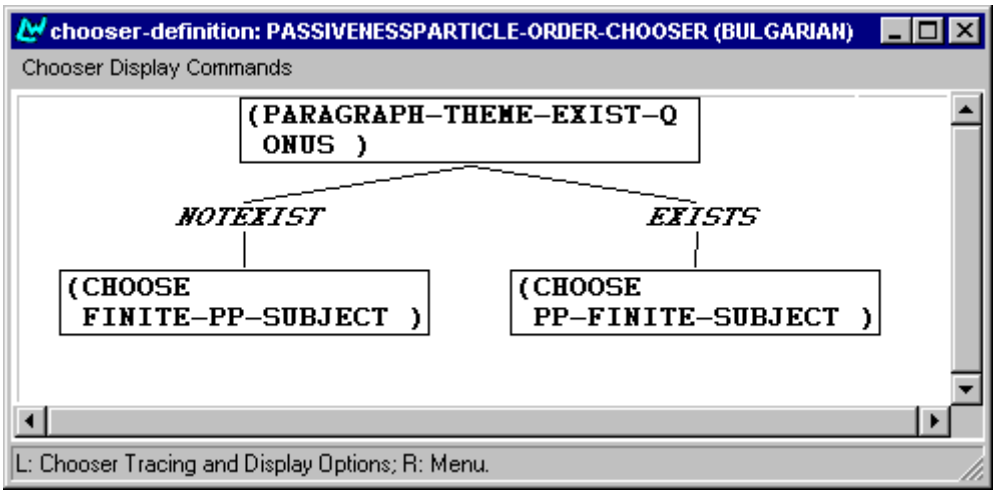
Passiveness Particle “se” appears, when the Process is passive and in medio-passive construction. The Finite is given the characteristic “reflexive-verb”.





Medio-reflexive passive construction can be realized either by Clitic^Finite^Subject order or by Finite^Clitic^Subject order. The later is natural and presents the default case, when Finite precedes Subject.



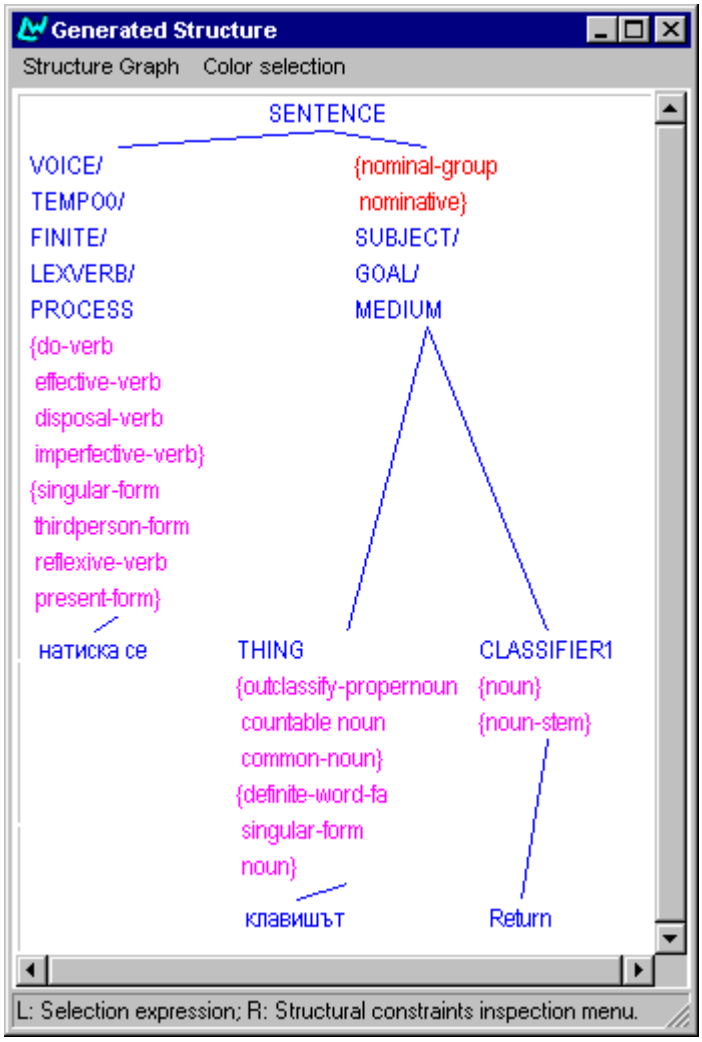


Given the feature **reflexive-verb** Morphological module returns the specified verb form (person and number) and “se” particle is added at the end. Given one more feature “pp-in-front” the resulting verb form begins by particle “se”.

The SPLs and generation results for the examples (97) and (98) are given bellow.

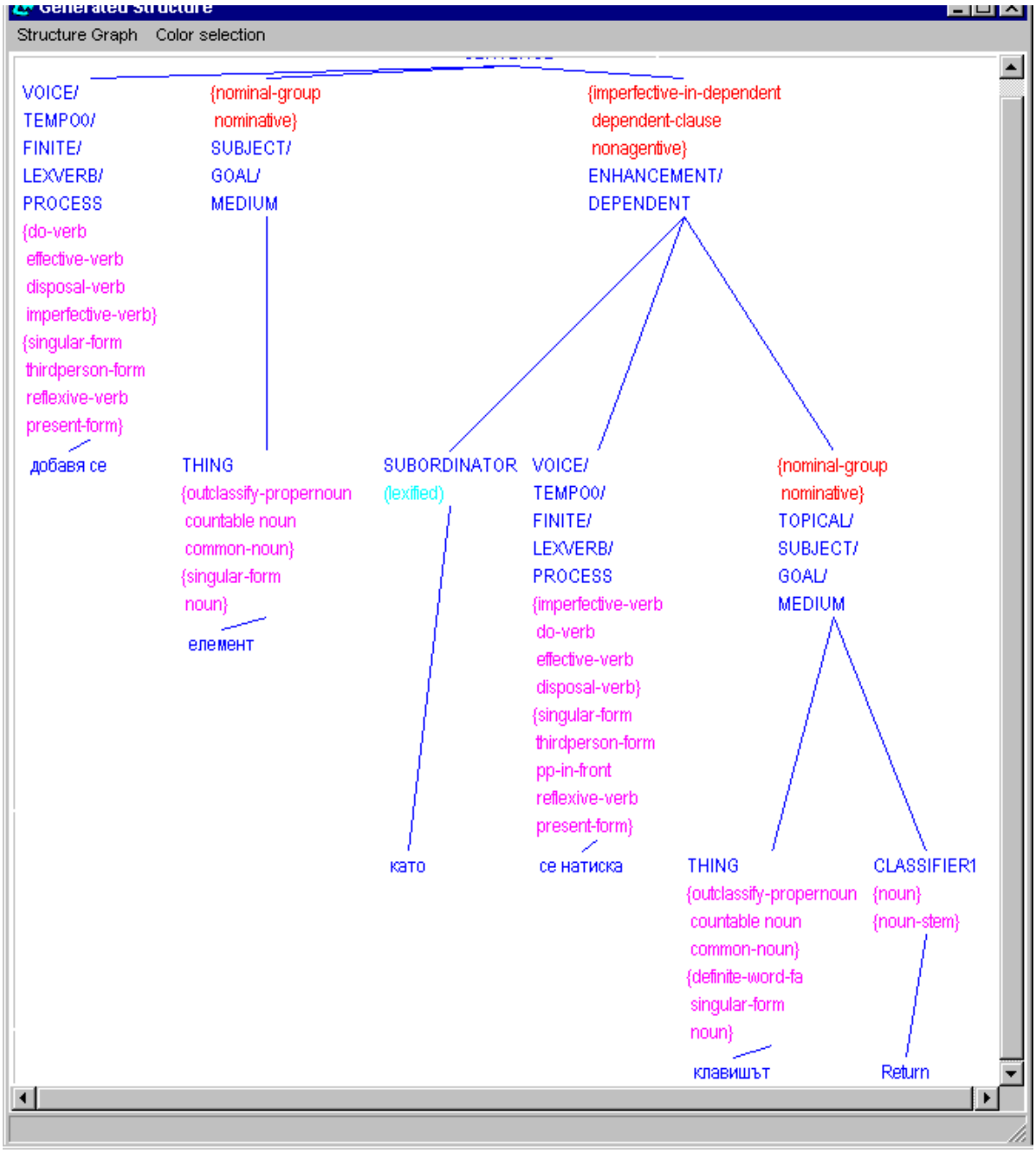
(96)

```
(|a1| / DM::PRESS
 :PREFER-MENTION-AGENT-Q WITHHOLD
 :ACTEE  (|a2| / DM::KEY
:CLASS-ASCRIPTION (|a3| / DM::GUI-RETURN))
      :ACTOR  (HEARER / DM::USER :IDENTIFIABILITY-Q DM::IDENTIFIABLE)
)
```



(97) *Натиска* *се* *клавишът* *Return.*
Press-medio-passive Sg3 refl.particle key Return

```
(S / (DM::RST-MEANS DM::MANNER)
       :PREFER-MENTION-AGENT-Q WITHHOLD
   :DOMAIN (|a1| / DM::ADD-TO
        :ACTEE  (|a2| / DM::OBJECT
:LEX DM::ELEMENT
:NUMBER DM::PLURAL)
                 :ACTOR  (HEARER / DM::USER
:IDENTIFIABILITY-Q DM::IDENTIFIABLE))
       :RANGE  (|a3| / DM::PRESS
        :ACTEE  (|a4| / DM::KEY
              :CLASS-ASCRIPTION (|a5| / DM::GUI-RETURN))
                 :ACTOR  (HEARER / DM::USER
:IDENTIFIABILITY-Q DM::IDENTIFIABLE))
)
```



(98) *Добавя се елемент, като се натиска клавишът* Return
Add-m.passive Sg3 refl element by refl press-M.passiveSg3 key Return
An element is added by pressing the Return key.

3.5.5 Concluding remarks

The fact that word order is relatively free in Slavonic languages, but at the same time not arbitrary with respect to a given communicative intention and the current discourse context, brings about two important issues for the process of automatic generation of continuous texts:

- **coverage**: the generator (the grammar) needs to be able to generate all the grammatically well-formed orderings possible in a given language
- **control**: since word order is not arbitrary, the generator must choose the most appropriate grammatically well-formed ordering with respect to the communicative intention and the context

From a cross-linguistic perspective, or from the viewpoint of building a multilingual NLG system, an additional issue comes in:

- **flexibility**: we want to describe word order in different languages in uniform terms, such that we are able to draw cross-linguistic generalizations;
- furthermore, a multilingual system needs a **uniform mechanism** handling word order, which can be adjusted to handling WO in the particular languages by setting a few parameters, or by providing language-specific preferences.

The approach to word ordering we have developed in Agile takes all these issues into account, and addresses them in a principled way.

3.6 Textual conjunction

3.6.1 Description of the Phenomenon

In order for a reader of an instructional text to easily understand how to accomplish the task to which the instructions pertain, the instructional text needs to reflect in a perspicuous fashion the hierarchical organization of the tasks at hand as well as the sequence of steps to be carried out. The hierarchical organization of the tasks is reflected straightforwardly by the hierarchical organization of the text, or by the hypotactic relationships between clauses within clause complexes. The former is an issue of layout, while the latter is a part and parcel of aggregation (see TEXM3, Sections 3.3.1 and 3.3.2). The latter requires the generated text to reflect the sequencing of the bits of content that are being expressed, and mark the sequencing of steps overtly when necessary (see TEXM3, Section 3.5).

The bits of content as specified in an A-box are inherently sequentially related. The assumption is that the order of sub-steps specified in an A-box corresponds to the order in which the steps are to be carried out. Therefore, the generated instructions need to reflect this underlying "sequencing" of steps. The straightforward way of reflecting the step sequences is by ordering of the sentences in the output. However, when the input content gets more complex, more elaborate structuring of the generated text becomes needed. Two alternatives of realising such more complex content by a well-structured text are demonstrated in Figure 62 and Figure 63 (repeated from TEXS3:3.2.1, where they are discussed in more detail).

To draw a line and arc combination polyline

1. Draw a line segment.
   First start the PLINE command using one of these METHODs:
   Windows: From the Polyline flyout on the Draw toolbar, choose Polyline.
   DOS and UNIX: From the Draw menu, choose Polyline.
   Then specify the start point of the line segment and the endpoint of the line segment.
2. Draw an arc segment.
   First switch to Arc mode by entering **a**. The Arc mode confirmation dialog box appears. Select OK.
   Then specify the endpoint of the arc.
3. Draw another line segment.
   First return to Line mode by entering **l**. The Line mode confirmation dialog box appears. Select OK.
   Then enter the distance and angle of the line in relation to the endpoint of the arc.
4. Press Return to end the polyline.

Figure 62: A combination of sequence styles: using numbered list for the top-level GOALs, and explicit sequence discourse markers for the lower-level GOALs, along with aggregation.

To draw a line and arc combination polyline
First draw a line segment.
1. Start the PLINE command using one of these METHODs:
Windows: From the Polyline flyout on the Draw toolbar, choose Polyline.
DOS and UNIX: From the Draw menu, choose Polyline.
2. Specify the start point of the line segment.
3. Specify the endpoint of the line segment.

Then draw an arc segment.
1. Enter **a** to switch to Arc mode. The Arc mode confirmation dialog box appears.
2. Select OK.
3. Specify the endpoint of the arc.

Then draw another line segment.
1. Enter **l** to return to Line mode. The Line mode confirmation dialog box appears.
2. Select OK.
3. Enter the distance and angle of the line in relation to the endpoint of the arc.

Finally, press Return to end the polyline.

Figure 63: A combination of sequence styles: using explicit sequence markers for the top-level GOALs within the list of steps, and a numbered list for the lower-level steps.

We discussed in detail various alternative ways of realizing complex content in Section 3.2 in the TEXS3 deliverable. In TEXS3:3.2.1, we substantiated the need for explicit marking of sequences as a means to facilitate a reader's orientation in a text which describes complex structured tasks. Essentially, we proposed to distinguish between the following *sequence realization styles*:

- Running text sequence
  - Unmarked running text sequence (realised by a continuous paragraph where the elements in the sequence do not include any overt sequence markers)
  - Linguistically marked running text sequence (realised by a continuous paragraph where the elements in the sequence include overt sequence discourse markers)
- List sequences
  - Unmarked list sequences (realised by bullet lists in which the elements in the sequence do not include any overt sequence discourse markers)
  - Numbered list sequences (realised by numbered lists in which the elements in the sequence do not include any overt sequence discourse markers in addition to the numbering, which in itself constitutes sufficient overt sequence marking)
  - Linguistically marked list sequences (realised by lists of elements in which the elements in the sequence include overt sequence discourse markers)

By *(linguistic) sequence discourse markers* we mean expressions like 'firstly', 'secondly', 'first', 'second', 'third', etc., 'now', 'then', 'finally', 'lastly', 'further', etc. (and their Bulgarian, Czech and Russian counterparts, see Figure 64 below). The following examples (99), (100) and (101) illustrate simple sequences employing the three different sequence realisation styles of the same fragment of content in English, Czech, Bulgarian and Russian.

- **English:**
  2 elements: *First, Now*
  3 and more elements: *First*, (*Then*)[+23], *Finally*
- **Bulgarian:**
  2 elements: *Отначало, Сега*
  3 and more elements: *Отначало,* (*След това*)[+], *Накрая*
- **Czech:**
  2 elements: *Nejprve, Nyní*
  3 and more elements: *Nejprve,* (*Potom*)[+], *Nakonec*
- **Russian:**
  2 elements: *Сначала, Затем* (*Теперь* is also possible, but the context of its usage is restricted to the here-and-now interaction style)
  3 and more elements: *Сначала ,*(*Затем*)[+]*, И наконец*

Figure 64: Different linguistic markers depending on the number of elements in a sequence

(99) Numbered list

(a) English

1. Choose Element Properties. The Element Properties dialog box appears.
2. Enter the offset of the multiline element in the Element Properties dialog box.
3. Select Add to add the element.

(b) Czech

1. Vyberte Element Properties. Objeví se dialogové okno Element Properties.
2. V dialogovém okně Element Properties zadejte offset elementu multičáry.
3. Pro přidání elementu vyberte Add.

(c) Bulgarian

1. Изберете Element Properties. Появява се диалоговият прозорец Element Properties.
2. Въведете отместването на елемента на мултилинията в диалоговия прозорец Element Properties.
3. Изберете Add, за да добавите елемента.

(d) Russian

1. Нажмите кнопку Element Properties. Появится диалоговое окно Element Properties.
2. В диалоговом окне Element Properties введите смещение элемента мультилинии
3. Нажмите кнопку Add, чтобы добавить элемент.

(100) Linguistically marked list

(a) English

First choose Element Properties. The Element Properties dialog box appears.
Then enter the offset of the multiline element in the Element Properties dialog box.
Finally select Add to add the element.

[23] (X)+ means one or more occurrences of 'then'.

(b) Czech

Nejprve vyberte Element Properties. Objeví se dialogové okno Element Properties.
Potom v dialogovém okně Element Properties zadejte offset elementu multičáry.
Nakonec vyberet Add pro přidání tohoto elementu.

(c) Bulgarian

Отначало изберете Element Properties. Появява се диалоговият прозорец Element Properties.
След това въведете отместването на елемента на мултилинията в диалоговия прозорец Element Properties.
Накрая изберете Add, за да добавите елемента.

(d) Russian

Сначала нажмите кнопку Element Properties. Появится диалоговое окно Element Properties.
Затем диалоговом окне Element Properties введите смещение элемента мультилинии.
И наконец Нажмите кнопку Add, чтобы добавить элемент.

(101) Linguistically marked running text

(a) English

First choose Element Properties. The Element Properties dialog box appears. Then enter the offset of the multiline element in the Element Properties dialog box. Finally select Add to add the element.

(b) Czech

Nejprve vyberte Element Properties. Objeví se dialogové okno Element Properties. Potom v dialogovém okně Element Properties zadejte offset elementu multičáry. Nakonec vyberte Add pro přidání tohoto elementu.

(c) Bulgarian

Отначало изберете Element Properties. Появява се диалоговият прозорец Element Properties. След това въведете отместването на елемента на мултилинията в диалоговия прозорец Element Properties. Накрая изберете Add, за да добавите елемента.

(d) Russian

Сначала нажмите кнопку Element Properties. Появится диалоговое окно Element Properties. Затем диалоговом окне Element Properties введите смещение элемента мультилинии. И наконец нажмите кнопку Add, чтобы добавить элемент.

As is apparent from the classification of the sequence styles above, issues of *layout* and *linguistics realisation* are intertwined there in the area of sequence realisation: While the distinction between a list and a running text is a matter of layout choice, the distinction between numbering and discourse markers is a matter of linguistic realisation choice. The text and sentence planning issues involved in making these choices were discussed in detail in the TEXS3 and TEXM3 deliverables.

In the present section, we concentrate on the explicit sequence marking in separate sentences in a procedural text, using linguistic markers. In general, linguistic markers express **non-structural relations** where the related content is realised in different sentences, but the relationship is made explicit. The relationship (e.g., sequence, cause) reflected by a linguistic marker constitutes a **cohesive bond** between the two (or more) sentences.

In SFG, this type of cohesion is known as **conjunction** (Halliday 1985, pp. 323--330). In order to avoid potential terminological confusion, we use the term **textual conjunction**

henceforth. Halliday mentions a range of possible meanings within the domains of **elaboration, extension and enhancement**[24] which can be expressed by the choice of a conjunctive Adjunct (i.e., an adverbial group or prepositional phrase), or by one of a set of conjunctions (e.g., *and, or, nor, but, yet, so, then*). As for their placement in a sentence, the conjunctive adjuncts typically and the conjunctions obligatorily occupy a position within the Theme at the beginning of a sentence (not only in English as discussed by Halliday, but also in Czech, Russian and Bulgarian).

The classification of textual conjunction proposed by Halliday is summarised below:

- Elaboration
  - Apposition (expository, exemplifying)
  - Clarification (corrective, distractive, dismissive, particularizing, resumptive, summative, verificative)
- Extension
  - Addition (positive, negative)
  - Adversative
  - Variation (replacive, subtractive, alternative)
- Enhancement
  - Spatio-temporal
    - Simple external (following, simultaneous, preceding, conclusive)
    - Complex (immediate, interrupted, repetitive, specific, durative, terminal, punctiliar)
    - Simple internal (following, simultaneous, preceding, conclusive)
  - Manner
    - Comparison (positive, negative)
    - Means
  - Causal
    - General
    - Specific (result, purpose, reason)
    - Conditional (positive, negative, concessive)
  - Matter (positive, negative)

From this wealth of different types of textual conjunction, one particular subtype is relevant for the full instructions in procedural style generated in the Final Prototype of the Agile system, namely **simple external temporal enhancement**, which is the type of textual conjunction employed in explicit linguistic marking of sequences. 'External' means that the time referred to by the conjunctions is the temporal sequence of the processes referred to, as opposed to 'internal' where it would be the temporal unfolding of the discourse itself.

An additional classification of textual conjunction is discussed by Martin (Martin 1992). He distinguishes the following different types of **sequence regulation**:

---

[24] Cf. the discussion of clause complexity in the LSPEC2 deliverable for an explanation of elaboration, extension and enhancement.

- Presentational
- Temporal
- Numerical
- Logical

In the full instructions in procedural style in the Agile project, we are making use of two of these types, namely numerical temporal sequence regulation and temporal sequence regulation. **Numerical temporal sequence** regulation is manifested by the use of explicit numbering of steps in a sequence. However, as described in the TEXM3 deliverable, numbering is not realised by the lexicogrammar, but it is obtained through inserting the appropriate HTML mark-up and its subsequent interpretation using a suitable viewer (e.g., Microsoft Explorer in the Final Prototype).

The class of textual conjunction needed for the explicit marking of sequences using linguistic discourse markers is specified as **temporal sequence regulation**, more precisely, the cross-classification of **sequence regulation** and **simple external temporal enhancement**. Unlike numbering, this type of temporal sequence regulation needs to be realised by the lexico-grammars, because it involves the use of language-specific expressions as textual conjunctions.

In the next section, we present linguistic specifications pertaining to the temporal sequence regulation type of textual conjunction as developed for the Agile Final Prototype.

3.6.2 Formal Specifications

In this section we provide the formal linguistic specifications for the temporal sequence regulation type of textual conjunction described in the previous section. The corresponding implementations are presented in the next section.

The specification of temporal sequence regulation starts by distinguishing a **conjuncted** clause from a **nonconjuncted** one.

```
System: CONJUNCTION →
        [NONCONJUNCTED]
        [CONJUNCTED]
Chooser: CONJUNCTION
    Check whether the semantics specifies a conjunctive relation
```

The purpose of the first few systems which actually deal with temporal sequence regulation is to distinguish this type of textual conjunction from other types. First, the **non-structural** conjunction is distinguished from **structural** conjunction.

```
System: STRUCTURAL-CONJUNCTION →
         [NONSTRUCTURAL-CONJUNCTED]
         [STRUCTURAL-CONJUNCTED]
Chooser: STRUCTURAL-CONJUNCTION-CHOOSER
    For an additional conjunctive relation, choose structural.
```

Next, it is decided whether the conjunctive relation can be lexified.

```
System: LEXIFIED-CONJUNCTION →
        [LEXIFIED-CONJUNCTIVE]
        [OPENCHOICE-CONJUNCTIVE]
```

```
Chooser: LEXIFIED-CONJUNCTION-CHOOSER
    If the conjunction can be lexified,
    then choose lexified-conjunctive
    else choose openchoice-conjunctive
```

Now, a **process-regulated** textual conjunction is distinguished.

```
System: CONJUNCTIVE-PROCESS-REGULATION →
        [PROCESS-REGULATED]
        [NOT-PROCESS-REGULATED]
Chooser: CONJUNCTIVE-PROCESS-REGULATION-CHOOSER
    If the conjunction relationship arises from a progression or a
    logical derivation,
    then choose process-regulated
    else choose not-process-regulated
```

Finally, the choice of textual conjunction type is made.

```
    System: PROCESS-REGULATED-TYPE →
        [NECESSITY-REGULATION]
        [SEQUENCE-REGULATION]
        [TEMPORAL-REGULATION]
        [SPATIAL-REGULATION]
Chooser: PROCESS-REGULATED-TYPE-CHOOSER
   If the conjunction relationship is a logical consequence
   then choose necessity-regulation
   else
     if it is a presentational, numerical, temporal or logical sequence
     then choose sequence-regulation
     else
       if it is a relationship in time,
       then choose temporal-regulation
       else choose spatial-regulation
```

Once the feature **sequence-regulation** has been reached, a distinction is made between an **absolute** and a **relative** position in the sequence.

```
    System: SEQUENCE-CONJUNCTION  →
        [RELATIVE-SEQUENCE]
        [ABSOLUTE-SEQUENCE]
Chooser: SEQUENCE-CONJUNCTION-CHOOSER
If the conjunction relationship corresponds to an absolute position in
the sequence
   then choose absolute-sequence
   else choose relative-sequence
```

In English, the realization corresponding to **relative-sequence** would be "further". As for the absolute position, there is another decision made:

```
    System: ABSOLUTE-SEQUENCE-CONJUNCTION →
        [FIRST-POSITION]
        [LAST-POSITION]
        [OTHER-POSITION]
Chooser: ABSOLUTE-SEQUENCE-CONJUNCTION-CHOOSER
If the conjunction relationship specifies that the proposition is in
the first position in the sequence,
   then choose first-position
   else if it specifies that it is in the last position,
```

```
then choose last-position
else choose other-position
```

In English, **first-position** would correspond to "first(ly)", **last-position** to "finally/last(ly)/ultimately" and **other-position** to "secondly". The Czech, Bulgarian and Russian counterparts of "further", "firstly", "finally" and "secondly" are shown in Figure 64, which is repeated below for convenience.

- **English:**
  2 elements: *First, Now*
  3 and more elements: *First*, (*Then*)[+25], *Finally*
- **Bulgarian:**
  2 elements: *Отначало, Сега*
  3 and more elements: *Отначало,* (*След това*)[+], *Накрая*
- **Czech:**
  2 elements: *Nejprve, Nyní*
  3 and more elements: *Nejprve,* (*Potom*)[+], *Nakonec*
- **Russian:**
  2 elements: *Сначала, Затем* (*Теперь* is also possible, but the context of its usage is restricted to the here-and-now interaction style)
  3 and more elements: *Сначала ,*(*Затем*)[+]*, И наконец*

Figure 65: Different linguistic markers depending on the number of elements in a sequence

3.6.3 Implementation

We have adapted the CONJUNCTION region from the Nigel grammar for English to satisfy the needs of generating textual conjunction of the temporal sequence regulation type needed in the full instructions in procedural style in Czech, Russian and Bulgarian n the Agile Final Prototype. In particular, as discussed above, we need to generate the Czech, Russian and Bulgarian counterparts of the textual conjunctions "first", "secondly", "then" and "finally" (cf. Figure 65). The implementations as we present them in this section have been fully implemented in all three grammars.

Temporal sequence regulation is implemented in the CONJUNCTION region in our grammars. The first few systems in this region differentiate temporal sequence regulation from other types of textual conjunction as non-structural, lexifiable and process-regulated (cf. Figure 66 through Figure 75). These systems are the same in all three grammars.

Once the feature **sequence-regulation** has been reached, a distinction is made between an **absolute** and a **relative** position in the sequence (cf. Figure 76 through Figure 79). The relative position corresponds to English "further". The absolute position splits further (cf. Figure 80 through Figure 83). While the Nigel implementation distinguished only between "ultimately" and "secondly" for the absolute position, we are using a three-way distinction between "first position", "final/last/ultimate position" and "other position". While "first" is an

[25] (X)+ means one or more occurrences of 'then'.

added distinction, "other" replaces the original "secondly" by something more general, corresponding to "now".

These are all the systems needed to generate the textual conjunctions in the Agile Final Prototype. In (102), we present an example of a simple sentence with a textual conjunctive, in the Czech, Russian and Bulgarian variant, and the corresponding SPL. Then we show the outputs generated by our grammars.

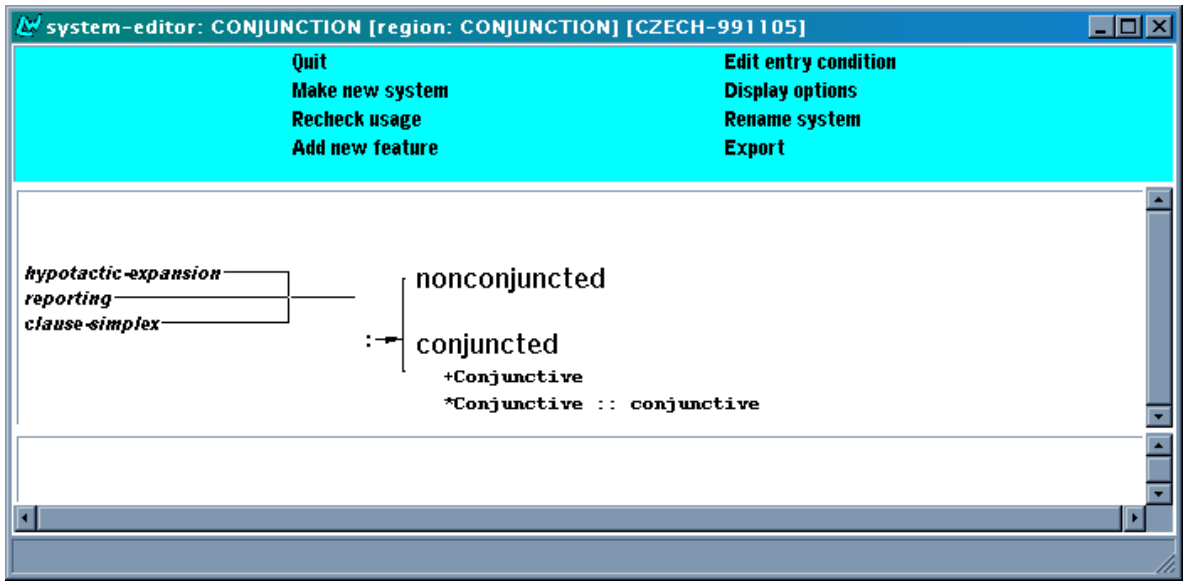


Figure 66: CONJUNCTION system (Cz, Bg, Ru)

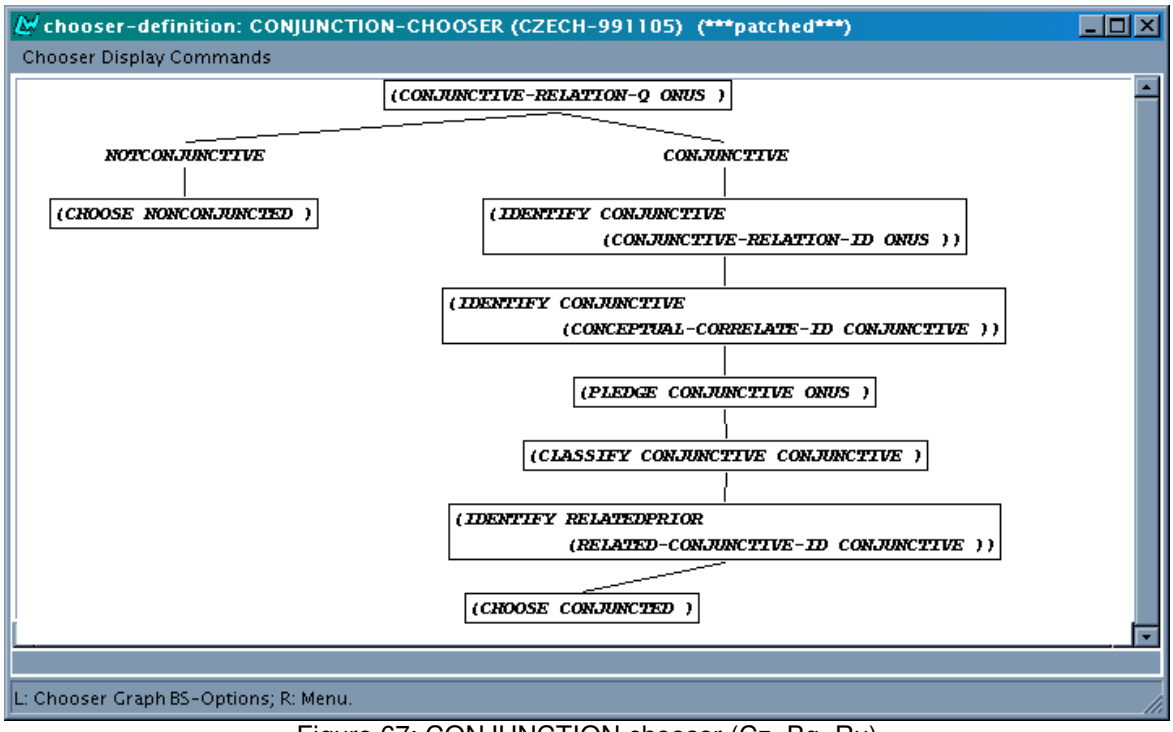


Figure 67: CONJUNCTION chooser (Cz, Bg, Ru)

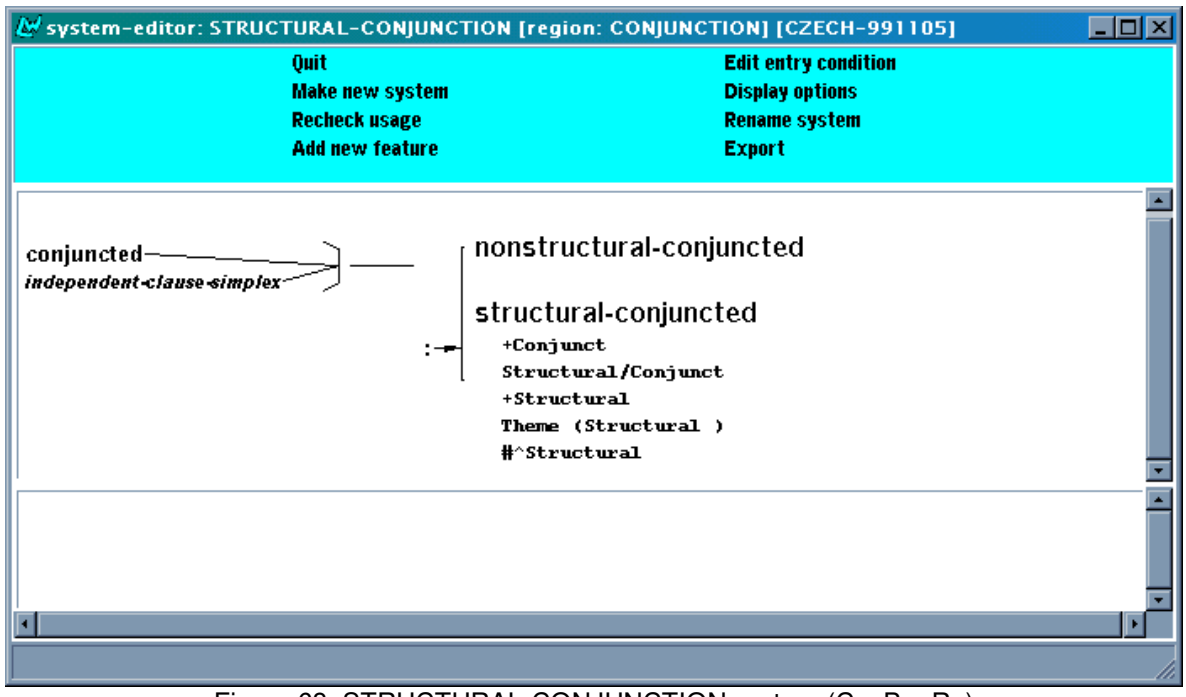


Figure 68: STRUCTURAL-CONJUNCTION system (Cz, Bg, Ru)

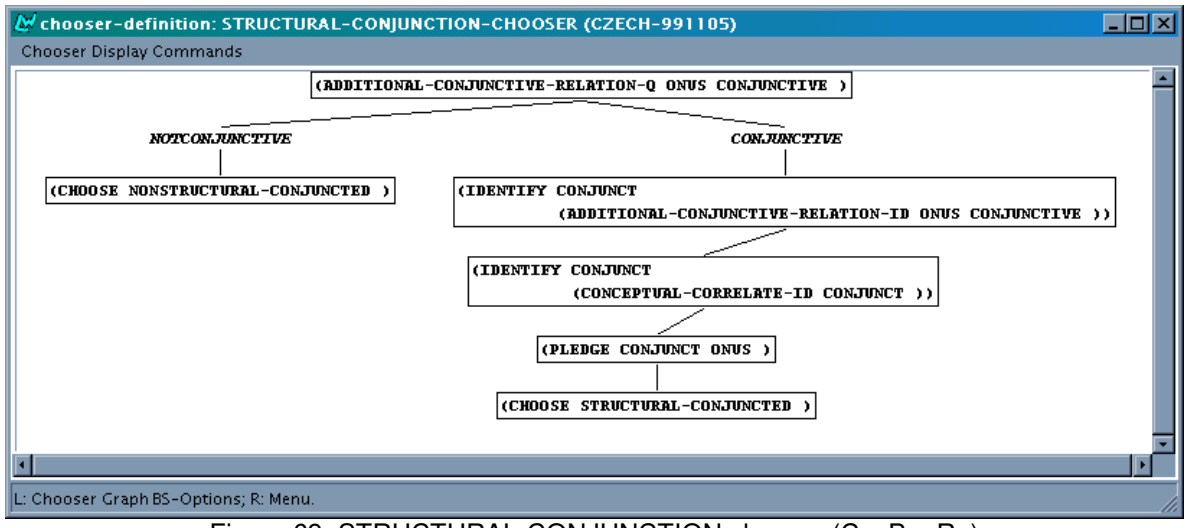


Figure 69: STRUCTURAL-CONJUNCTION chooser (Cz, Bg, Ru)

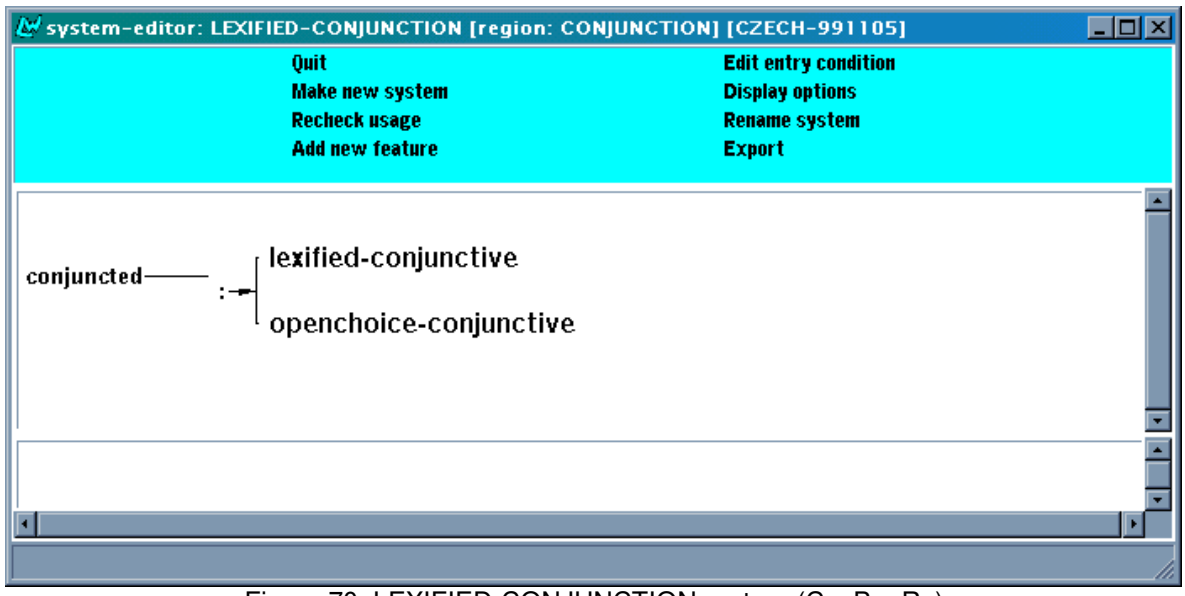


Figure 70: LEXIFIED-CONJUNCTION system (Cz, Bg, Ru)

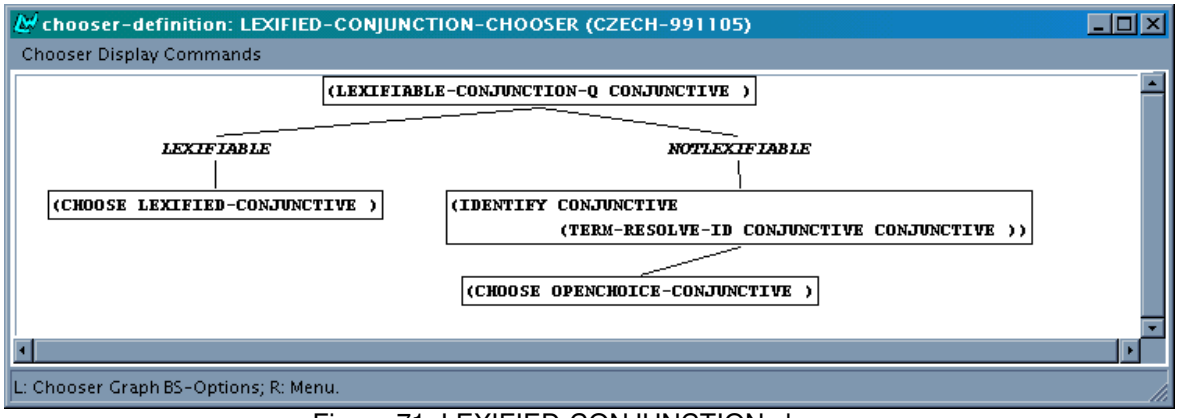


Figure 71: LEXIFIED-CONJUNCTION chooser

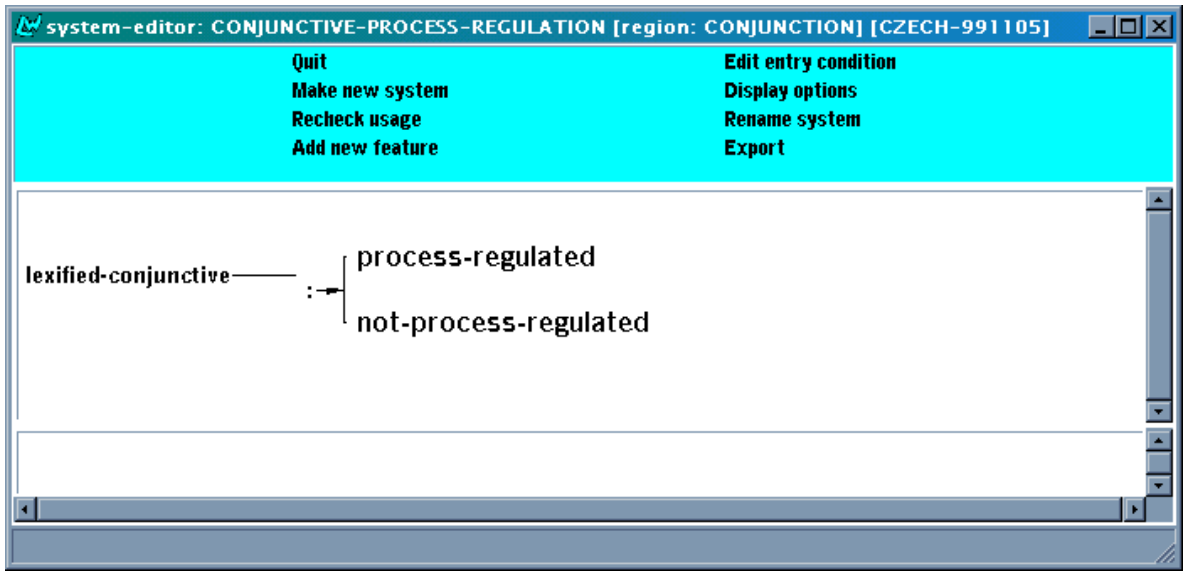


Figure 72: CONJUNCTIVE-PROCESS-REGULATION system (Cz, Bg, Ru)

