AGILE

Automatic Generation of Instructions in Languages of Eastern Europe

Title ***Implementation of grammatical resources for the intermediate demonstrator***

Authors
Elena Andonova
John Bateman
Silvia Hansen
Jiri Hana
Ivana Kruijff-Korbayová
Geert-Jan Kruijff
Kamenka Staykova
Elena Sokolova
Elke Teich

Deliverable *IMPL2-BAS, IMPL2-CU, IMPL2-RU*

Status *Final*

Availability *Public*

Date *July 1999*

INCO COPERNICUS PL961104

Abstract:

This document comprises the deliverables IMPL2-BAS, IMPL2-CU and IMPL2-RU of work package 7, task 7.2 of the AGILE project. We present the implementation of the grammatical resources for Bulgarian, Czech and Russian that are necessary to generate texts with stylistic variation. The resource implementation has followed the linguistic specification of a broader range of phenomena as described in deliverables SPEC2-BAS, SPEC2-CU and SPEC2-RU, so that a more general coverage has been achieved.

Table of Contents

List of Figures

1. Introduction

This document comprises the deliverables IMPL2-BAS, IMPL2-CU and IMPL2-RU of work package 7, task 7.2 of the AGILE project. We present the implementation of the grammatical resources for Bulgarian, Czech and Russian necessary to generate texts with stylistic variation in the CAD/CAM domain. The variation covered mainly concerns the parameters of explicitness and personal/impersonal (cf. deliverable TXSM; Kruijff-Korbayova *et al.*, 1999). The target text we agreed upon for the intermediate demonstrator (henceforth: ImD) are given in the appendix.

To counteract a sublanguage-bias, the resource implementation has followed the linguistic specification of a broader range of phenomena as described in deliverables SPEC2-BAS, SPEC2-CU and SPEC2-RU (Andonova *et al.*, 1999). Thus, a broader coverage of linguistic phenomena has been achieved. The phenomena in focus have been the following:

- clause complexity, transitivity, mood, voice, tense, aspect, theme (clause level)
- minor transitivity (prepositional phrase level)
- determination (nominal group and clause levels)

This has resulted in a coverage of regions of the grammars of Bulgarian, Czech and Russian that can be synoptically displayed by a table as follows.[1]

		Ideational			Interpersonal	Textual
		Logical		Experiential		
Clause		COMPLEXITY		**Transitivity, Circumstance, Voice, Tense, Aspect**	**Mood**, Polarity, Attitude, Modality	**Theme**, Culmination, Conjunction, **Determination**
groups/phrases	**prep**			**Minor transitivity**		
	nom		Classif-ication	Noun-type, Epithet, Qualification	Person, Attitude	**Determination**
	adj		Modifi-cation	Quality-type		
	quant		Modifi-cation	Quantity-type		
	adv		Modifi-cation	Circumstantial-type	Comment	Conjunctive
		complexes	simplexes			

Each of these areas is described in a separate section (see Section 2 below).

[1] The regions in focus are given in bold face; other regions that have been treated as well, however not systematically, in order to be able to generate the target texts, are given in grey shade. Note that determination is located in the nominal group as well as the clause because in Slavic languages determination is predominantly, but not solely, reflected by word order in the clause.

In contradistinction to the first deliverable in this work package (IMPL1-BAS, IMPL1-CU, IMPL1-Ru; Bateman *et al.*, 1998), in the present document we explicitly adopt an organization that is not language-specific. Each section is dedicated to the description of one particular phenomenon rather than to one particular language (Section 2.1: Transitivity, Section 2.2: Diathesis, Section 2.3: Minor Transitivity, Section 2.4: Mood, Section 2.5: Tense, Section 2.6: Aspect, Section 2.7: Clause-Complexity, Section 2.8: Determination, Section 2.9: Word Order). Each phenomenon is dealt with for all three languages.[2] This reflects the implementation strategy we adopted for task 7.2: responsibilities were distributed across sites according to phenomena rather than languages so as to support a truly contrastive-linguistic method of work. The grammatical resources implemented have thus not only been built up by transfer comparison with the English grammar Nigel (cf. Bateman *et al.*, 1998, Section 3.1) that is part of the KPML implementation platform, but crucially by continuous contrastive-linguistic comparison across Bulgarian, Czech and Russian. Only with this strategy has it been possible to implement rather general accounts of the grammars of Bulgarian, Czech and Russian in a year's time. It is thus demonstrated once again that the methodology of resource sharing we have argued for in (Bateman *et al.*, 1998) leads to fast, but nevertheless principled results.

Before we start the presentation of the system network implementations, let us give an overview of the main notational conventions used in this document.

1.1 Notational conventions

Below we give the notational conventions necessary to follow the description of the grammar implementation. Figure 1 presents the general notational conventions used in Systemic Functional Grammar, Figure 2 shows the notation we adopt for system networks, Figure 3 provides the syntax of computational system network specifications, and Figure 4 shows the syntax of multilingual system network specifications. In the text, feature names are given in bold face, system names are given in capitals, and also the names of regions are given in capitals.

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:	

[2] Each section describing the phenomena in focus presents the system implementation and the corresponding chooser. Inquiries and inquiry implementations are not given because they are usually taken over from the NIGEL grammar (cf. IMPL1 (Bateman *et al.*, 1998) for the concepts of choosers and inquiries). SPL specifications, i.e., input specifications for tactical generation, are given as examples only. For the full sets of SPLs for each language for the ImD texts please visit the individual sites' webpages:

http://fairway.ms.mff.cuni.cz/~agile/

http://www.iinf.acad.bg/agile

http://www.aha.ru/~sharoff//Agile

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 of generated structures)

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

```
(system
 :name SYSTEM-NAME
 :inputs (:Language-1 :Language-2 OR feature-x
          (AND feature-y feature-z))
 :outputs ((:language-1 language-2 0.5 feature-a
                (:language-1 insert Function-1))
                (:language-2 insert Function-4))
           (:language-1 language-2 0.5 feature-b
                (:language-2
                     conflate Function-2 Function-3)
                (:language-1 :language-2
                     preselect Function-2 feature-c)))
 :chooser SYSTEM-NAME-CHOOSER
 :region REGION-NAME
   :metafunction METAFUNCTION)
```

Figure 4: Syntax for computational multilingual system network specifications

2. Implementation of grammatical resources

2.1 Transitivity (nuclear and circumstancial)

In this chapter we describe the implementation of the transitivity resources for Russian, Czech and Bulgarian. The implementation is based upon the English Nigel grammar. Transitivity manifests itself structurally as a configuration of a process, the participants involved in the process (nuclear transitivity) and the attendant circumstances (circumstancial transitivity). This configuration forms the basis for the interpretation of the transitivity structure of the clause as a constituent structure, such as e.g., Actor + Process + Locative. In Nigel, nuclear transitivity is divided into relational and nonrelational transitivity. Here, nonrelational transitivity is described in Section 2.1.1, relational transitivity is described in Section 2.1.2. Included in Section 2.1.1 is a solution to the problem of agency (ergativity vs. transitivity), which has been discussed in the SPEC2 deliverable (Andonova *et al*., 1999). Circumstantial transitivity is desribed in Section 2.1.3.

2.1.1 Nonrelational transitivity

The most general transitivity systems are part of the region NONRELATIONAL-TRANSITIVITY. The system PROCESS-TYPE (see below) distinguishes between **material**, **mental**, **verbal** and **relational** processes. This system is common to the three Slavic languages. In the ImD we focus on two types of processes - material and relational. Verbal and mental are hardly used in this register, according to our corpus analyses. The PROCESS-TYPE system is shown in Figure 1.

PROCESS-TYPE:
(transitivity-unit) →
[material] (Process::do-verb),
[mental] (Process::experience-verb),
[verbal] (Process::symbolic-verb),
[relational] (Process::relational-verb).

```
(SYSTEM
    :NAME   PROCESS-TYPE
    :INPUTS   TRANSITIVITY-UNIT
    :OUTPUTS    ((0.333333 MATERIAL
                (CLASSIFY PROCESS DO-VERB))
                (0.333333 MENTAL
                (CLASSIFY PROCESS EXPERIENCE-VERB))
                (0 VERBAL
                (CLASSIFY PROCESS SYMBOLIC-VERB))
                (0.333333 RELATIONAL
                (CLASSIFY PROCESS RELATIONAL-VERB)))
    :REGION   NONRELATIONALTRANSITIVITY
    :METAFUNCTION   EXPERIENTIAL
    :CHOOSER   PROCESS-TYPE-CHOOSER
```

```
)

(CHOOSER
    :NAME   PROCESS-TYPE-CHOOSER
    :DEFINITION   ((ASK (STATIC-CONDITION-Q PROCESS)
                  (STATIC
                  (ASK (MENTAL-PROCESS-Q PROCESS)
                  (MENTAL (IDENTIFY SENSER
                  (SENSER-ID PROCESS))
                  (COPYHUB SENSER MEDIUM)
                  (CHOOSE MENTAL))
                  (NONMENTAL
                  (ASK (VERBAL-PROCESS-Q PROCESS)
                  (VERBAL (IDENTIFY SAYER
                  (SAYER-ID PROCESS))
                  (CHOOSE VERBAL))
                  (NOTVERBAL (CHOOSE RELATIONAL))))))
                  (NONSTATIC
                  (ASK (VERBAL-PROCESS-Q PROCESS)
                  (VERBAL (IDENTIFY SAYER
                  (SAYER-ID PROCESS))
                  (CHOOSE VERBAL))
                  (NOTVERBAL
                  (ASK (MENTAL-PROCESS-Q PROCESS)
                  (MENTAL (IDENTIFY SENSER
                  (SENSER-ID PROCESS))
                  (COPYHUB SENSER MEDIUM)
                  (CHOOSE MENTAL))
                  (NONMENTAL (CHOOSE MATERIAL)))))))))
```