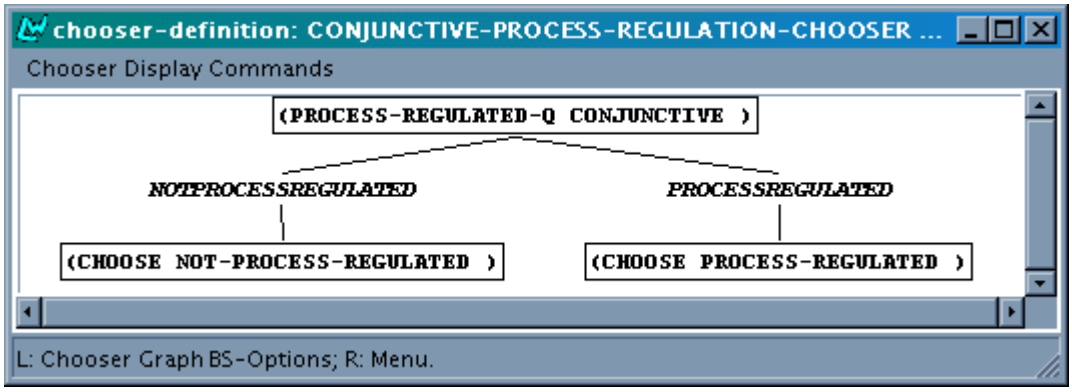


Figure 73: CONJUNCTIVE-PROCESS-REGULATION chooser (Cz, Bg, Ru)

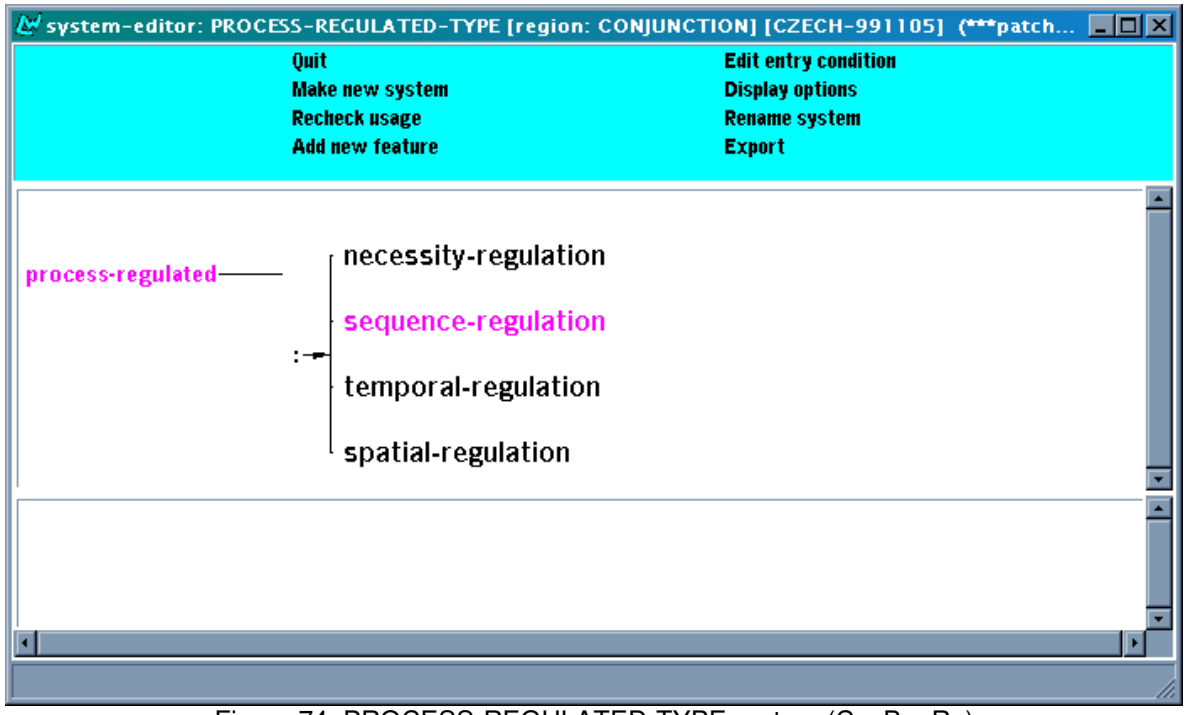


Figure 74: PROCESS-REGULATED-TYPE system (Cz, Bg, Ru)

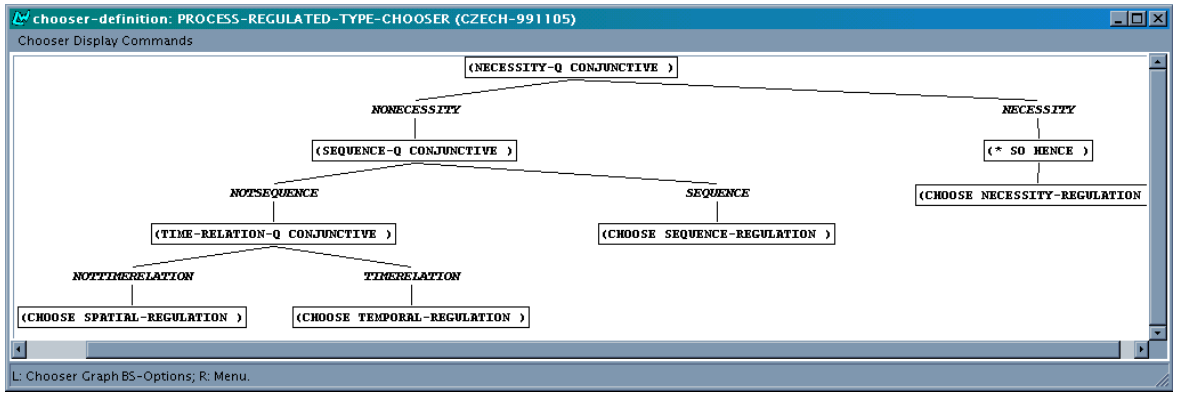


Figure 75: PROCESS-REGULATED-TYPE chooser (Cz, Bg, Ru)

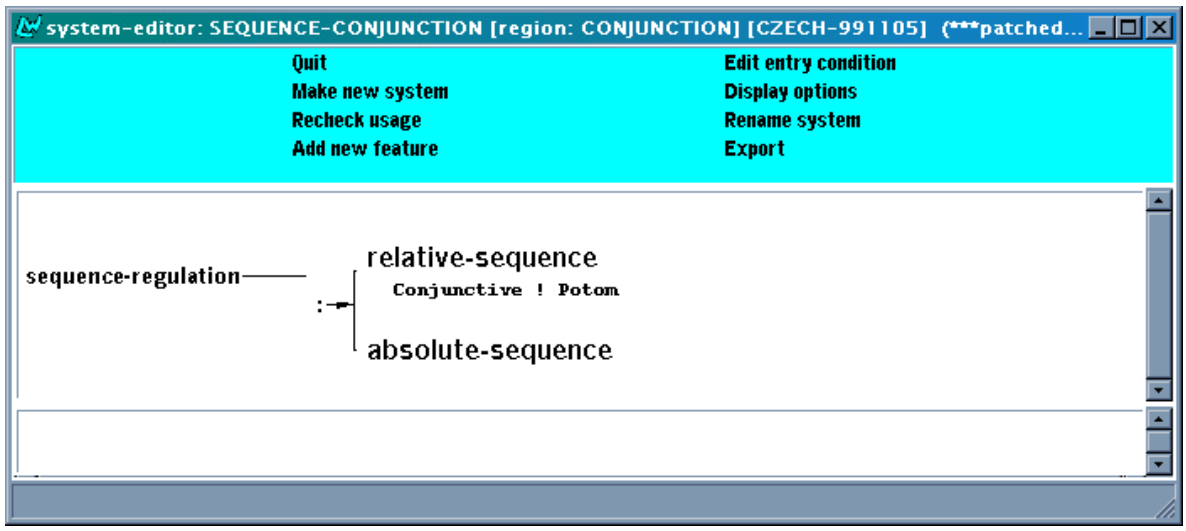


Figure 76: SEQUENCE-CONJUNCTION system (Cz)

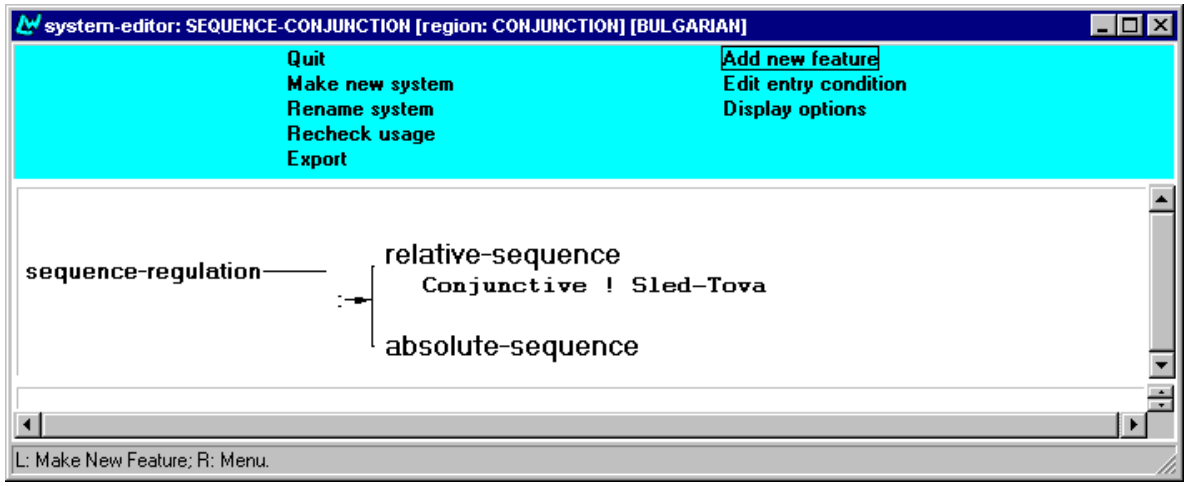


Figure 77: SEQUENCE-CONJUNCTION system (Bg)

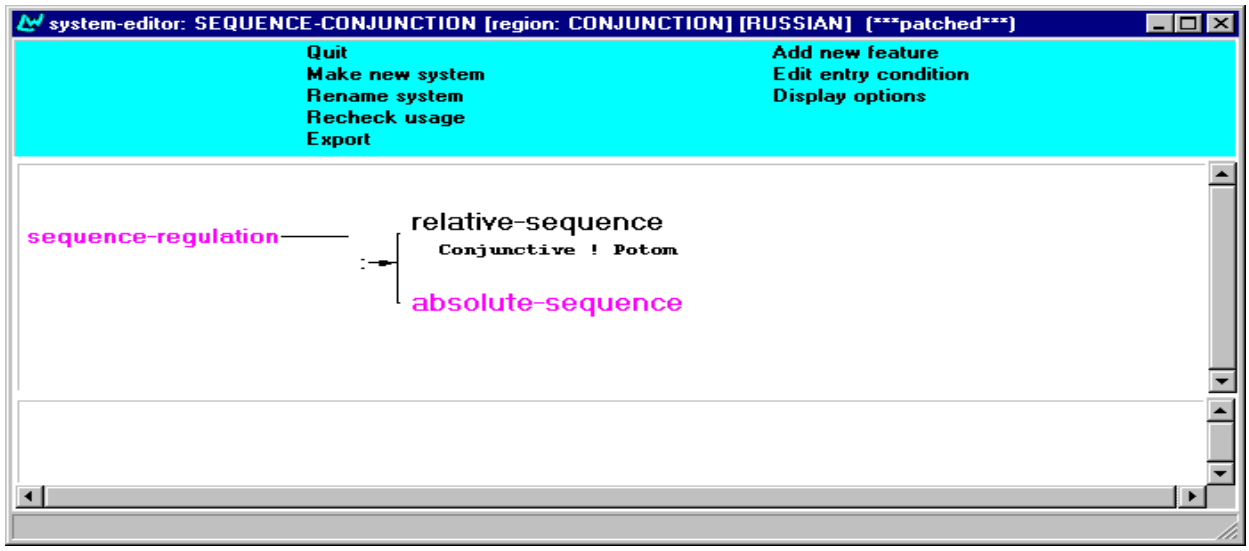


Figure 78: SEQUENCE-CONJUNCTION system (Ru)

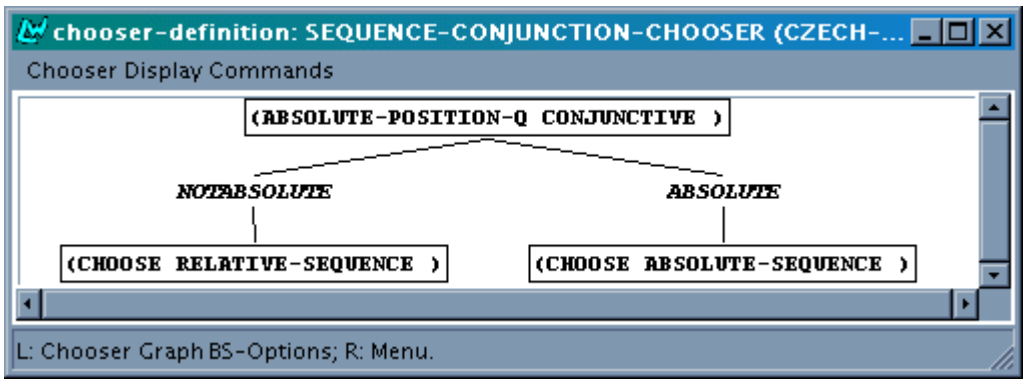


Figure 79: SEQUENCE-CONJUNCTION chooser (Cz, Bg, Ru)

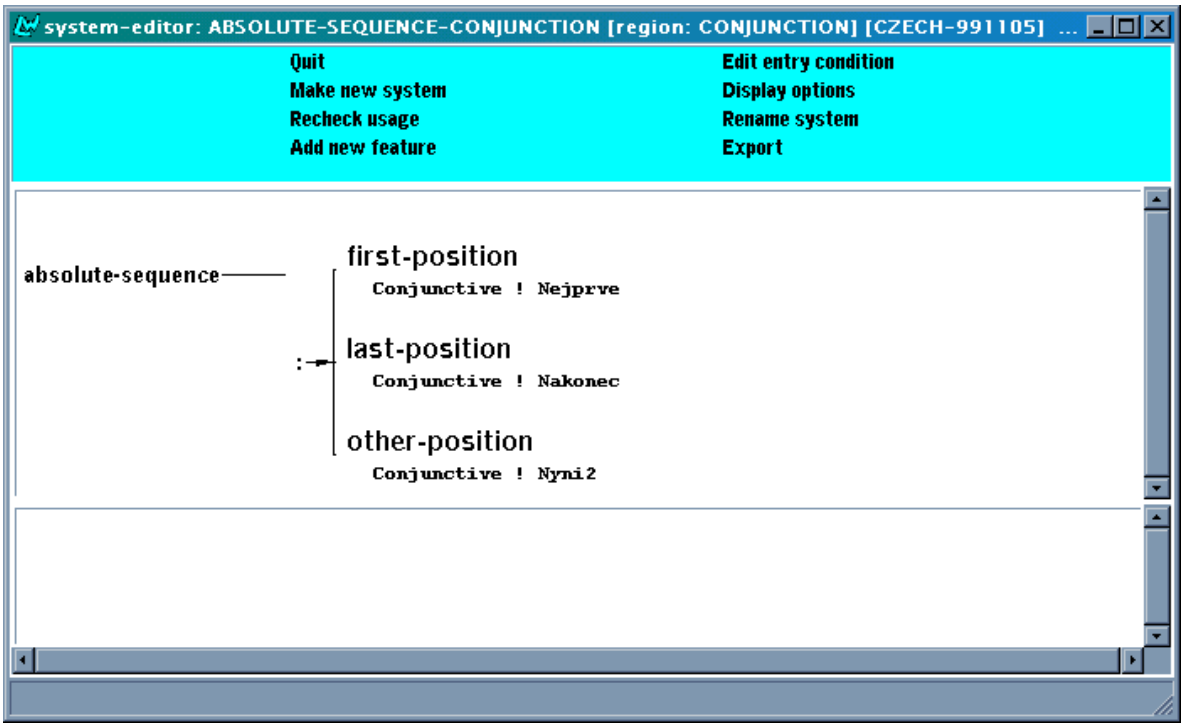


Figure 80: ABSOLUTE-SEQUENCE-CONJUNCTION system (Cz)

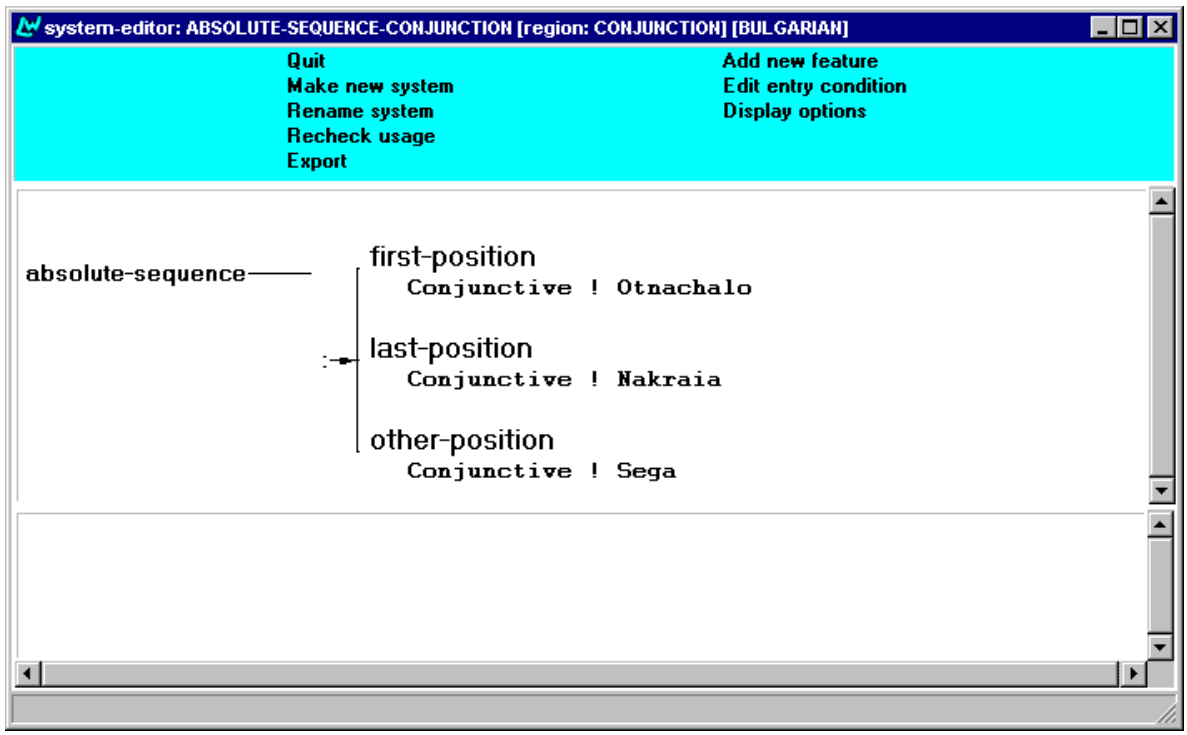


Figure 81: ABSOLUTE-SEQUENCE-CONJUNCTION system (Bg)

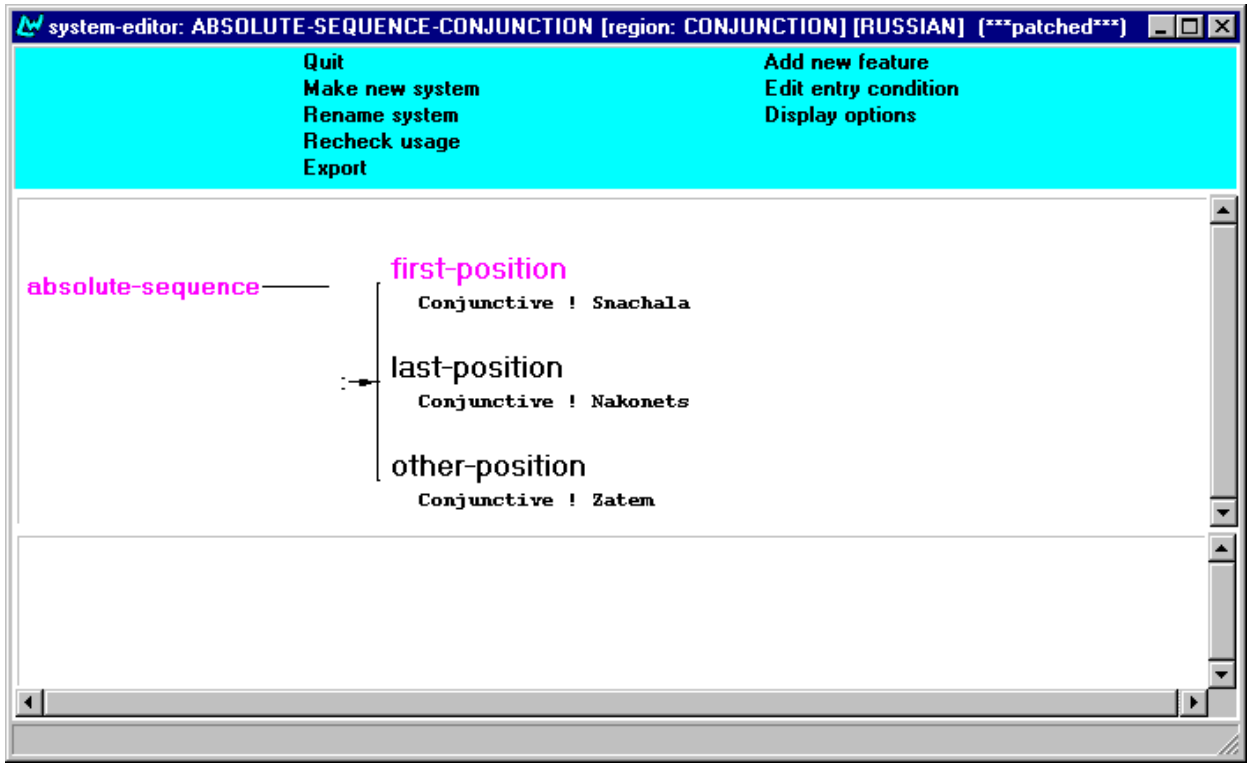


Figure 82: ABSOLUTE-SEQUENCE-CONJUNCTION system (Ru)

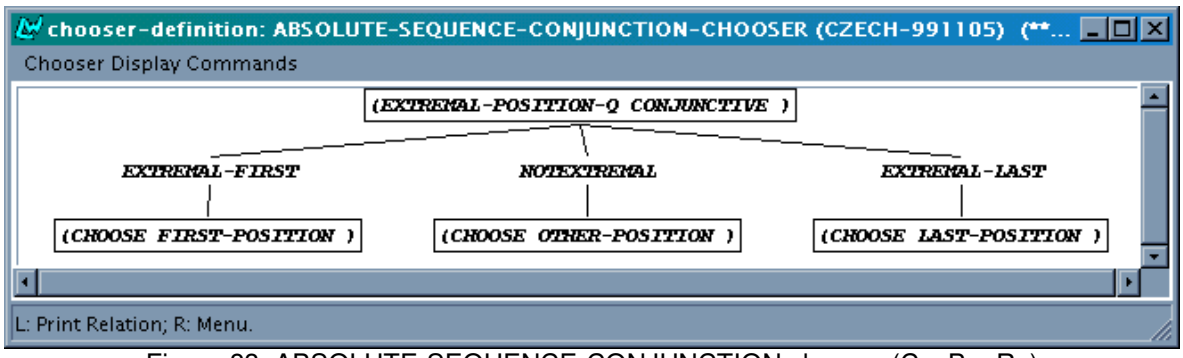


Figure 83: ABSOLUTE-SEQUENCE-CONJUNCTION chooser (Cz, Bg, Ru)

The following example illustrates the use of a textual conjunctive in a simple sentence. In Figure 84 we present the corresponding SPL. Figure 86 shows the generated grammatical structure using the Czech grammar, Figure 87 shows the generated grammatical structure using the Bulgarian grammar, and Figure 88 shows the generated grammatical structure using the Russian grammar.

(102) First define a multiline.

(a) Cz: *Nejprve* *definujte* *multičáru.*
First define-imp multiline-Acc

(b) Bg: *Отначало* *дефинирайте мултилинията.*
First define-imp multiline

(c) Ru: *Сначала* *определите* *мультилинию.*
First define-imp multiline-Acc

```
(|a35| / DM::DEFINE
:SPEECHACT IMPERATIVE
:CONJUNCTIVE SEQUENCE-FIRST
:ACTEE (|a38| / DM::MULTILINE :IDENTIFIABILITY-Q NOTIDENTIFIABLE)
:ACTOR (HEARER / DM::USER :IDENTIFIABILITY-Q DM::IDENTIFIABLE)
)
```

Figure 84: SPL for the sentences in (102)

The SPL statement `:conjunctive first` in the SPL in Figure 84 is an abbreviation using an SPL macro. The SPL macros for the conjunctives needed in the Agile Final Prototype texts are as shown in Figure 85:

```
Conjunctive-first =
          (:conjunctive-relation-q conjunctive
           :conjunctive-relation-id (?rr / rhetorical-relation)
           :process-regulated-q processregulated
           :necessity-q nonecessity
           :sequence-q sequence
           :absolute-position-q absolute
           :extremal-position-q extremal-first)
Conjunctive-last =
         (:conjunctive-relation-q conjunctive
          :conjunctive-relation-id (?rr / rhetorical-relation)
          :process-regulated-q processregulated
          :necessity-q nonecessity
          :sequence-q sequence
          :absolute-position-q absolute
          :extremal-position-q extremal-last)
Conjunctive-other =
          (:conjunctive sequence-other
           :conjunctive-relation-q conjunctive
           :conjunctive-relation-id (?rr / rhetorical-relation)
           :process-regulated-q processregulated
           :necessity-q nonecessity
           :sequence-q sequence
           :absolute-position-q absolute
           :extremal-position-q nonextremal)
Conjunctive-relative =
          (:conjunctive-relation-q conjunctive
           :conjunctive-relation-id (?rr / rhetorical-relation)
           :process-regulated-q processregulated
           :necessity-q nonecessity
           :sequence-q sequence
           :absolute-position-q notabsolute
           :relative-position-q immediate)
```

Figure 85: SPL macros for textual conjunctives

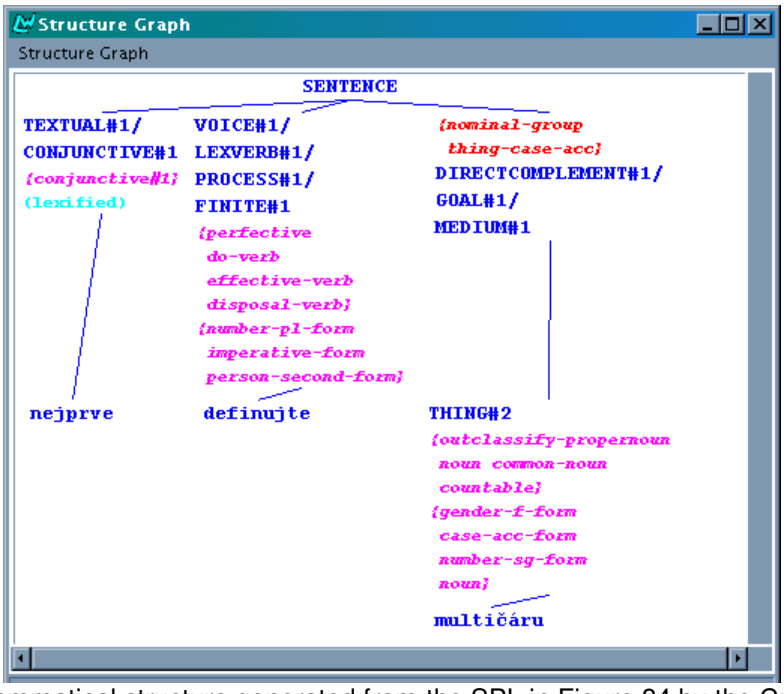


Figure 86: Grammatical structure generated from the SPL in Figure 84 by the Czech grammar

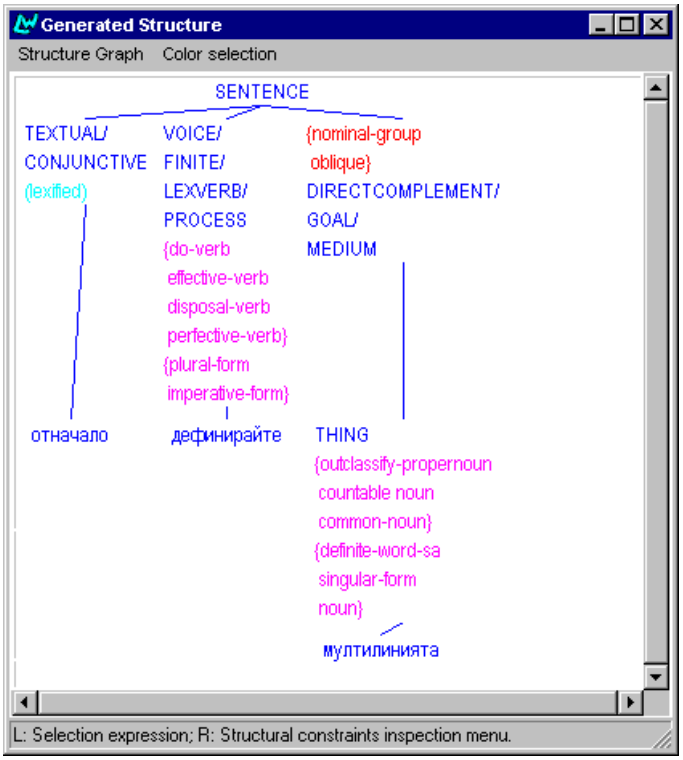


Figure 87: Grammatical structure generated from the SPL in Figure 84 by the Bulgarian grammar

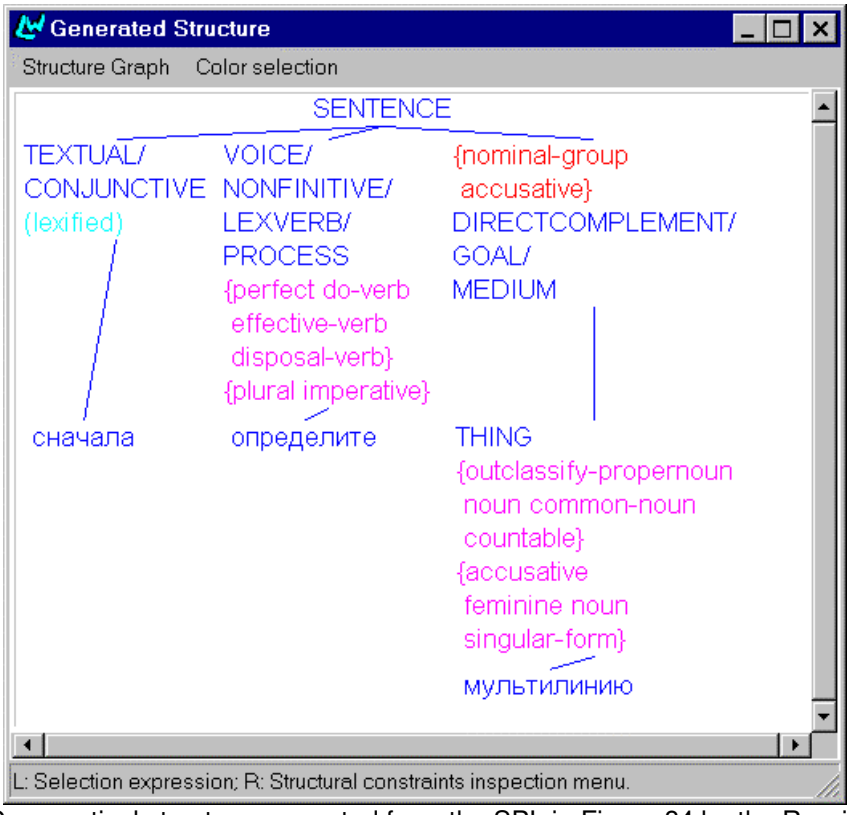


Figure 88: Grammatical structure generated from the SPL in Figure 84 by the Russian grammar

3.7 Agreement

3.7.1 Introduction

Agreement or congruence can be described as two (or more) syntactical units sharing particular grammatical features, e.g., case, number, gender or person. In Czech, Bulgarian and Russian, we can distinguish three kinds of agreement:

1. Subject – predicate.

(103) En: *The line disappeared.*
Bg: *Линията е изчезнала.*
L: line-FSg is-Sg3 disappeared-FSg
Cz: *Úse•ka zmizela.*
L: Line-FSg disappeared-FSg3.

2. Subject – predicative adjective agreement.

(104) En: *Command is accessible.*
Bg: *Командата е достъпна.*
L: command-FSg is-Sg3 accessible-FSg
Cz: *P•íkaz je dostupný.*
L: Command-ISgNom is-Sg3 accessible-NomISg.
Ru: *Команда доступна.*
L: Command-FSg is-accessible-FSg.

3. Agreement within the nominal group.

(105) En: *Enter the fifth external point.*
Bg: *Задайте петата външна точка.*
L: Enter-Pl2 fifth-FSg external-FSg point-FSg
Cz: *Zadejte pátý externí bod.*
L: Enter-Pl2 fifth-ISgAcc external-ISgAcc point-ISgAcc
Ru: *Введите пятую внешнюю точку.*
L: Enter-Pl2 fifth-FSgAcc external-FSgAcc point-FSgAcc

Agreement can be classified also from another perspective: syntactical or semantic. Although semantic agreement is also present in described languages, it is quite rare and not important for our domain, therefore in the following we will deal only with the syntactical one.

3.7.1.1 Morphological abbreviations

Throughout this chapter, in word-by-word translation to English we use following abbreviations of morphological categories. Categories present in the abbreviations are in the

following order: POS[26], Gender, Number, Case and Person. A category is omitted if it is not relevant (e.g. case for finite verb) or not interesting in the given context. The possible values for each category are the following (of course, not all are present in all languages):

- POS: Adj (adjective), PastPart (past participle), etc.
- Gender: M (masculine, in Czech masculine animate), I (masculine inanimate[27]), F (feminine), N (neuter)
- Number: Sg (singular), Pl (plural)
- Case[28]: Nom (nominative), Gen (genitive), Dat (dative), Acc (accusative), Voc (vocative), Loc (locale) and Ins (instrumental)
- Person: 1, 2, 3

Therefore, for example: FSg means feminine singular, Sg3 means Singular third person and NSgNom means neuter, singular, nominative.

<table>
<tr><th></th><th>Czech</th><th>Russian</th><th>Bulgarian</th></tr>
<tr><td rowspan="2">Gender</td><td colspan="3">M, F, N</td></tr>
<tr><td>I</td><td colspan="2"></td></tr>
<tr><td>Number</td><td colspan="3">Sg, Pl</td></tr>
<tr><td rowspan="2">Case</td><td colspan="2">Nom, Gen, Dat, Acc, Loc, Ins</td><td rowspan="2">–</td></tr>
<tr><td>Voc</td><td></td></tr>
<tr><td>Person</td><td colspan="3">1,2,3</td></tr>
</table>

Table 3 – Comparison of morphological features relevant for agreement

3.7.2 Subject-Predicate Agreement

3.7.2.1 Analysis

In Czech, Bulgarian and Russian, a predicate[29] usually agrees with its nominative subject in person, number and gender (if applicable).

(106) En: *Command was accessible.*

[26] Sometimes more detailed than classical divison to 9 or 10 POS categories, e.g. PastPart (past participle). This category is also omitted if it is the same for the English word.

[27] Present only in Czech

[28] Present only in Czech and Russian

[29] By that we mean finite verb for simple verbal forms and all parts of compound verbal forms (See 3.7.2.1.1 for more details)

Bg: *Командата беше достъпна.*
L: Command-FSg was-Sg3 accessible-FSg
Cz: *P•íkaz byl*[30] *dostupný.*
L: Command-ISg was-ISg3 accessible-ISg.

(107) En: *Command was accessible.*
Bg: *Командата е достъпна.*
L: Command-FSg is-Sg3 accessible-FSg
Cz: *P•íkaz je dostupný.*
L: Command-ISg is-Sg3 accessible-ISg.

(108) En: *The system enables you to create a multiline style …*
Bg: *Системата позволява да създадете...*
L: System-FSg enable-Sg3 create-inf.
Cz: *Systém umož•uje vytvá•et styly multi•ár ...*
L: System-ISg enables-3Sg to-create styles of-multilines ...

This holds even if this subject is realized by a zero pronoun (so called pro-drop)[31].

(109) En: *Enter the distance between …*
Bg: *Задайте разстоянието между...*
L: Enter-Pl2 distance between…
Cz: *Zadejte vzdálenost mezi ...*
L: Enter-Pl2 distance between ....
Ru: *Введите расстояние между …*
L: Enter-2Pl distance between ....

(110) En: *Enter the distance between …*
Bg: *Вие задайте разстоянието между...*
L: You-Pl2 enter-Pl2 distance between…
Cz: *Vy zadejte vzdálenost mezi ...*
L: You enter-2Pl distance between ....

If the subject is in a case different from nominative[32] (e.g., in genitive)

(111) En: *Five points disappeared.*
Cz: *P•t bod• zmizelo.*
L: Five points-IPlGen disapeared-NSg3.

or the category of case is inappropriate for the subject (infinitival or sentential subjects)[33].

---

[30] It is in fact past participle. See 3.7.2.1.1 for more details

[31] In Czech and Bulgarian (in Russian in imperative), if the subject is not stressed it is often realized as zero pronoun (or, looking from a different perspective, the personal pronoun is omitted on the surface level). It is true in both indicative and imperative. If the pronominal subject is to be stressed, the personal pronouns must be explicitly expressed.

[32] This is present only in Czech and Russian

(112) En: *To open a drawing is simple.*
Cz: *Otev•ít kresbu je jednoduché.*
L: To-open drawing-FsgAcc is simple.

or if the verb has no subject at all (e.g. meteorological verbs or certain feelings verbs)[34]

(113) En: *It rains.*
Bg: *Вали.*
L: Rains-Sg3
Cz: *Prší.*
L: Rains-Sg3

(114) En: *I am cold.*
Bg: *Студено ми е.*
L: Cold I-Dat is-Sg3
Cz: *Je mi zima.*
L: Is-3Sg I-Dat cold

(115) En: *The button will be clicked*[35].
Cz: *Klepne se na tla•ítko.*
L: Click-Sg3 refl on button.

then the verb is assigned the default category of gender, number and person, which is neuter, singular and 3rd person.

Number of the predicate is determined by grammatical number of the subject, no matter if it denotes single object or set of objects.

(116) En: *The scissors disappeared.*
Cz: *Nužky zmizely.*
L: Scissors-IPlNom disappeared-IPl3.

3.7.2.1.1 Compound verbal forms

Compound verbal forms consist of finite forms of auxiliary verb and nonfinite forms (infinitive, participle) of the meaningful verb. For example, in Czech there are following compound verbal forms:

future tense	aux + infinitive	já budu volat
past tense	aux + past participle[36]	já jsem volal
present conditional	aux + past participle	já bych volal

[33] Currently not present in our domain.

[34] Not present in our domain.

[35] "na tlačítko" is adjunct in Czech and "klepnout" is intransitive verb, therefore when transformed into reflexive passive, there is no subject.

[36] In Czech, the auxiliary verb is not present in the third person

past conditional	present cond. of aux[37] + past participle	já bych byl volal
passive	aux + passive participle	já jsem volán

For detailed description of compound verbal forms (see Chapter 4. Mood and modality). All these words (except infinitive) have to agree with the subject in the same way as finite verb does[38]. The only difference is the set of morphological categories the word accepts:

Language	Verbal form	Gender	Number	Person
Bulgarian, Czech, Russian	finite	–	+	+
	participles	+[39]	+	–
Bulgarian	infinitive (da construction)	+[40]	+	–

(117) En: *You can save the line.*
Bg: *Вие можете да запазите линията.*
L: you-Pl2 can-Pl2 save-Pl2DaConstr line

3.7.2.1.2 Coordinated subject

In Czech and Russian, agreement with coordinated subject is rather complicated. For our domain, we can simplify the problem by assuming that number of predicate with coordinated subject is always plural and that person has to be uniform across the nominal group. For more detailed description of this problem for Czech see [Bémová 1995]

For Czech, gender of predicate will be minimal gender of participants of coordination, computed under following order[41]: m < i < f < n for Czech. This covers also trivial case

[37] That means: be + past part. of be

[38] For Bulgarian, it seems to be more natural to say that only finite agree with subject and other parts (infinitive, participle) agree with the finite

[39] Russian and Bulgarian do not distinguish gender of past participles in plural.

[40] Simple da construction do not distinguish gender.

[41] E.g.

Subject	Finite verb	Why
m+m	m	m is the only thing to select
m+f	m	m < f
m+f+f	m	m < f
m+f+n	m	m < f & m < n
m+n	m	m < n
f+n	f	f < n
f+I	i	i < f

Plural verbal and adjectival forms for feminine (f) and masculine inanimate (i) are the same, therefore it does not matter if we consider i to be smaller than f or vice versa.

when the gender of all participants is the same. For Bulgarian and Russian, this is not important because gender is not distinguished in plural.

(118) En: *The line and the box were deleted.*

Cz:	*Úse•ka*	*a*	*polí•ko*	*byly*	*smazány.*
L:	*Line-FSg*	*and*	*field-NSg*	*were-FPl3*	*deleted-FPl*
Ru:	*Линия*	*и*	*окно*	*были*	*удалены.*
L:	*Line-FSg*	*and*	*field-NSg*	*were-Pl*	*deleted-Pl*

However in Czech, there is an exception: if all participants have neuter gender and at least one is in singular then the gender of the predicate is feminine[42]:

(119) En: *The button and the box were enabled.*

Cz:	*Tla•ítko*	*a*	*polí•ko*	*byly*	*povolené.*[43]
L:	*Button-NSg*	*and*	*field-NSg*	*were-FPl3*	*enabled-FPl*

For more detailed description of agreement in Czech see [Kopečný 1962]

3.7.2.2 Implementation

The main problem with implementation is that number and especially gender of subject are not known when it is possible to inflectify finite, and it is not possible to inflectify finite when they are known. Therefore, it is necessary to use agreement operator (in simplified notation =). To make the treatment consistent we will use the same mechanism also for person.[44]

Second problem is that we do not know how to implement linguistically plausible notion of default values for finite. Therefore, we will handle this case in the same way as normal agreement (determine the values in subject side systems and pass them by agreement operators to predicate). Of course, this does not work for sentences without subject, however such sentences are not in our domain.

---

[42] Just to make things looking more complicated (obě in the second clause has to be in neuter, therefore also the second verb has to be in neuter):

En: *The button and the box were enabled and both disappeared.*

Cz:	*Tla•ítko*	*a*	*polí•ko*	*nebyly*	*povolené*	*a*	*ob•* *zmizela.*
L:	*Button-NSg*	*and*	*field-NSg*	*not-were-FPl3*	*enabled-FPl*	*and*	*both-**NPl*** *disappeared-**NPl**.*

[43] It does not mean that the feminine and neuter plural forms of verbs are the same. The verb is really in feminine form. The sentence (incorrect) with verb in neuter plural would look like:

Cz:*	*Tla•ítko*	*a*	*polí•ko*	*nebyla*	*povolená.*
L:	*Button-NSg*	*and*	*field-NSg*	*not-were-NPl3*	*enabled-NPl*

[44] Even for person there are some cases when the person of predicate is different from semantically derived person of subject:

En: *Five of you came.*

Cz:	*P•t*	*vás*	*p•išlo.*
L:	*Five*	*you-PlGen2*	*came-NSg3*

Of course, each language uses only systems that it needs (Bulgarian omits systems dealing with case, Bulgarian and Russian omits inanimate gender, etc.)

3.7.2.2.1 Inflection of the Thing

The agreement systems are all heavily dependent on the inflectional properties of the noun or pronoun (even not inserted) realizing the subject. Therefore, we present the features for these properties first:

For case:

```
THING-CASE-<C>
      where <C> ∈ {Nom, Gen, Dat, Acc, Voc, Loc, Ins}
```

For gender:

```
THING-GENDER-<G>
      where <G> ∈ {M, I, F, N}
```

For number:

```
THING-NUMBER-<N>
      where <N> ∈ {Sg, Pl}
```

Not all of these properties are present in all languages and properties for one category need not to be in one system.

3.7.2.2.2 Subject side systems

These systems determine the categories of predicate depending on the categories of the subject. We can distinguish two cases – the predicate does (`SVAgreement`) or does not (`SVNoAgreement`) agree with its subject. It does when the subject is in nominative, it does not otherwise (genitive subject[45])

```
SVAGREEMENT(Thing-Case-Nom)
    [SVAgreement]

SVNoAgreement(Thing-Case-Gen)
    [SVNoAgreement]
```

Systems determining gender of the predicate (neuter is default):

```
SUBJ-AGR-GENDER-<G> (Thing-Gender-<G> & SVAgreement)
    [Subj-Agr-Gender-<G>]
    where <G> ∈ {M,I,F}

Subj-Agr-Gender-N (Thing-Gender-N or SVNoAgreement)
    [Subj-Agr-Gender-N]
```

Systems determining number of the predicate (singular is default):

```
SUBJ-AGR-NUMBER-SG (Thing-Number-Sg or SVNoAgreement)
    [Subj-Agr-Gender-Sg]
```

[45] There are no infinitives or clauses in subject in our domain. However, in the future, appropriate feature is just simply added into `SVNoAgreement` after `Thing-Case-Gen`

```
Subj-Agr-Number-Pl (Thing-Number-Pl & SVAgreement)
    [Subj-Agr-Number-Pl]
```

Systems determining person of the predicate (3rd person is default):

```
SUBJ-AGR-PERSON-<P> (Pronoun-Person-<P> & SVAgreement)
    [Subj-Agr-Person-<P>]
    where <P> ∈ {1,2}

Subj-Agr-Person-3
    (Pronoun-Person-3 or nominal-term-resolution or SVNoAgreement)
    [Subj-Agr-Person-3]
```

3.7.2.2.3 Information passing systems

These systems are used to pass information determined by subject systems to appropriate words of predicate.

System passing information to finite:

```
SUBJECT-FINITE-AGREEMENT (Finite-Inserted & Subject-Inserted)
    [Subject-Finite-Agreement]
          (Subject = Finite
                      (Subj-Agr-Number-Sg ~ :::Number-Sg-Form)
                      (Subj-Agr-Number-Pl ~ :::Number-Pl-Form)

                      (Subj-Agr-Person-1 ~ :::Person-1-Form)
                      (Subj-Agr-Person-2 ~ :::Person-2-Form)
                      (Subj-Agr-Person-3 ~ :::Person-3-Form))
```

This system ensures that when Subject side systems determine number and person of the predicate (i.e. enters feature `Subj-Agr-*-*`), finite is inflectified appropriately.

System passing information to past or passive participles:

```
SUBJECT-AUXSTEM-AGREEMENT
    ( (Past-Participle-Inserted | Participle-Passive) &
          Subject-Inserted)
    [Subject-AuxStem-Agreement]
          (Subject = AuxStem
                      (Subj-Agr-Number-Sg ~ :::Number-Sg-Form)
                      (Subj-Agr-Number-Pl ~ :::Number-Pl-Form)

                      (Subj-Agr-Gender-M ~ :::Gender-M-Form)
                      (Subj-Agr-Gender-I ~ :::Gender-I-Form)
                      (Subj-Agr-Gender-F ~ :::Gender-F-Form)
                      (Subj-Agr-Gender-N ~ :::Gender-N-Form))
```

This system ensures that when Subject side systems determine number and gender of the predicate, participle is inflectified appropriately.

Bulgarian resources use similar system to ensure agreement with infinitive (da construction), it connects it by agreement with finite:

```
AUXSTEM-INSERT (Modal | P-Future | Da-phase)
    [Auxstem-Inserted]
          + Auxstem
          + Da-particle
          ! Da-particle da
          ^ Da-particle ^ Auxstem
          (Finite = AuxStem
          (Person-First-Form  ~ :::Person-First-Form)
          (Person-Second-Form ~ :::Person-Second-Form)
          (Person-Third-Form  ~ :::Person-Third-Form)
          (Number-Sg-Form ~ :::Number-Sg-Form)
          (Number-Pl-Form ~ :::Number-Pl-Form))
```

3.7.2.2.4 Coordination

Even if we simplify the problem by assuming that number of predicate with coordinated subject is always plural and that person has to be uniform across the nominal group, it still remains to determine gender of the predicate and than inflectify it appropriately[46]. The former is possible by determining minimal value of gender by comparing two adjacent members of coordinated subject each time. However the latter seems to be impossible in current version of KPML – we need to pass information up (similar to gender and person for simple subject), but we need to pass it across more than one rank – therefore we cannot use agreement operator.

As a (very inelegant) solution we apply the feminine gender to the predicate by default:

1. In plural, forms of participles, etc. in feminine gender are the same as forms in masculine inanimate
2. Feminine and masculine inanimate forms are the most probable in texts of CAD/CAM domain
3. In our corpus, there are no coordinated subjects

3.7.2.2.5 Language differences

3.7.2.2.5.1 Czech

Previous implementation (except system `Auxstem-insert` for Bulgarian) describes Czech, because it has all morphological features present in other two languages.

3.7.2.2.5.2 Bulgarian

Most of the differences in Bulgarian are implications of the fact that Bulgarian does not have cases. In Bulgarian Subject is always agreed with the predicate, so some of Subject side systems described above (and in particular, fork for applying default no-agreement cases (`SVNoAgreement`)) are not needed. The predicate in Bulgarian can contain "da-construction" and the system dealing with Subject-"da-construction" agreement in person and number is presented above.

[46] This is necessary only for Czech, Bulgarian and Russian do not distinguish gender in plural.

3.7.2.2.5.3 Russian

Russian resources model Subject – predicate agreement as far as agreement in Nominal group. It is very similar to Czech language – both keep their inflectional character. In agreement Russian is the same as Czech (besides it does not have vocative and masculine inanimate gender). The implementation is similar to the Czech implementation. So we do not consider the technical details here. The resulting graph structure with grammar form for 2nd person plural indicative is shown in Figure 90. In Figure 91 we also show passive construction where the zero auxiliary verb with grammar characteristics is shown and agreement in gender with passive participle. The zero auxiliary is a very specific character of Russian influencing in particular the non pro-dropping feature of the language in difference to Bulgarian and Czech.

3.7.2.2.6 Example of generation

The Figure 89 depicts structure graph of the following sentence:

(120) En: *You enter the command.*
Cz: *Vy zadáte p•íkaz.*
L: You-Pl2 enter-2Pl command

From inflection features (in boxes), you can see that `Finite` has the same number and person as subject has.

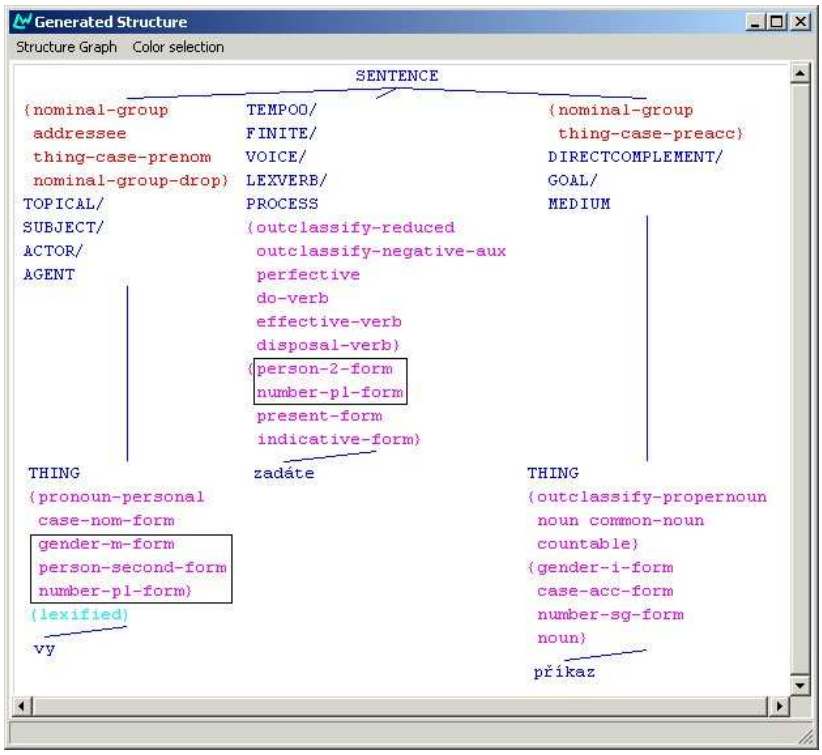


Figure 89: Subject – predicate agreement (Cz)

The Figure 90 presents the structure graph of the following sentence:

(121) En: *You draw an arc.*
Ru: *Вы нарисуете дугу*
L: You-Pl2 draw-2Pl arc

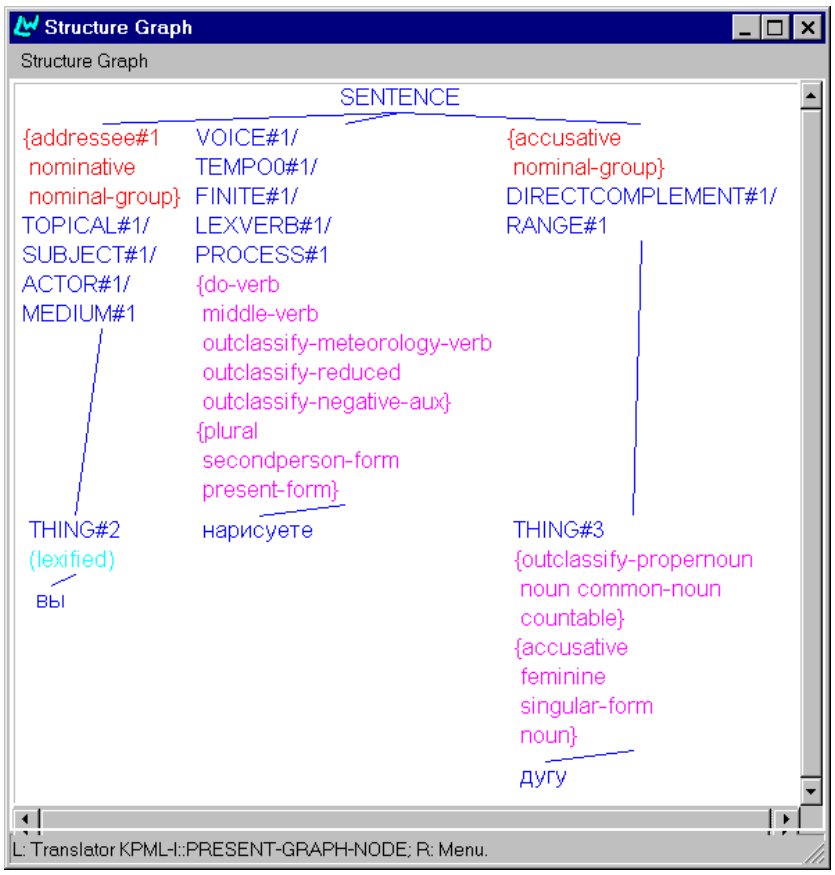


Figure 90: Subject – predicate agreement (Ru)

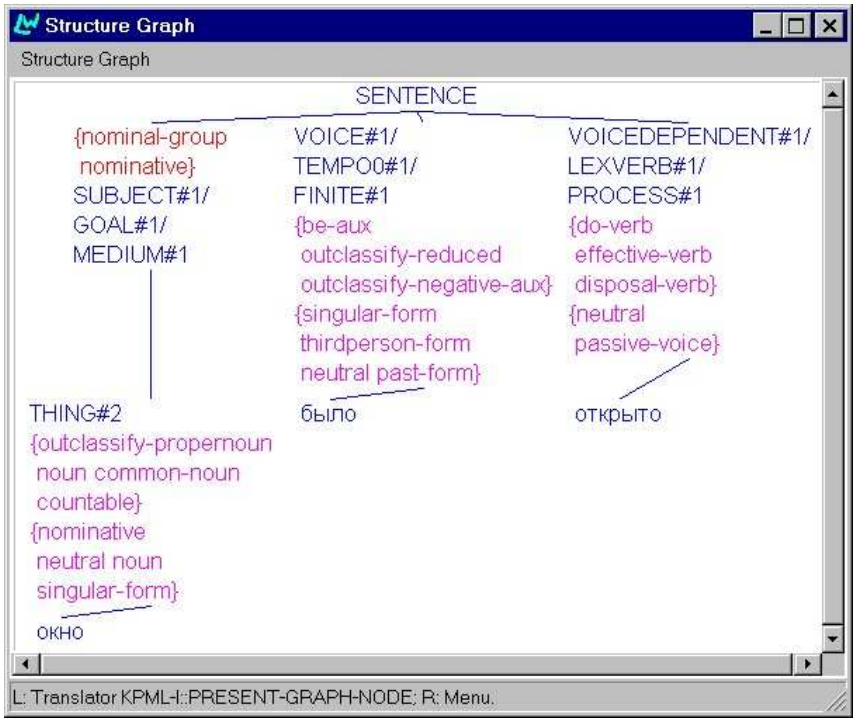


Figure 91: Passive construction with zero auxiliary verb agreement (Ru)

3.7.3 Agreement within the nominal group

3.7.3.1 Analysis

Within the nominal group, there is agreement between the head (pro)noun (Thing) and premodifiers, i.e., deictics and qualities, such as `Status`, `Provenance`, `Age`, `Size` and `Colour`.

3.7.3.2 Implementation

Each language uses only systems that it needs (Bulgarian omits systems dealing with case, Bulgarian and Russian omits inanimate gender, etc.)

For discussion about implementation of agreement with coordinated subject see 3.7.2.2.4

3.7.3.2.1 Status, provenance, etc.

3.7.3.2.1.1 Higher rank – Nominal group

In the same way as Nigel does, we distinguish five types of qualities: `Status`, `Provenance`, `Age`, `Size` and `Colour`. The systems accounting for the types of possible qualities take the following form:

```
<X>-MODIFICATION (Nominal)
     [<X>-Modified]
          + <X>
          <X>:Adjectival-group
          <X>:Congruent
    [Not-<X>-Modified]
    Chooser   <X>-Modification-Chooser
    where <X> ∈ {Status, Provenance, Age, Size, Colour}
```

As and example, we show the system for `Status` (`<X>` = `Status`):

```
STATUS-MODIFICATION (Nominal)
     [Status -Modified]
          + Status
          Status:Adjectival-group
          Status:Congruent
    [Not-Status-Modified]
    Chooser   Status-Modification-Chooser
```

Preselection of `<X>` as `Congruent` ensures that on the lower rank (adjectival group rank) it is known if the adjectival group should agree (be congruent) with its head.

Inflection of adjectival group is driven by preselections in systems described by following template:

```
<X>-<C>-<V>-PR (Thing-<C>-<V> & <X>-Modified)
    [<X>-<C>-<V>-Pr] <X>:Quality-<C>-<V>
    where
          <X> ∈ {Status, Age, Provenance, Size, Colour}
          <C> ∈ {Case, Gender, Number}
          <V> ∈ {Nom, Gen, Dat, Acc, Voc, Loc, Ins} for <C> = Case
          <V> ∈ {M, I, F, N} for <C> = Gender
          <V> ∈ {Sg, Pl} for <C> = Number
```

Therefore, there is 5*(7+4+2) = 65 systems. If we added more complicated cases of agreement (e.g. dual number), there would much more systems. Unfortunately, there is not

easily possible to generate all of these systems from some template similar to the one shown. Example of the system described by the template:

```
STATUS-NUMBER-PL-PR (Thing-Number-Pl & Status-Modified)
    [Status-Number-Pl-Pr] Status:Quality-Number-Pl
```

3.7.3.2.1.2 Lower rank – Adjectival group

```
CONGRUENT-FORK (Adjectival-Group)
    [Congruent]
    [Not-Congruent]
    :Chooser  Under-Status-Chooser
```

Inflection of `Quality` is realized by following three systems:

```
QUALITY-CASE (Congruent)
    [Quality-Case-Nom] Quality:::Case-Nom-Form
    [Quality-Case-Gen] Quality:::Case-Gen-Form
    [Quality-Case-Dat] Quality:::Case-Dat-Form
    [Quality-Case-Acc] Quality:::Case-Acc-Form
    [Quality-Case-Voc] Quality:::Case-Voc-Form
    [Quality-Case-Loc] Quality:::Case-Loc-Form
    [Quality-Case-Ins] Quality:::Case-Ins-Form

Quality-Gender (Congruent)
    [Quality-Gender-M] Quality:::Gender-M-Form
    [Quality-Gender-I] Quality:::Gender-I-Form
    [Quality-Gender-F] Quality:::Gender-F-Form
    [Quality-Gender-N] Quality:::Gender-N-Form

Quality-Number (Congruent)
    [Quality-Number-Sg] Quality:::Number-Sg-Form
    [Quality-Number-Pl] Quality:::Number-Pl-Form
```

3.7.3.2.2 Ordinal numerals

3.7.3.2.2.1 Higher rank – nominal group

Preselections on this rank drive inflections on lower rank.

```
Numerative-<C>-<V>-PR (Thing-<C>-<V> & Numerified)
    [Numerative-<C>-<V>-Pr] Numerative:Temperer-<C>-<V>
    where
          <C> ∈ {Case, Gender, Number }
          <V> ∈ {Nom, Gen, Dat, Acc, Voc, Loc, Ins} for <C> = Case
          <V> ∈ {M, I, F, N} for <C> = Gender
```

3.7.3.2.2.2 Lower rank

Following systems inflectify ordinal numeral depending on preselections form higher rank.

```
ORDINAL-CASE (Congruent)
    [Ordinal-Case-Nom] Ordinal:::Case-Nom-Form
    [Ordinal-Case-Gen] Ordinal:::Case-Gen-Form
    [Ordinal-Case-Dat] Ordinal:::Case-Dat-Form
    [Ordinal-Case-Acc] Ordinal:::Case-Acc-Form
    [Ordinal-Case-Voc] Ordinal:::Case-Voc-Form
    [Ordinal-Case-Loc] Ordinal:::Case-Loc-Form
    [Ordinal-Case-Ins] Ordinal:::Case-Ins-Form
```

```
Ordinal-Gender (Congruent)
    [Ordinal-Gender-M] Ordinal:::Gender-M-Form
    [Ordinal-Gender-I] Ordinal:::Gender-I-Form
    [Ordinal-Gender-F] Ordinal:::Gender-F-Form
    [Ordinal-Gender-N] Ordinal:::Gender-N-Form

Ordinal-Number (Congruent)
    [Ordinal-Number-Sg] Ordinal:::Number-Sg-Form
    [Ordinal-Number-Pl] Ordinal:::Number-Pl-Form
```

3.7.3.2.3 Cardinal numerals

3.7.3.2.3.1 Higher rank – nominal group

Preselections on this rank drive inflections on lower rank.

```
NUMERATIVE-<C>-<V>-PR (Thing-<C>-<V> & Numerified)
    [Numerative-<C>-<V>-Pr] Numerative:Temperer-<C>-<V>
    where
          <C> ∈ {Case, Gender }
          <V> ∈ {Nom, Gen, Dat, Acc, Voc, Loc, Ins} for <C> = Case
          <V> ∈ {M, I, F, N} for <C> = Gender
```

3.7.3.2.3.2 Lower rank

Following systems inflectify cardinal numeral depending on preselections form higher rank.

```
TEMPERER-CASE (Simplex-Cardinal)
    [Temperer-Case-Nom] Temperer:::Case-Nom-Form
    [Temperer-Case-Gen] Temperer:::Case-Gen-Form
    [Temperer-Case-Dat] Temperer:::Case-Dat-Form
    [Temperer-Case-Acc] Temperer:::Case-Acc-Form
    [Temperer-Case-Voc] Temperer:::Case-Voc-Form
    [Temperer-Case-Loc] Temperer:::Case-Loc-Form
    [Temperer-Case-Ins] Temperer:::Case-Ins-Form

Temperer-Gender (Simplex-Cardinal)
    [Temperer-Gender-M] Temperer:::Gender-M-Form
    [Temperer-Gender-I] Temperer:::Gender-I-Form
    [Temperer-Gender-F] Temperer:::Gender-F-Form
    [Temperer-Gender-N] Temperer:::Gender-N-Form

Temperer-Number (Simplex-Cardinal)
    [Temperer-Number-Sg] Temperer:::Number-Sg-Form
```

3.7.3.2.4 Deictic

Main difference between deictic and previous parts of sentence is, that deictic does not have its own rank – it is on the same level as `Thing` is.

```
DET-<C>-<V> (Thing-<C>-<V> & Explicit-Deictic)
    [Det-<C>-<V>-Pr] Deictic:::<C> -<V>-Form>
    where
          <C> ∈ {Case, Gender, Number }
          <V> ∈ {Nom, Gen, Dat, Acc, Voc, Loc, Ins} for <C> = Case
          <V> ∈ {M, I, F, N} for <C> = Gender
          <V> ∈ {Sg, Pl} for <C> = Number
```

3.7.3.2.5 Language differences

3.7.3.2.5.1 Czech

Previous implementation describes Czech, because it has all morphological features present in other two languages.

3.7.3.2.5.2 Russian

In agreement within nominal group, Russian is the same as Czech (besides it does not have vocative and masculine inanimate gender). The implementation is similar to the Czech implementation. So we do not consider the technical details here.

3.7.3.2.5.3 Bulgarian

The same is true for Bulgarian (besides it does not have cases and masculine inanimate gender). Bulgarian also has different treatment of deictics

In Bulgarian the (nominal group's) Deictic is realized as function of the whole nominal group, so the scheme of preselections in their rank (NG) and inflections on the lower rank is kept here.

In Bulgarian when Deictic of nominal group is `SPECIFIC`, `DEMONSTRATIVE` and `NONSELECTIVE` (in Nigel terms), which is analogue to English Deictic "the", it is realized as a morphological marker by the morphological module. This marker (the Deictic) could be carried by different element of the nominal group (Numerative, Quality, Thing). When the Thing is inflectified the following system is used:

```
NOMINATIVE-NONSELECTIVE-NOUN
    (Nonselective & Nominative &
          Not-Status-Modified & Not-Colour-Modified & Not-Age-Modified
&
          Not-Size-Modified & No-Post-Deictic )
    [Full-Article]  Thing:::Definite-Word-FA
```

When the Deictic is demonstrated by the element of the adjectival group we use the system shown bellow to transform the Deictic function to preselection of adjectival group:

```
ADJECTIVAL-GR-DETERMINATION-FA
    (Nominative & Nonselective &
          (Status-Modified | Colour-Modified Age-Modified |
          Size-Modified | Post-Deictic-Modified))
    [Full-Article-AG]  AG-Deictic:FA-Determination
```

Further the characteristic `FA-determination` (full-article-determination) is associated with a particular element of the adjectival group by the realization statement of the next system:

```
ADJECTIVAL-GR-ARTICLE-REALIZATION (Adjectival-Group)
    [FA-Determination]
          Quality:::Definite-Word-FA
          Numerative:::Definite-Word-FA
          Ordinal:::Definite-Word-FA
```

Same mechanism is used for NONSPECIFIC, NONSELECTIVE, SINGULAR Deictic, which is in Bulgarian a morphological marker corresponding to English Deictic "a(n)".

All other types of deictics in nominal group (specific and non-specific) have the feature `Explicit-Deictic` and for their agreement with the Thing element in gender and number are used systems of the type `DET-<C>-<V>`

3.7.3.2.6 Example of generation

The Figure 92 depicts structure graph of the following sentence:

(122) En: *Enter the fifth external point.*
Cz: *Zadejte pátý externí bod.*
L: Enter-Pl2 fifth-ISgAcc external-ISgAcc point-ISgAcc

From inflection features (in boxes), you can see, that `Ordinal` under `Ordinator` and `Quality` under `Status` have the same gender (masculine inanimate – `gender-i-form`), number (singular – `number-sg-form`) and case (accusative – `case-acc-form`) as `Thing`. This is ensured by preselections marked by ellipses.

3.7.4 Agreement of subject and predicative adjective

3.7.4.1 Analysis

From some point of view agreement with predicative adjective is mixture of subject-verb agreement and agreement within nominal group. Predicative adjective agrees with subject in gender, number and case[47] (only nominative or genitive are possible).

(123) En: *The command is accessible.*
Bg: *Командата е достъпна.*
L: Command-FSg is-Sg3 accessible-FSg
Cz: *P•íkaz je dostupný.*
L: Command-ISgNom is-Sg3 accessible-NomISg.

(124) En: *Lines are visible.*
Bg: *Линиите са видими.*
L: Line-Pl are-Pl3 visible-Pl
Cz: *Úse•ky jsou viditelné.*
L: Line-FPlNom are-Pl3 visible-FPlNom.

(125) En: *Five lines are visible.*
Cz: *P•t úse•ek je viditelných.*
L: Five-Nom line-FPlGen[48] *is-Sg3 visible-FPlGen.*

[47] In Czech and Russian, not in Bulgarian.

[48] Genitive instead of nominative is required by the numerals higher than four. See [Chapter ##10. Quantification]

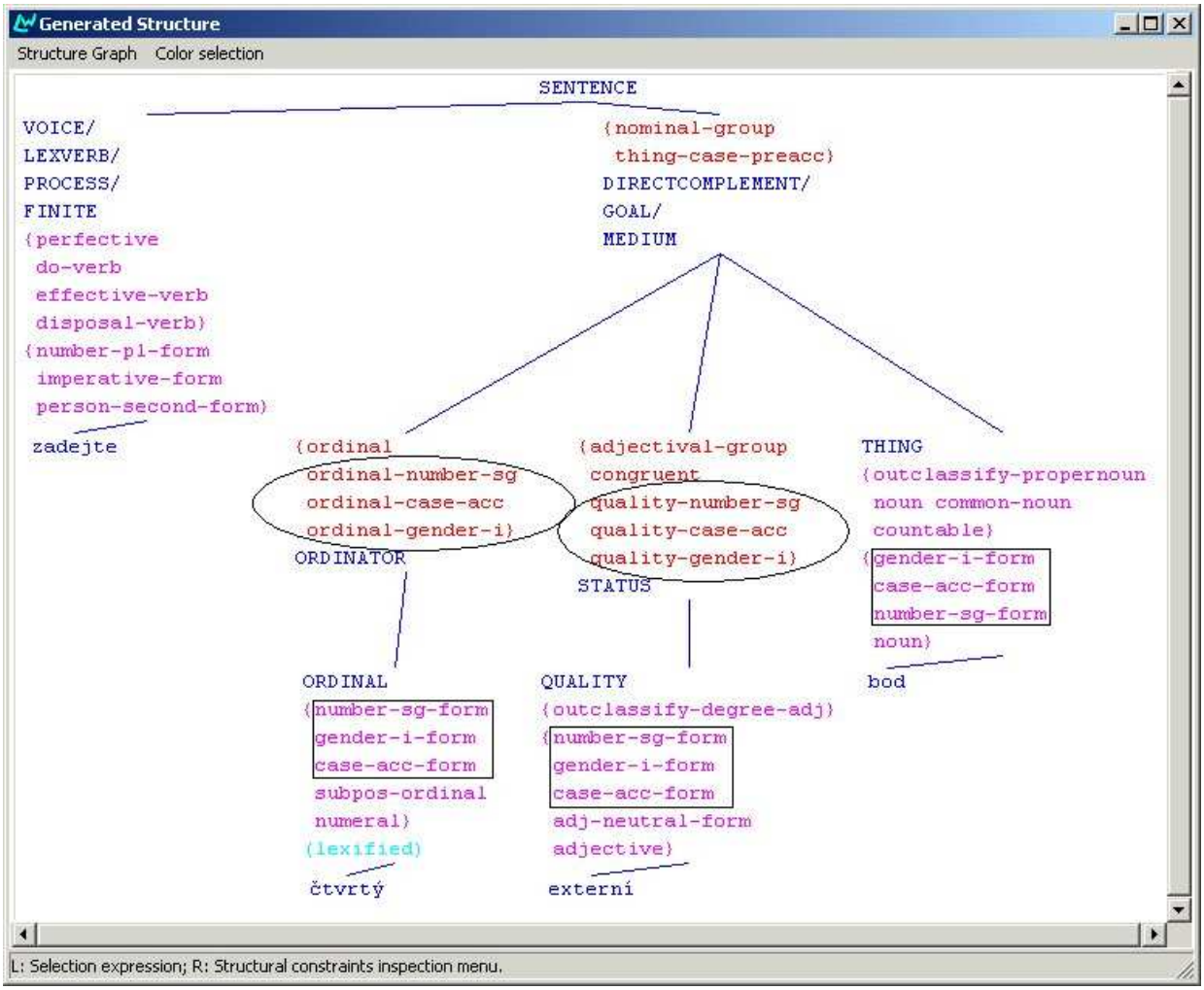


Figure 92 - Agreement within nominal group

3.7.4.2 Implementation

Predicative adjective in Nigel is realized as `Quality` under `Attribute` (See Figure 92).

3.7.4.2.1 Preselections on the clause rank

Preselection of `Attribute` is performed by the following system. Agreement operator ensures that `Attribute` is preselected[49] for gender number and case if appropriate feature is entered in `Subject`.

```
SUBJECT-PREDICATIVEADJ-AGREEMENT (Ascriptive & Subject-Inserted)
    [Subject-Predicativeadj-Agreement]
          (Subject ~ Attribute
                (Thing-Gender-M = Quality-Gender-M)
                (Thing-Gender-I = Quality-Gender-I)
                (Thing-Gender-F = Quality-Gender-F)
                (Thing-Gender-N = Quality-Gender-N)

                (Thing-Number-Sg = Quality-Number-Sg)
                (Thing-Number-Pl = Quality-Number-Pl)
```

[49] Keyword preselection is omitted in agreement operator: (A = B) in fact means (A = (:B))

```
                (Thing-Number-Nom = Quality-Number-Nom)
                (Thing-Number-Gen = Quality-Number-Gen))
```

Current version of KPML (3.0) does not show preselections done by agreement operator in the structure graph (C.f. Figure 93).

3.7.4.2.2 Adjective group rank

The inflection of Quality inserted under Attribute is done by systems QUALITY-CASE, QUALITY-GENDER, QUALITY-NUMBER, described in chapter 3.7.3.2.1.2 above.

3.7.4.2.3 Example of generation

The Figure 93 depicts structure graph of the following sentence:

(126) En: *Commands are accessible.*
Cz: *P•íkazy jsou dostupné.*
L: Commands are-Pl3 accessible-IPl.

You can see, that `Quality` under `Attribute` has the same gender and number as subject (masculine inanimate – `gender-i-form` and plural – `number-pl-form`). Preselections done by agreement operator are not displayed in Structure graph.

3.7.4.2.4 Language differences

3.7.4.2.4.1 Czech

Previous implementation describes Czech, because it has all morphological features present in other two languages.

3.7.4.2.4.2 Bulgarian

Bulgarian omits in system SUBJECT-PREDICATIVEADJ-AGREEMENT lines responsible for agreement in inanimate gender and case:

```
SUBJECT-PREDICATIVEADJ-AGREEMENT (Ascriptive & Subject-Inserted)
    [Subject-Predicativeadj-Agreement]
          (Subject ~ Attribute
                (Thing-Gender-M = Quality-Gender-M)
                (Thing-Gender-F = Quality-Gender-F)
                (Thing-Gender-N = Quality-Gender-N)

                (Thing-Number-Sg = Quality-Number-Sg)
                (Thing-Number-Pl = Quality-Number-Pl))
```

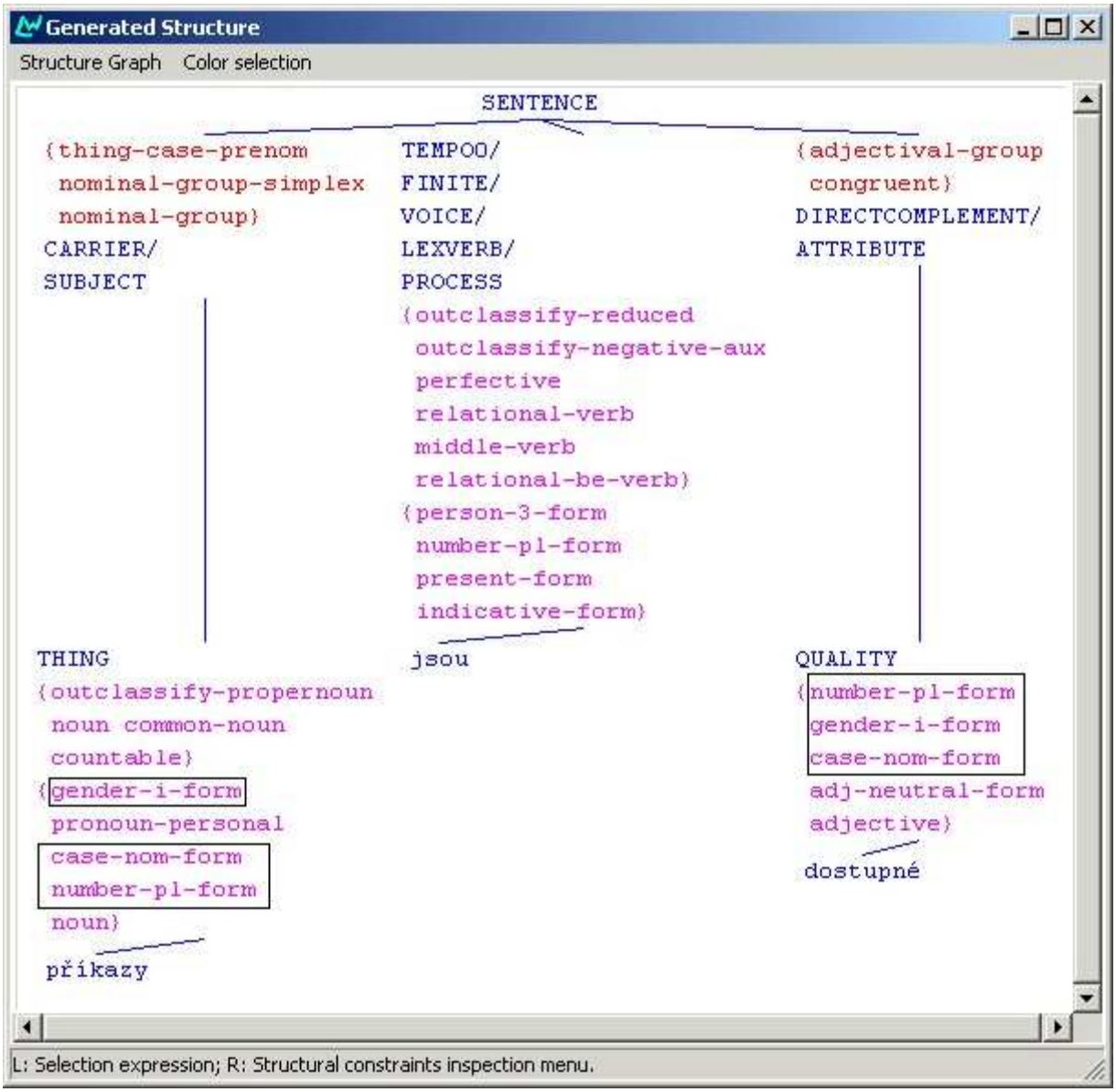


Figure 93: Agreement of subject and predicative adjective

3.7.5 Conclusion

Agreement in Bulgarian, Czech and Russian is more complicated than similar phenomenon in English. It is driven mostly by syntactical properties of agreeing units. Implementation described above covers all agreement necessary for final corpora of Agile; moreover it implements many cases not covered by the corpora. Modularity and overall design of all systems allows easy enhancement for more special cases in the future. The only exception is agreement with coordinated subject that is impossible to be fully covered in current version of KPML.

3.8 Quantification

In this section we describe two phenomena – the realization of quantitative construction in Slavic languages and quantity selection construction. We consider cardinal numerals presenting the quantity of objects realized by a noun, which can be modeled straightforwardly. The quantity selection construction (e.g., *one of the following methods*) presents a more complicated case, especially for Czech and Russian. In difference to English and Bulgarian, Czech and Russian have more complex government patterns and the cardinal acts as head of the construction.

3.8.1 Cardinal numerals

3.8.1.1 Analysis

In Czech, when the numeral is above 5, the counted object is not in nominative or accusative but in genitive case.

In Russian the realisation of quantity has the following specifics. The cardinal *odin* (one) is realised as an adjective that agrees with the Thing in gender, number and case. Others are realised by cardinals having the Thing in genitive case, when the whole nominal group is in nominative or accusative. We neglect the animate / inanimate opposition, which presents a yet more complex case, and consider only inanimate objects as presented in numeral constructions. We do not have different quantities of animate persons in the register. When the nominal group is in nominative or in accusative and we have the cardinals *два, три, четыре* (*two, three, four*) (not considering *оба (both)* and *полтора (half)*) the Thing is in singular and the cardinals above 4 have the Thing in plural. In indirect cases – genitive, dative, instrumental, locative - the cardinals are in plural and agree with the Thing in case. The cardinal 'two' agrees with the noun in gender as well.

(127) *Two points disappeared*

Cz: *Dva body zmizely.*
Two-INom points-IPlNom disappeared-IPl3.

Ru: ••• ••••• •••••••
Two-FNom points-FPlNom disappeared-Pl3.

Bg: ••• ••••• •••••••••
Two points-Pl disappeared-Pl3

(128) *Five points disappeared*

Cz: *P•t bod• zmizelo.*
Five-INom points-IPlGen disappeared-NSg3.

Ru: •••• ••••• •••••••.
Five-Nom points-FPlGen disappeared-NSg3.

3.8.1.2 Implementation for Czech

Initial system for determining case of a nominal group is the following:

```
THING-CASE (Nominal-Group)
   [Thing-Case-PreNom]
   [Thing-Case-PreGen]
   [Thing-Case-PreDat]
   [Thing-Case-PreAcc]
   [Thing-Case-PreVoc]
   [Thing-Case-PreLoc]
   [Thing-Case-PreIns]
   [GENITIVE)
   CHOOSER: Thing-Case-Chooser
```

Idea behind features `Thing-Case-PreNom/Gen/..`: It will be nominative/genitive/…, if nothing special happens (Now changes can happen only for Nom and Acc). And also, it is not the only way, how realize this case can (Now only for genitive).

Currently we need this special behaviour only for nominative, accusative and genitive, therefore the system could be as following:

```
THING-CASE (Nominal-Group)
   [Thing-Case-PreNom]
   [Thing-Case-PreGen]
   [Thing-Case-Dat]
   [Thing-Case-PreAcc]
   [Thing-Case-Voc]
   [Thing-Case-Loc]
   [Thing-Case-Ins]
   [GENITIVE)
   CHOOSER: Thing-Case-Chooser
```

However, the previous treatment is more consistent. The case is driven by preselection from higher ranks. Therefore, it is necessary to use preselections with `Thing-Case-PreNom` instead `Thing-Case-Nom`, etc. as in previous versions of Nigel.