Figure 5:Type of process: system and chooser (En, Cz, Ru, Bg)

Another very general system in the NONRELATIONALTRANSITIVITY region is is the AGENCY system. For English, this system reflects the ergative pattern of English, in which an Agent + Medium configuration (transitive construction) has a Medium-only variant (middle or ergative construction). While the grammars of Slavic languages have the middle variant as well, its construction needs more complex morpho-syntactic means. Example (1a) illustrates this pattern for English, and examples (1b) – (1d) illustrate which additional means are needed in Bulgarian, Czech and Russian, i.e., reflexivization of the Process.

(1)

(a) En: The lion woke the tourist – The tourist woke.

(b) Cz: Lev probudil turistu - Turista se probudil.

Lion-nom woke tourist-acc – Tourist-nom woke-refl

(c) Bg: Лъвът събуди туриста. - Туристът се събуди.

Lion woke tourist – Tourist woke-refl

(d) Ru: Лев разбудил туриста – Турист проснулся.

Lion-nom woke tourist-acc – Tourist-nom woke-refl

The English system of AGENCY accounts for this kind of alternation as follows:

AGENCY:

(transitivity-unit) →

[middle] (Process::middle-verb),

[effective] (Process::effective-verb).

This is hard to apply for Slavic languages because they are transitively rather than ergatively organized, i.e., we find the productive alternation of transitive vs. intransitive. Since both patterns exits, we decided to change the AGENCY system to a system that distinguishes between the number of participant roles, on the one hand, and proper agency, on the other hand. The system then looks as follows, where **middle** now stands for process types with one participant only, **middle-transitive** corresponds to the English middle, and **effective** stays the same.

AGENCY:

(transitivity-unit) →

[middle] (Process:: intransitive-verb),

[middle-transitive] (Process::middle -verb),

[effective] (Process::transitive-verb).

Middle processes in the Slavic languages do not have Agents – they are *inherently* Agent-less (see examples (2a-d)); **middle-transitive** processes involve a Medium, but may have an Agent. An example from Russian would be "okno otkroetsja" (The window will open). There are no occurrences of this type in the ImD texts. What we have instead is - "okno pojavitsja" (The window will appear) – which is a **middle** configuration. Finally the **effective** option accounts for the transitive variant, and thus has the normal voice potential (active – passive), which the other two have not.

(2) *Slavic intransitive verbs in middle configuration*

(a) En: Switch to Arc mode.

(b) Bg: Превключите в режим Arc.

Switch to mode Arc

(c) Cz: Přepnutí do režimu Kreslení oblouků.

Switch to the mode of the arc drawing

(d) Ru: Перейдите в режим Arc

Switch to mode Arc

Middle processes can be realized by intransitive verbs (as in 2b-d) or by reflexive verbs. See examples (3b-d).

(3) *Slavic reflexive verbs in middle configuration*

(a) En: The dialog box appears.

(b) Bg: Диалоговият прозорец се появява.

The dialog box appears.

(c) Cz: Objeví se dialogové okno

Appear- refl dialog box-nom.

(d) Ru: На экране появится диалоговое окно

On the screen appear- refl dialog box-nom.

Effective processes are realized by lexical verbs that are transitive, i.e., that have a Direct Complement realized in accusative case. Effective processes are the most frequent kind of process in the ImD texts. See example (4).

(4) *Effective = Slavic transitive*

(a) En: The user starts the command. (to specify (the point), to enter (the symbol), etc.)

(b) Bg: Потребителят стартира командата. (задавам (точка), въвеждам (символ), etc.)

The user starts the command

(c) Cz: Uživatel spustí příkaz. (určit (bod), zadat (symbol), etc.)

User-nom starts the command-acc

(d) Ru: Пользователь запускает команду. (задать (точку), ввести (символ), etc.)

User-nom starts the command-acc

Finally, what we called middle-transitive processes are realized by reflexive forms of transitive verbs. See example (5).

(5) *Middle-transitive*

(a) En: The command starts.