Nominative and accusative can be changed to genitive if certain conditions occurs, to simplify input conditions in the rest of systems we merge features `Thing-Case-PreNom` and `Thing-Case-PreAcc` into a feature `NumerativableCases`:

```
NumerativableCases (Or Thing-Case-PreNOM Thing-Case-PreAcc)
   [NumerativableCases]
   :REGION   NounInflection
```

Is the object numerified (This system differs from the original Nigel system `Numeration` in the input condition.):

```
NUMERATIONX (NumerativableCases)
   [NumerifiedX]
   [NonNumerifiedX]
   CHOOSER: NumerationX-Chooser
```

If the object is numerified, the following system will determine if the number is higher that 4:

```
AboveFour (NumerifiedX)
   [More-Than-Four]
   [Not-More-Than-Four]
   CHOOSER:   More-Than-Four-Chooser
```

Following two systems partition Numerativable cases into two path:

1) `Numerative` – numerative case (genitive) will be used
2) `NonNumerative` – proposed case will be normally used

```
Numerative (NumerativableCases & More-Than-Four)
   [Numerative]


NonNumerative (NumerativableCases &
      (NonNumerifiedX | Not-More-Than-Four))
   [NonNumerative]
```

Following systems determine really applied cases:

```
Thing-Case-Nom (Thing-Case-PreNOM & NonNumerative)
   [Thing-Case-Nom Thing:::Case-Nom-Form


Thing-Case-Gen   (Thing-Case-PreGen | Numerative)
   [Thing-Case-Gen] Thing:::Case-Gen-Form


Thing-Case-Acc (Thing-Case-PreAcc & NonNumerative)
   [Thing-Case-Acc] Thing:::Case-Acc-Form
```

Rest of the systems are simple:

```
Thing-Case-<C> (Thing-Case-Pre<C>)
   [Thing-Case-<C>] Thing:::Case-<C>-Form
where <C> ∈ {Dat, Voc, Loc, Ins}
```

3.8.1.3 Implementation for Russian

Here are main systems appropriate for agreement in nominal group between thing and ordinator. Code addition is highlighted using bold font. All the changes limit the systems input according the algorithm described earlier.

```
(GATE
    :NAME PLURAL-TO-THING
    :INPUTS (AND PLURAL (OR NONNUMERIFIED MORE-THAN-FOUR
                        (and LESS-THAN-FIVE
       (or genitive dative INSTRUMENTAL PREPOSITIONAL))))
    :OUTPUTS ((1.0 PLURAL-TO-THING (INFLECTIFY THING PLURAL-FORM)))
     :REGION   COUNTNUMBER
     :METAFUNCTION   IDEATIONAL
)
(GATE
     :NAME   ACCUSATIVE-CASE
     :INPUTS   (AND ACCUSATIVE NOMINAL-GROUP-SIMPLEX
              (OR INDIVIDUAL-NAME SINGULAR NONPLURAL NONNUMERIFIED eq-one))
     :OUTPUTS   ((1.0 ACCUSATIVE-CASE-T
       (INFLECTIFY THING ACCUSATIVE)))
     :REGION   NOMINAL-PERSON
     :METAFUNCTION   INTERPERSONAL
)
(GATE
     :NAME   RGENITIVE-CASE
     :INPUTS   (OR (AND RGENITIVE NOMINAL-GROUP-SIMPLEX)
```

```
            (AND (OR ACCUSATIVE NOMINATIVE) (or MORE-THAN-FOUR LESS-THAN-
   FIVE)
                  NOMINAL-GROUP-SIMPLEX)
        )
    :OUTPUTS   ((1.0 GENITIVE-CASE-T
      (INFLECTIFY THING GENITIVE)))
    :REGION   NOMINAL-PERSON
    :METAFUNCTION   INTERPERSONAL
)
```

Description: implements all events of appearance THING in Nominal group in genitive case. The following two systems implements transmission of genitive case from the nominal group to the terminal ORDINAL node.

```
(GATE
    :NAME   RGENITIVE-CASE-ORDINATOR
    :INPUTS   (AND RGENITIVE ORDINATIVE)
    :OUTPUTS   ((1.0 RGENITIVE-CASE-O
      (PRESELECT ORDINATOR RGENITIVE)))
    :REGION   NOMINAL-PERSON
)

(GATE
    :NAME   GENITIVE-CASE-ORDINAL
    :INPUTS   (AND RGENITIVE ORDINAL)
    :OUTPUTS   ((1.0 GENITIVE-CASEO1
      (INFLECTIFY ORDINAL GENITIVE)))
    :REGION   NOMINAL-PERSON
)
```

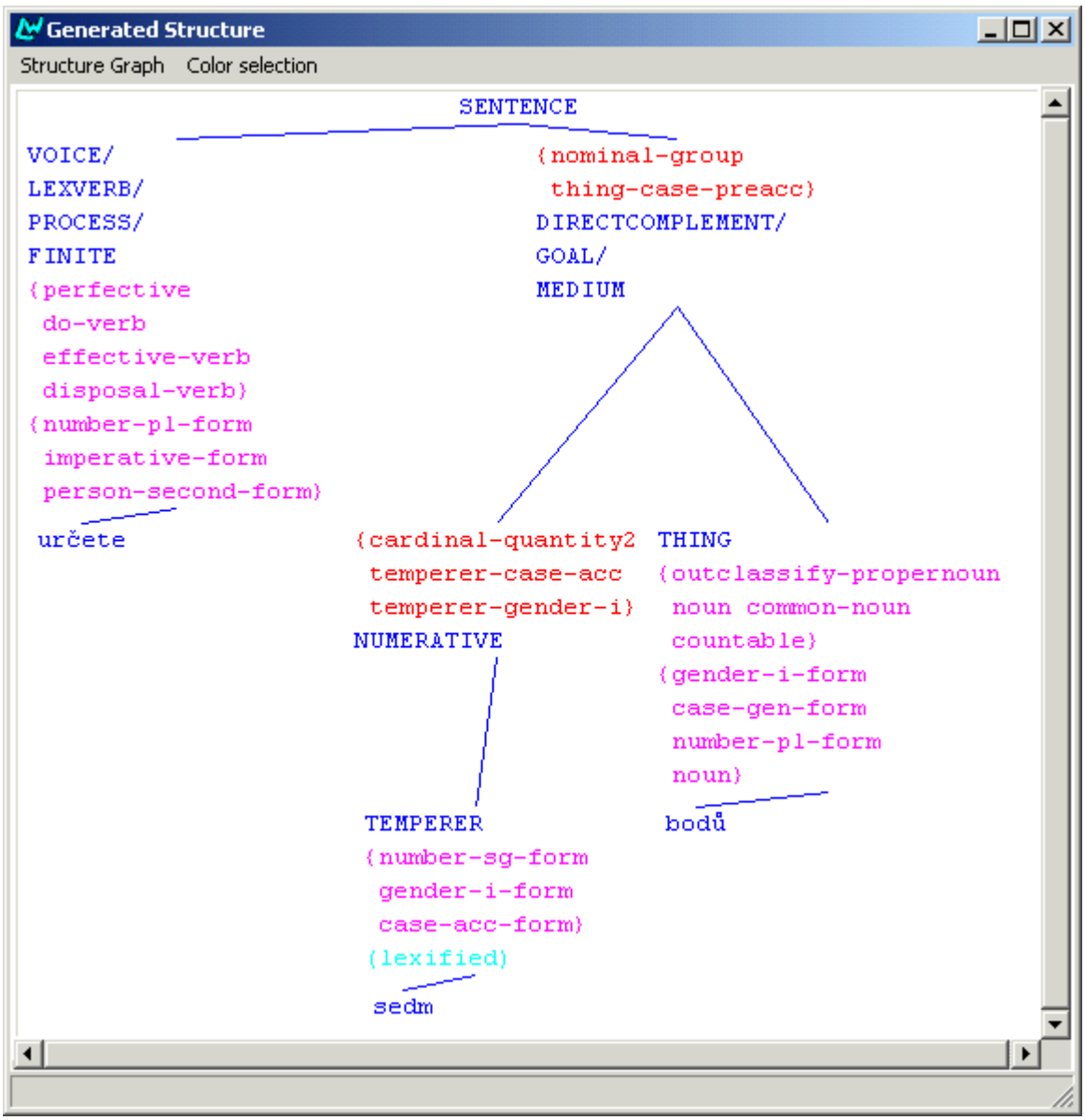


Figure 94: Accusative with numeral above 4

Cz: *Ur•ete sedm bod•.*
Specify seven-Acc points-Gen.

En: *Specify seven points.*

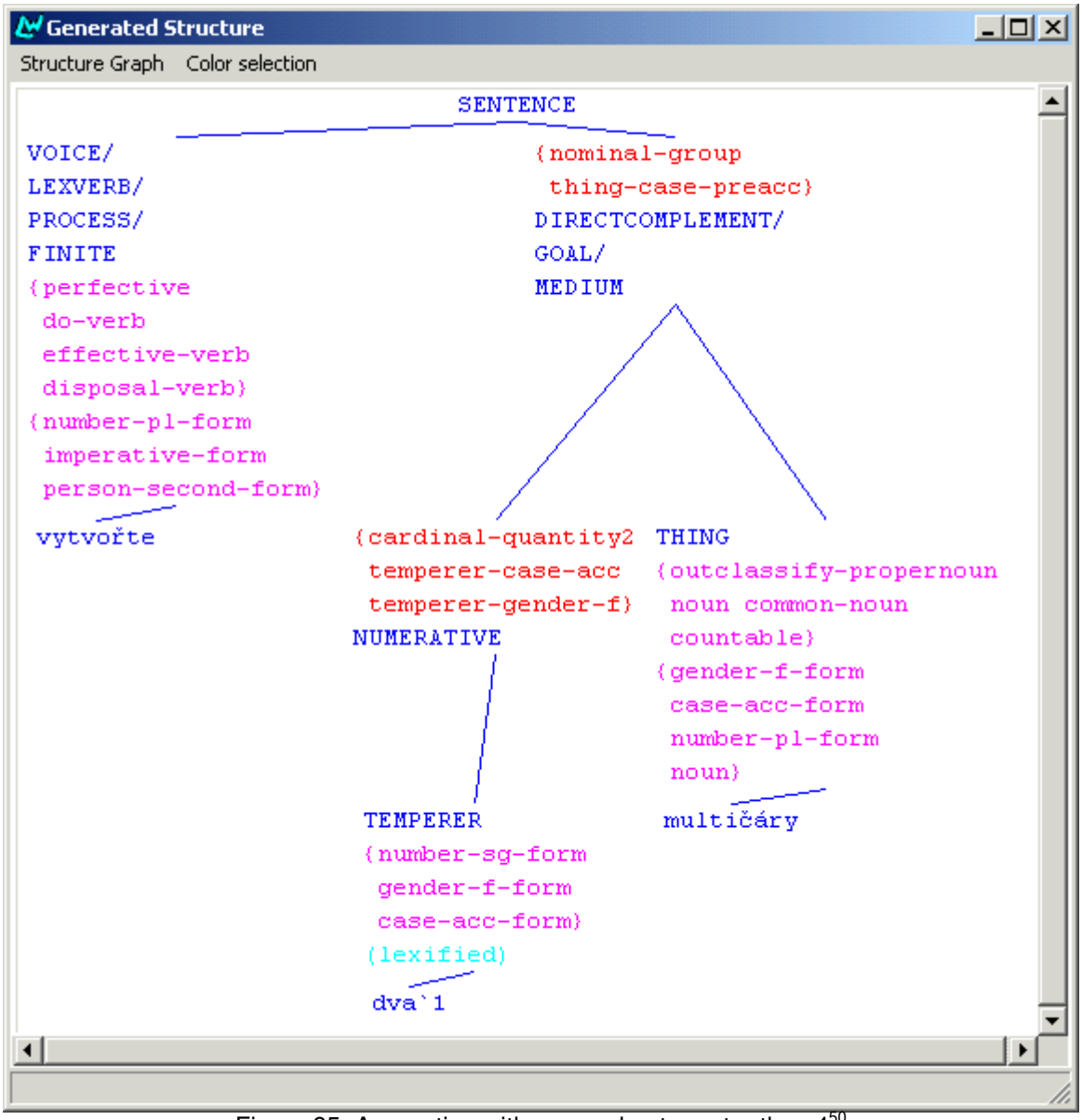


Figure 95: Accusative with numeral not greater than 4[50]

[50] The numeral dva (two) is not correctly inflectified by Czech morphology.

Cz: *Vytvo•te dv• multi•áry..*
Create two-Acc multiline-Acc.

En: *Create two multilines.*

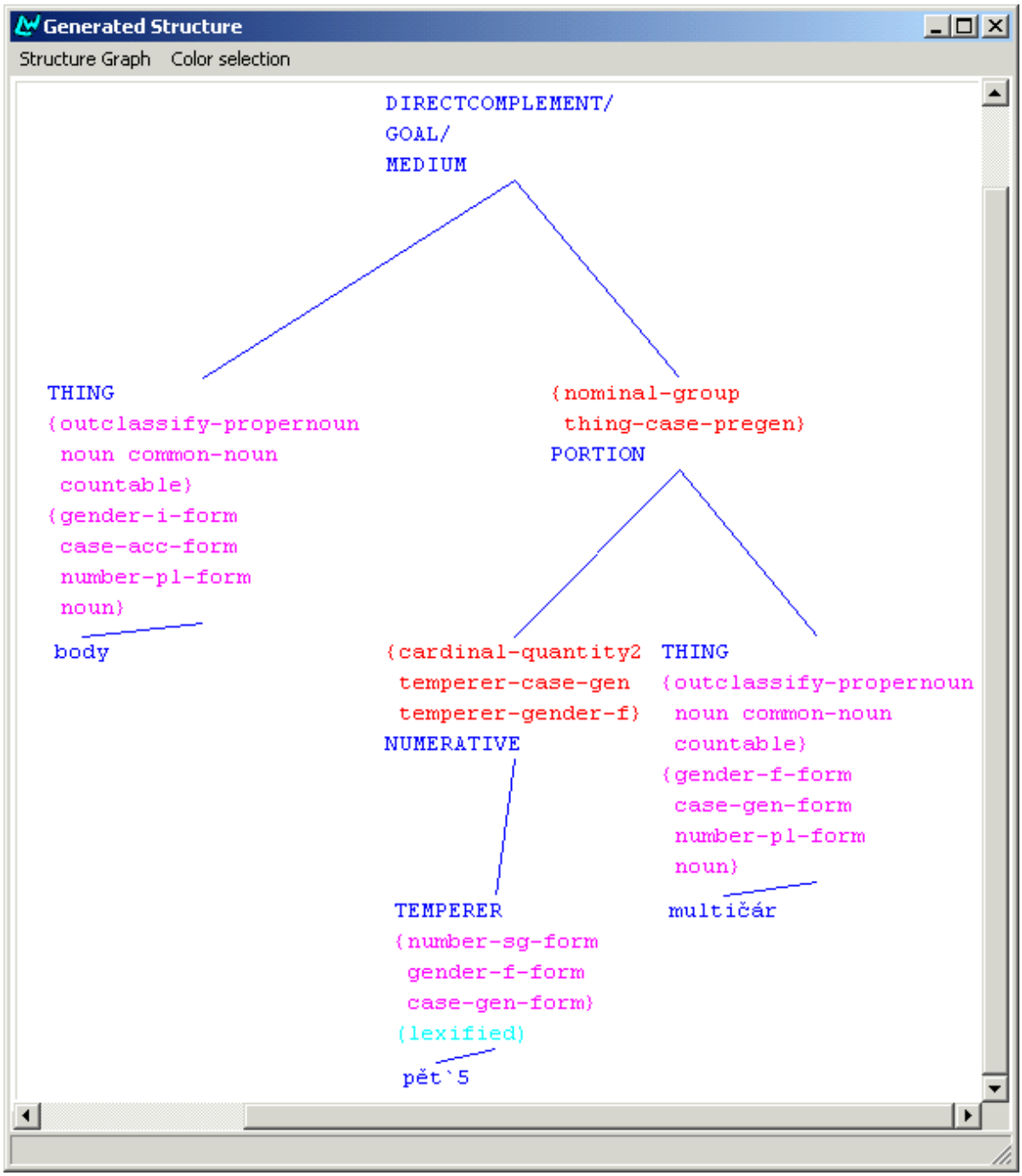


Figure 96: Fragment of a sentence with genitive with a numeral

Cz: .. *body p•ti multi•ár..*
.. *points-Acc five-Gen multilines-Gen*

En: .. *points of five multilines.*

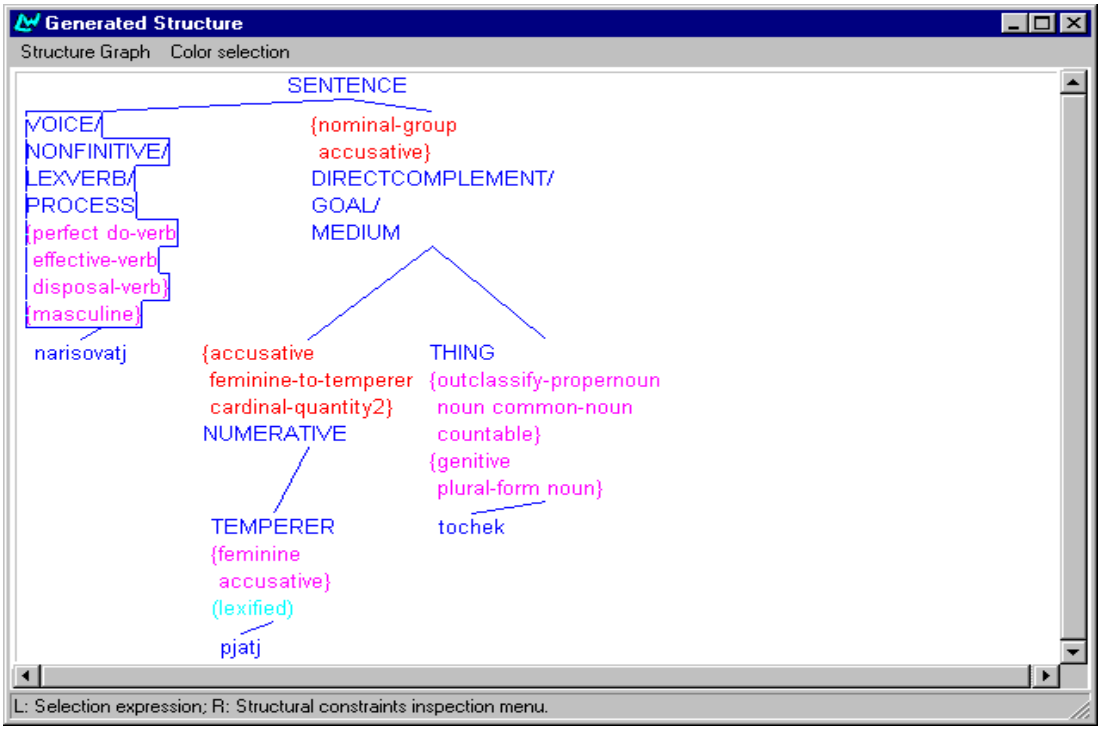


Figure 97: The numeral “5” having Acc of the all nominal group (Ru)

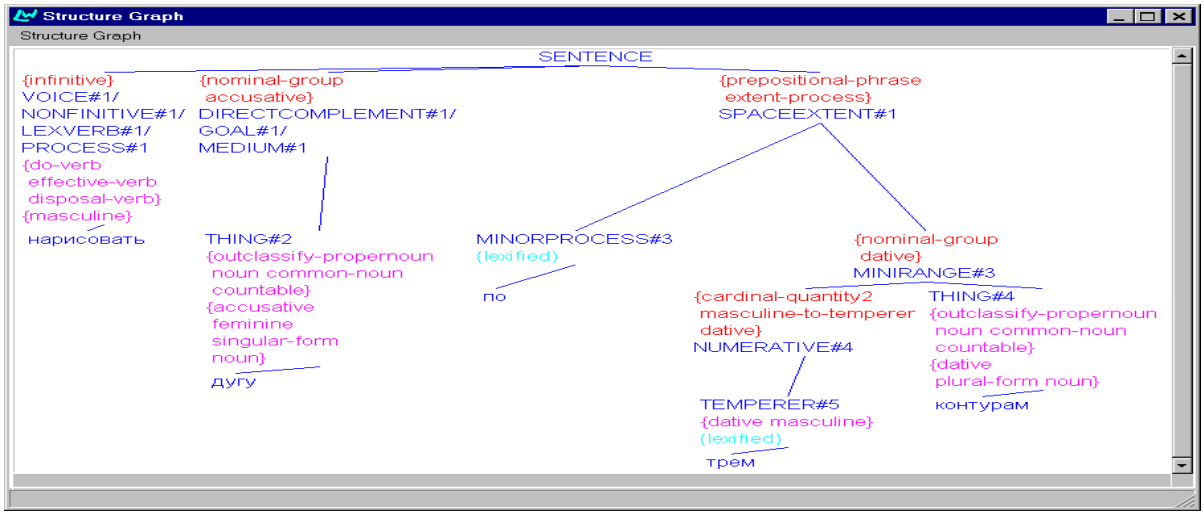


Figure 98: Cardinal agreement (Ru)

In Figure 98 an indirect case of the nominal group is presented – the noun is in Pl and in appropriate case (Dat), the cardinal agrees with it.

3.8.2 Quantity selection

3.8.2.1 Linguistic specifications

We describe a model for the selective quantitative construction. An example from our sample texts is given below:

(129) En: *Open the dialog box using one of following methods*

(a) Ru: *Откройте диалоговое окно одним из следующих* способов:
Open dialog-Adj box-N-Acc one-Ins of following-Adj methods-Gen

(b) Bg: *Отворете прозореца с един от тези методи.*
Open window-Det by one of these methods.

(c) Cz: *Otevřete dialogové okno jednou z následujících metod.*
Open dialogue-Adj box-Acc one-Ins from following-Adj methods-Gen

As can be seen from the example, comparing English with Russian for instance, is that there are two cases assigned in the Russian and Czech equivalent of the phrase "*one of the following methods*", instrumental case to *odnim* (*one*) and genitive case to *sleduyushich sposobov* (*following methods*). In English and Bulgarian, in contrast, there is no case. Therefore, the syntactic modelling of this kind of structure is quite straightforward. Figure 99 shows the structural representation adopted in Bulgarian from Nigel for modelling this kind of quantification phenomenon. As can be seen from the tree structure, the phrase structure organization is rather flat, with Quantifier, Qselector, Deictic and Thing being arranged at the same level in the tree, the Thing implicitly acting as the head of the construction. As we see from the gloss at the example (129) English and Bulgarian structures are identical.

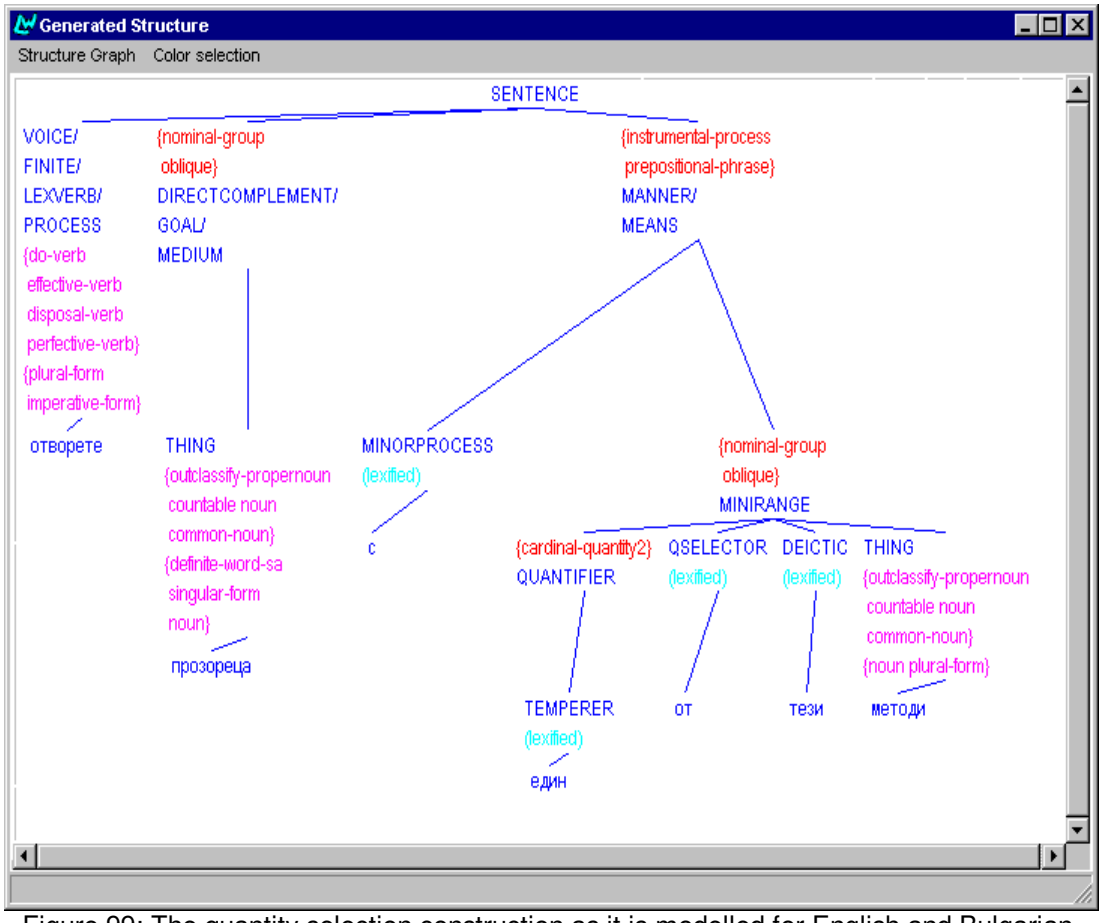


Figure 99: The quantity selection construction as it is modelled for English and Bulgarian.

This representation is not adequate for Czech and Russian. First, in Czech and Russian, the Quantifier (here: *odnim*) is a more likely candidate for headship , the rest of the phrase acting as Postmodifier. Second, within the Postmodifier (here*: iz sludiyushich sposobov*), the preposition *iz* case-governs its argument (here: *sposobov*). Thus, we deal with a more

complex structure here than in English and Bulgarian. This structure is displayed in Figure 100. It is not that we meet another type of conceptualisation of the phenomenon in Czech or Russian compared to English or Bulgarian. However, the realizations are cross-linguistically different.

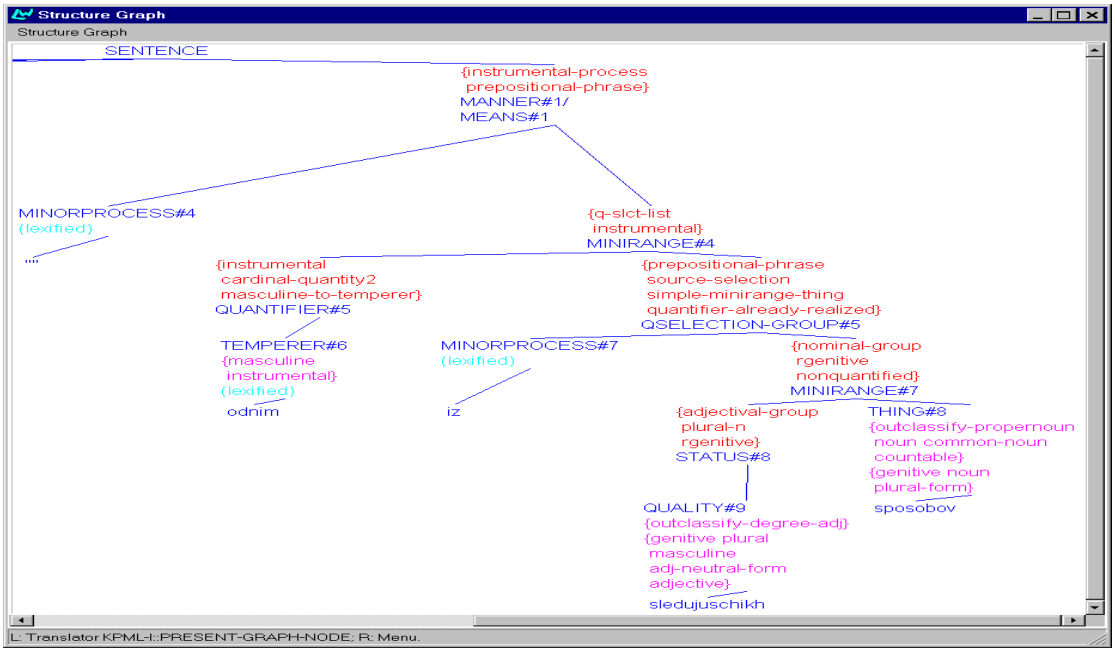


Figure 100: Structure of quantification construction realized for Czech and Russian grammar

3.8.2.2 Implementation

To be able to generate this kind of structure, we modified the existing NIGEL systems in various ways. In Figure 100 we show the graph structure relevant for the Russian and Czech construction realisation. To realize selective quantification construction the new phrase **quantity selection list** (**q-slct-list**) was entered. It was decided that it reflects the linguistic phenomena of the phrase in the more adequate way. Nominal group construction that covered the whole structure in the Nigel model has moved now to deeper level of the structure and corresponds to Thing and Status collocation.

In Figure 101 we show SPL for the quantity selective construction that realises the Russian sentence from the example above.

```
(|a11| / DM::OPEN-SCREEN-OBJECT :SPEECHACT IMPERATIVE
   :ACTEE (|a13| / OBJECT :LEX OKNO
          :PROPERTY-ASCRIPTION (J / QUALITY :LEX DIALOGOVYJ)))
   :GENERALIZED-MEANS (|a14| / DM::METHOD*
          :DETERMINER  THIS
          :QUANTITY-SELECTION-Q QUANTITY
          :QUANTITY-SELECTION-id 1 :NUMBER PLURAL
          :PROPERTY-ASCRIPTION (S / QUALITY :LEX SLEDUJUSCHIJ)))
```

Figure 101 SPL for the quantity selective construction (Ru)

The two SPL expressions :QUANTITY-SELECTION-Q QUANTITY & QUANTITY-SELECTION-id 1 are sufficient for the generation in Russian and Czech and :DETERMINER THIS is needed for English and Bulgarian and doesn't influence the generation process for Russian and Czech being irrelevant.

The implementation for Bulgarian follows the NIGEL implementation.

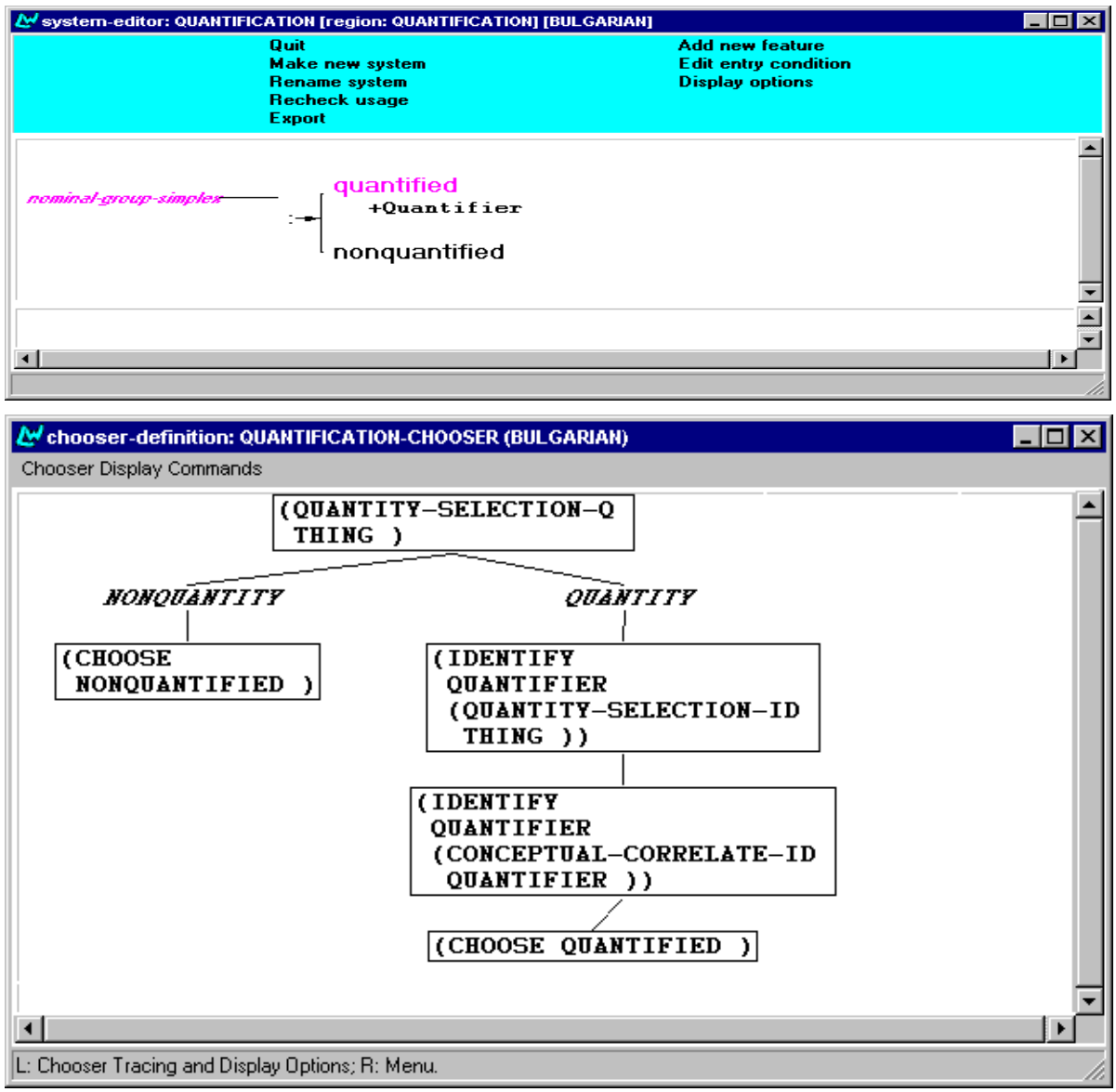


Figure 102: Quantification system and chooser (Bg)

Then the quantity selection construction is finished by the Quantity-type and Quantity-selection systems that are shown in Figure 103

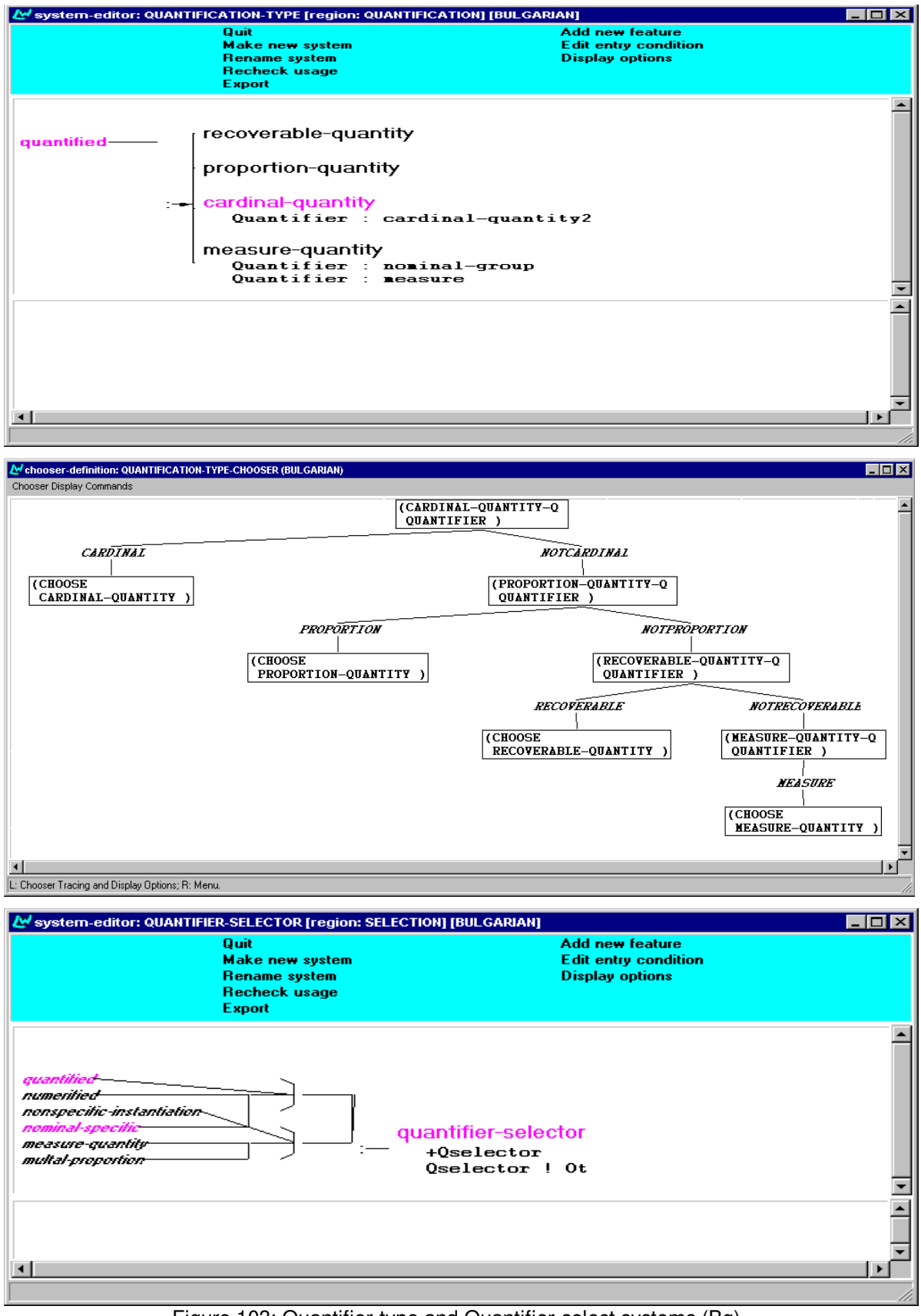


Figure 103: Quantifier-type and Quantifier-select systems (Bg)

For Russian and Czech some reorganisation of the generation process is needed. The new group starts with setting **q-slct-list** preselect to Minirange grammatical function. For this purpose system MINIRANGE-TYPE (region PPOTHER) was changed (Figure 104:).

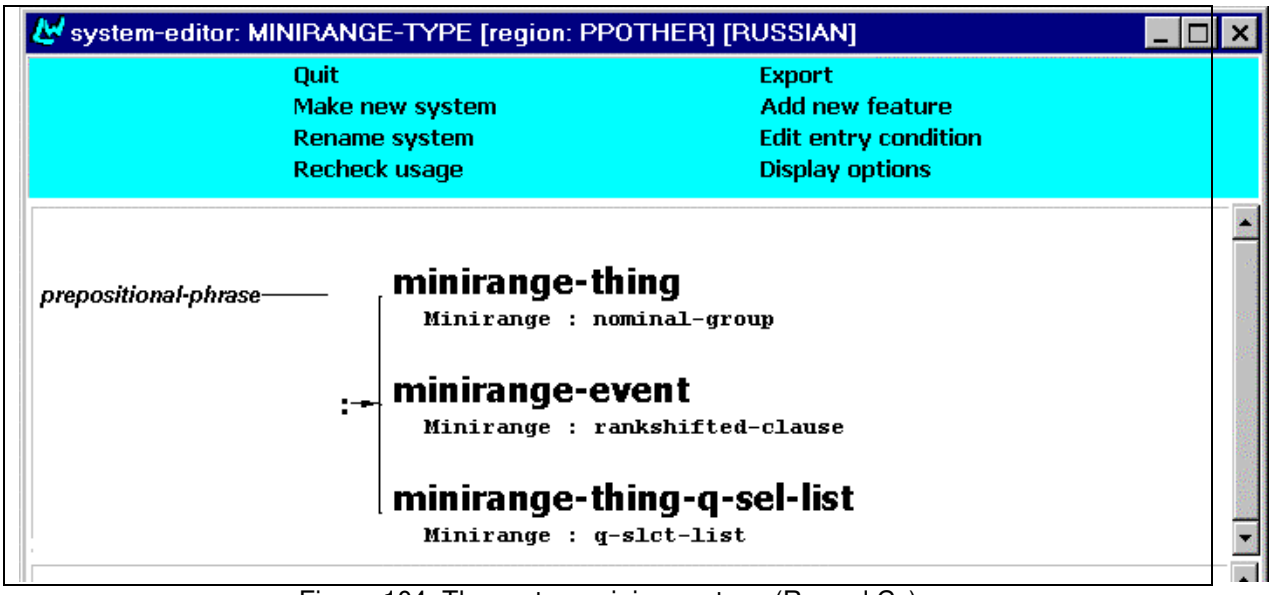


Figure 104: The system minirange-type (Ru and Cz)

For Quantification group we should separate the selective quantification case from MINIRANGE-THING one. Therefore we insert the new out-feature MINIRANGE-THING-Q-SEL-LIST to OUTPUTS of the system.

The choice between MINIRANGE-THING and MINIRANGE-THING-Q-SEL-LIST is made on the grounds of presence in the SPL the QUANTITY-SELECTION-Q QUANTITY feature. In the Figure 105 we show an inquiry-q-code of MINIRANGE-THING-TYPE system (‘quality corresponds here to minirange-thing-q-sel-list feature and ‘nonquality – to simple-minirange-thing).

```
(DEFUN QUANTITY-UP-Q-ru-CODE (ONUS)
(IF (FETCH-SUBC-FEATURE 'RANGE ONUS)
    (IF (EQ (FETCH-SUBC-FEATURE 'QUANTITY-SELECTION-Q (FETCH-SUBC-FEATURE
    'RANGE ONUS))
       'QUANTITY)
     'QUANTITY 'NONQUANTITY)
 'NONQUANTITY)
)
```

Figure 105 The inquiry-q-code of MINIRANGE-THING-TYPE system (Ru and Cz)

Then we should tune the development of Minirange grammatical function be appropriate to the preselection made earlier. Preselection should be transformed to the function feature. For that we need to add the new output feature Q-SLCT-LIST to the system NOMINAL-LIKE-GROUP-CLASS (region RANKING) since our classification of nominal-like groups became more delicate. In Figure 106 we show the system.

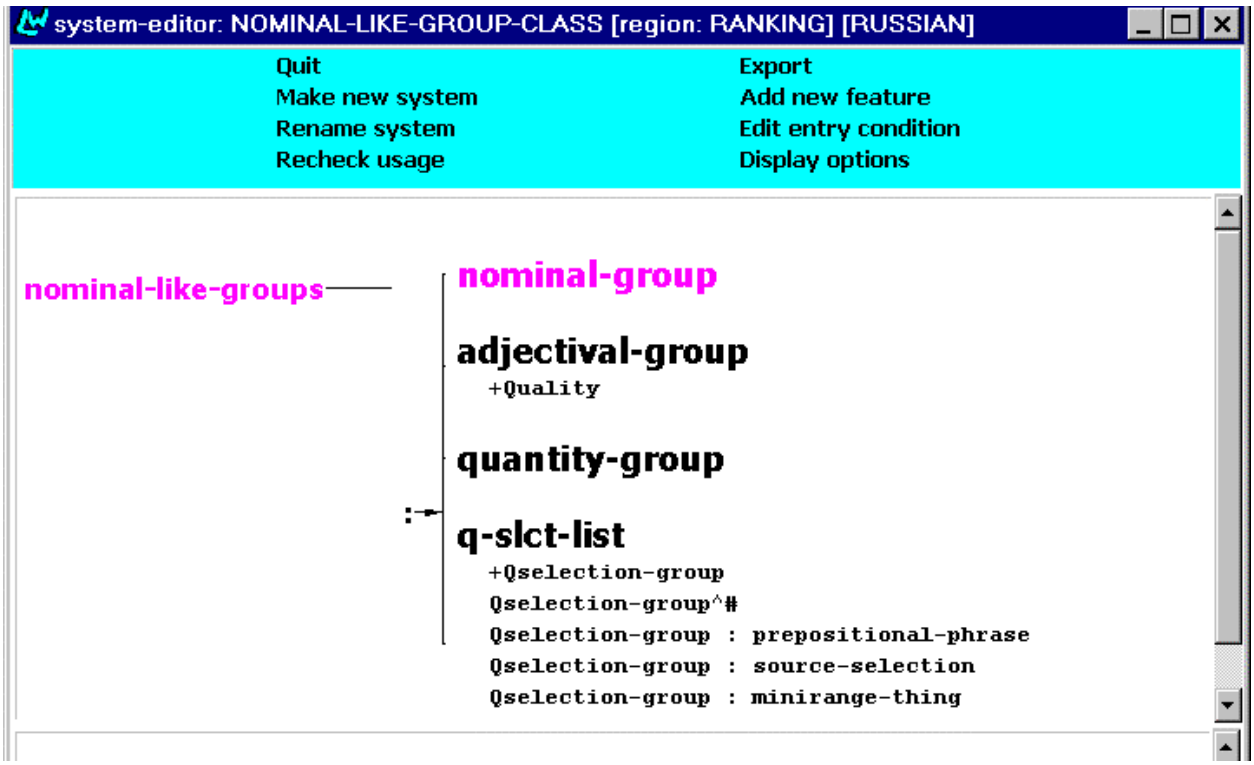


Figure 106: The NOMINAL-LIKE-GROUP-CLASS system (Ru and Cz)

As we can see in Figure 107 the NOMINAL-LIKE-GROUP-CLASS-CHOOSER should also be changed. To minimize changes in resource source code we make the decision SELECTION-LIST vs. NOT-SELECTION-LIST on the top layer of the chooser structure.