(b) Bg: Командата се стартира

The command starts-refl

(c) Cz: Příkaz se spustí

The command-nom starts-refl

(d) Ru: Команда запускается

The command-nom starts-refl

The implementations of the AGENCY system and its chooser are shown in Figure 6.

```
(SYSTEM
    :NAME   AGENCY
    :INPUTS   (OR MATERIAL MENTAL VERBAL)
    :OUTPUTS   (
(:ENGLISH (0.5 MIDDLE (CLASSIFY PROCESS MIDDLE-VERB))
          (0.5 EFFECTIVE(CLASSIFY PROCESS EFFECTIVE-VERB)))
(:RUSSIAN (0.333 MIDDLE (CLASSIFY PROCESS MIDDLE-VERB))
          (0.333 MIDDLE-TRANSITIVE
                (CLASSIFY PROCESS MIDDLE-VERB)
                (CONFLATE PROCESS VOICE)
                (PRESELECT PROCESS REFLEXIVE-FORM))
          (0.333 EFFECTIVE(CLASSIFY PROCESS EFFECTIVE-VERB)))
                )
    :CHOOSER   AGENCY-CHOOSER
    :REGION   NONRELATIONALTRANSITIVITY
    :METAFUNCTION    EXPERIENTIAL
)

(CHOOSER
    :NAME   AGENCY-CHOOSER
    :DEFINITION   ((ASK (VERBAL-PROCESS-Q PROCESS)
      (VERBAL
      (ASK (ADDRESSEE-ORIENTED-Q PROCESS)
      (ADDRESSEEORIENTED (COPYHUB SAYER AGENT)
      (CHOOSE EFFECTIVE))
      (NOTADDRESSEEORIENTED (CHOOSE MIDDLE))))
      (NOTVERBAL
      (ASK (CAUSED-PROCESS-Q PROCESS)
      (INDEPENDENT (CHOOSE MIDDLE))
      (CAUSED (IDENTIFY AGENT
      (CAUSER-ID PROCESS))
      (ASK (CAUSATIVE-MENTION-Q AGENT PROCESS)
   (:ENGLISH (WITHHOLD (CHOOSE MIDDLE)))
   (:RUSSIAN (WITHHOLD (CHOOSE MIDDLE-TRANSITIVE)))
      (MENTION (CHOOSE EFFECTIVE)))))
        ))))
```

Figure 6: Agency: system and chooser implementation (Cz, Bg, Ru)

The most frequent process type in the ImD texts is **effective-material.** Choice of this type is wired in the gate EFFECTIVE-MATERIAL. For **middle-transitive** agency we also need this gate (see Figure 7).

```
(GATE
    :NAME   EFFECTIVE-MATERIAL
    :INPUTS   (:ENGLISH (AND EFFECTIVE MATERIAL))
              (:RUSSIAN (OR (AND EFFECTIVE MATERIAL) MIDDLE-TRANSITIVE)))
    :OUTPUTS   ((1.0 EFFECTIVE-MATERIAL))
    :CHOOSER   EFFECTIVE-MATERIAL-CHOOSER
    :REGION   NONRELATIONALTRANSITIVITY
    :METAFUNCTION    EXPERIENTIAL
)

(CHOOSER
    :NAME   EFFECTIVE-MATERIAL-CHOOSER
    :DEFINITION   ((IDENTIFY GOAL
           (AFFECTED-ID PROCESS))
           (COPYHUB GOAL MEDIUM)
           (COPYHUB AGENT ACTOR)
           (CHOOSE EFFECTIVE-MATERIAL)
 ))
```

Figure 7: Effective-material gate: gate and chooser (Cz, Bg, Ru)

An example of an SPL input expression generating a middle-transitive clause with a reflexivized verb is shown in Figure 8.

```
(EXAMPLE
    :NAME   VOICE-51-RU
    :GENERATEDFORM   ""
    :TARGETFORM   "Okno otkroetsja"
    :LOGICALFORM
          (P / DIRECTED-ACTION :LEX OTKRYTJ
   :CAUSATIVE-MENTION-Q WITHHOLD
   :PREFER-MENTION-AGENT-Q WITHHOLD
   :ACTOR (P1 / PERSON :LEX poljzovatelj)
   :ACTEE (D1 / OBJECT :LEX OKNO))
   :SET-NAME   VOICE
)
```

Figure 8: Semantic form for generating **middle-transitive** clauses (Ru)

Material processes may also include a Beneficiary as participant in the configuration. In relation to this participant in the English model the processes are divided into dispositive and creative by the system DOING-TYPE. In the ImD texts we have both, but there is no difference in their use concerning the Beneficiary. We nevertheless conserve the distinction of RECIPIENCY (for **dispositive** processes) and CLIENCY (for **creative** processes). Recipient and Client are subtypes of Beneficiary. In English they differ formally by the preposition (*to* for Recipient and *for* for Client). In Russian and Czech the Beneficiary is marked by dative case with both dispositive and creative processes. Creative processes have a variant with a preposition: (Ru) *для,* (Cz) *pro,* (En) *for.* Bulgarian shows the same distinctions as English: (Bu) *na* - (En) *to,* (Bu) *za* - (En) *for.* For example:

(6) *Realizing Beneficiary*

(a) En: She sent her best wishes to John - She sent John her best wishes.

(b) Cz: Poslala pozdrav (Acc) Johnovi

(She) sent best wishes-acc John-dat

Poslala Johnovi (dat) pozdrav

(She) sent John-dat best wishes-acc.

(c) Bg: Тя изпрати поздрави на Джон

She sent best wishes to John

Тя изпрати на Джон поздрави

She sent to John best wishes

(d) Ru: Она передала привет Джону

She sent best wishes-acc John-dat

Она передала Джону привет

She sent John-dat best wishes-acc

The examples show that for dispositive processes the Slavic languages have only one way of expressing the Beneficiary – dative case for Russian and Czech and one preposition for Bulgarian. English has two different prepositions and different participant roles. But in all the cases we have two variants of word order (reflecting differences in communicative structure).

For creative processes the situation is the same as for dispositive for Bulgarian and English. For Russian and Czech there are two variants – realization by a nominal group in dative case or by a prepositional phrase with the prepositions *dlja* and *pro*, respectively.In the two variants, two variants of word order are possible as shown in examples (7):

(7) *Realizing Cliency*

(a) En: Fred bought a present for his wife - Fred bought his wife a present.

(b) Cz: Fred koupil dárek pro svou ženu - Fred koupil pro svou ženu dárek

Fred bought a present-acc for his wife-gen (different in communicative structure)

Fred koupil dárek (acc) své ženě (dat) - Fred koupil své ženě dárek

Fred bought a present-acc for his wife-dat (different in communicative structure)

(c) Bg: Фред купи подарък за жена си - Фред купи за жена си подарък

Fred bought a present for his wife

(d) Ru: Фред купил подарок для своей жены - Фред купил для своей жены подарок

Fred bought a present-acc for his wife-gen (different in communicative structure)

Фред купил подарок своей жене - Фред купил своей жене подарок

Fred bought a present-acc for his wife-dat (different in communicative structure)

We do not look at the problems any further, since the output features will be negative because there are no Beneficiary roles in the ImD texts. In Figure 9 we show the systems relating to the choice of Beneficiary.

```
(SYSTEM
    :NAME   DOING-TYPE
    :INPUTS   EFFECTIVE-MATERIAL
    :OUTPUTS   ((0.5 CREATIVE
               (CLASSIFY PROCESS CREATION-VERB))
               (0.5 DISPOSITIVE
               (CLASSIFY PROCESS DISPOSAL-VERB)))
    :CHOOSER   DOING-CHOOSER
    :REGION   NONRELATIONALTRANSITIVITY
    :METAFUNCTION   EXPERIENTIAL
)
(CHOOSER
    :NAME   DOING-CHOOSER
    :DEFINITION   ((ASK (PREEXIST-Q GOAL PROCESS)
                  (NONEXISTENT (CHOOSE CREATIVE))
                  (EXISTED (CHOOSE DISPOSITIVE))))
)
```

```
(SYSTEM
    :NAME   CLIENCY
    :INPUTS   CREATIVE
    :OUTPUTS   ((0.5 CLIENCY
               (INSERT BENEFICIARY)
               (PRESELECT BENEFICIARY NOMINAL-GROUP))
               (0.5 NONCLIENCY))
    :CHOOSER   CLIENCY-CHOOSER
    :REGION   NONRELATIONALTRANSITIVITY
    :METAFUNCTION   EXPERIENTIAL
)
(CHOOSER
    :NAME   CLIENCY-CHOOSER
    :DEFINITION   ((ASK (CLIENCY-Q PROCESS)
                  (CLIENCY (IDENTIFY BENEFICIARY
                  (CLIENT-ID PROCESS))
                  (IDENTIFY BENEFICIARY
                  (CONCEPTUAL-CORRELATE-ID BENEFICIARY))
                  (CHOOSE CLIENCY))
                  (NONCLIENCY (CHOOSE NONCLIENCY)))
 ))

(SYSTEM
    :NAME   RECIPIENCY
    :INPUTS   DISPOSITIVE
    :OUTPUTS   ((0.9 NONRECIPIENCY)
               (0.1 RECIPIENCY
               (INSERT BENEFICIARY)
               (PRESELECT BENEFICIARY NOMINAL-GROUP)))
    :CHOOSER   RECIPIENCY-CHOOSER
    :REGION   NONRELATIONALTRANSITIVITY
    :METAFUNCTION   EXPERIENTIAL
)
(CHOOSER
    :NAME   RECIPIENCY-CHOOSER
    :DEFINITION   ((ASK (POSSESSION-ONSET-Q PROCESS)
                  (NOPOSSESSIONCREATED (CHOOSE NONRECIPIENCY))
                  (POSSESSIONCREATED
                  (ASK (POSSESSION-ONSET-SPECIFICATION-Q PROCESS)
                  (SPECIFIED (CHOOSE RECIPIENCY)
                  (IDENTIFY BENEFICIARY
                  (BENEFICIARY-ID PROCESS))
                  (IDENTIFY BENEFICIARY
                  (CONCEPTUAL-CORRELATE-ID BENEFICIARY)))
                  (NOTSPECIFIED (CHOOSE NONRECIPIENCY)))))
 ))

(GATE
    :NAME   NONBENEFACTION
    :INPUTS   (OR NONRECIPIENCY NONCLIENCY
              (AND EFFECTIVE-VERBAL NONADDRESS))
    :OUTPUTS  ((1.0 NONBENEFACTIVE))
    :REGION   NONRELATIONALTRANSITIVITY
    :METAFUNCTION   EXPERIENTIAL
)
```

```
(GATE
   :NAME   NONOBLIQUE-COMPLEMENTED
   :INPUTS (OR MED-BEN AG-BEN NONBENEFACTIVE NONCLIENCY MENTAL RELATIONAL
           (AND MIDDLE VERBAL))
   :OUTPUTS   ((1.0 NONOBLIQUE-COMPLEMENTED))
   :REGION   UNIFYINGGATES
   :METAFUNCTION   UNIFYING
)
```