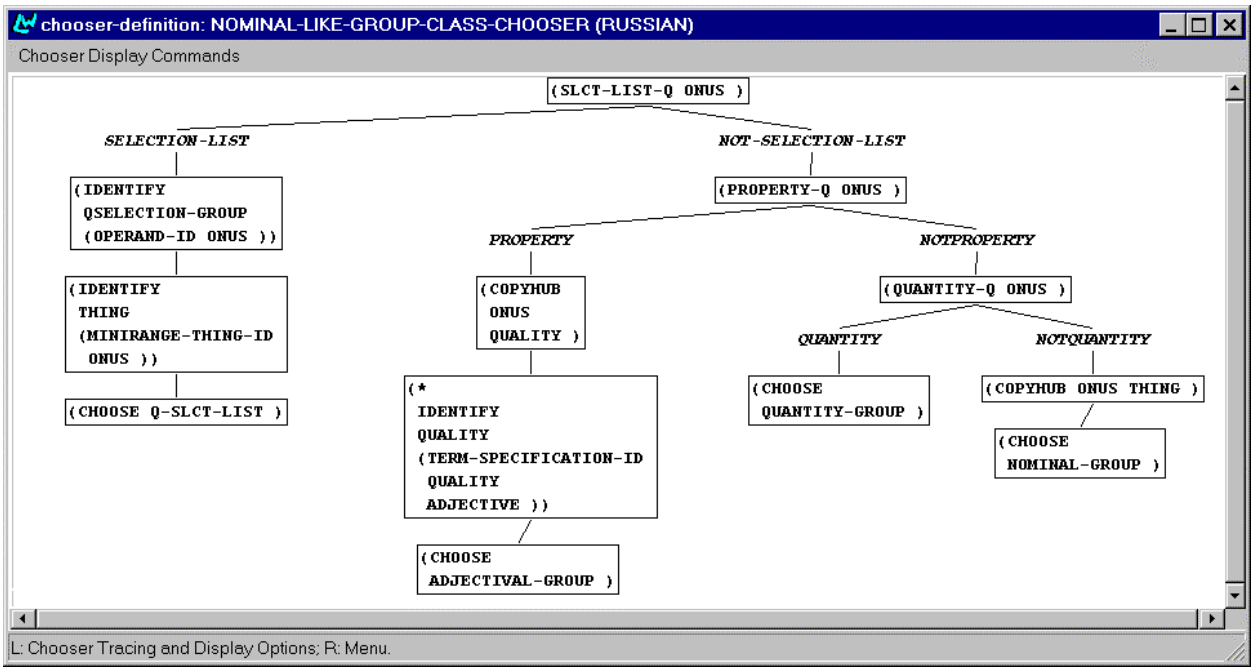


Figure 107: NOMINAL-LIKE-GROUP-CLASS-CHOOSER chooser (Ru and Cz)

All information about choosing alternatives of top structure level is taking from preselections. Therefore appropriate inquiry is written using TRIVIALDEFAULTCODE

and PRESELECTIONGUIDANCE where among other strings the following one is present: (Q-SLCT-LIST . SELECTION-LIST). To insert QSELECTION-GROUP function we set essential restrictions to out-feature Q-SLCT-LIST of the system NOMINAL-LIKE-GROUP-CLASS and the chooser for the system using different procedures: INSERT, ORDERATEND, PRESELECT and IDENTIFY.

Using feature Q-SLCT-LIST the MINIRANGE grammatical function should develop two daughter functions: QUANTIFIER and QSELECTION-GROUP. This is realised by QUANTIFICATION (region QUANTIFICATION) system, which is shown in Figure 108. To insert the QUANTIFIER function we added to the input restriction of the system QUANTIFICATION, which were NOMINAL-GROUP-SIMPLEX feature, a new one - Q-SLCT-LIST.

```
QUANTIFICATION
  {nominal-group-simplex; q-slct-list} ->
    [quantified] +quantifier
    [nonquantified]
```

Figure 108 Quantification system

To insert the QSELECTION-GROUP function we created a new system QSELECTION-GROUP-INSERTION in Ranking region. This system creates and tunes in a proper way the function QSELECTION-GROUP.

The function QSELECTION-GROUP is preselected as prepositional phrase, so it has two dependant functions - MINOPROCESS and MINIRANGE (see Graph Structure in Figure 100). Natural way to realize the semantics of selection is to lexify MINOPROCESS with appropriate preposition – Ru: “iz”; Cz: “z” and Bg: “ot”.

To activate appropriate insertion and lexifying of MINOPROCESS function and also the putting the case marker of the preposition to the nominal group we should set the restriction IZ (OF for English grammar) (region PPOTHER) to QSELECTION-GROUP grammatical function. This function needs features SOURCE-SELECTION and SIMPLE-PREPOSITION as it’s input.

To activate the feature SOURCE-SELECTION we add extra output-feature to MINOR-PROCESS-TYPE system (PPOTHER region). The system is shown in Figure 109 . To do this we use the method we applied earlier to system NOMINAL-LIKE-GROUP-CLASS.

```
MINOR-PROCESS-TYPE
 (prepositional-phrase) →
   [source-selection]
   [spatio-temporal-process]
   [instrumental-process]
   [comparative-process]
   [causal-process]
   [accompaniment-process]
   [matter-process]
   [role-process]
   [portion-process]
```

Figure 109: The MINOR-PROCESS-TYPE system.

The realized structure of systemic choices demonstrate stable behavior on other instances of the selective quantification groups realisation, for example, with other meaning of the :QUANTITY-SELECTION-id, for example:

(130)

(a) En: *Five of the eight segments constitute one polyline.*

(b) Ru: *Пять из (этих) восьми отрезков образуют одну полилинию*
Five-Nom of (these) eight-Gen fragments-Gen form one-Acc polyline-Acc

(c) Cz : *Pět z (těchto) osmi úseků tvoří čáru.*
Five-Nom of (these) eight-Gen segments-Gen form polyline-Acc

(d) Bg: *Пет от осемте отсечки образоват една полилиния*
Five of eight-Det line-fragments form one polyline-Acc

As we pointed out above the SPL construction representation remains universal for the three Slavic language and English.

3.9 Clause complexity

The aim of the current chapter is to describe how clause complexity is handled in the grammars of Czech, Russian and Bulgarian in the Final Prototype of the Agile system. A detailed discussion of the notions of clause complexity as understood in Systemic Functional Grammar (SFG, Halliday 1985) was presented in the LSPEC2 deliverable, and the implementations achieved for the Intermediate Prototype were described in the IMPL2 deliverable.

Even though a variety of types clause complexity were encountered in the Agile corpus, only a subset thereof appeared in the Intermediate Prototype target texts, and we therefore had a narrower focus in the Intermediate Prototype grammars. The coverage in this phase included the following clause complexity types:

- paratactic extension (addition type)
- paratactic enhancement (result circumstantials)
- hypotactic enhancement (manner and purpose circumstantials)

The following examples illustrate the types covered in the Intermediate Prototype:

(131) paratactic extension: positive addition (conjunction)

(a) En: *Specify the internal point and press Return.*

(b) Cz: *Určete vnitřní bod a stiskněte Enter.*
Specify-Pl2 internal-Acc point-Acc and press-Pl2 Return.

(c) Bg: *Въведете вътрешна точка и натиснете Return.*
Specify-Pl2 internal point and press-Pl2 Return.

(d) Ru: *Укажите внутреннюю точку и нажмите Return*
Specify-Pl2 internal-Acc point-Acc and press-Pl2 Return.

(132) paratactic enhancement: result circumstantial

(a) En: *Choose OK and close the dialog box.*

(b) Cz:. *Vyberte OK a uzavřete dialogový panel.*
Choose-Pl2 OK and close-Pl2 dialog-Acc box-Acc

(c) Bg: *Изберете ОК и затворете диалоговия прозорец*
Choose-Pl2 OK and close-Pl2 dialog box

(d) Ru: *Нажмите кнопку ОК и закройте диалоговое окно*
Choose-Pl2 OK and close-Pl2 dialog-Acc box-Acc

(133) hypotactic enhancement: manner (means) circumstantial

(a) En: *Start the PLINE command using one of the following methods.*

(b) Cz: (dependent: nominalization)
Spusťte příkaz PLINE použitím jedné z následujících metod.
Start-Pl2 command-Acc PLINE using-ins one-Gen of following-Gen methods-Gen

(c) Bg: (dependent: finite)
Стартирайте командата PLINE, като използвате един от следните методи
Start-Pl2 command-the PLINE, by use-Pl2 one of following methods

(d) Ru: (dependent: nonfinite)
Запустите команду PLINE, воспользовавшись одним из следующих способов
Start-Pl2 command-Acc PLINE using-gerund one-Ins of following-Gen methods-Gen

(134) hypotactic enhancement: purpose circumstantial

(a) En: *Press Return to end the polyline.*

(b) Cz: (dependent: nominalization)
Stiskněte Return, pro ukončení křivky.
Press-Pl2 Return for ending polyline

(c) Bg: (dependent: finite)
Натиснете Return, за да завършите полилинията.
Press-Pl2 Return so that end-Pl2 polyline

(d) Ru: (dependent: nonfinite)
Нажмите клавишу Return, чтобы завершить рисование полилинии.
Press-Pl2 key Return, in-order-to end drawing polyline-Gen

In the Final Prototype, the following additional coverage is needed:

(135) paratactic extension: positive variation (disjunction)

(a) En: *Specify the internal point or press Return.*

(b) Cz: *Určete vnitřní bod nebo stiskněte Enter.*
Specify-Pl2 internal-Acc point-Acc or press-Pl2 Return.

(c) Bg: *Въведете вътрешна точка или натиснете Return.*
Specify-Pl2 internal point o r press-Pl2 Return.

(d) Ru: *Укажите внутреннюю точку или нажмите Return*
Specify-Pl2 internal-Acc point-Acc or press-Pl2 Return.

(136) paratactic enhancement: temporal circumstantial

(a) En: *Choose OK and then close the dialog box.*

(b) Cz:. *Vyberte OK a potom uzavřete dialogový panel.*
Choose-Pl2 OK and then close-Pl2 dialog-Acc box-Acc

(c) Bg: *Изберете ОК и след това затворете диалоговия прозорец*
Choose-Pl2 OK and after that close-Pl2 dialog box

(d) Ru: *Нажмите кнопку ОК и затем закройте диалоговое окно*
Choose-Pl2 OK and then close-Pl2 dialog-Acc box-

(137) hypotactic enhancement: conditional circumstantial

(a) En: *If you use Windows, enter the Draw command.*

(b) Cz: (dependent: finite)
Pokud používáte Windows, zadejte příkaz Draw.
If use-Pl2 Windows enter-Pl2 command -Acc Draw

(c) Ru: (dependent: nonfinite)
Если вы используете Windows, введите команду Draw.
If you use Windows enter-Pl2 command-Acc Draw

In addition, the Czech grammar now covers two types of hypotactic purpose enhancement: besides the type with the dependent realised by a nominalization generated in the Intermediate Prototype, we can also generate the dependent realised by a finite dependent clause in conditional mood, illustrated below:

(138) Cz: purpose dependent: finite
Vyberte OK, abyste uložili vlastnosti multičáry
Choose-imp-Pl2 OK would-Pl2 save-pastparticiple properties-Acc multilne-Gen
Choose OK in order to save the multilane properties.

As for Bulgarian, it also allows for two types of realisation of a purpose dependent: besides the non-finite version covered in the Intermediate Prototype grammar, Bulgarian also allows for a realisation using nominalization, similarly to Czech:

(139) Bg: purpose dependent: nominalisation
Натиска се Return за завършване на полилинията
Press-3sg refl Return for ending of polyline
Return is pressed for ending the polyline.

These types were discussed in LSPEC2, but were not fully implemented in the Czech and Bulgarian grammars in the Intermediate Prototype. We generated only the type of purpose dependent realized by a nominalization.

The outline of the present chapter is as follows. First, we provide detailed formal specifications concerning those types of clause complexity which occur in the target texts of the Final Prototype (Section 3.9.1). This involves a brief recapitulation of the general specification of clause complexity adopted from SFG, and a detailed specification concerning the types of clause complexity covered in our grammars. We pay special attention to the additional coverage of clause complexity that is offered in the Final Prototype grammars in comparison to the grammars for the Intermediate Prototype.

Secondly, we describe the implementations of the clause complexity region in the Final Prototype grammars, and we present examples of the generated sentences (Section 3.9.2).

3.9.1 Formal Specifications

The principle idea behind clause complexity in SFG as described in (Halliday,1985) is that we can view a sentence as a complex of clauses, just like a group can be perceived of as a **word complex**. And, similarly to word complexes, a **head clause** can be distinguished from clauses that **modify** that head clause. This notion of clausecomplexity enables one to cover the entire spectrum of a sentence's (possible) functional organizations. As a matter of fact, Halliday defines a sentence *as a clause complex*, and the clause complex is the only grammatical unit recognized above the clause (cf. p.216, *ibidem*).

The complexity in clausal organization arises from the different ways in which clauses can be related. SFG interprets relations between clauses in terms of the 'logical' component of the linguistic system. In this interpretation two dimensions are distinguished, to be able to provide more detail on how exactly clauses can "modify" a head clause. The first dimension concerns the system of interdependency, being that of **parataxis** and **hypotaxis**[51]. The other dimension is the system of logico-semantics, regarding **expansion** and **projection**. Because neither the Agile corpus (cf. the CORP and LSPEC1 deliverables) nor the Agile target texts show any clause complexes that can be classified as projections, we only consider expansion in the discussion below.

In the remainder of this section we will describe the dimensions of taxis and expansion in more detail. Crucial thereby is to bear in mind that clause complexity arises out of the interplay of both dimensions.

3.9.1.1 Interdependency or Taxis between Clause Complexes

The idea behind **taxis** is to elucidate the relative status of the head clause and the modifying clause(s):

- **hypotaxis** is the relation between a dependent element and its dominant, on which the former is dependent, whereas
- **parataxis** is a relation between two elements in which neither is dependent on the other (i.e. they stand on equal footing). It is the contention of SFG that all "logical" relations in language are either hypotactic or paratactic.

The following specification captures this distinction for the expansion subtype of clause complexity (we are not considering projection, as noted above):

```
System: EXPANSION-TAXIS →
    [general-hypotactic-expansion],
    [paratactic-expansion]
Chooser:
If one of the parts is more prominent, then choose hypotaxis
Else choose parataxis
```

[51] As a matter of fact, the tactic dimension is general to all complexes, not just to clause complexes - in the same spirit it applies to word, group, and phrase.

We can consider a pair of clauses as a **clause nexus** relating a **primary clause** and a **secondary clause**. The following table associates these terms to taxis:

	Primary clause	Secondary clause
Parataxis	1 (initiating)	2 (continued)
Hypotaxis	α (dominant)	β (dependent)

Figure 110 - Clause nexus / tactic dimensions

In the next subsections we discuss hypotaxis and parataxis in some more detail.

3.9.1.1.1 Hypotaxis

Hypotaxis is the binding of elements of unequal status, namely a **dominant** and a **dependent** subordinate to the dominant. Due to the unequal status of participants involved in a hypotactic relation, the relation is logically speaking *asymmetric* and *non-transitive*.

The elements in a hypotactic structure are ordered in dependence, but this order is largely independent of the linearity in the sentence. Thus, one can have a dependent clause (a) following the dominant, (b) preceding the dominant, (c) enclosed in the dominant, or (d) enclosing the dominant (cf. *ibidem*, pp.222-223), see Figure 111.

(a)	The file is not stored	until you Save it.		
	α	β		α^β
(b)	If you Save the file,	then it will be stored.		
	β	α		β^α
(c)	Store,	if you want,	the file.	
	α	β	α	α «β»
(d)	You might,	the manual says,	save the file.	
	β	α	β	β «α»

Figure 111: Ordering of Dominant and Dependent in a hypotactic clause complex

The ordering of clauses in a hypotactic complex is a matter of textual organization. We discussed the motivations for ordering the dependent before or after the dominant in the TEXS3 deliverable. In most cases in the texts generated in Agile, the default ordering of the dominant preceding the dependent is appropriate. When the opposite ordering is required, the dependent needs to be specified as Thematic, and it then becomes fronted.

The specification of hypotactic complexes is complicated by the fact that multiple dependent elements can modify the same dominant element. The distinction between a "simple" and a "multiple" dependent is made in the HYPOTAXIS-ALPHA-COMPLEXITY system specified below. There are several systems in the implementation that take care of the cases of hypotactic extension, which we do not present here in detail, in the interest of space.

```
System: HYPOTAXIS-ALPHA-COMPLEXITY →
    [complex-alpha-hypotactic-expansion]
       (+Dependent,
        +Terminant,
        Terminant : full),
    [hypotactic-expansion]
Chooser:
If there are multiple dependents
    Then choose complex-alpha-hypotactic-expansion
    Else choose hypotactic-expansion
```

3.9.1.1.2 Parataxis

In **parataxis**, the primary and secondary element of a clause nexus are of equal status. Both the **initiating** (primary) and **continuing** (secondary) element could stand as an individually functional whole - contrary to the case of hypotaxis, where the dependent cannot be considered outside the dominant it relies on. We also see this relative autonomy expressed in the logical characteristics of the paratactic relation, being *symmetry* and *transitivity*. The specification of parataxis is as follows:

```
System: PARATAXIS  →
    [parataxis]
       (+Initiating,
        +Continuing,
        Initiating ... Continuing)
```

Because the initiating and continuing clause in a nexus are not dependent on one another, there is no ordering other than the linearity of the sentence - in contrast to hypotaxis, as we saw above. The ordering of the parts within a paratactic complex corresponds to the ordering of the corresponding pieces of content as specified in the task model (A-box). For steps in a sequence, this ordering models the actual temporal ordering of the steps, as discussed in the TEXM3 deliverable.

3.9.1.2 Logico-Semantic Relations between Clause Complexes

We already indicated above that we characterize the modificationary relation between clauses in a clause complex not only in terms of the tactic dimension, but also in terms of **logico-semantic relationships**. SFG maintains a classification of logico-semantic relations that categorizes them using a small number of abstract types, in order to be able to handle the wide variety of relationships that can be distinguished as to hold between a primary and secondary clause in a clause nexus.

The classification is based on two fundamental classes, being **expansion** and **projection**. The notion of **expansion** expresses the idea that a clause, in its linguistic function as process, exchange and message, can enter into construction with another clause which augments it. The two clauses together then form a clause complex. The notion of **projection**, on the other hand, signifies those constructions in which the second clause is projected through the primary clause. The latter *instates* it as a locution, or an idea. The following specification captures the distinction between projection and expansion:

```
System: TYPE-OF-INTERDEPENDENCE →
       [expansion],
       [projection]
```

As we mentioned already, we restrict ourselves to expansion, because neither the Agile corpus (cf. the CORP and LSPEC1 deliverables) nor the Agile target texts show any clause complexes that can be classified as projections.

A secondary clause can expand the primary clause by **elaborating** it, **extending** it, or **enhancing** it. The following specification reflects this distinction:

```
System: EXPANSION-TYPE →
    [enhancement]   (+Enhancement),
    [extension]     (+Extension),
    [elaboration]   (+Elaboration).
Chooser:
If the expansion develops the same meaning
    then choose elaboration
    else if the expansion presents a circumstantial qualification
       then choose enhancement
       else choose enxtension
```

The remaining subsections elaborate these subtypes.

3.9.1.2.1 Expansion - Elaboration

In **elaboration**, the secondary clause develops further the meaning of the primary clause by specifying or describing it in more detail. As such, the secondary clause does not introduce a new referent in the discourse but rather provides a more elaborate description of an already present referent. Thereby it is not essential that the secondary clause is logically related to the primary clause as a whole - it may elaborate just a part of it (i.e. on one or more constituents).

In combination with parataxis, elaboration gives rise to the following three types: **exposition** – "in other words", **examplification** – "for example", **clarification** – "to be precise". In combination with hypotaxis, elaboration gives the category of **non-defining relative clauses** in SFG.[52] The meaning of the secondary clause is non-restrictive, or descriptive, with respect to the thesis expressed in the primary clause.

Since none of the elaboration types are present in the Final Prototype target texts, we do not consider them any further.

3.9.1.2.2 Expansion - Extension

In **extension** the secondary clause *adds* something new to the meaning expressed in the primary clause - contrary to the case of elaboration, above. What is added may either be a simple **addition**, **replacement** or a **variation.** Again, subtypes of extension are discerned, depending on whether the relation occurs in combination with parataxis or hypotaxis.

---

[52] Note that **defining relative clauses**, unlike the non-defining ones, realize **embedding**. Embedding is functionally different from taxis as defined in SFG: *"Whereas parataxis and hypotaxis are relations BETWEEN clauses or other ranking elements, embedding is not. Embedding is a mechanism whereby a clause or phrase comes to function as a constituent WITHIN the structure of a group, which itself is a constituent of a clause. Hence there is no direct relationship between an embedded clause and the clause within which it is embedded."* (Halliday 1994, p.242).

Paratactic extension is also known as **coordination** between clauses. The different types of coordination are distinguished in the following specification:

```
System: EXTENDING-COORDINATION-TYPE →
    [additive-coordination]
    [alternative-coordination]
    [contrastive-coordination]
Chooser:
    If the relation between the parts is contrastive
       then choose contrastive-coordination
       else if the relation is disjunctive
          then choose alternative-coordination
          else choose additive-coordination
```

The target texts of the Final Prototype contain occurrences of two types of paratactic extension: addition and variation. Addition is realised by **conjunction**, while variation is realised by **disjunction**. The following examples (repeated from the introduction for convenience's sake) illustrate these types:

(140) paratactic extension: positive addition (conjunction)

(a) En: *Specify the internal point and press Return.*

(b) Cz: *Určete vnitřní bod a stiskněte Enter.*
Specify-Pl2 internal-Acc point-Acc and press-Pl2 Return.

(c) Bg: *Въведете вътрешна точка и натиснете Return.*
Specify-Pl2 internal point and press-Pl2 Return.

(d) Ru: *Укажите внутреннюю точку и нажмите Return*
Specify-Pl2 internal-Acc point-Acc and press-Pl2 Return.

(141) paratactic extension: positive variation (disjunction)

(a) En: *Specify the internal point or press Return.*

(b) Cz: *Určete vnitřní bod nebo stiskněte Enter.*
Specify-Pl2 internal-Acc point-Acc or press-Pl2 Return.

(c) Bg: *Въведете вътрешна точка или натиснете Return.*
Specify-Pl2 internal point or press-Pl2 Return.

(d) Ru: *Укажите внутреннюю точку или нажмите Return*
Specify-Pl2 internal-Acc point-Acc or press-Pl2 Return.

3.9.1.2.3 Expansion - Enhancement

Finally, **enhancement** is that type of expansion where one clause enhances the meaning of another by qualifying it by reference to time, place, manner, cause or condition (i.e., **circumstantial**). We list the principal types of enhancement in the table in Figure 112 below (cf. (Halliday,1994), p.234).

Category		Meaning
Temporal		
	same time	A meanwhile B
	different time: later	A subsequently B
	different time: earlier	A previously B
Spatial		
	same place	C there D
Manner		
	Means	N by means of M
	Comparison	N is like M
Causal-conditional		
	cause: reason	because P so result Q
	cause: purpose	because intention Q so action P
	condition: purpose	If P then Q
	condition: negative	If not P then Q
	condition: concessive	If P then contrary to expectation Q

Figure 112 - Types of enhancement

Like the other two types of expansion, enhancement can be combined with either parataxis or hypotaxis. As mentioned above, we cover a subset of the enhancement types in the Final Prototype grammars:

- Paratactic enhancement: causal (result), temporal
- Hypotactic enhancement: temporal, manner (means), causal (purpose) and conditional (condition).

We discuss them below in detail.

3.9.1.2.3.1 Paratactic enhancement

Paratactic enhancement yields a kind of coordination in which a circumstantial feature is incorporated. The following specification reflects the distinctions:

```
System: QUALIFYING-COORDINATION-TYPE →
    [causal-coordination]
    [temporal-coordination]
Chooser:
    If the relation between the parts is a causal sequence
       then choose causal-coordination
       else choose temporal-coordination
```

The examples below illustrate this type of enhancement. The first example is from the Intermediate Prototype target texts and illustrates causal coordination (namely, result).

(142) paratactic enhancement: result circumstantial

(a) En: *Choose OK and close the dialog box.*

(b) Cz:. *Vyberte OK a uzavřete dialogový panel.*
Choose-Pl2 OK and close-Pl2 dialog-Acc box-Acc

(c) Bg: *Изберете ОК и затворете диалоговия прозорец*
Choose-Pl2 OK and close-Pl2 dialog box

(d) Ru: *Нажмите кнопку ОК и закройте диалоговое окно*
Choose-Pl2 OK and close-Pl2 dialog-Acc box-Acc

This example illustrates the realization of a Goal by a result-circumstantial in a paratactic clause complex. A similar realization is possible for a Side-effect. While our grammars are capable of generating this type of clause complex, the text planner does not at the moment plan such realizations.

The next example is from the Final Prototype target texts. It illustrates a temporal coordination. This type of coordination is planned within the full instructions text type. It realises explicit sequence marking, as discussed in the TEXM3 deliverable.

(143) paratactic enhancement: temporal circumstantial

(a) En: *Choose OK and then close the dialog box.*

(b) Cz:. *Vyberte OK a potom uzavřete dialogový panel.*
Choose-Pl2 OK and then close-Pl2 dialog-Acc box-Acc

(c) Bg: *Изберете ОК и след това затворете диалоговия прозорец*
Choose-Pl2 OK and after that close-Pl2 dialog box

(d) Ru: *Нажмите кнопку ОК и затем закройте диалоговое окно*
Choose-Pl2 OK and then close-Pl2 dialog-Acc box-Acc

3.9.1.2.3.2 Hypotactic enhancement

The target texts for the Final Prototype contain several types of hypotactic enhancement: temporal, manner (means), causal (purpose) and conditional (condition). The following specification reflects the distinctions:

```
System: QUALIFYING-CONDITION →
       [QUALIFYING-CONDITION]
System: QUALIFYING-CONDITION-TYPE →
    [concessive-qualifying-condition]
    [conditional-qualifying-condition] →
    [purposive-qualifying-condition]
    [manner-qualifying-condition]
    [spatial-qualifying-condition]
    [causal-qualifying-condition]
    [temporal-qualifying-condition]
Chooser:
```

```
If the dependent presents a manner
 then choose manner-condition
 else if the dependent presents a cause
   then choose causal-condition
   else if the dependent presents a logical condition
     then choose logical-condition
     else if the dependent presents a concession
       then choose concessive-condition
       else if the dependent presents a purpose
         then choose purposive-condition
         else if the dependent presents a space
           then chose spatial-condition
           else if the dependent presents a time
             then choose temporal-condition
```

The examples below illustrate the types of hypotactic enhancement present in the Final Prototype target texts:

(144) hypotactic enhancement: manner (means) circumstantial

(a) En: *Start the PLINE command using one of the following methods.*

(b) Cz: (dependent: nominalization)
Spusťte příkaz PLINE použitím jedné z následujících metod.
Start-Pl2 command-Acc PLINE using-ins one-Gen of following-Gen methods-Gen

(c) Bg: (dependent: finite)
Стартирайте командата PLINE, като използвате един от следните методи
Start-Pl2 command-the PLINE, by use-Pl2 one of following methods

(d) Ru: (dependent: nonfinite)
Запустите команду PLINE, воспользовавшись одним из следующих способов
Start-Pl2 command-Acc PLINE using-gerund one-Ins of following-Gen methods-Gen

(145) hypotactic enhancement: purpose circumstantial

(a) En: *Press Return to end the polyline.*

(b) Cz: (dependent: nominalization)
Stiskněte Return, pro ukončení křivky.
Press-Pl2 Return for ending polyline

(c) **Cz**: (dependent: finite)
Vyberte OK, abyste uložili vlastnosti multičáry
Choose-imp-Pl2 OK would-Pl2 save-pastparticiple properties-Acc multilne-Gen
Choose OK in order to save the multilane properties.

(d) Bg: (dependent: finite)
Натиснете Return, за да завършите полилинията.
Press-Pl2 Return so that end-Pl2 polyline

(e) Ru: (dependent: nonfinite)
Нажмите клавишу Return, чтобы завершить рисование полилинии.
Press-Pl2 key Return, in-order-to end drawing polyline-Gen

(146) hypotactic enhancement: conditional circumstantial

(a) En: *If you use Windows, enter the Draw command.*

(b) Cz: (dependent: finite)
Pokud používáte Windows, zadejte příkaz Draw.
If use-Pl2 Windows enter-Pl2 command -Acc Draw

(c) Ru: (dependent: nonfinite)
Если вы используете Windows, введите команду Draw.
If you use Windows enter-Pl2 command-Acc Draw

In principle, there are the following possibilities for the realisation of the dependent in hypotactic enhancement in our three languages:

- The dependent can be a clause, in which case there the following possibilities
  - finite
  - non-finite
- The dependent can be a gerund.
- The dependent can be a nominalization, in which case there the following possibilities
  - nominal group
  - prepositional phrase

We consider the cases involving a nominalization under the heading of clause complexes, because the nominalizations are obtained from processes, and because they seem to substitute for non-finite dependents. For example in Czech, any hypotactic enhancement is either realised by a finite clause or by a nominalization, there is no possibility involving non-finite clauses.

What realisation possibilities are available in any given language is language specific. So, for example, purpose dependent in Russian is a non-finite clause, in Czech and Bulgarian it is a finite clause or a nominalization (prepositional phrase). A means dependent is a gerund in Russian, a finite clause in Bulgarian and either a finite clause or a nominalization (nominal group in instrumental case) in Czech.

So, for each of the enhancement types, we need to specify the possible realisations. The following specifications hold for the Czech grammar:

```
System: QUALIFYING-CONDITION-CAUSAL-DEPENDENT →
       [CAUSAL-DEPENDENT-CLAUSE]
       [CAUSAL-DEPENDENT-GROUP]

System: QUALIFYING-CONDITION-CONCESSIVE-DEPENDENT  →
       [CONCESSIVE-DEPENDENT-CLAUSE]

System:  QUALIFYING-CONDITION-CONDITIONAL-DEPENDENT  →
       [CONDITIONAL-DEPENDENT-CLAUSE]

System: QUALIFYING-CONDITION-MANNER-DEPENDENT  →
    [MANNER-DEPENDENT-CLAUSE]
    [MANNER-DEPENDENT-GROUP]

System: QUALIFYING-CONDITION-PURPOSE-DEPENDENT →
       [PURPOSE-DEPENDENT-CLAUSE]
       [PURPOSE-DEPENDENT-GROUP]

System: QUALIFYING-CONDITION-SPATIAL-DEPENDENT →
       [SPATIAL-DEPENDENT-CLAUSE]
       [SPATIAL-DEPENDENT-GROUP]

System: QUALIFYING-CONDITION-TEMPORAL-DEPENDENT  →
       [TEMPORAL-DEPENDENT-CLAUSE]
       [TEMPORAL-DEPENDENT-GROUP]
```

In the cases where a choice is available, the decision is for the time being driven by a statement in the SPL, or by a default. Eventually, there should be criteria for deciding between a clause or a group-realization. These criteria could use temporal specificity of the dependent or complexity of the dependent to decide.

3.9.2 Implementation

The implementations handling clause complexes constitute the CLAUSECOMPLEX region of our grammars. The systems of low delicacy in this region are the same for our three languages as well as for English. Differences occur at the more delicate levels, as was pointed out in the formal specifications, and will be reminded below.

The topmost system of the CLAUSECOMPLEX region is the CLAUSECOMPLEXITY system. This system has as outputs the grammatical features [clausecomplex] and [clausesimplex]. The feature [clausecomplex] serves then as input for the system TYPE-OF-INTERDEPENDENCE (see the system in Figure 113 and the chooser in Figure 115).

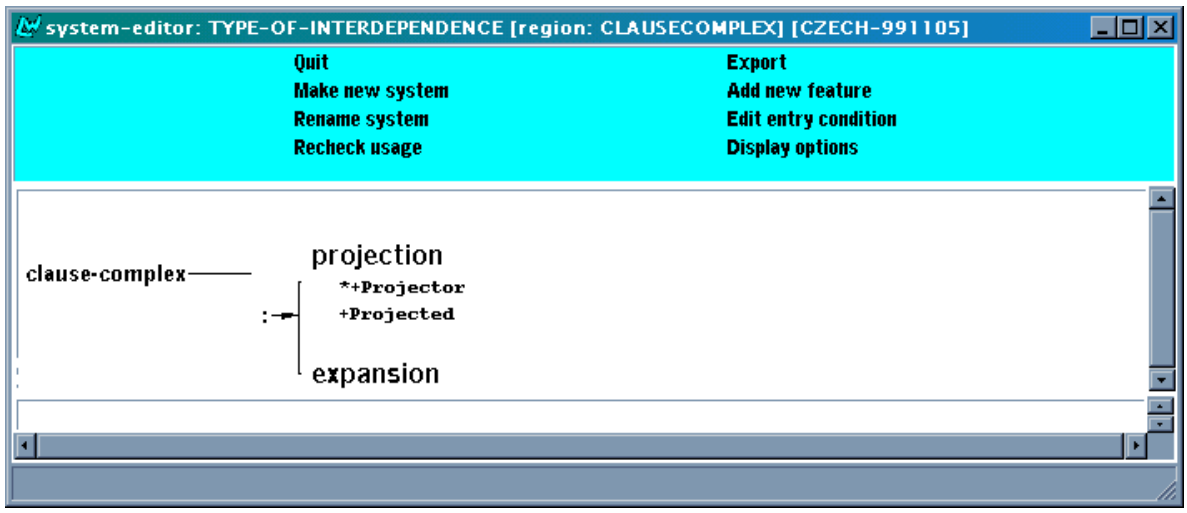


Figure 113 Type of interdependence: system (Cz, Ru, Bg)

Our Intermediate Prototype texts do not show any occurrence of projection, and neither does the AGILE corpus. Therefore, we do not consider that branch of the network any further, and we concentrate on the expansion branch.

Through the **expansion** feature, two systems are triggered, namely EXPANSION-TAXIS (see the system in Figure 114 and the chooser in Figure 116) and EXPANSION-TYPE (see the system in Figure 117 and the chooser in Figure 118).The system EXPANSION-TAXIS ensures an *interplay* of taxis and logico-semantic relations.

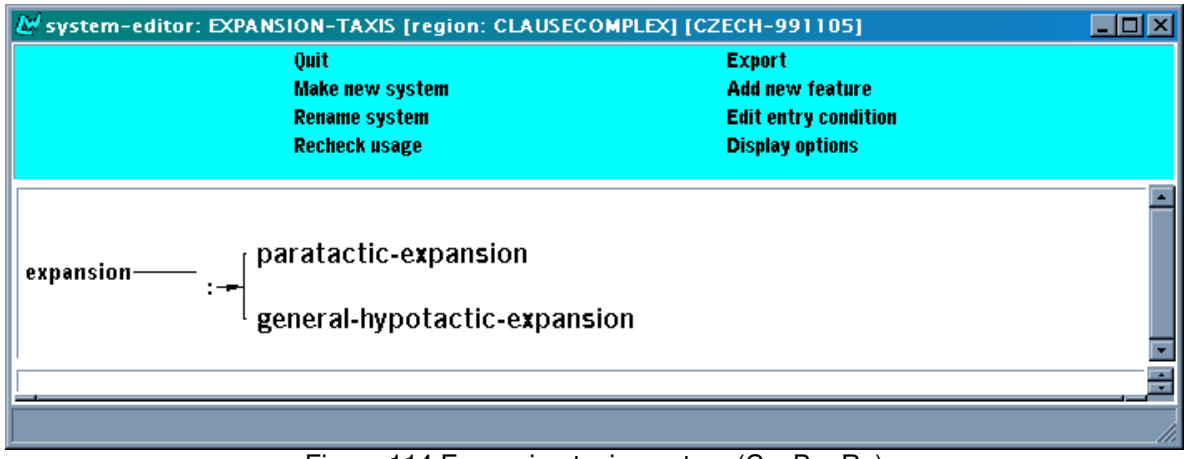


Figure 114 Expansion taxis: system (Cz, Bg, Ru)

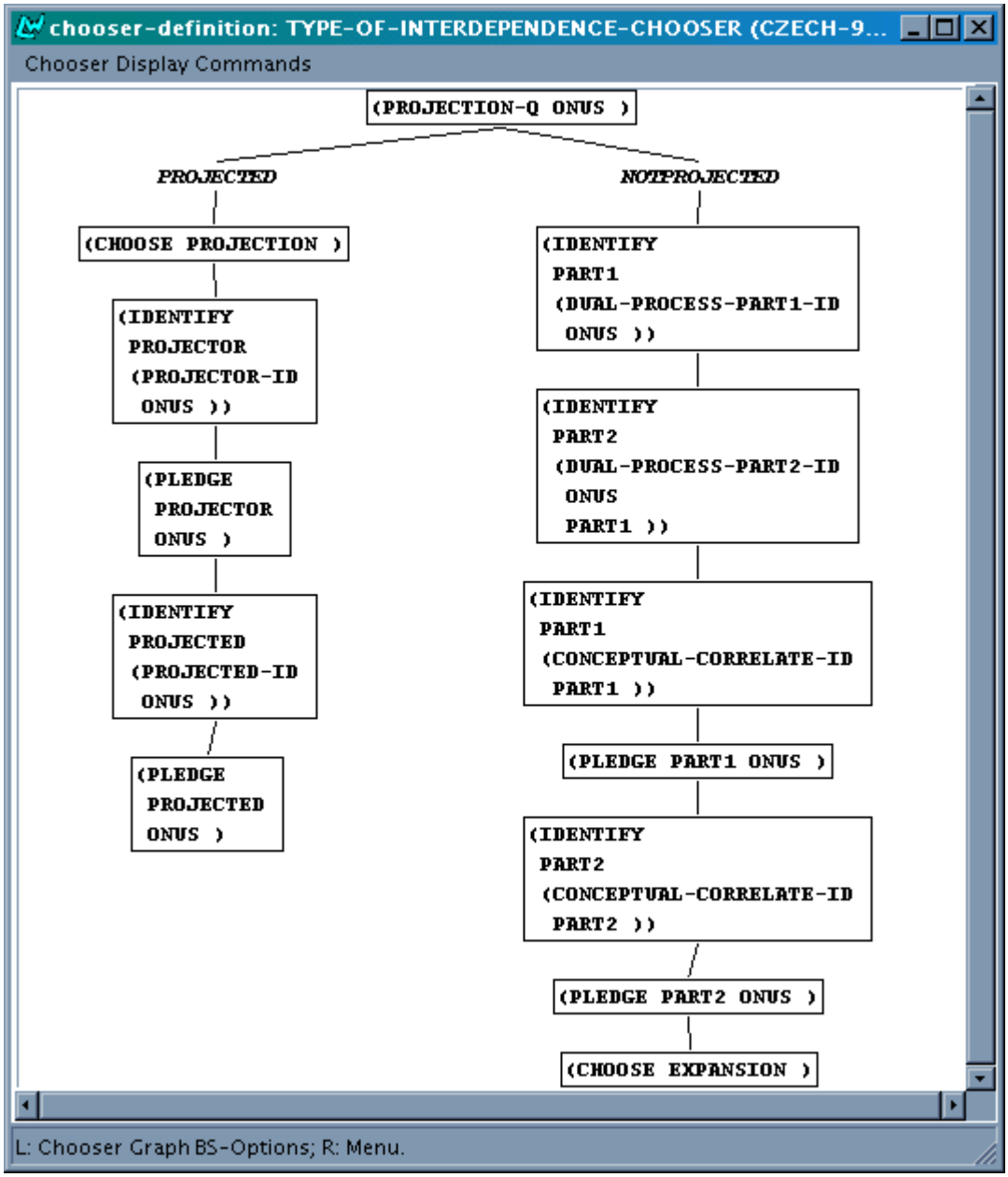


Figure 115 Type of interdependence: chooser (Cz, Ru, Bg)

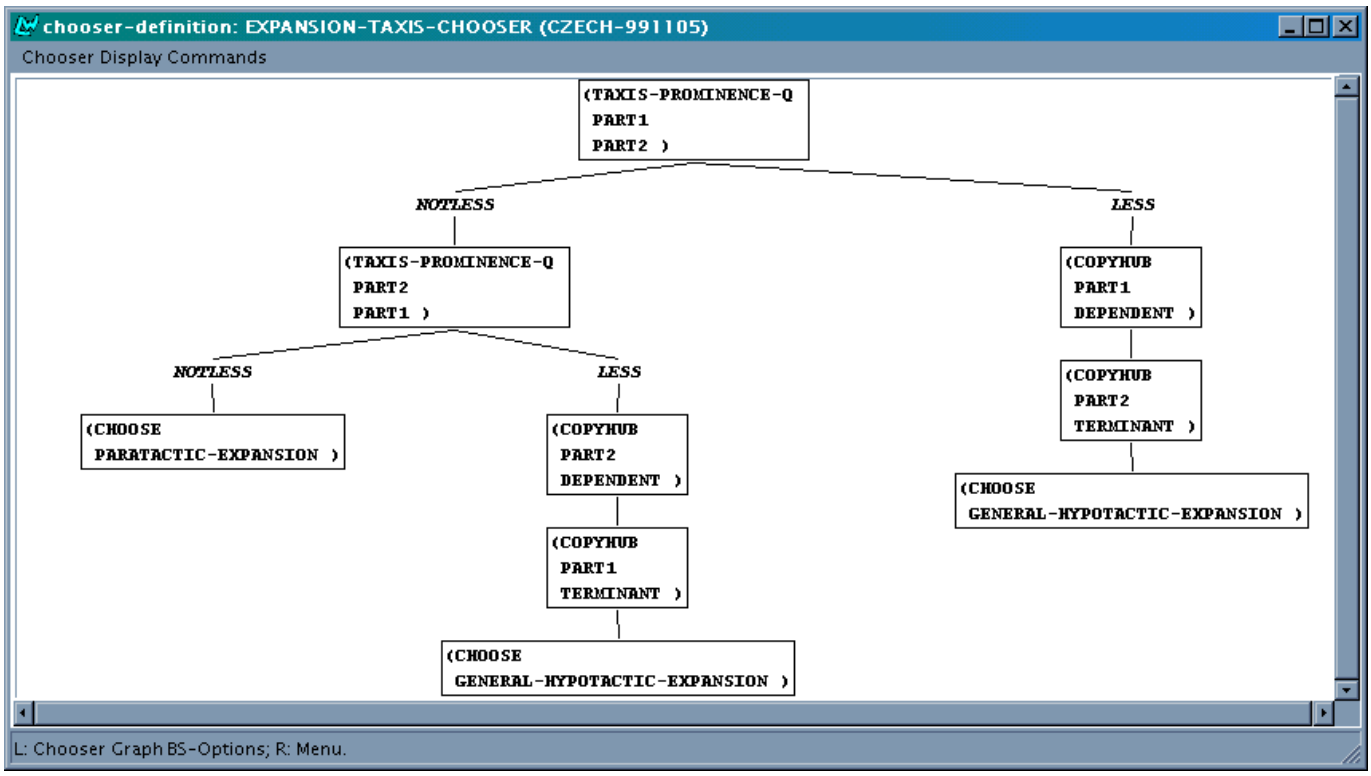


Figure 116 Expansion taxis: chooser (Cz, Bg, Ru)

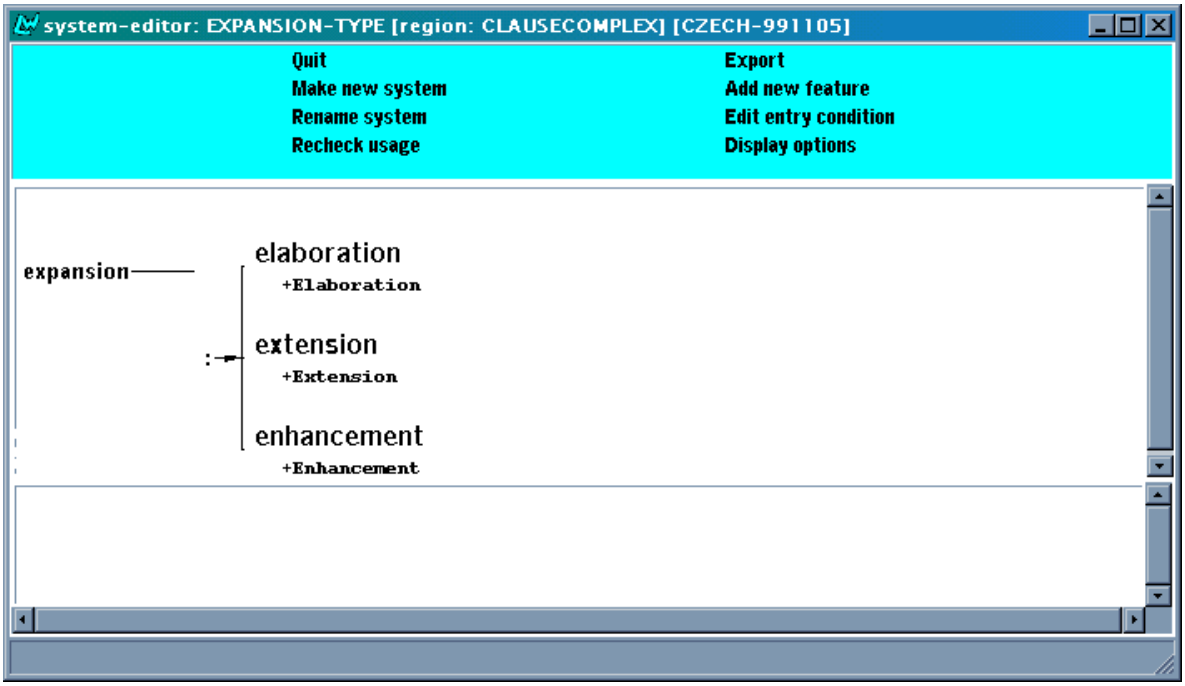


Figure 117 Expansion type: system (Cz, Bg, Ru)

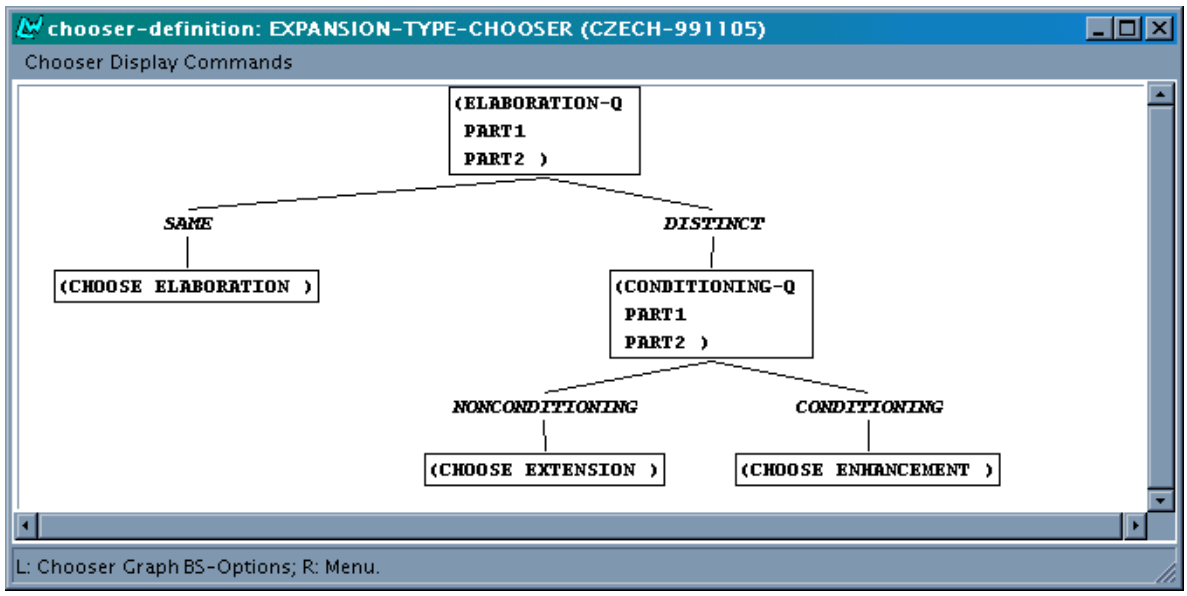


Figure 118 Expansion type chooser (Cz, Bg, Ru)

The tactic dimension has its reflection in the grammatical structure through the systems PARATAXIS (Figure 119) and HYPOTAXIS-ALPHA-COMPLEXITY (see Figure 120 and Figure 123 for the system and chooser implementations).