Figure 9: Selecting the Beneficiary participant (Cz, Bg, Ru)

As it is shown in the Figure 9 both **nonrecipiency** and **noncliency** lead to the feature **nonbenefactive** and then to the feature **nonoblique-complemented** defined by the gates shown in Figure 9. With these systems, the selection of participants is complete. Their realization by syntactic functions (Subject, Directcomplement etc.) is described in Section 2.2 on diathesis.

2.1.2 Relational transitivity

Relational processes are processes of being. In the intermediate demonstrator texts we have only one type of relational processes - the existential process, semantically a one-place relation. In English, existential constructions have the following constituents:

There (Subject) + are (Process of being) + Existent.

The Theme position is filled by the formal Subject in English (*there*). Slavic languages do not need to insert a formal Subject, the Subject position can stay empty or be occupied by a Circumstance (temporal or spatial). This gives rise to the following kinds of structure:

На экране (Locative) + появится (Process of being) + окно (Existent/Subject) (Nom)

On the screen will appear a window

The topmost system of the RELATIONALTRANSITIVITY region which accounts for the general types of relational process is the system TYPE-OF-BEING (see Figure 10).

```
(SYSTEM
    :NAME   TYPE-OF-BEING
    :INPUTS   RELATIONAL
    :OUTPUTS   ((0.33333334 IDENTIFYING
            (INSERT TOKEN)
            (INSERT VALUE)
            (INSERT IDENTIFIED)
            (INSERT IDENTIFIER)
            (CONFLATE IDENTIFIED SUBJECT)
            (CONFLATE IDENTIFIER DIRECTCOMPLEMENT))
    ((:ENGLISH    0.33333334 EXISTENTIAL
             INSERT EXISTENT)
            (LEXIFY SUBJECT THERE)
             PRESELECT EXISTENT NOMINAL-GROUP))
    (:RUSSIAN    0.33333334 EXISTENTIAL
            (INSERT EXISTENT)
            (CONFLATE EXISTENT SUBJECT)
```

```
              (ORDER PROCESS SUBJECT)
              (PRESELECT EXISTENT NOMINAL-GROUP))
     (0.33333334 RELATIONAL-OTHER))
    :CHOOSER   TYPE-OF-BEING-CHOOSER
    :REGION   RELATIONALTRANSITIVITY
    :METAFUNCTION   EXPERIENTIAL
)
```

Figure 10: TYPE-OF-BEING system (Cz,Ru)

An SPL input expression for generating existential clauses is given in Figure 11.

```
(EXAMPLE
    :NAME   D1-TEXT4-3A2-RU
    :TARGETFORM   "Na ekrane pojavitcja dialogovoje okno
                   Multiline Properties."
    :LOGICALFORM
    (P / EXISTENCE :LEX POJAVITJSJA :TENSE PRESENT :THEME C1
    :ACTOR
    (D / OBJECT :LEX OKNO :PROPERTY-ASCRIPTION
    (D1 / QUALITY :LEX DIALOGOVYJ)
    :LABEL-ASCRIPTION
    (D2 / OBJECT :NAME Multiline-Properties))
    :SPATIAL-LOCATING
    (C1 / ONE-OR-TWO-D-LOCATION :LEX EKRAN))
    :SET-NAME   D1-TEXT4
)
```

Figure 11: Semantic representation with Existent function (Ru)

For Bulgarian the process *to appear* is described as motion process, so is is not covered here.

2.1.3 Circumstantial transitivity

Circumstantial transitivity is the resource for representation of phenomena attendant on (rather than involved in) the process (Adjuncts).