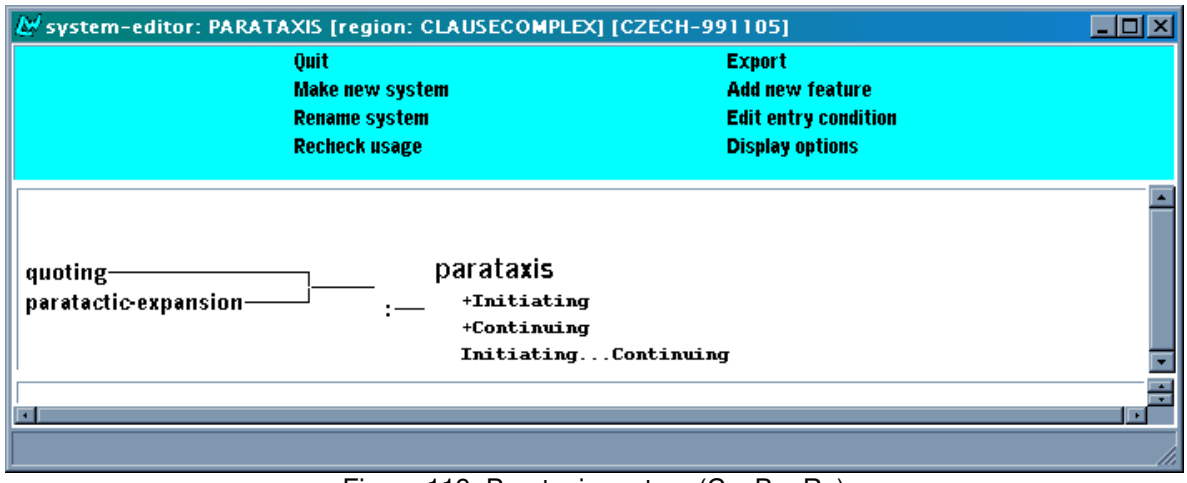


Figure 119: Parataxis system (Cz, Bg, Ru)

As we said earlier, the HYPOTAXIS-ALPHA-COMPLEXITY system serves to handle the cases of multiple dependents. For an illustration of sentences containing multiple dependent elements see Section 3.9.2.4. For a single dependent, the feature **hypotactic-expansion** leads to the system HYPOTAXIS (Figure 121) which inserts the Dependent, and to the system CONJUNCTION (Figure 122) which decides whether there is an explicit conjunction. The latter system is already in a different region, namely the CONJUNTION region.

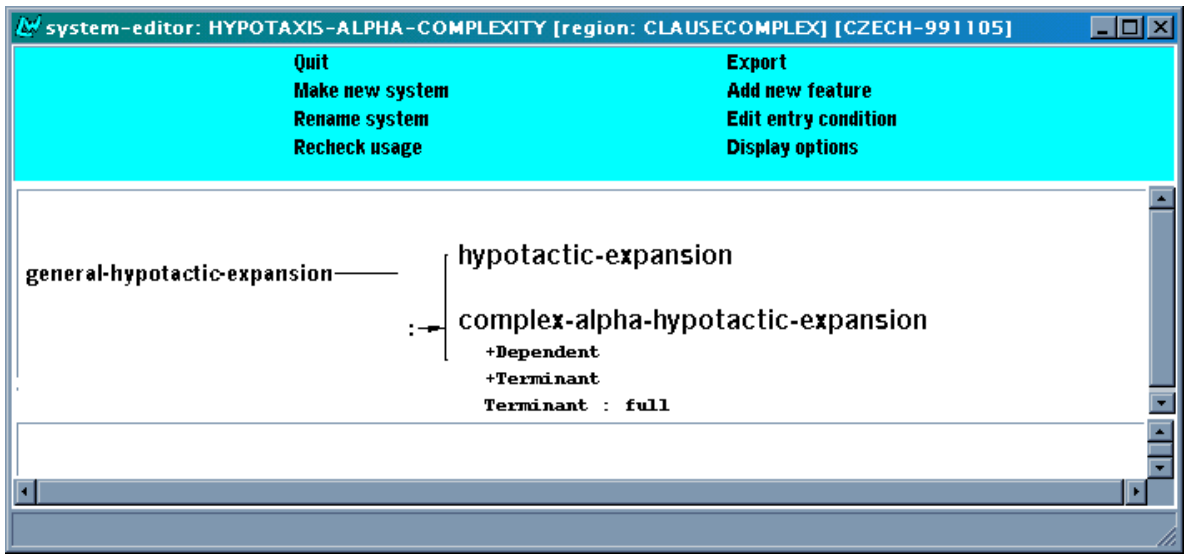


Figure 120: Hypotaxis-alpha-complexity system

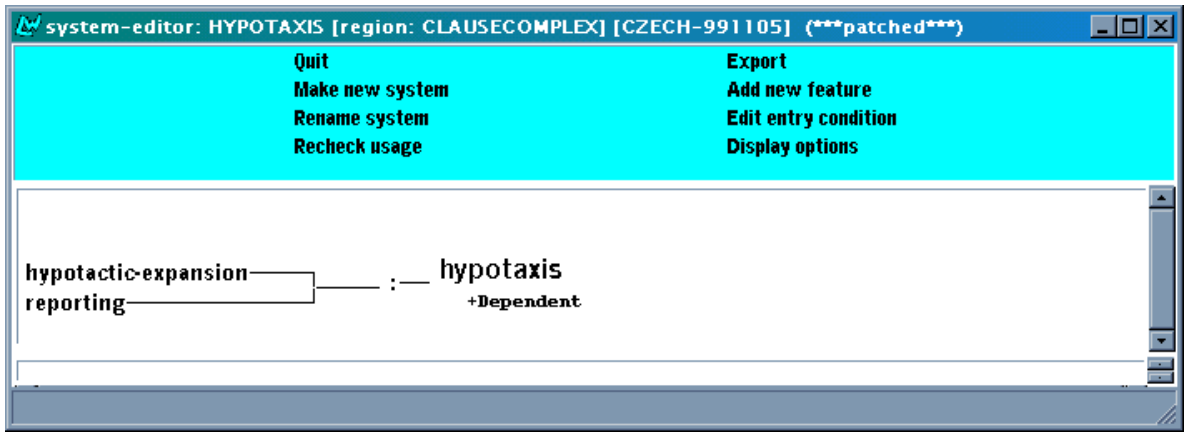


Figure 121: Hypotaxis system (Cz, Ru, Bg)

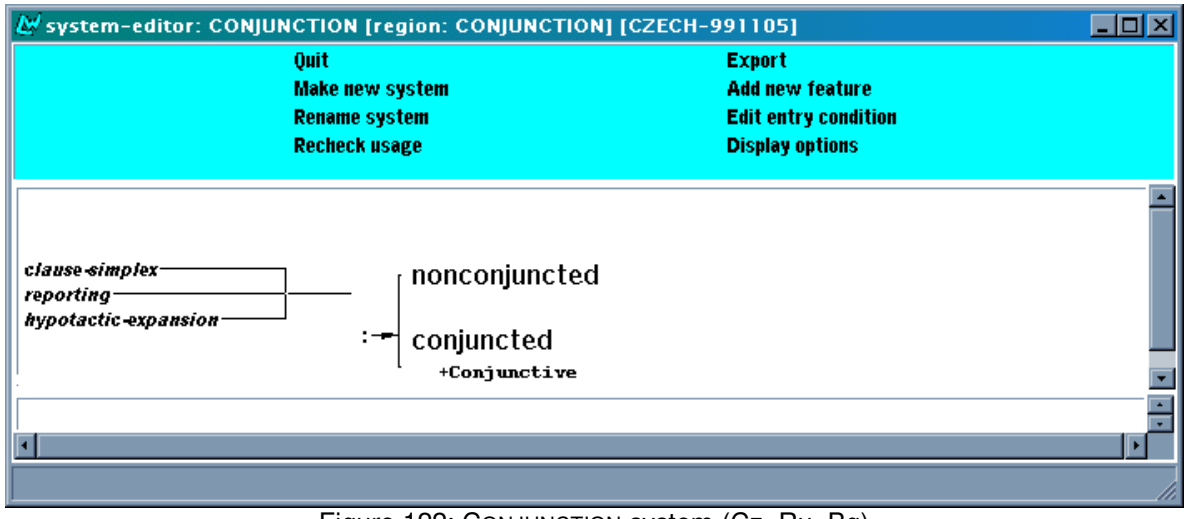


Figure 122: CONJUNCTION system (Cz, Ru, Bg)

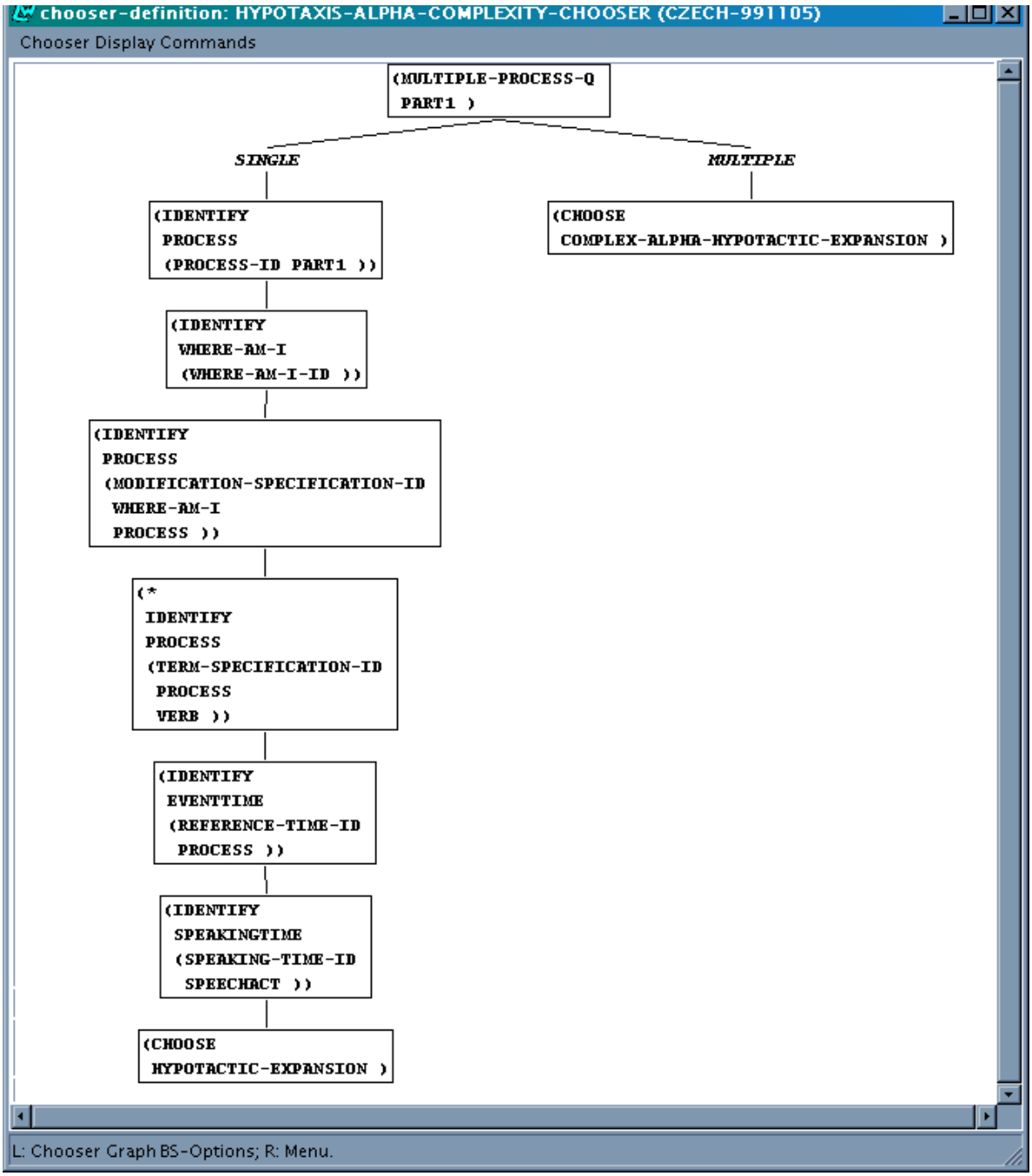


Figure 123: Hypotaxis-alpha-complexity chooser (Cz, Ru, Bg)

The generation of a clause complex proceeds as follows. On the first traversal through these systems, the basic type of expansion relation is determined, and the type of their interdependence. For the selected expansion relation a corresponding grammatical function is inserted into the grammatical structure, as is clear from the definition of the EXPANSION-TYPE system above, whereas the interdependency systems insert the elements relevant for the selected type of taxis (e.g. Initiating and Continuing for parataxis). On a subsequent traversal, these grammatical features are combined to act as inputs to systems detailing out the exact kind of clause complexity.

In most cases, a further traversal through the region determines then the exact kind of clause complexity, for enhancement for example in terms of circumstantials (QUALIFYING-CONDITION). These systems introduce grammatical features with which constraints for

systems in regions of lower rank are associated, concerned for example with dependent clauses (DEPENDENCY region) the complexity of nominal groups (NOMINALGROUPCOMPLEXITY region) or circumstantials (CIRCUMSTANCE region). These regions are not described in detail here. What is important about this observation is that the CLAUSECOMPLEX region is thus primarily concerned with *classifying* clauses (or groups and phrases, in the case of nominalization), which to some extent influences the grammatical categories of the clauses involved, but leaving the more detailed realization to other regions.

3.9.2.1 Paratactic Extension

The system and chooser relevant for the classification of paratactic clause complexes are shown in Figure 124 and Figure 125, respectively. The appropriate connectives are inserted and lexified in this system. In order to generate the appropriate connectives in each language, one can either modify the insertions in each grammar, or one can use some universal names for the connectives, which are used in each of the grammars, and create the corresponding lexical entries in each lexicon.

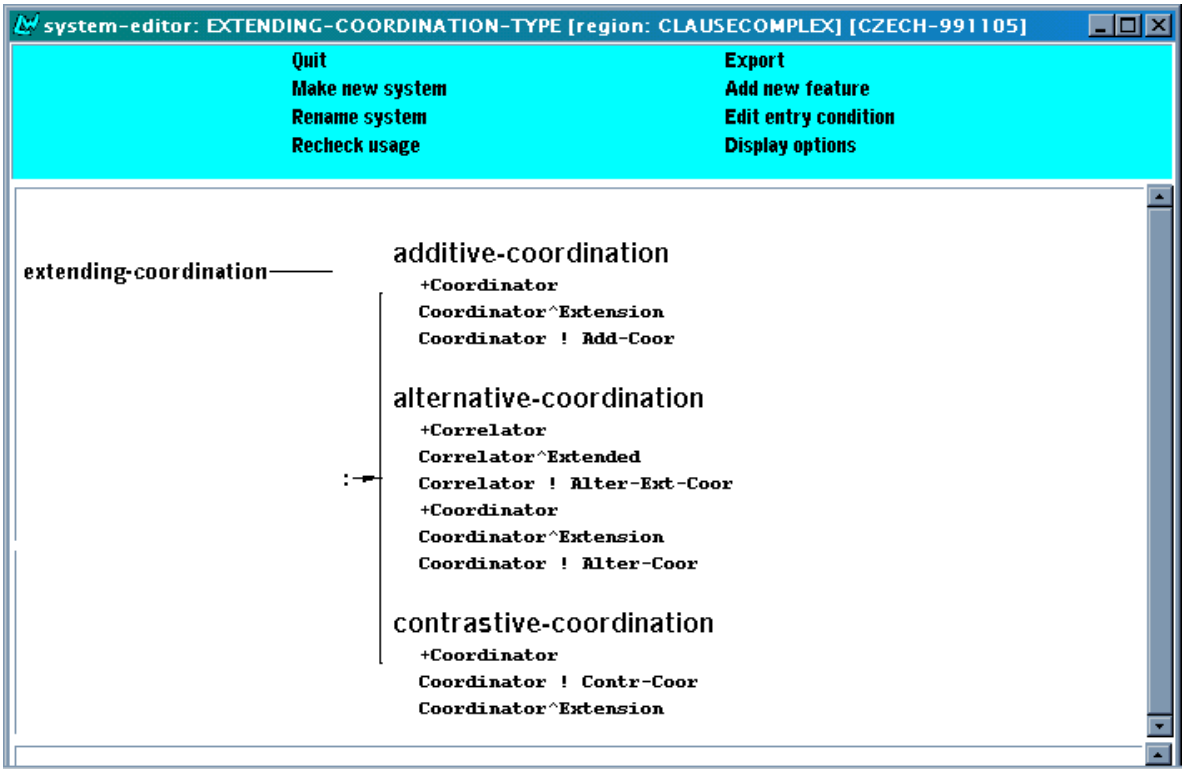


Figure 124 Extending coordination type: system (Cz, Ru, Bg)

The appropriate connectives are as follows:

- additive coordination: En: "and", Cz: "a", Bg: "и", Ru: "и",
- alternative coordination:
- correlator: En: "either", Cz: "buď", Bg: "или", Ru: "или"
- coordinator: En: "or", Cz: "nebo", Bg: "или", Ru: "или"
- contrastive coordination: En: "but", Cz: "ale", Bg: "но", Ru: "но"

Note that while English and Czech have a pair of connectives for alternative coordination, one for the Extended and another for the Extension, both Russian and Bulgarian use the same word in both functions. So, in English, the pair is *either-or*, in Czech it is *buď-nebo*, and Russian as well as Bulgarian it is *или-или*.

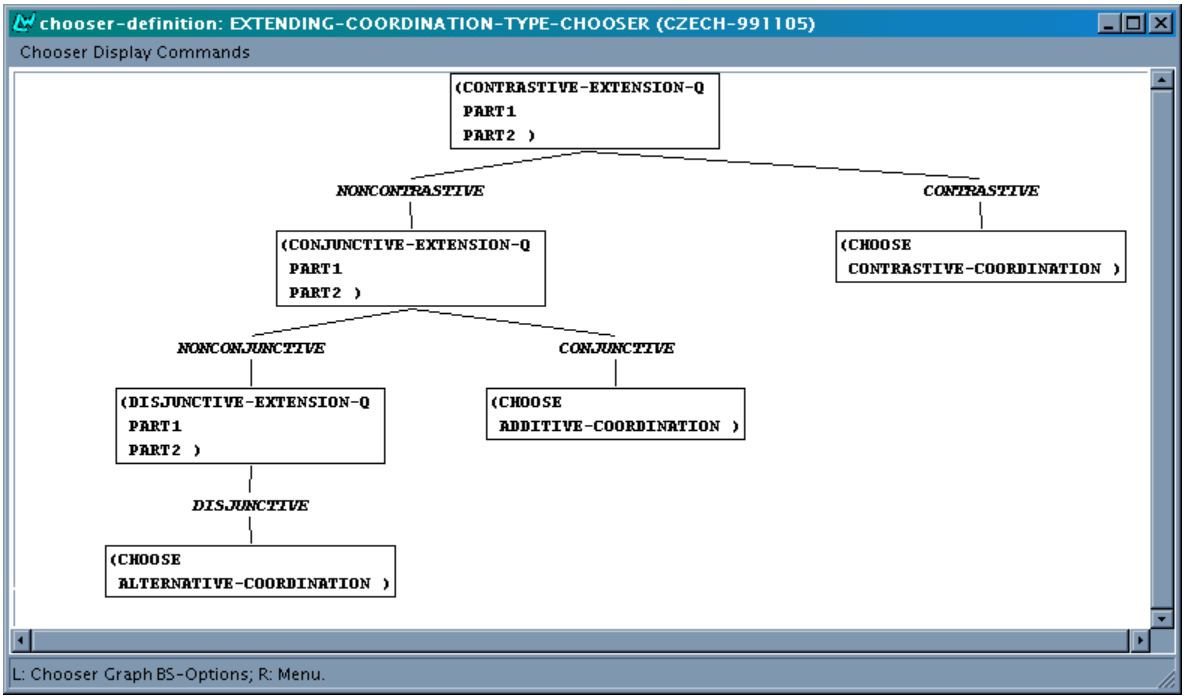


Figure 125 Extending coordination type: chooser (Cz, Ru, Bg)

The following examples illustrate the generation input and output for additive and alternative coordination using the Czech grammar (for examples of disjunction and conjunction involved in a combination of clause complexity types also in Russian and Bulgarian see Section 3.9.2.4):

(147) Paratactic extension: additive

(a) Input SPL:

```
(c / conjunction
    :Domain
    (S1 / DM::SPECIFY-COMPONENT
        :SPEECHACT IMPERATIVE
        :ACTEE (P1 / OBJECT :LEX bod
                            :IDENTIFIABILITY-Q notIDENTIFIABLE
          :Property-aScription (P / Quality :lex vnitr3ni2) ))
    :Range
    (S2 / DM::PRESS
        :ACTEE (A / DM::key :IDENTIFIABILITY-Q NOTIDENTIFIABLE))))
```

(b) Generated output: (see Figure 126 for the grammatical structure)

Cz: *Určete vnitřní bod a stiskněte klávesu.*
Specify-Pl2 internal-Acc point-Acc and press-Pl2 key.
Specify an internal point and press a key.

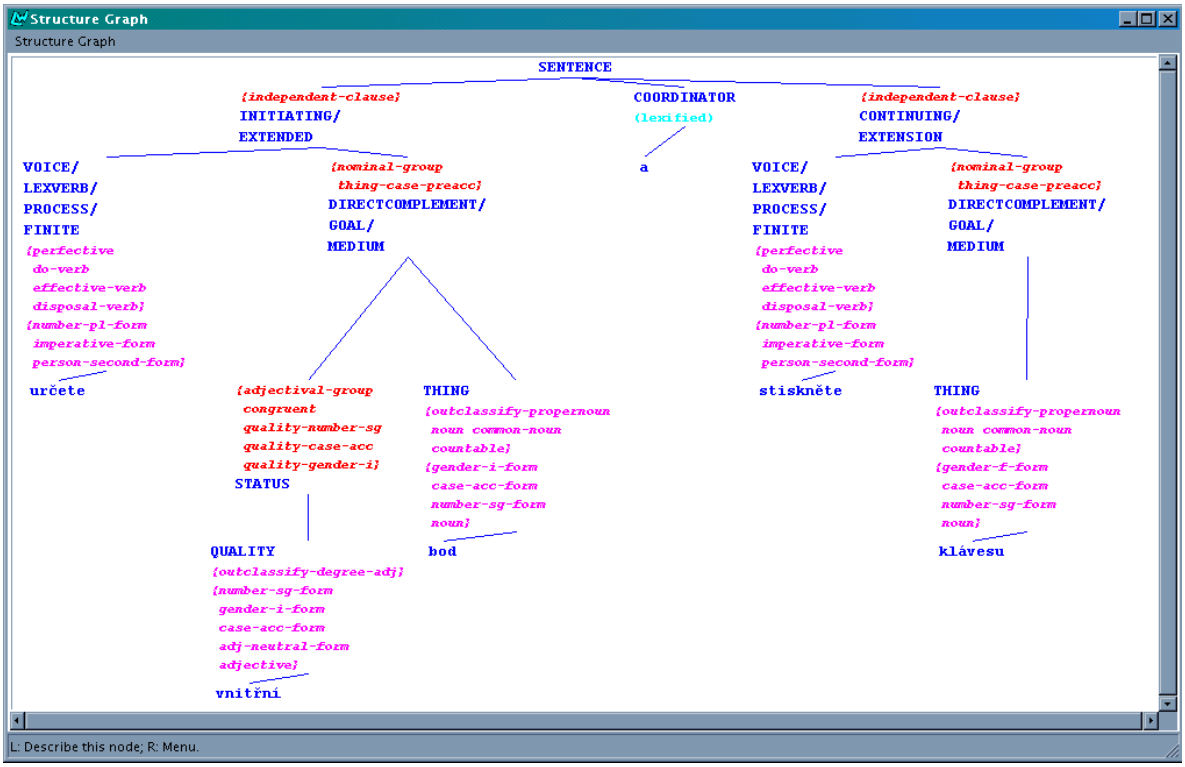


Figure 126: Grammatical structure for (147), Czech

(148) Paratactic extension: alternative

(a) Input SPL:

```
(c / disjunction
    :Domain
    (S1 / DM::SPECIFY-COMPONENT
        :SPEECHACT IMPERATIVE
        :ACTEE (P1 / OBJECT :LEX bod
                            :IDENTIFIABILITY-Q notIDENTIFIABLE
          :Property-aScription (P / Quality :lex vnitr3ni2) ))
    :Range
    (S2 / DM::PRESS
        :ACTEE (A / DM::key :IDENTIFIABILITY-Q NOTIDENTIFIABLE))))
```

(b) Generated output: (see Figure 127 for the grammatical structure)

Cz: *Určete vnitřní bod nebo stiskněte klávesu.*
Specify-Pl2 internal-Acc point-Acc or press-Pl2 key.
Specify an internal point or press a key.

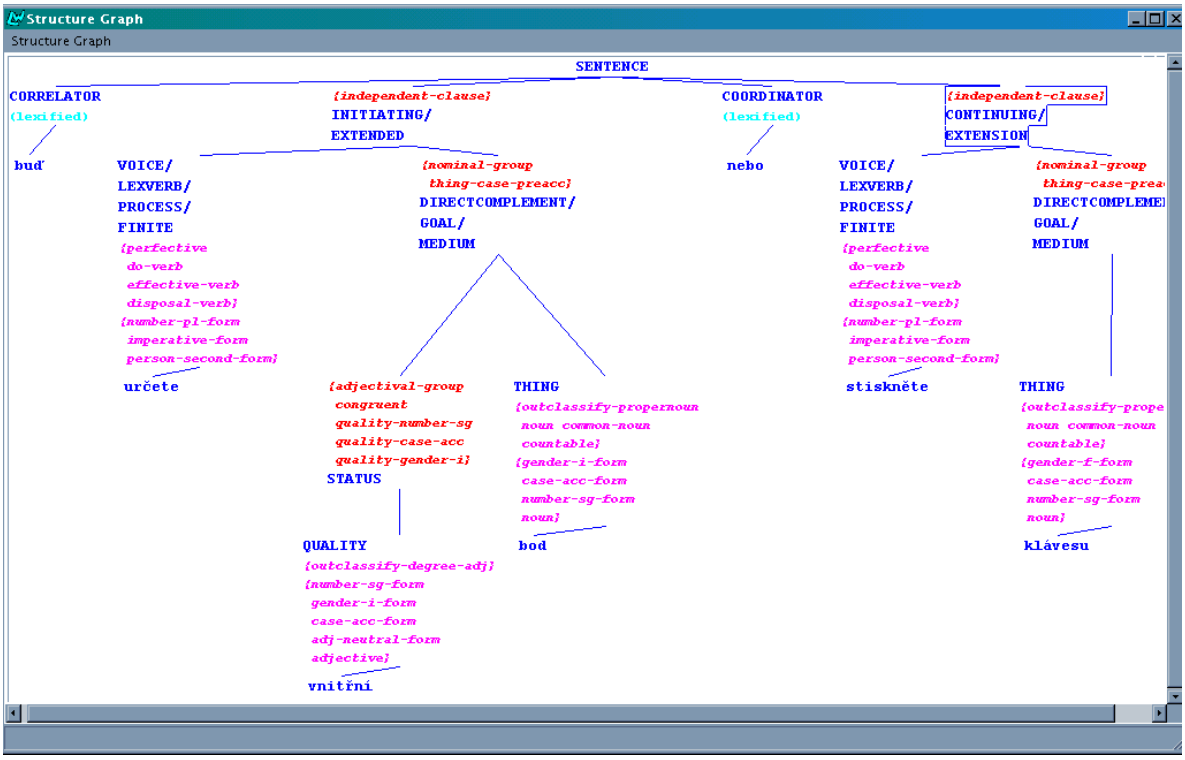


Figure 127: Grammatical structure for (148), Czech

3.9.2.2 Paratactic Enhancement

The relevant implementations are shown in Figure 128 and Figure 129. Note that also in this case the respective connectives are inserted in the system. The lexifications are shown below:

- Causal coordination: En: "and", Cz: "a", Bg: "и", Ru: "и"
- Temporal coordination: En: "then", Cz: "pak", Bg: "след това", Ru: "затем"

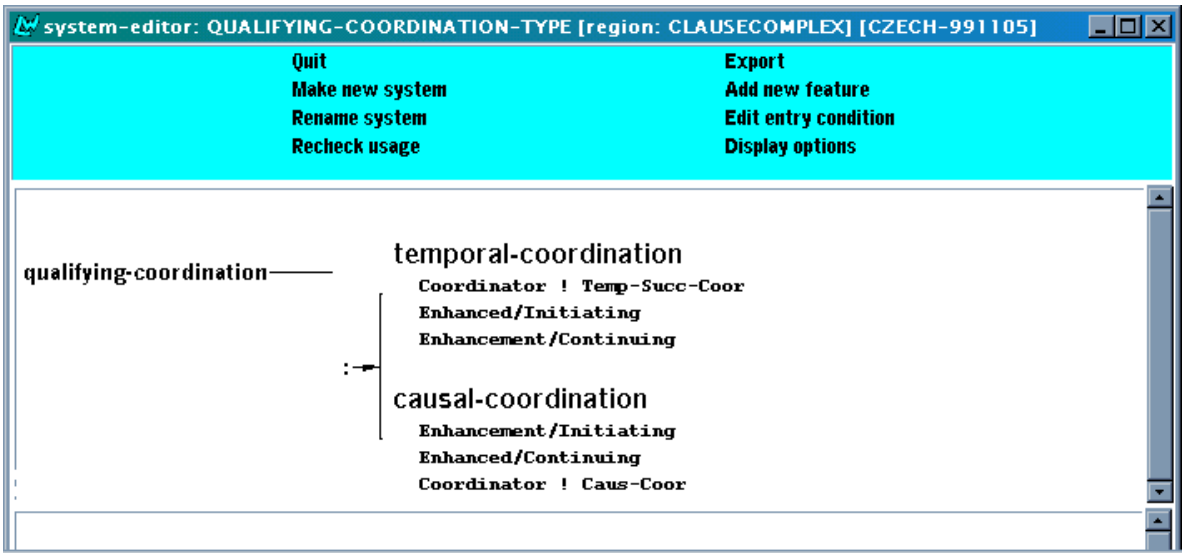


Figure 128 Qualifying-coordination type: system (Cz, Ru, Bg)

```
(CHOOSER
 :NAME                 QUALIFYING-COORDINATION-TYPE-CHOOSER
 :DEFINITION           ((ASK (TEMPORAL-ENHANCEMENT-Q PART1 PART2)
                             (NOTTEMPORALSUCCESSION
                              (ASK (CAUSAL-ENHANCEMENT-Q PART1 PART2)
                                   (CAUSAL
                                    (CHOOSE CAUSAL-COORDINATION)
                                    (ASK
                                     (CAUSAL-SEQUENCE-Q PART1 PART2)
                                     (NOTCAUSES
                                      (ASK
                                       (CAUSAL-SEQUENCE-Q PART2 PART1)
                                       (CAUSES
                                        (COPYHUB PART2 INITIATING)
                                        (COPYHUB PART1 CONTINUING))))
                                     (CAUSES
                                      (COPYHUB PART1 INITIATING)
                                      (COPYHUB PART2 CONTINUING))))))
                             (TEMPORALSUCCESSION
                              (CHOOSE TEMPORAL-COORDINATION)
                              (IDENTIFY PROCESS1 (PROCESS-ID PART1))
                              (IDENTIFY PROCESS1TIME
                                        (REFERENCE-TIME-ID PROCESS1))
                              (IDENTIFY PROCESS2 (PROCESS-ID PART2))
                              (IDENTIFY PROCESS2TIME
                                        (REFERENCE-TIME-ID PROCESS2))
                              (ASK (PRECEDE-Q PROCESS1TIME PROCESS2TIME)
                                   (NOTPRECEDES
                                    (ASK
                                     (PRECEDE-Q
                                      PROCESS2TIME
                                      PROCESS1TIME)
                                     (PRECEDES
                                      (COPYHUB PART2 INITIATING)
                                      (COPYHUB PART1 CONTINUING))
                                     (NOTPRECEDES
                                      (COPYHUB PART1 INITIATING)
                                      (COPYHUB PART2 CONTINUING))))
                                   (PRECEDES
                                    (COPYHUB PART1 INITIATING)
                                    (COPYHUB PART2 CONTINUING))))))
```

Figure 129 Qualifying-coordination type: chooser (Cz, Ru, Bg)

The following example illustrates the generation of a temporal paratactic coordination.

(149) Paratactic enhancement: temporal

(a) Input SPL:

```
(c / RST-sequence
    :Domain
    (S1 / DM::SPECIFY-COMPONENT
        :SPEECHACT IMPERATIVE
        :ACTEE (P1 / DM::line-segment
                        :IDENTIFIABILITY-Q notIDENTIFIABLE ))
    :Range
    (S2 / DM::PRESS
        :ACTEE (A / DM::key :IDENTIFIABILITY-Q NOTIDENTIFIABLE))))
```

(b) Generated output: (see Figure 130 for the grammatical structure)
Cz: *Určete úsečku a potom stiskněte klávesu.*
Specify-Pl2 line-Acc and then press-Pl2 key.
Specify a line segment and then press a key.

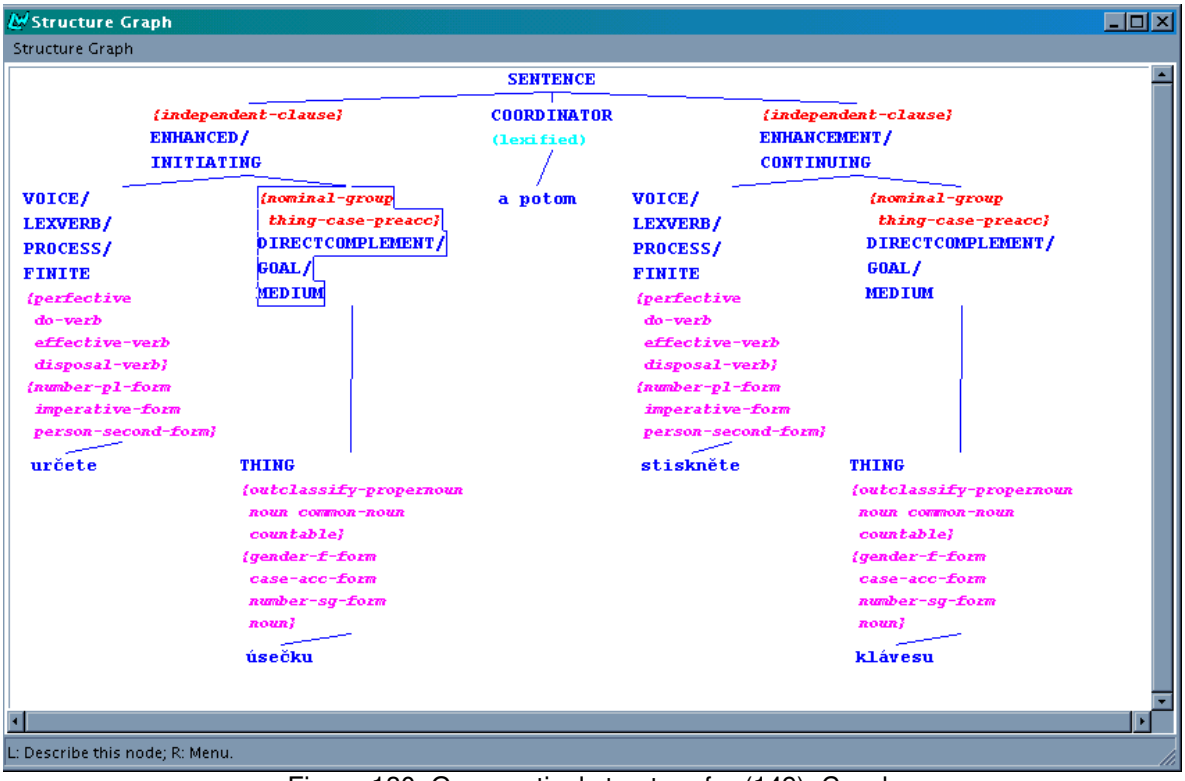


Figure 130: Grammatical structure for (149), Czech

3.9.2.3 Hypotactic enhancement

The system making the choice between the various types of hypotactic enhancement is shown in Figure 131. We do not show the chooser, because of its size (cf. Section 3.9.1.2.3.2).

Then, as we said above, the appropriate choice has to be made as for the realisation of the dependent. We do not show all the relevant systems here, but only the ones for purposive condition (Figure 132, Figure 133), means condition (Figure 134) and conditional condition (Figure 135) as implemented in the Czech grammar. Similar systems are implemented in the Russian grammar. In the Bulgarian grammar, only the implementation of hypotactic purpose enhancement has been fully implemented, means enhancement is incomplete, and conditional enhancement has not been implemented.

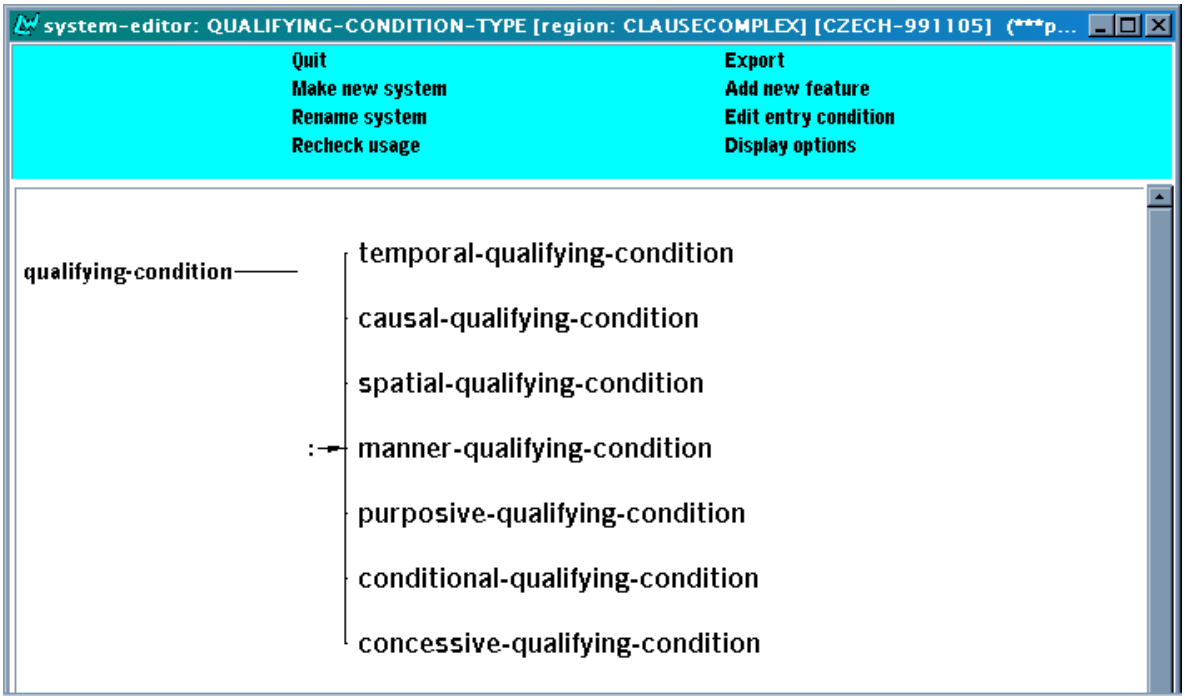


Figure 131: Qualifying condition type system (Cz, Ru, Bg)

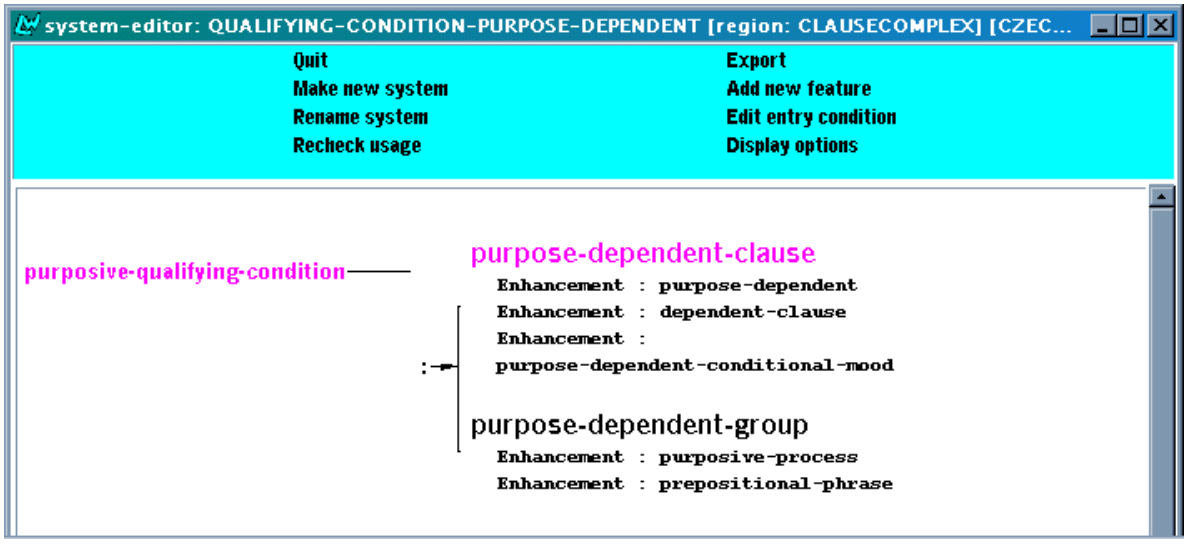


Figure 132 Qualifying-condition-purpose-dependent: system (Cz)

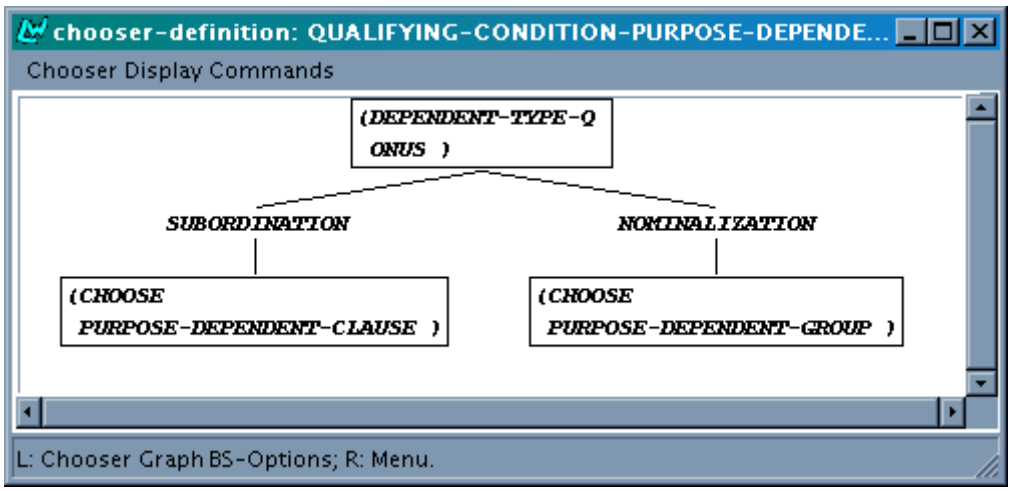


Figure 133: Qualifying-condition-purpose-dependent chooser (Cz)

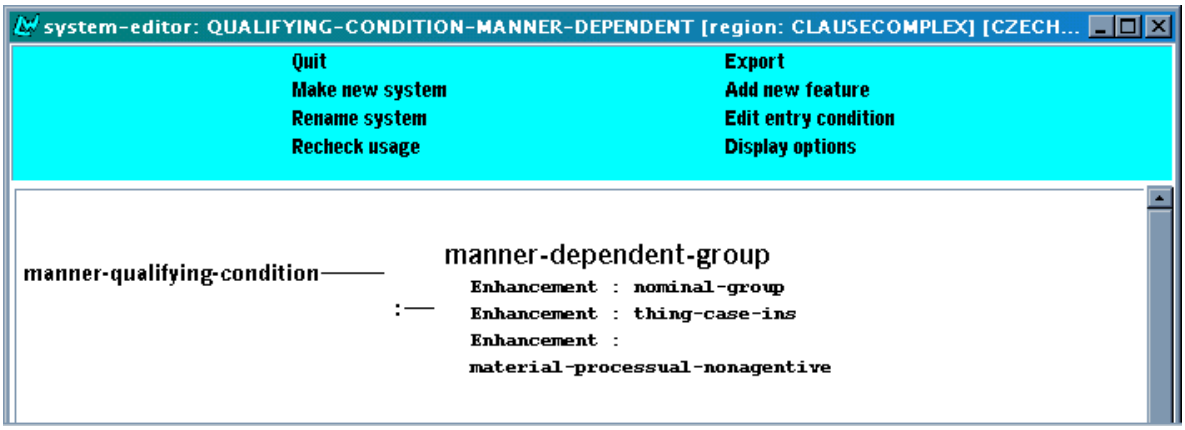


Figure 134 Qualifying-condition-means-dependent: system (Cz)

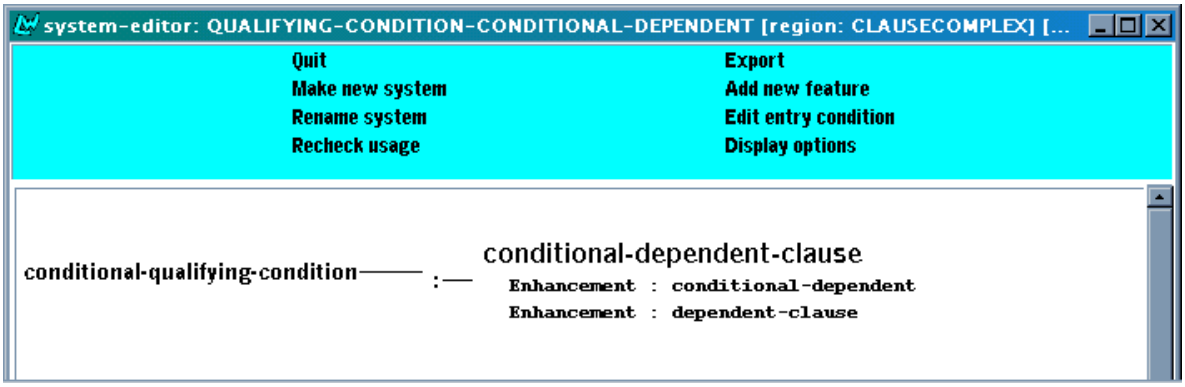


Figure 135: Qualifying-condition-conditional-dependent: system (Cz)

The following examples demonstrate the generation of purpose hypotactic enhancement clauses in Russian, Bulgarian and Czech. As shown above, Russian realizes the purpose dependent clauses as non-finite, Bulgarian allows for finite or nominalization, but only finite has been implemented, and Czech allows for finite or nominalization, and both have been implemented.

(150) Hypotactic enhancement: purpose (Ru: non-finite)

(a) Input SPL:

```
(R / RST-PURPOSE :DOMAIN
   (P / DIRECTED-ACTION :LEX NAZHATJ :SPEECHACT IMPERATIVE :ACTEE
      (A1 / OBJECT :LEX KLAVISHA :LABEL-ASCRIPTION
          (L1 / OBJECT :NAME RETURN)))
   :RANGE
   (E / DIRECTED-ACTION :LEX ZAVERSHITJ :ACTEE
      (D / ABSTRACTION :LEX RISOVANIE :META-ACTANT
         (M1 / OBJECT :LEX POLILINIJA))))
```

(b) Generated output: (see Figure 136 for the grammatical structure)

Ru: *Нажмите клавишу Return, чтобы завершить рисование полилинии.*

Press key-Acc Return in-order end drawing polyline-Gen

Press the Return key in order to quit drawing the polyline.

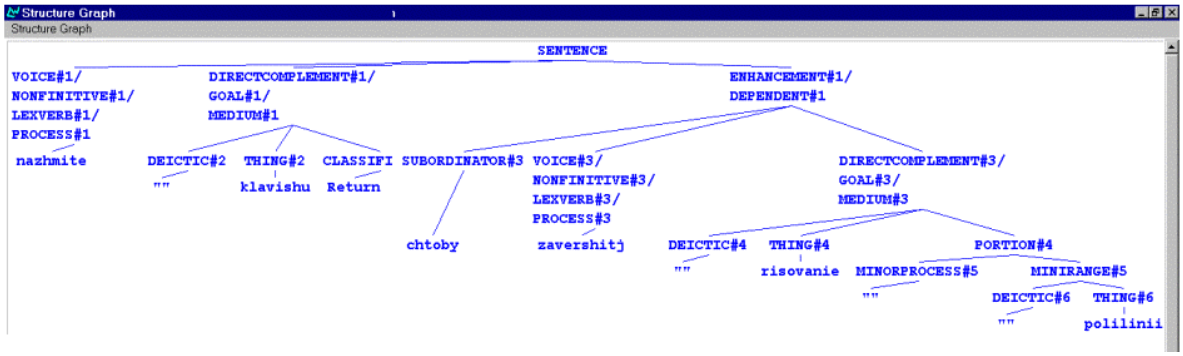


Figure 136: Generated structure for (150), Russian

(151) Hypotactic enhancement: purpose (Bg: finite)

(a) Input SPL:

```
(R / RST-PURPOSE
 :DOMAIN
   (P / DIRECTED-ACTION :LEX NATISNA
   :SPEECHACT IMPERATIVE
   :ASPECT-Q PERFECTIVE-ASPECT
   :ACTEE (D / OBJECT :NAME RETURN))
   :RANGE
   (E / DIRECTED-ACTION :LEX ZAVURSHA
  :ASPECT-Q PERFECTIVE-ASPECT
 :ACTEE
      (P2 / OBJECT :LEX POLILINIA :IDENTIFIABILITY-Q IDENTIFIABLE)))
```

(b) Generated output: (see Figure 137 for the grammatical structure).

Bg: *Натиснете Return, за да завършите полилинията*

Press Return so-that end polyline

Press Return so that you end the polyline.

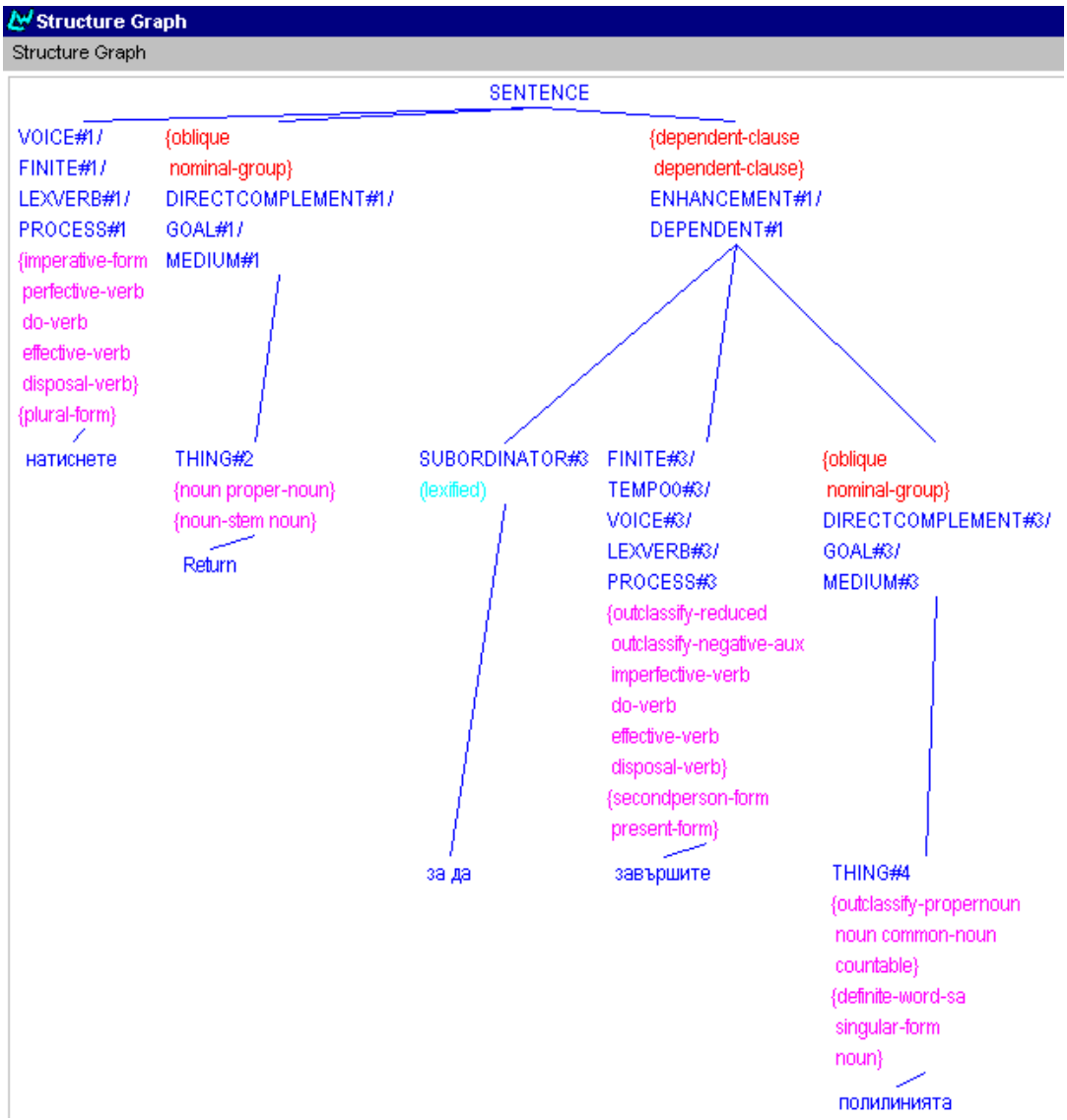


Figure 137: Generated structure for (151), Bulgarian

(152) Hypotactic enhancement: purpose (Cz: finite)

(a) Input SPL:

```
(R / RST-PURPOSE
 :dependent-type-q subordination
 :DOMAIN
   (P / DM::PRESS
   :SPEECHACT IMPERATIVE
   :ACTEE (D / DM::key))
   :RANGE
   (E / DM::END-LINE
   :ACTOR (os / DM:operating-system :lex syste2m
           :identifiability-q identifiable
           :subj-minimal-attention-q nonminimal))
   :ACTEE
      (P2 / DM::POLYLINE :IDENTIFIABILITY-Q IDENTIFIABLE)))
```

(b) Generated output: (see Figure 138 for the grammatical structure).
Cz: *Stiskněte klávesu, aby křivku systém zavřel.*
Press key-Acc so-that polyline system closed
Press a key so that the system would close the polyline

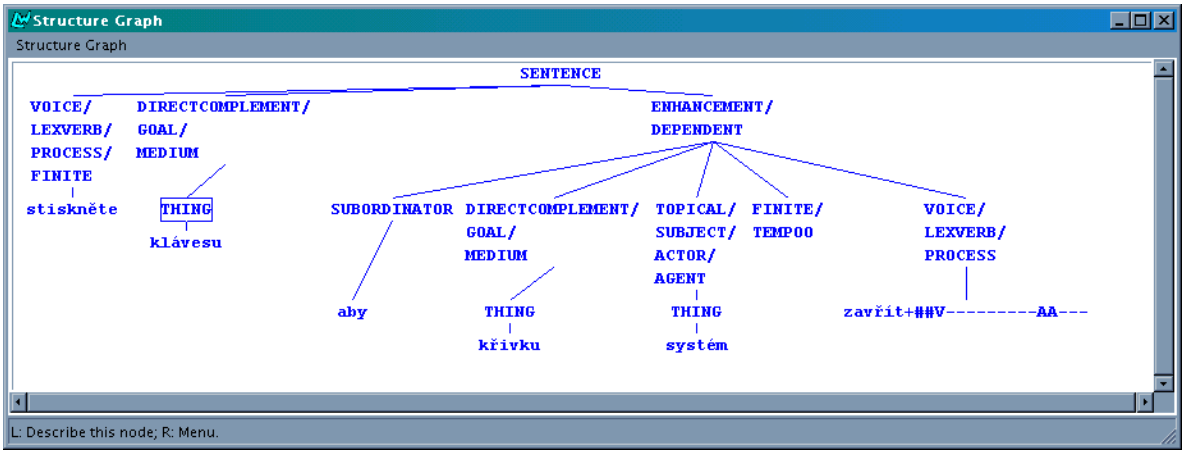


Figure 138: Grammatical structure for (152), Czech

Note that the implementation in the Czech grammar currently has a flaw concerning the proper realisation of the verb form in the dependent clause. The dependent clause is in conditional mood, which involves a Finite element derived from the auxiliary verb "být" (be) and a Voie element corresponding to the content-verb, which is in past participle form. Conditional mood in a suboordinate clause is a new construction in the grammar, and we have not yet entirely completed it.

The next example shows the version of purpose dependent using a prepositional phrase witha nominalisation.

(153) Hypotactic enhancement: purpose (Cz: nominalization)

(a) Input SPL:

```
(R / RST-PURPOSE
 :DOMAIN
   (P / DM::PRESS
   :SPEECHACT IMPERATIVE
   :ACTEE (D / DM::key))
   :RANGE
   (E / DM::END-LINE
   :ACTOR (os / DM:operating-system :lex syste2m
          :identifiability-q identifiable
          :subj-minimal-attention-q nonminimal))
   :ACTEE
      (P2 / DM::POLYLINE :IDENTIFIABILITY-Q IDENTIFIABLE)))
```

(b) Generated output: (see Figure 139for the grammatical structure).
Cz: *Stiskněte klávesu, pro zavření křivky.*
Press key-Acc for closing-Acc polyline-Gen
Press a key closing the polyline.

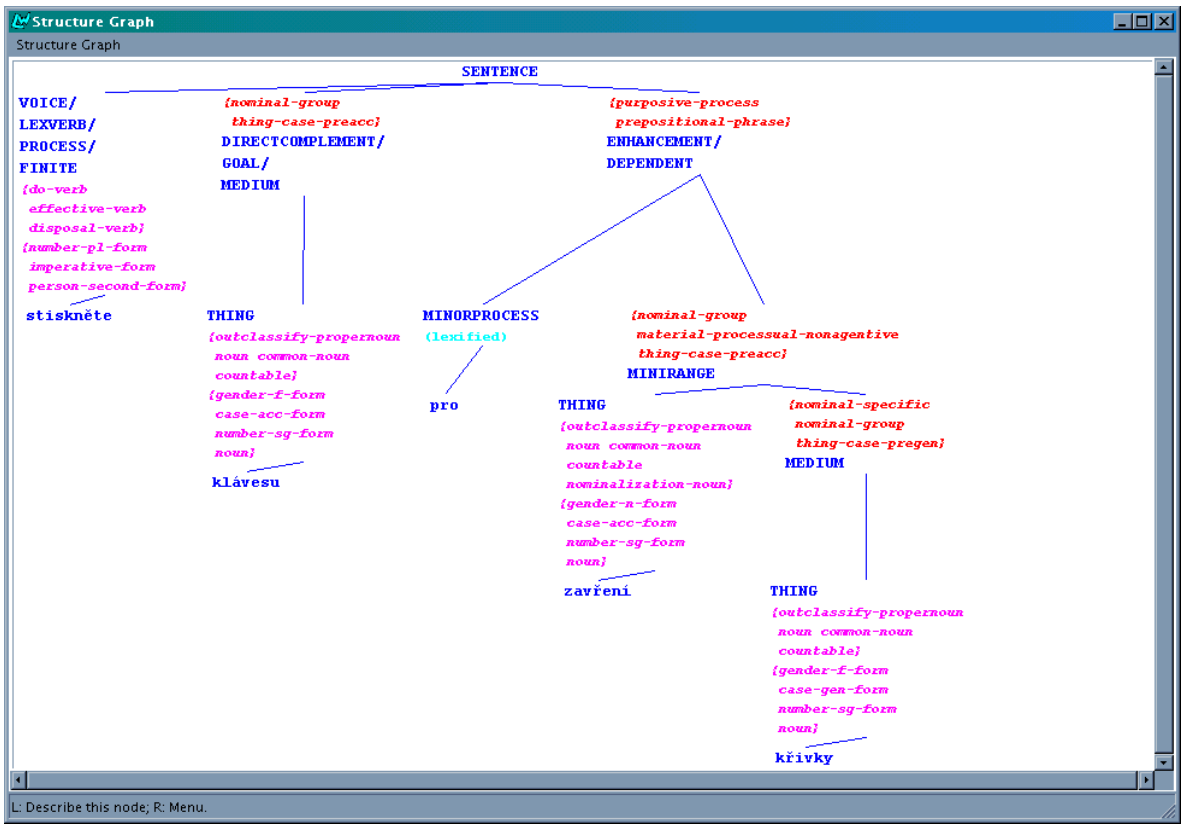


Figure 139: Grammatical structure for (153), Czech

We have so far showed examples of purpose dependent, because the realizations thereof involve the greatest number of variation across the three languages, as well as within. Examples of means and conditional dependent are shown in the next section, where we present the generation input and output for several quite complex sentences, involving a combination of parataxis hypotaxis.

3.9.2.4 Examples of Combined Types of Clause Complexity

In order to demonstrate that our grammars are capable of generating complex sentences involving multiple types of clause complexes, we now show examples involving a combination of parataxis and hypotaxis of different types (namely: purpose, means and conditional) in this section. In each example of clause complex combination, we show an SPL, the generated sentences and the corresponding grammatical structures. The input SPL which is used for generating the sentences is the same for the three languages, except rare cases of specifying lexical items explicitly (in case no corresponding domain model concept has been defined, e.g. USE).

(154) Additive parataxis and hypotactic purpose enhancement

(a) Input SPL:

```
(G7571 / RST-PURPOSE
:DOMAIN
  (D / conjunction
  :domain  (|a19| / DM::ENTER
  :speechact imperative
    :ACTEE (|a21| / DM::COMMAND-LINE :CLASS-ASCRIPTION
              (|a22| / DM::GUI-DRAW :CONTEXTUAL-BOUNDNESS DM::YES)
                     :CONTEXTUAL-BOUNDNESS DM::YES)
    :ACTOR (HEARER / DM::USER
           :IDENTIFIABILITY-Q DM::IDENTIFIABLE))
  :range  (|a19| / DM::click
  :speechact imperative
    :ACTEE (|a21| / DM::button :identifiability-q notidentifiable
   :CLASS-ASCRIPTION
              (|a22| / DM::GUI-DRAW :CONTEXTUAL-BOUNDNESS DM::YES)
                     :CONTEXTUAL-BOUNDNESS DM::YES)
      :ACTOR (HEARER / DM::USER
           :IDENTIFIABILITY-Q DM::IDENTIFIABLE)))
:RANGE
 (|a11| / DM::START-TOOL
    :ACTEE (|a13| / DM::PROGRAM)
    :ACTOR (HEARER / DM::USER :IDENTIFIABILITY-Q DM::IDENTIFIABLE)))
```

(b) Generated output:

Cz: (see Figure 140 for the grammatical structure).
Zadejte řetězec Draw a klikněte na tlačítko Draw pro spuštění programu.
Enter string Draw and click on button Draw for starting program

Bg: (see Figure 141for the grammatical structure).
Въведете командата Draw и щракнете бутона OK,
enter-imp.Pl2 command Draw and click-imp.Pl2 button OK,
за да стартирате програма.
in order to start-Pl2 program

Ru: (see Figure 142 for the grammatical structure).
Введите команду Draw и нажмите кнопку Draw, чтобы запустить программу.

En: *Enter the Draw command line and click the Draw button to start the program.*

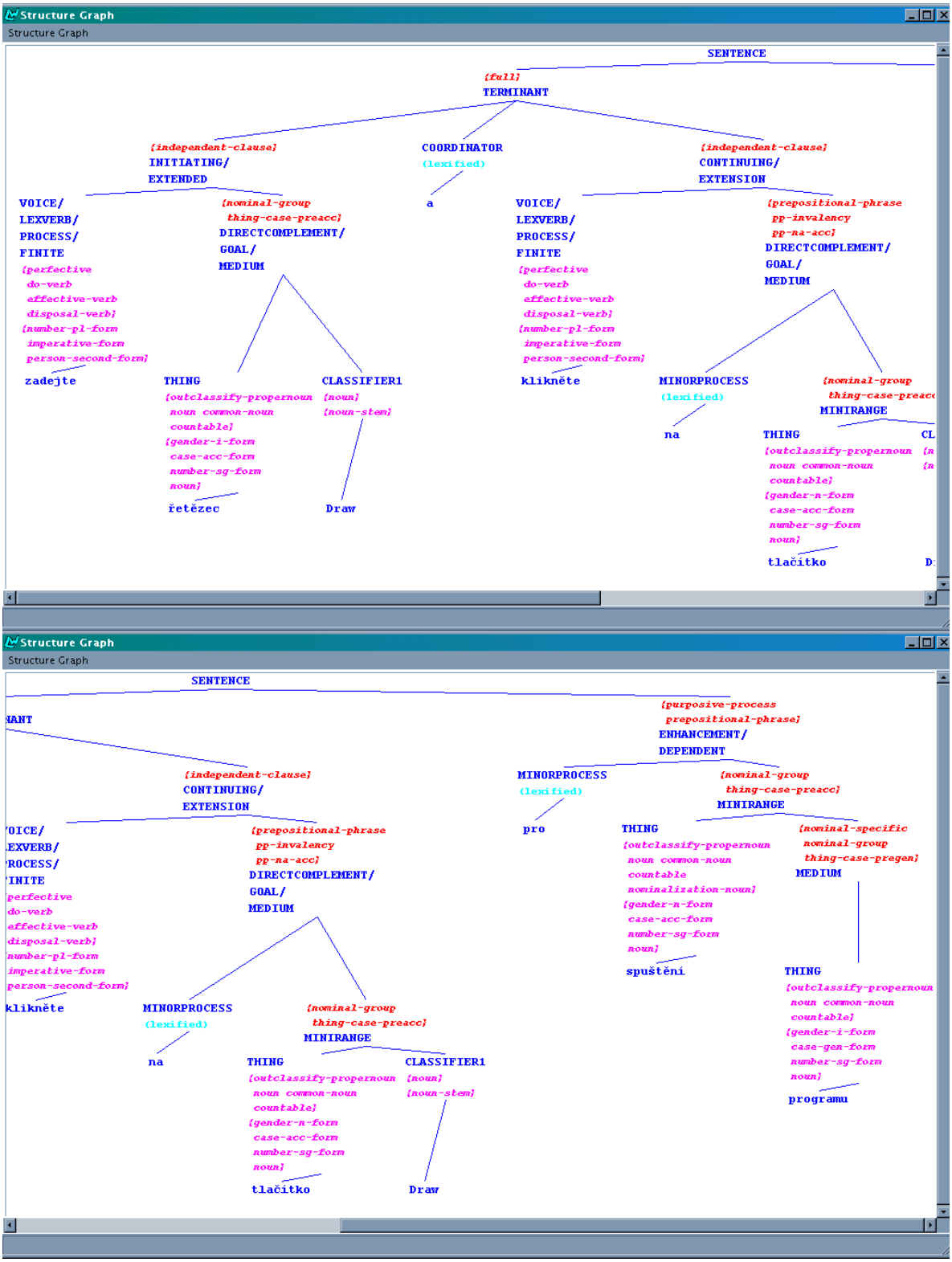


Figure 140: Grammatical structure for (154), Czech

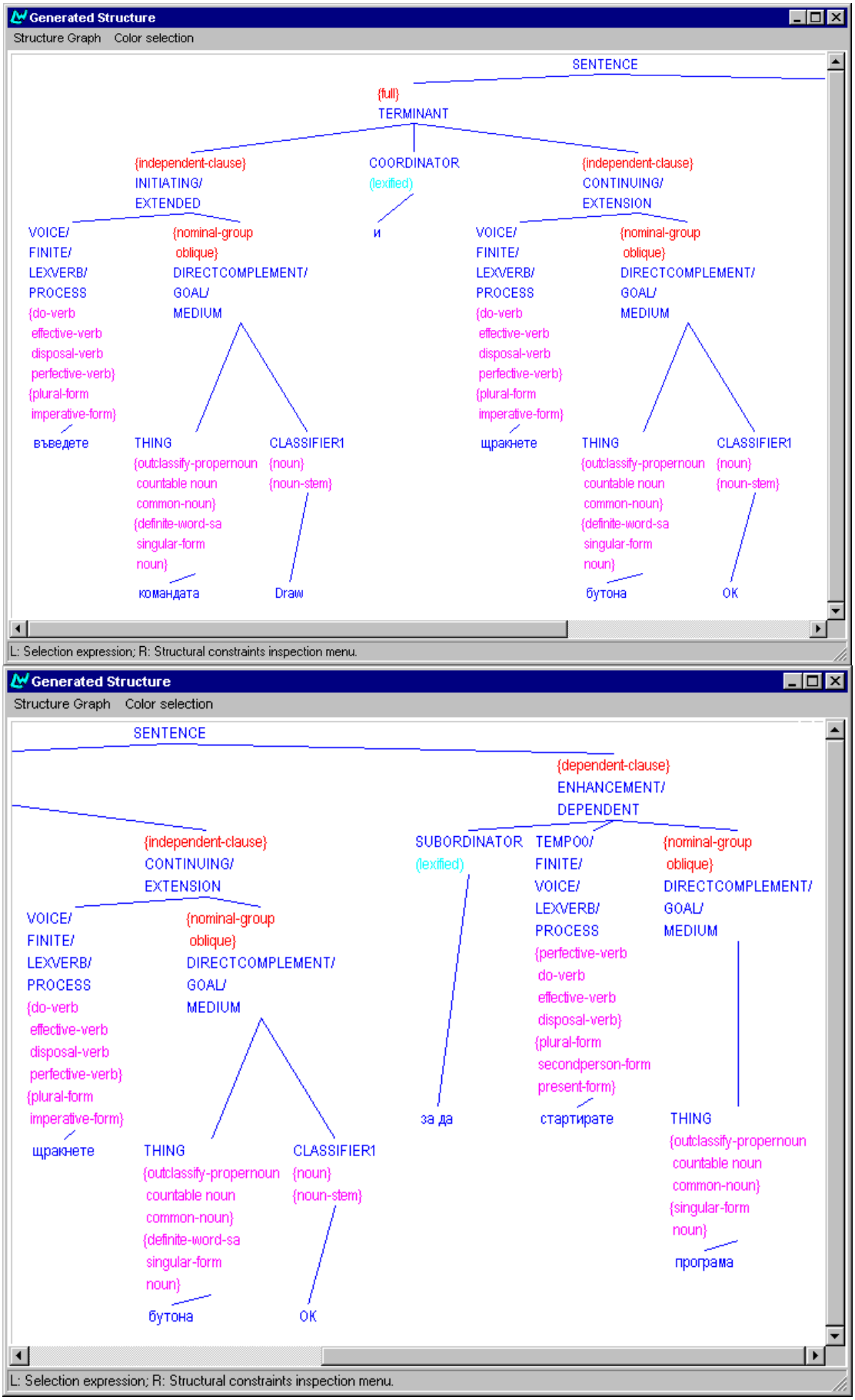


Figure 141: Grammatical structure for (154), Bulgarian

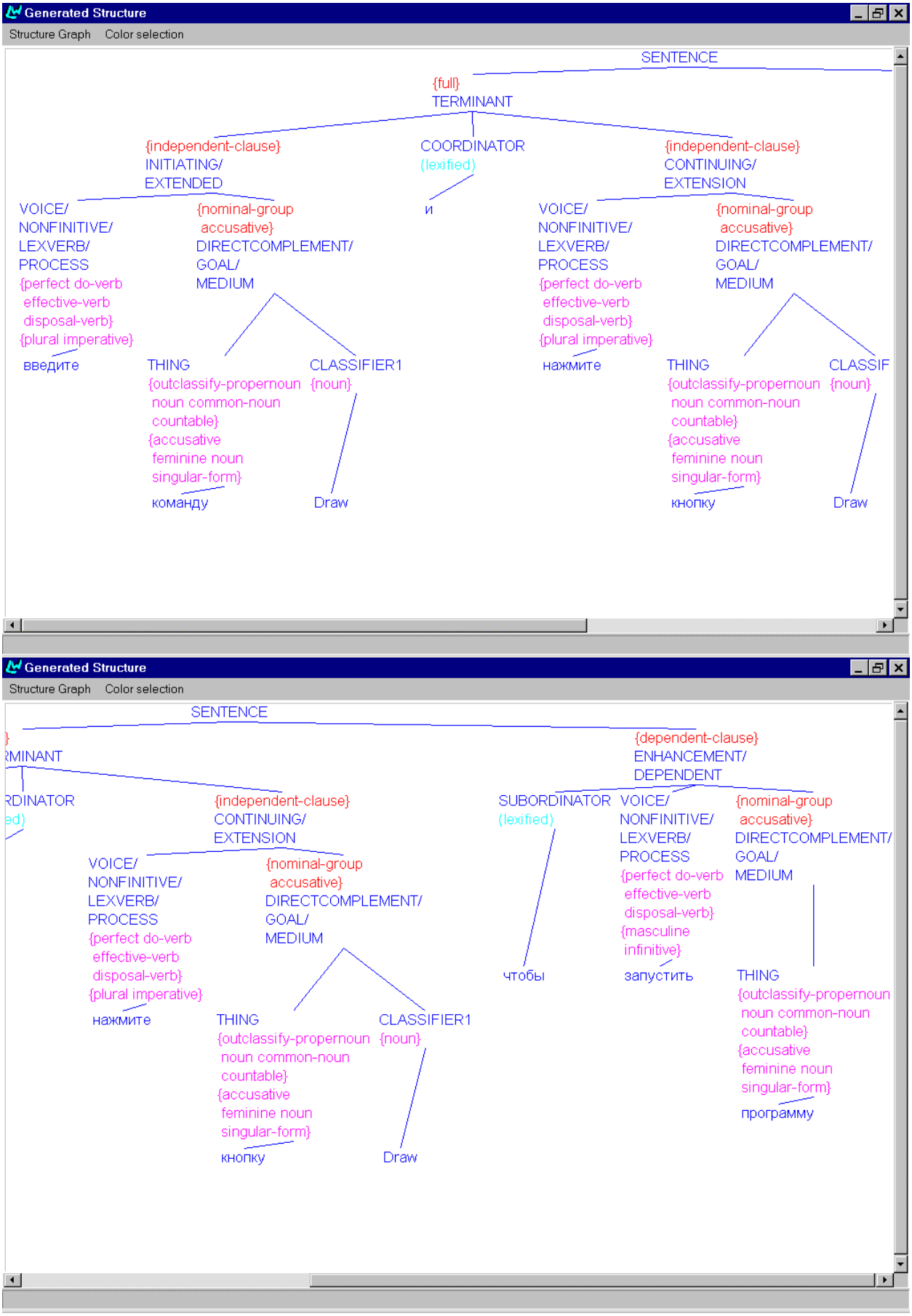


Figure 142: Grammatical structure for (154), Russian

(155) Variation parataxis and hypotactic means enhancement

(a) Input SPL:

```
(G7571 / (RST-Means Manner)
:RANGE
  (D / disjunction
  :domain  (|a19| / DM::ENTER
  :speechact imperative
    :ACTEE (|a21| / DM::COMMAND-LINE :CLASS-ASCRIPTION
              (|a22| / DM::GUI-DRAW :CONTEXTUAL-BOUNDNESS DM::YES)
                     :CONTEXTUAL-BOUNDNESS DM::YES)
    :ACTOR (HEARER / DM::USER
           :IDENTIFIABILITY-Q DM::IDENTIFIABLE))
  :range  (|a19| / DM::click
  :speechact imperative
    :ACTEE (|a21| / DM::button :identifiability-q notidentifiable
   :CLASS-ASCRIPTION
              (|a22| / DM::GUI-DRAW :CONTEXTUAL-BOUNDNESS DM::YES)
                     :CONTEXTUAL-BOUNDNESS DM::YES)
      :ACTOR (HEARER / DM::USER
           :IDENTIFIABILITY-Q DM::IDENTIFIABLE)))
:DOMAIN
 (|a11| / DM::START-TOOL
    :ACTEE (|a13| / DM::PROGRAM)
    :ACTOR (HEARER / DM::USER :IDENTIFIABILITY-Q DM::IDENTIFIABLE)))
```

(b) Generated output:

Cz: (see Figure 143for the grammatical structure).
Spusťte program zadáním řetězce Draw nebo kliknutím na tlačítko Draw.
Start program entering string Draw or clicking on button Draw

Ru: (see Figure 144for the grammatical structure).
Запустите программу, введя команду Draw или нажав кнопку Draw

Bg: (see Figure 145 for the grammatical structure).
Стартирайте програма или като въведете командата Draw
Start-imp.Pl2 program either by enter-Pl2 command Draw
или като щракнете бутона OK.
Or by click-Pl2 button OK
En: *Start the program by entering the Draw command line or by clicking the Draw button.*

In Bulgarian, the verb forms of processes ENTER (въведете) and CLICK (щракнете) are not realized. In Bulgarian, the construction is Finite - “da-construction” in Pl2, and the implementation for this type of hypotaxis is not completed yet.