The types of Circumstances are described in the CIRCUMSTANCE region.The region covers a number of simultaneous systems each describing one type of Circumstance. Circumstances can be realized as prepositional phrases or adverbial groups. Realization by prepositional phrases is described in Section 2.3 (Minor Transitivity). The types of Circumstances are described by the system TYPE-OF-CIRCUMSTANCE as follows:

```
(TYPE-OF-CIRCUMSTANCE):
      (transitivity-unit) →
      [nonaccompaniment]
      [accompaniment] (+Accompaniment
                        Accompaniment:prepositional-phrase
                        Accompaniment:accompaniment-process)
      [noncause]
      [cause] (+Cause)

      [nonmanner]
      [manner] (+Manner)

      [nonmatter]
     [matter] (+Matter     Matter:matter-process)
```

```
 [nonrole]
 [role] (+Role      Role:role-process)

[no-spatial-extent]
[spatial-extent] (+Spaceextent
                  Spaceextent:prepositional-phrase
                 Spaceextent:extent-process)
[no-spatial-location]
[spatial-location] (+Spacelocative)

[no-temporal-extent]
[temporal-extent] (+Timeextent
                     Timeextent:prepositional-phrase
                     Timeextent:extent-process)
[no-temporal-location]
[temporal-location] (+Timelocative)
```

Matter, Role and Timeextent do not occur in the ImD texts. We therefore focus on the other types in the remainder of this section.

In Figure 12 we display the system's implementation for the types of Circumstances that are present in our corpus: Accompaniment, Cause, Manner, Spaceextent, Spacelocative and Timelocative.

```
(SYSTEM
    :NAME   ACCOMPANIMENT-ADJUNCT
    :INPUTS   TRANSITIVITY-UNIT
    :OUTPUTS   ((0.9 NONACCOMPANIMENT)
               (0.1 ACCOMPANIMENT
               (INSERT ACCOMPANIMENT)
               (PRESELECT ACCOMPANIMENT PREPOSITIONAL-PHRASE)
               (PRESELECT ACCOMPANIMENT ACCOMPANIMENT-PROCESS)))
    :CHOOSER   ACCOMPANIMENT-CHOOSER
    :REGION   CIRCUMSTANTIAL
    :METAFUNCTION   EXPERIENTIAL
)

(SYSTEM
    :NAME   CAUSE-ADJUNCT
    :INPUTS   TRANSITIVITY-UNIT
    :OUTPUTS   ((0.9 NONCAUSE)
               (0.1 CAUSE
               (INSERT CAUSE)))
    :CHOOSER   CAUSE-CHOOSER
    :REGION   CIRCUMSTANTIAL
    :METAFUNCTION   EXPERIENTIAL
)
```

```
(SYSTEM
    :NAME   MANNER-ADJUNCT
    :INPUTS   TRANSITIVITY-UNIT
    :OUTPUTS   ((0.9 NONMANNER)
               (0.1 MANNER
               (INSERT MANNER)))
    :CHOOSER   MANNER-CHOOSER
    :REGION   CIRCUMSTANTIAL
    :METAFUNCTION   EXPERIENTIAL
)

(SYSTEM
    :NAME   SPATIAL-EXTENT-ADJUNCT
    :INPUTS   TRANSITIVITY-UNIT
    :OUTPUTS   ((0.9 NO-SPATIAL-EXTENT)
               (0.1 SPATIAL-EXTENT
               (INSERT SPACEEXTENT)
               (PRESELECT SPACEEXTENT PREPOSITIONAL-PHRASE)
               (PRESELECT SPACEEXTENT EXTENT-PROCESS)))
    :CHOOSER   SPATIAL-EXTENT-CHOOSER
    :REGION   CIRCUMSTANTIAL
    :METAFUNCTION   EXPERIENTIAL
)

(SYSTEM
    :NAME   SPATIAL-LOCATION
    :INPUTS   TRANSITIVITY-UNIT
    :OUTPUTS   ((0.9 NO-SPATIAL-LOCATION)
               (0.1 SPATIAL-LOCATION
               (INSERT SPACELOCATIVE)))
    :CHOOSER   SPATIAL-LOCATION-CHOOSER
    :REGION   CIRCUMSTANTIAL
    :METAFUNCTION   EXPERIENTIAL
)

(SYSTEM
    :NAME   TEMPORAL-LOCATION
    :INPUTS   TRANSITIVITY-UNIT
    :OUTPUTS   ((0.9 NO-TEMPORAL-LOCATION)
               (0.1 TEMPORAL-LOCATION
               (INSERT TIMELOCATIVE)))
    :CHOOSER   TEMPORAL-LOCATION-CHOOSER
    :REGION   CIRCUMSTANTIAL
    :METAFUNCTION   EXPERIENTIAL
)
```

Figure 12: Types of circumstances (Cz, Bg, Ru)

We comment on these systems in the order they appear in Figure 12.

In the system ACCOMPANIMENT the Circumstance Accompaniment is inserted and preselected for prepositional-phrase and accompaniment-process. The preposition realizing this type of Circumstance is „in relation to“ (see Section 2.3).