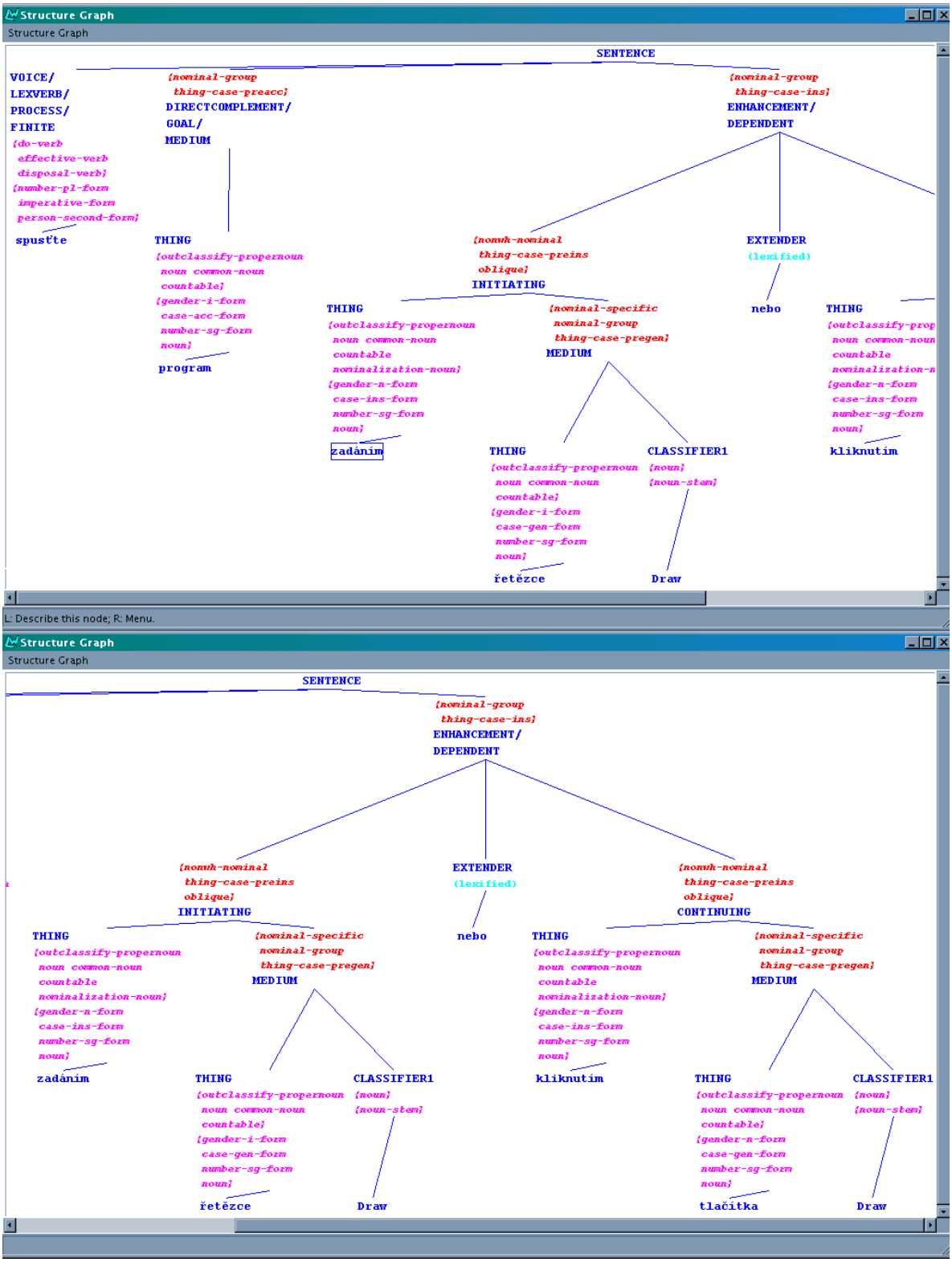


Figure 143: Grammatical structure for (155), Czech

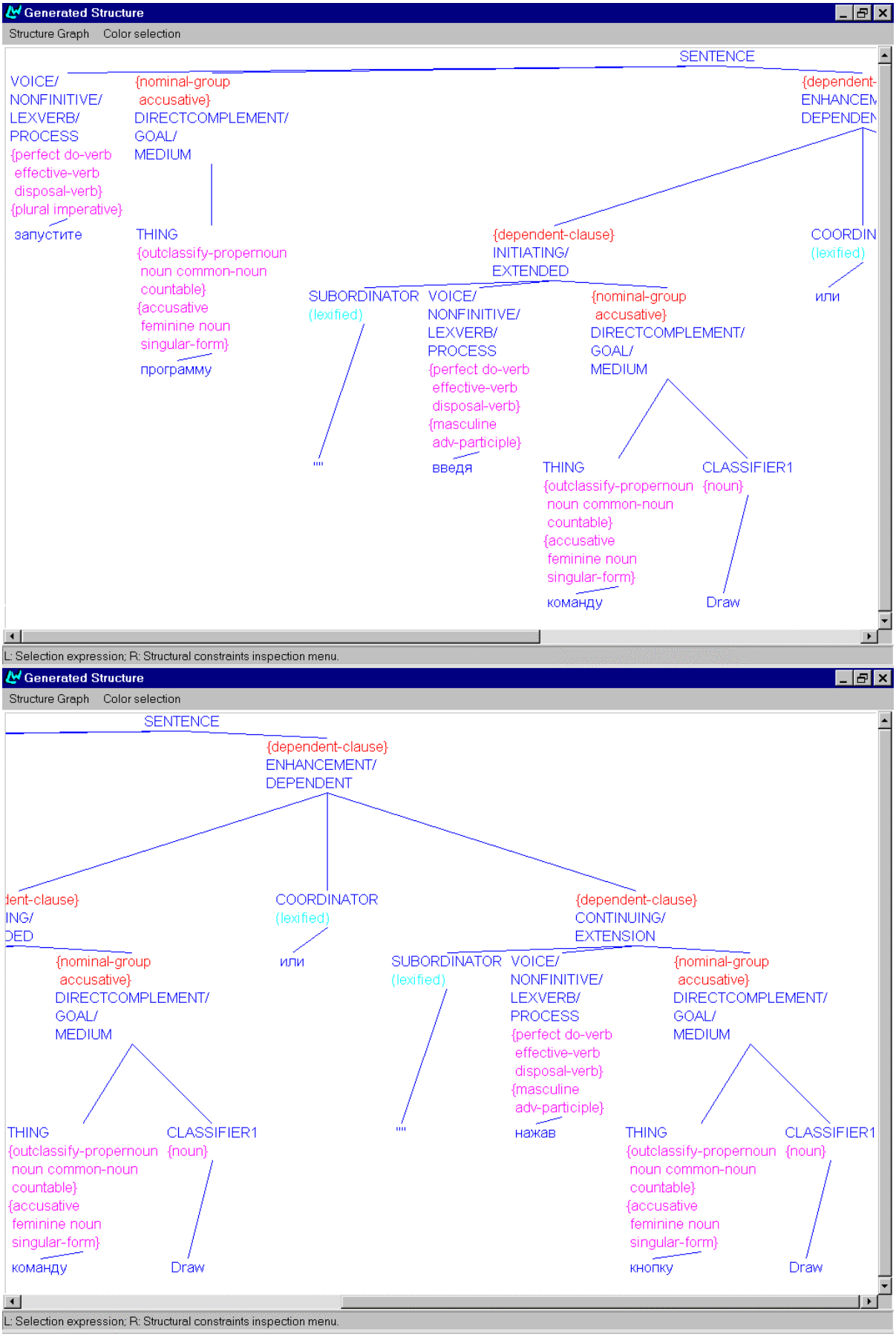


Figure 144: Grammatical structure for (155), Russian

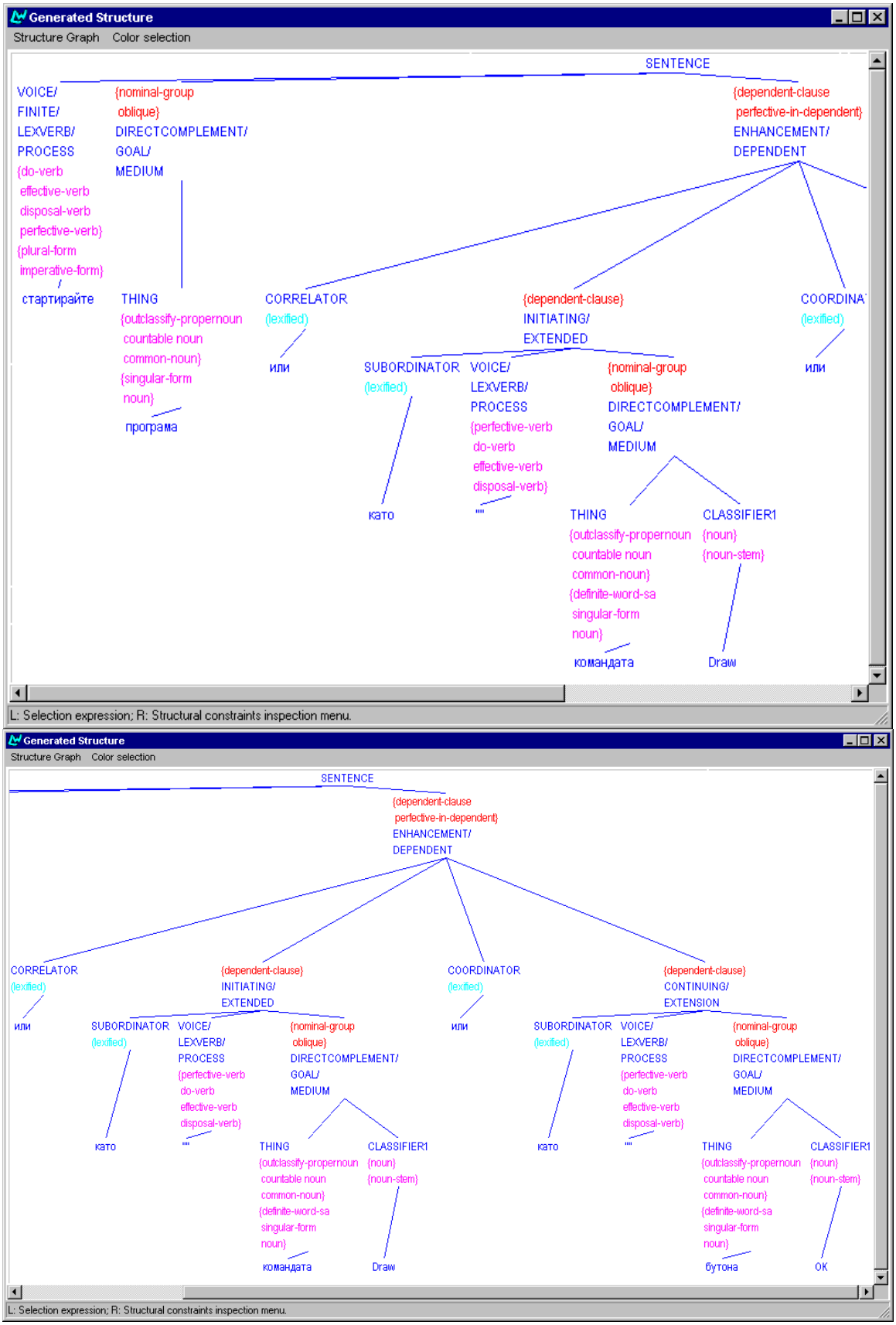


Figure 145: Grammatical structure for (155), Bulgarian

(156) Additive parataxis and hypotactic conditional enhancement

(a) Input SPL:

```
(G7571 / RST-LOGICAL-CONDITION
:DOMAIN
  (D / conjunction
  :domain  (|a19| / DM::ENTER
  :speechact imperative
    :ACTEE (|a21| / DM::COMMAND-LINE :CLASS-ASCRIPTION
              (|a22| / DM::GUI-DRAW :CONTEXTUAL-BOUNDNESS DM::YES)
                     :CONTEXTUAL-BOUNDNESS DM::YES)
    :ACTOR (HEARER / DM::USER
           :IDENTIFIABILITY-Q DM::IDENTIFIABLE))
  :range  (|a19| / DM::click
  :speechact imperative
    :ACTEE (|a21| / DM::button :identifiability-q notidentifiable
   :CLASS-ASCRIPTION
              (|a22| / DM::GUI-DRAW :CONTEXTUAL-BOUNDNESS DM::YES)
                     :CONTEXTUAL-BOUNDNESS DM::YES)
      :ACTOR (HEARER / DM::USER
           :IDENTIFIABILITY-Q DM::IDENTIFIABLE)))
:RANGE
 (|a11| / dispositive-material-action :lex USE
  :ACTEE (|a13| / DM::operating system :LEX ELLIPSISZERO
         :CLASS-ASCRIPTION
       (|a19| / DM::GUI-WINDOWS :CONTEXTUAL-BOUNDNESS DM::YES))
  :ACTOR (HEARER / DM::USER :IDENTIFIABILITY-Q DM::IDENTIFIABLE)))
```

(b) Generated output:

Cz: (see Figure 146 for the grammatical structure).
Pokud používáte Windows, zadejte řetězec Draw a klikněte na tlačítko Draw,
If use-Pl2 Windows enter string Draw and click on button Draw

Ru: (see Figure 147 for the grammatical structure).
Если вы используете Windows, введите команду Draw и нажмите кнопку Draw.
If you use-Pl2 Windows, enter command Draw and click button Draw
En: *Enter the Draw command line and click the Draw button if you use Windows.*

Conditional hypotactic enhancement has not been implemented for Bulgarian.

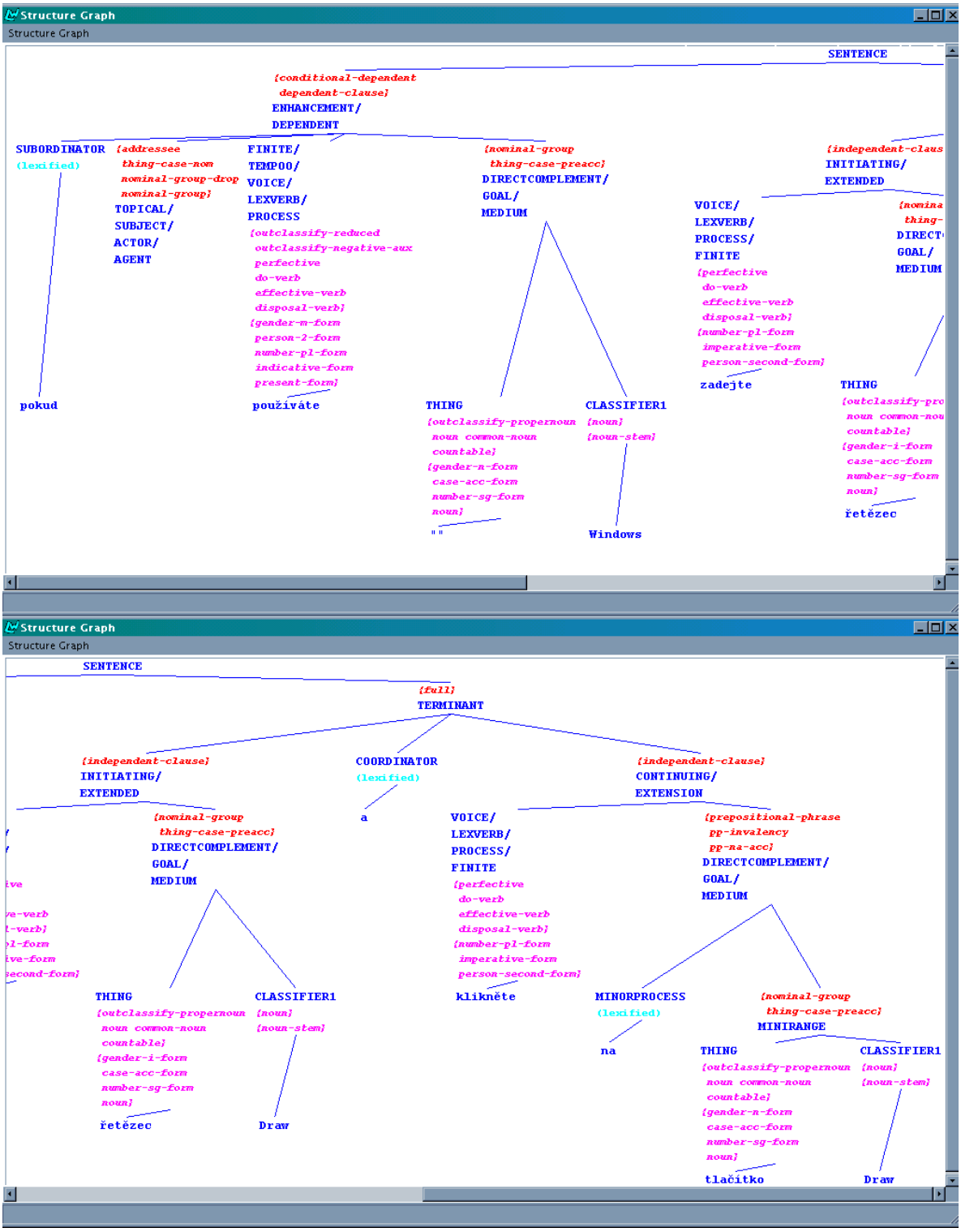


Figure 146: Grammatical structure for (156), Czech

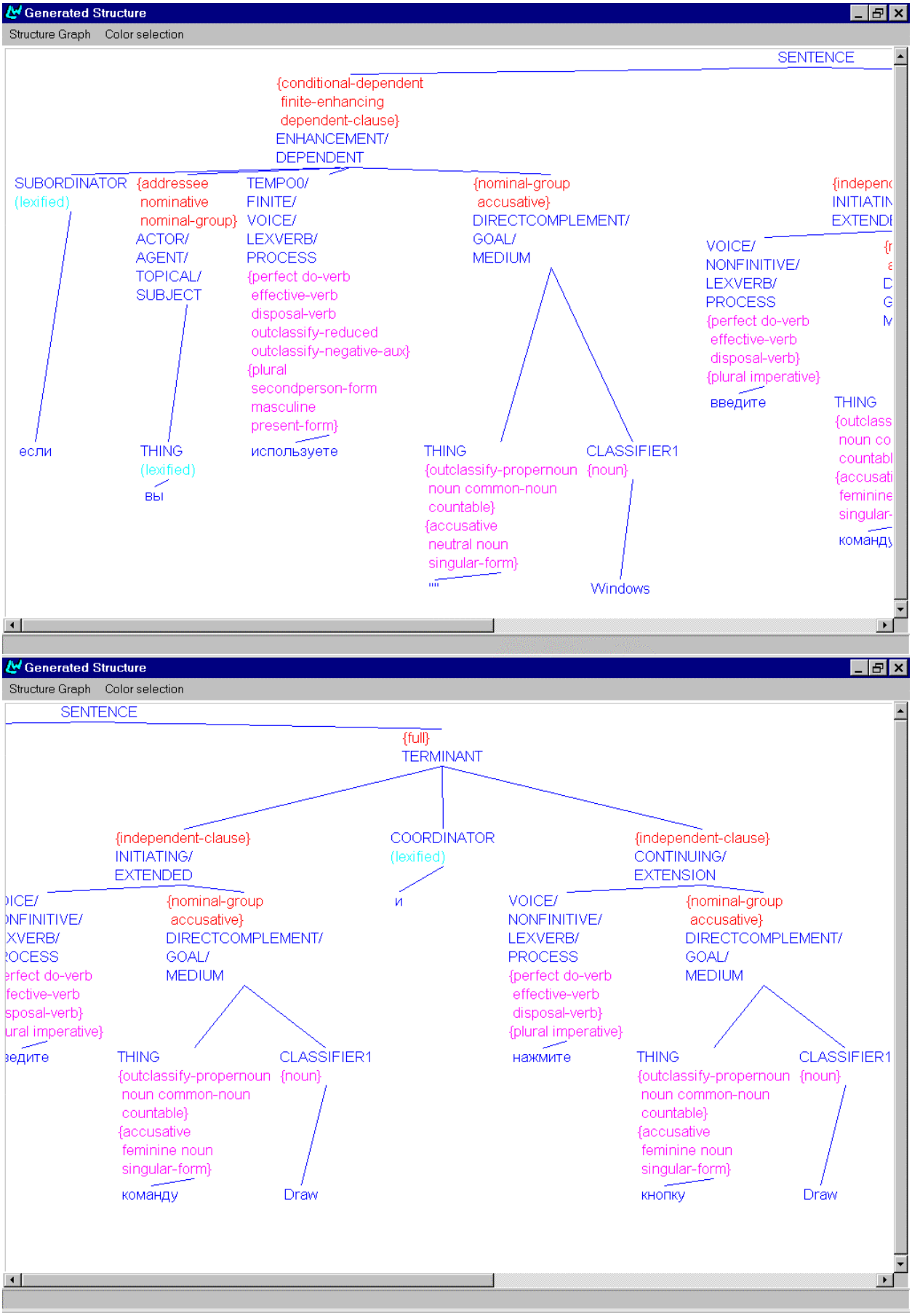


Figure 147: Grammatical structure for (156), Russian

3.10 Summary and conclusions

One of the primary goals of the AGILE project has been the development of lexico-grammatical resources suitable for multilingual generation:

> "The overall goal of AGILE is to make available a *generic* set of tools and *linguistic resources* for generating Czech, Bulgarian and Russian: no such resources are currently available." (Agile, 1997; our emphasis).

The AGILE project thus represents the first attempt ever at a computational account of Bulgarian, Czech and Russian for the purpose of natural language generation. The size and complexity of this task, i.e., the development of generic linguistic resources for three languages from scratch, made us consider the question of which computational linguistic framework to choose for resource development especially carefully.

We were looking for a framework for computational grammar development that would help reduce the size and complexity of the task by

- being easily accessible to the partners in terms of the linguistic concepts it uses;
- being easily adaptable to the languages to be accounted for in the project;
- being easily interfaceable with other components of the application system to be developed in the project.

Given these desiderata, the framework most suitable appeared to be the Komet-Penman MultiLingual system (KPML). KPML met the needs formulated above in the following ways:

- KPML is based on a functional theory of language, Systemic Functional Linguistics (SFL), a linguistic approach that shares many of its characteristics with the Eastern European tradition of functional linguistics. This allowed us, for instance, a straightforward integration of the functional account of word order emanating from the Prague School in the form of a new word order algorithm that enhances KPML's linear ordering techniques (see Section 3.4 of the present report).
- KPML is especially geared towards the development of multilingual grammars, implementing a quite sophisticated model of multilinguality and supporting a maximal sharing of resources among different languages. This allowed us to apply a method of resource development with maximal sharing of development efforts among the partners (see Section 2.6 of the present report).
- KPML is also a well-established tactical generator, which has been used in numerous generation projects in which it has been interfaced with a variety of other computational linguistic components, notably text planners and domain models. Since KPML is built on general linguistic principles, in AGILE it has been possible to use KPML's basic representational means (system networks, choosers and inquiries) for modelling one part of our text planning resources, thus providing a natural connection between the grammar and text structure (see e.g., TEXS2).

Other frameworks and systems for the development of generation resources exist, of course, such as for example the SURGE system mentioned in Chapter 2, but none of them

seemed to be as straightforwardly adaptable to the languages we have been concerned with in AGILE.

Especially attractive in our particular context have been the techniques of *resource sharing* available in KPML. The idea of resource sharing is based on the insight that when contrastively analysing any two languages, one will always detect differences *and* commonalities and that it would be helpful to make use of cross-linguistic commonality in building up new linguistic resources. Accordingly, KPML offers various ways of sharing the computational description of an existing grammar with new languages added to the system. Also, it supports a truly contrastive-linguistic way of building up multilingual resources for new languages, such as merging grammars of different languages into one resource or writing out grammars of individual languages from a common resource. KPML thus appeared the best-suited candidate for handling the complex task of developing generic linguistic resources for Bulgarian, Czech and Russian, the goal of Work Packages 6 and 7.

After a training phase for the partners in using KPML, grammar development in the early stages of the project was carried out by sharing large parts of the English grammar implemented in KPML, Nigel, with Bulgarian, Czech and Russian. This led to a fast prototyping of the Initial Demonstrator. In later stages of the project, we made use of KPML's facilities of sharing resources among Bulgarian, Czech and Russian and organized grammar development in such a way that work on the tasks involved in work packages 6 and 7 was distributed according to linguistic phenomena rather than individual languages. This was only possible because KPML makes available various facilities for multilingual resource management. Even if this strategy was costly in terms of organizational effort, we strongly believe that doing it in any other way would not have allowed the maximal sharing of efforts and resources among the three languages accounted for here and we would not have gained as many insights into the cross-linguistic variation among them (as described in deliverables SPEC2 and IMPL2 and the present report).

The coverage of the grammars we have achieved can be described on four different levels of specificity/genericity (graphically depicted in Figure 148):

1. Coverage of target texts
2. Coverage of the sublanguage of the domain of software instructions
3. Coverage of the sublanguage of general instructional texts
4. Coverage of the general grammar of the three languages

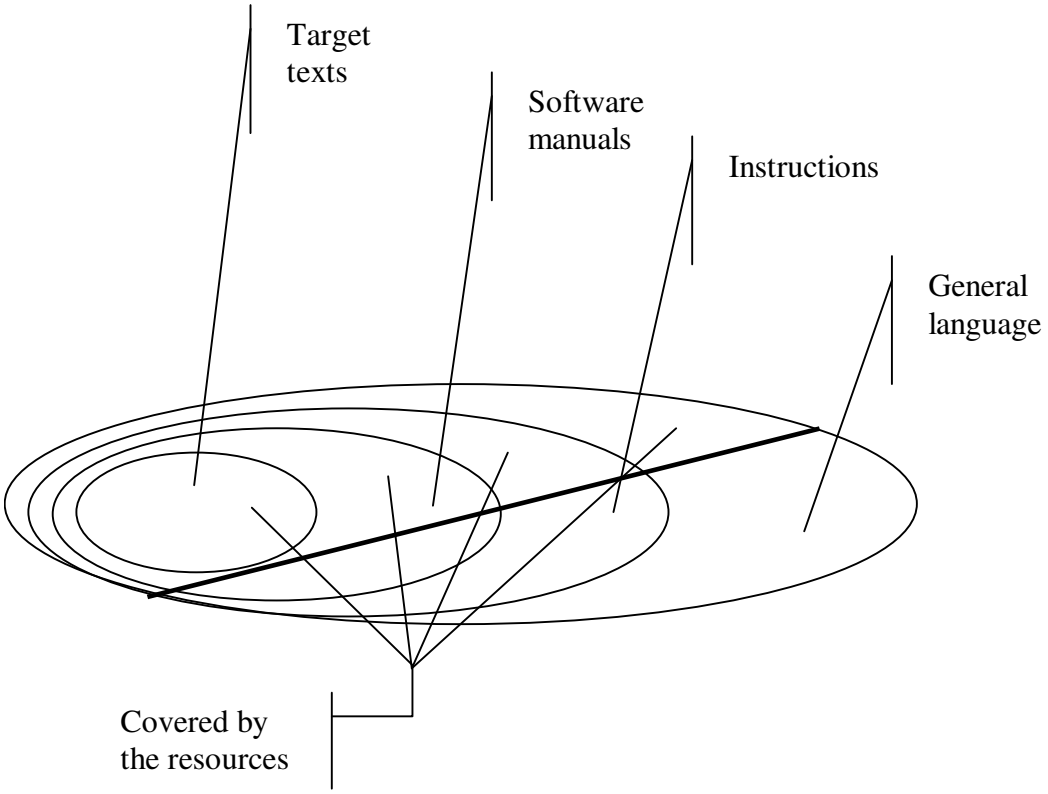


Figure 148 Coverage of lexico-grammatical resources

At the most specific level, the grammars cover the grammatical phenomena of the target texts (for the target texts of the various stages of the project see deliverables IMPL1, TEXS1 and TEXS2). At a slightly less specific level, the grammars cover the grammatical phenomena occurring in the software manuals investigated, in particular in the CAD/CAM domain. At a more general level, we have attempted to include grammatical phenomena that pertain to instructional texts more generally, irrespective of the domain (field). At the most general level, we have made an effort to build up the grammars in such a way that the sublanguage biases that are implied by the other three levels are compensated for.

We have taken the following measures to ensure that coverage at all of these levels was considered at all times:

1. The target texts provided the immediate goal to work towards in all three phases of linguistic description, specification and implementation. SPLs for the sentences in these texts were manually constructed (later, they were automatically produced) and used to test the grammars.
2. A corpus of instructional texts, both from the software domain and from other domains, was analyzed to determine the kinds of phenomena that had to be treated by the grammars.
3. By virtue of the inherent property of Systemic Functional Grammar of looking at whole grammatical paradigms (systems) rather than just fragments, we were forced to consider whole paradigms rather than just parts of them. Also, we took the implementation of Systemic Functional Grammar in KPML, which is organized according to functional regions which group individual grammatical systems into larger classes, as a reference resource for general language coverage

throughout the project (see Table 4 below which is reproduced from Chapter 2 for convenience).

We have thus employed a method of grammatical resource development that is a combination of *instance-oriented* and *system-oriented.* 'System-oriented' means building up a computational resource with a view to the language system as a whole. 'Instance-oriented' means building up a computational resource on the basis of a corpus-analysis of texts of the given register or sublanguage.

		Ideational			Interpersonal	Textual
		Logical		Experiential		
Clause		complexity		Transitivity Circumstance	Mood, Polarity, Attitude, Modality, Tense	Theme, Culmination, Conjunction, Voice
groups/phrases	prep			Minor transitivity		
	nom		Meta-actant	Nominal-type, Epithet, Qualification, Selection	Person, Attitude	Determinat-ion
	adj.		Modi-fier	Quality-type		
	quant		Modi-fier	Quantity-type		
	adv.		Modi-fier	Circumstantial	Comment	Conjunctive
		complexes	simplexes			

Table 4: Functional regions in KPML grammars

The list of linguistic phenomena found relevant for the generation of instructional texts and the list of linguistic phenomena of the target texts was merged into the classification of general language functional regions used in KPML. This list included:

Clause rank

- **nuclear transitivity**: realization of Upper Model processes of dispositive, creative and non-directed material actions, realization of existential relations, as well as of several domain-specific types, like SNAP, SWITCH-MODE and SAVE;
- **circumstances**: different types of spatial relations;
- **aspect** choice;
- **mood** (imperative and declarative, plus conditional for Czech)
- **diathesis (voice)** (active, medio-passive and passive)

- realization of some **modality** options, notably possibility, plus necessity and ability for Czech;
- **word order** constraints;
- **hypotactic clause complexity** (restricted to purpose, means, logical-condition relations);
- **paratactic clause complexity** (conjunction and disjunction);
- **Subject-Finite agreement**
- **Selection of case** for functional elements that are realized by case-bearing grammatical categories (e.g., nominal cases for NGs realizing participants at clause rank)
- **Subject Dropping** in Czech and Bulgarian

Nominal group rank

- **agreement** (between Deictic, Quantity, Status and Thing elements);
- selection of **lexical gender** for lexical items;
- **determination**;
- expressions with cardinal and ordinal numbers and **quantification** constructions of the type "*one of X*"
- **nominalization** of actions;
- **nominal group complexity**: conjunction and disjunction, generalized possessive relations.

Prepositional phrase rank

- language-specific **extensions of prepositional phrase systems** (case government of prepositions)

Other areas of implementation concern punctuation, vocalization in prepositional phrases and pronouns, and interfacing the lexico-grammatical resources with external morphological modules.

Combining the above-mentioned instance-orientation and system-orientation in grammar development, basic general-language grammars *and* sublanguage grammars for CAD/CAM instructional texts have been created. Due to the system-orientation, these grammars are less restricted than sublanguage grammars of a particular domain; and due to the instance-orientation, these grammars are adequate for the domain at hand as well. In Table 5, reproduced below from Chapter 2 for convenience, the functional regions treated focally in Work Packages 6 and 7 are highlighted.

			Ideational		Interpersonal	Textual
			Logical	Experiential		
Clause		**Complexity** (clause and nom; CU)		**Transitivity, Circumstance,** **Aspect** (RRIAI)	**Mood,** Polarity, Attitude, **Modality, Tense** (BAS)	**Theme,** Culmination, **Conjunction,** **Determinat-ion** (CU) **Voice** (RRIAI)
groups/phrases	prep			**Minor transitivity** (RRIAI)		
	nom		Meta-actant	**Nominal-type, Epithet, Qualification, Selection** (CU)	**Person**, Attitude (CU)	**Determinat-ion** (CU)
	adj		Modi-fier	Quality-type		
	quant		Modi-fier	Quantity-type		
	adv		Modi-fier	Circumstantial	Comment	Conjunctive
		complex	simplex			

PLUS: **word order** and **agreement** (both: CU); interfacing external **morphology components** (RRIAI).

Table 5: Functional regions focally treated in WPs 6 & 7

We have thus achieved a fairly good grammatical coverage for Bulgarian, Czech and Russian. The most important thing to note here again is that the basic design of these grammatical resources is not sublanguage-specific, but based on general linguistic principles. Therefore, we expect that these grammars can easily be re-used in other domains and applications. Given the paradigmatic orientation of systemic functional grammars, a computational grammar that is organized in this way, even if it does not implement some system (paradigms) exhaustively, can be straightforwardly extended. Take for example the MOOD system, which distinguishes between indicative and imperative at the primary level and between declarative and interrogative for indicative at the secondary level. In AGILE, the options declarative and imperative have been implemented to the highest degree of delicacy so that clauses in declarative and imperative mood in all three languages can be

generated. The interrogative option is included in the system of MOOD, but is simply not further detailed out at the moment. For a complete implementation of MOOD, the only thing that would need to be done is to specify further possible subclassifications of interrogative for the three languages and their structural realizations.

Finally, the KPML system itself has been further developed in the course of the project as an immediate reaction to the requirements of Slavonic languages. First, external morphology modules can now be straightforwardly interfaced with KPML. Second, there is a new mechanism for syntactic agreement. And third, the flexible word order of Slavonic languages made it necessary to implement a new, more versatile word ordering algorithm which is made available with the present deliverable. This algorithm is mainly inspired by the notion of communicative dynamism developed within the Prague Schools and turned out to be quite compatible with existing systemic functional notions implemented in KPML in the textual metafunction.

In the last steps towards an integrated system (Final Prototype) we will have to test some of the implementations described here for one language for the other languages (e.g., the spatial location implementation described in Section 3.1.2 for Russian still has to be tested for Czech, or the textual conjunction specification described in Section 3.6 still has to be tested for Russian and Bulgarian). In the remainder of the project, we are going to fine-tune the grammar implementations of Bulgarian, Czech and run another round of multilingual grammar testing. Also, the individual grammars will be prepared for release as self-contained components equipped with test suites (sets of SPLs) that document their coverage.

References

Agile Project (1997) Automatic generation of instructions in languages of Eastern Europe. Technical Annex, Commission of the European Community, Brussels, DG XIII, 1997.

AutoCAD (1995) *AutoCAD Release 13. User's guide,* chapter 2. Autodesk Co.

Bateman J.A., Matthiessen C.M.I.M., Nanri K, Zeng L. (1991) The re-use of linguistic resources across languages in multilingual generation components. In: Proceedings of the 1991 International Joint Conference on Artificial Intelligence, Sydney, Australia, volume 2, (pp. 966—971). Morgan Kaufmann Publishers.

Bateman J. A. (1992) Grammar, systemic. In: S. Shapiro, editor, *Encyclopedia of Artificial Intelligence, Second Edition*, pages 583 -- 592. John Wiley and Sons, Inc., 1992.

Bateman J. A. (1995) Basic technology for multilingual theory and practise: The KPML development environment. In R. Kittredge, editor, *Proceedings of the IJCAI '95 Workshop on Multilingual Generation*, Montreal, Quebec, August 1995.

Bateman J. A. (1997a) KPML Development Environment: multilingual linguistic resource development and sentence generation. German National Center for Information Technology (GMD), Institute for Integrated Publication and Information Systems (IPSI), Darmstadt, Germany, March 1997. (Release 1.0).

Bateman J. A. (1997b) Enabling technology for multilingual natural language generation: the KPML environment. *Natural Language Engineering* 1(1):1-42, 1997.

Bateman J.A. (1999) The KPML multilingual natural language generation system, development environment and tools. At: http://purl.org/net/kpml.

Bateman J. A., Th.Kamps, J. Kleinz, K. Reichenberger (1998) Communicative goal-driven NL generation and data-driven graphics generation: An architectural synthesis for multimedia page generation. *In Proceedings of the International Workshop on Natural Language Generation*, Niagara-on-the-lake, Canada, 1998.

Bateman, J. A., R. T. Kasper, J. D. Moore, and R. A. Whitney (1990). A general organization of knowledge for natural language processing: the PENMAN Upper Model. Technical report, USC/Information Sciences Institute, Marina del Rey, California, 1990.

Bateman J. A., E. A. Maier, E. Teich, and L.Wanner (1991) Towards an architecture for situated text generation. In *Proceedings of the International Conference on Current Issues in Computational Linguistics*, pages 336--349, Penang, Malaysia, 1991.

Bateman, J. A. and Teich, Elke. (1995) Selective information presentation in an integrated publication system: an application of genre-driven text generation. In: *Information Processing and Management*, 31(5): 753--768. Elsevier Science Ltd. 1995.

Bémová A. et al. (1995): Linguistic problems of Czech, Project Peco 2924, Charles University Prague.

Comrie, B. (1976) *Aspect.* Cambridge University Press, Cambridge.

Daneš F. (1974) *Papers on Functional Sentence Perspective:* Academia, Prague, 1974.

Dik S. (1978) *Functional Grammar.* North Holland, Amsterdam, 1978.

DiMarco C., G. Hirst, L. Wanner, and J.Wilkinson (1995) Healthdoc: Customizing patient information and health education by medical condition and personal characteristics.In A. Cawsey, editor, *Proceedings of the Workshop on Patient Education*. University of Glasgow, Glasgow, 1995.

Elhadad M. and J. Robin. (1996) An overview of SURGE: A reusable comprehensive syntactic realization component. Technical Report 96-03, Dept of Mathematics and Computer Science, Ben Gurion University, Beer´Sheva, Israel, 1996.

Elhadad M. (1990) Types in functional unification grammars. *In Proceedings of the 28th. Annual Meeting of the Association for Computational Linguistics*, pages 157 --164. Association for Computational Linguistics, 1990.

Fawcett R. P. and G. H. Tucker (1989) Prototype generators 1 and 2. Technical Report COMMUNAL Report Number 10, Computational Linguistics Unit, University of Wales College of Cardiff, 1989.

Firbas J. (1966) On defining the theme in functional sentence analysis. *Travaux Linquistique de Prague*, 1:267--280, 1966.

Grosz Barbara, Joshi Aravind and Weinstein Scott [1995]. Centering: a framework for modeling the local coherence of discourse. In: Computational linguistics, Vol. 21, No. 2, pp. 203-226.

Grosz Barbara, and Sidner Candy L. [1986]. Attention, intentions and the structure of discourse. In: Computational linguistics, Vol. 12: pp. 175-204.

Firbas Jan [1992] *Functional sentence perspective in written and spoken communication.* Cambridge: Cambridge University Press.

Hajičová Eva [1993]. Issues of sentence structure and discourse patterns. Theoretical and computational linguistics, Vol. 2. Prague: Charles University.

Hajičová, Eva, Kruijff-Korbayová, Ivana, Sgall, Petr. [1999]. Prague Dependency Treebenk. In the Proceedings of the Conference on Text, Speech and Dialogue (TSD'99), Mariánské Lázně, Czech Republic. Matoušek et al. (eds.). Springer-Verlag.

Halliday M.A.K. (1973). *Explorations in the Functions of Language*. Edward Arnold, London, 1973.

Halliday, M.A.K. (1978) *Language as social semiotic: the social interpretation of language and meaning.* London: Edward Arnold.

Halliday, M.A.K. (1985), (1994, 2nd ed.) *Introduction to Functional Grammar,* London: Edward Arnold.

Halliday M.A.K. and C.M.I.M. Matthiessen (1999) *Construing experience through meaning: a language-based approach to cognition*. Cassell Academic, London, 1999.

Halliday M.A.K., A. McIntosh, and P. Strevens (1964) *The linguistic sciences and language teaching.* Longman, London, 1964.

Hartley, A., Paris, C. (1996) Two sources of control over the generation of software instructions. In: Proc.of the ACL Annual Meeting, Santa Cruz, June 1996. pp. 192-199.

Hartley A. F. and C. Paris (1997) Multilingual document production: from support for translating to support for authoring. *Machine Translation* (Special Issue on New Tools for Human Translators), 12(1-2):109--129, 1997.

Henschel R. and J. A. Bateman (1994) The merged upper model: a linguistic ontology for German and English. In *Proceedings of COLING '94*, Kyoto, Japan, August 1994.

Hjelmslev L. (1943) *Omkring sprogteoriens grundlaeggelse*. Akademisk Forlag, Kopenhavn, 1943.

Kasper R. T. (1989) A flexible interface for linking applications to PENMAN's sentence generator. In *Proceedings of the DARPA Workshop on Speech and Natural Language*, 1989. Available from USC/Information Sciences Institute, Marina del Rey, CA.

Kopečný František (1962): Základy české skladby, SPN Praha

Kruijff G.J.M.and Kruijff-Korbayová I. Text Structuring in a Multilingual System for Generation of Instructions. In: Proceedings of the Second Workshop on Text, Speech and Dialogue, Mariánské Lázně, September 1999. Springer and Verlag. (AGILE project)

Kruijff-Korbayová I., Kruijff G.J.M. and Bateman John. Contextually Appropriate Ordering of Nominal Expressions. Contribution to a volume based on the ESSLLLI¨"99 Workshop on Generating Nominal Expressions, Utrecht, August 1999. Kees van Deemter and Rodger Kibble (eds.).

(http://Kwetal.ms.mff.cuni.cz/~korbay/Public/agile-ordering-booksubm.ps).

Mann W. C. and C. M.I.M. Matthiessen (1983) Nigel: A systemic grammar for text generation. Technical Report RR-83-105, USC/Information Sciences Institute, February 1983. (Also appears in R. Freedle, ed.itor, *Systemic Perspectives on Discourse: Volume I*, published by Ablex)

Mann W. C. (1983) An overview of the PENMAN text generation system. In *Proceedings of the National Conference on Artificial Intelligence*, pages 261--265. AAAI, August 1983.

Mann W. & Matthiessen C.M.I.M.. (1985) A demonstration of the Nigel text generation computer program. In James D. Benson and William S. Greaves, editors, *Systemic Perspectives on Discourse,* vol 1. Ablex, Norwood, N.J.

Matthiessen C. M.I.M., I. Kobayashi, L. Zeng, and M. Cross (1995) Generating multimodal presentations: resources and processes. *In Proceedings of the Australian Conference on Artificial Intelligence*}, Canberra, 1995.

Matthiessen C. M.I.M., L. Zeng, M. Cross, I. Kobayashi, K. Teruya, C. Wu (1998) Communicative goal-driven NL generation and data-driven graphics generation: An architecture for multimedia page generation. *In Proceedings of the International Workshop on Natural Language Generation*, Niagara-on-the-lake, Canada, 1998.

Matthiessen C. M.I.M. (1988) Semantics for a systemic grammar: The chooser and inquiry framework. In M. Cummings, J. D. Benson, and W. S. Greaves, editors, *Systemic Perspectives on Discourse*. John Benjamins, Amsterdam,1988.

McKeown K., M. Elhadad, Y. Fukumoto, J. Lim, C. Lombardi, J. Robin, and F. Smadja (1990) Natural Language Generation in COMET. In R. Dale, C. Mellish, and M. Zock, editors, *Current Research in Natural Language Generation*, pages 103 -- 139. Academic Press, London, 1990.

Martin, James R. 1992. *English Text: Systems and Structure*. Benjamins, Amsterdam.

Mathesius Vilém [1939] O takzvaném aktuálním členění větném [On the so-called functional sentence perspective]. Slovo a slovesnost 5: 171-174.

Oliva Karel, and Tania Avgustinova. Wackernagel position and related phenomena in Czech. In Wiener Slavistiches Jahrbuch, pages 21--42. Verlag der Oesterreichischen Akademie der Wissenchaften, Wien, 1995.

Paris C., K. Vander Linden, M. Fischer, A. F. Hartley, L. Pemberton, R. Power, and D. Scott (1995) A Support Tool for Writing Multilingual Instructions. *In Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI) 1995*, pages 1398 -- 1404, Montreal, Canada, 1995.

Penman Project (1989). PENMMAN documentation: the Primer, the User Guide, the Reference Manual, and the Nigel manual. Technical report, USC/Information Sciences Institute, Marina del Rey, California, 1989.

Pollard C. and I. A. Sag (1987) *Information-based syntax and semantics*: *volume 1*. Chicago University Press, Chicago, 1987.

Pollard C. and I. A. Sag (1993*) Information-based syntax and semantics: volume 2*. Chicago University Press, Chicago, 1992.

Roesner D. and M. Stede (1992) TECHDOC: a system for the automatic production of multilingual technical documents. Technical Report FAW-TR-92021, Forschungsinstitut fuer anwendungsorientierte Wissenverarbeitung (FAW) an der Universitaet Ulm, Ulm, Germany, September 1992.

Roesner D. and M. Stede (1994) Generating multilingual documents from a knowledge base: the TECHDOC project. *In Proceedings of the 15th. International Conference onComputational Linguistics* (Coling 94), volume I, pages 339 -- 346, Kyoto, Japan, 1994.

Sgall P., E. Hajicova, and J.Panevova The Meaning of the Sentence in Its Semantic and Pragmatic Aspects. Reidel Publishing Company, Dordrecht, 1986.

Steedman, Mark. [1996]. *Surface Structure and Interpretation*. M.I.T. Press. Cambbridge, MA.

Strube, Michael. [1999]. Never look back. In the Proceedings of the Annual Meeting of the Association of Computational Linguistcs (ACL'99). ACL.

Teich E., L. Degand, and J.A. Bateman (1996) Multilingual textuality: Experiences from multilingual text generation. In G. Adorni and M. Zock, editors*, Trends in Natural Language Generation: an artificial intelligence perspective*, number 1036 in Lecture Notes in Artificial Intelligence, pages 331--349. Springer-Verlag, Berlin, New York, 1996.

Teich E. (1995) Towards a methodology for the construction of multilingual resources for multilingual generation. In R. Kittredge, editor, *Proceedings of the IJCAI '95 Workshop on Multilingual Generation*, Montreal, Quebec, August 1995.

Teich E. (1999) Systemic Functional Grammar in Natural Language Generation: Linguistic Description and Computational Representation. Cassell Academic, London, 1999.

Vendler, Z. (1967) Linguistics in Philosophy. Cornell University Press, Ithaca.

Wanner L. and E. H. Hovy (1996) The HealthDoc sentence planner. *In Proceedings of the 8th International Workshop on Natural Language Generation*, Herstmonceux, UK, June 1996.

Yang G., K. F. McCoy, and K. Vajay-Shanker (1991) From functional specification to syntactic structures: Systemic Grammar and Tree Adjoining Grammar. *Computational Intelligence*, 7(4):207 --219, 1991.

Зализняк А.А. (1977) *Грамматический словарь русского языка.* Москва: Русский язык.

Мельчук, И.А. (1974) Опыт теории лингвистических моделей "Смысл-Текст". Москва: Наука.

Падучева Е. В. (1996) *Семантические исследования* (Семантика времени и вида в русском языке. Семантика нарратива). Школа "Языки русской культуры", Москва.

Пешковский А. М. (1934) *Русский синтаксис в научном освещении.* Государственное учебно-педагогическое издательство, Москва.

Храковский, В.С., Володин, А.П. (1986) *Семантика и типология императива.* Русский императив. Наука, Ленинград.

AGILE deliverables referred in the present deliverable:

[CORP] Anthony Hartley, Danail Dochev, Nevena Gromova, Kamenka Staykova, Alla Bemova, Aleksandr Rosen, Jiri Trojanek, Elena Sokolova. *Tagging and analysis of instructional texts in the software domain.* June 1998. The deliverable for the WP3 of AGILE project PL961104.

[LSPEC2] Geert-Jan M. Kruijff, John Bateman, Alla Bémová, Danail Dochev, Ivana Kruijff-Korbayová, Serge Sharoff, Hana Skoumalová, Lena Sokolova, Kamena Stoikova, Elke Teich, Jirí Trojánek. *Modelling Lexical Resources in KPML for Generating Instructions in Slavonic Languages.* October 1998. The deliverable for the WP4-2 of AGILE project PL961104.

[SPEC2] Elena Andonova, John Bateman, Nevena Gromova, Anthony Hartley, Geert-Jan M. Kruijff, Ivana Kruijff-Korbayová, Serge Sharoff, Hana Skoumalová, Lena Sokolova, Kamenka Staykova, Elke Teich, *Formal specification of extended grammar models.* February 1999. The deliverable for the WP6-2 of AGILE project PL961104.

[TEXS2] I. Kruijff-Korbayová, G.J.M. Kruijff, J. Bateman, D. Dochev, N. Gromova, A. Hartley, E. Teich, S. Sharoff, L. Sokolova, and K. Staykova, *Specification of elaborated text structures*. AGILE deliverable 5.2, April 1999. (Deliverable comprises TEXS2-Cz, TEXS2-Bg, TEXS2-Ru)