For the selection of Cause, the system of CAUSE-PHORICITY offers the choice of realizing the Cause phorically (as an adverbial group) or nonphorically (as a prepositional phrase). In the ImD texts, we only have nonphoric Causes. Also, Cause has to be subclassified further (see the system CAUSE-TYPE presented in Figure 13).

```
(SYSTEM
    :NAME   CAUSE-TYPE
    :INPUTS   NONPHORIC-CAUSE
    :OUTPUTS    ((0.25 REASON
       (INSERT REASON)
       (CONFLATE CAUSE REASON)
       (PRESELECT REASON REASON-PROCESS))
       (0.25 PURPOSE
       (INSERT PURPOSE)
       (CONFLATE CAUSE PURPOSE)
       (PRESELECT PURPOSE PURPOSIVE-PROCESS))
       (0.25 BEHALF
       (INSERT BEHALF)
       (CONFLATE CAUSE BEHALF)
       (PRESELECT BEHALF CLIENT))
       (0.25 CONCESSION
       (INSERT CONCESSION)
       (CONFLATE CAUSE CONCESSION)
       (PRESELECT CONCESSION CONCESSIVE-PROCESS)))
    :CHOOSER   CAUSE-TYPE-CHOOSER
    :REGION   CIRCUMSTANTIAL
    :METAFUNCTION   EXPERIENTIAL
)
```

Figure 13: The system CAUSE-TYPE (Cz, Bg, Ru)

In the ImD, we only have Purpose Circumstances. See example (8).

(8) *Purpose circumstance (TEXT4) (simplified)*

(a) En: Select an arc for the startpoint of the multiline.

(b) Cz: Zvolte oblouk pro počáteční bod multičáry.

Select an arc-acc for startpoint-gen multiline-gen

(c) Bg: Изберете дъга за началната точка на мултилинията.

Select an arc for the startpoint of the multiline

(d) Ru: Выберите дугу для начальной точки мультилинии.

Select an arc-acc for startpoint-gen multiline-gen

An example of a semantic input representation (SPL) with information on purpose is shown in Figure 14.

```
(EXAMPLE
    :NAME   D1-TEXT4-5-1-RU
    :TARGETFORM   "Sadajte dugu dlja nachaljnoj tochki muljtilinii."
    :LOGICALFORM
           (P / DIRECTED-ACTION :LEX ZADATJ :SPEECHACT IMPERATIVE
    :ACTEE (A2 / OBJECT :NAME duga)
    :purpose (t / OBJECT :LEX tochka :PROPERTY-ASCRIPTION
           (t1 / QUALITY :LEX nachaljnyj)
    :meta-actant
           (t2 / OBJECT :NAME MULJTILINIJA)))
    :SET-NAME   D1-TEXT4
)
```

Figure 14: Semantic representation with purpose function (Ru)

In terms of MANNER-TYPE, the only Circumstance that occurs in the ImD texts is Means. Russian and Czech realize the Means Circumstance by a nominal group in Instrumental case (see example (9)). Russian has the additional option of realization with a prepositional phrase. This depends on whether the Means is interpreted as some sort of an agentive instrument or as a manipulating instrument (see example (10)).

(9) *Means circumstance*

(a) Cz: Spusťte příkaz KŘIVKA jedním z následujících způsobů

Start the PLINE command one-instr of following-gen methods-gen

(b) Ru: Запустите команду PLINE одним из следующих способов

Start the PLINE command by one-instr of following-gen methods-gen

This kind of realization is not possible for Bulgarian. In Bulgarian it can be only expressed by a clause complex.

```
(SYSTEM
    :NAME   MANNER-TYPE
    :INPUTS   MANNER
    :OUTPUTS   ((0.33333334 MEANS
               (INSERT MEANS)
               (:CZECH (PRESELECT MEANS INSTRUMENTAL-CASE)
                       (PRESELECT MEANS NOMINAL-GROUP))
               (CONFLATE MANNER MEANS))
               (0.33333334 QUALITY
               (INSERT QUALITY)
               (PRESELECT MANNER ADVERBIAL-GROUP MANNER-ADVERBIAL)
               (CONFLATE MANNER QUALITY))
               (0.33333334 COMPARISON
               (INSERT COMPARISON)
               (CONFLATE MANNER COMPARISON)
               (PRESELECT COMPARISON PREPOSITIONAL-PHRASE)
               (PRESELECT COMPARISON COMPARATIVE-PROCESS)))
    :CHOOSER   MANNER-TYPE-CHOOSER
    :REGION   CIRCUMSTANTIAL
    :METAFUNCTION   EXPERIENTIAL)
```

Figure 15: Manner type system (Cz, Ru)

(10) *Means circumstance realized as preposition phrase and as nominal group*

(a) Ru: следующим методом

(by the) following method-instr (agentive-instrument)

(b) Ru: с помощью молотка

(by means of a hammer-gen (manipulating-instrument)

For Russian, we therefore have to introduce a more fine-grained subsystem for **means**, which distinguishes between the agentive and the manipulating function of a Means circumstance.

Spatial-location Circumstances are differentiated according to whether they express a rest process or a motion process, and if they are motion processes whether the motion is towards an object or way from it. The system implementations of these choices are shown in Figure 16. Examples and realizations of these alternatives are described in Section 2.3.

```
(SYSTEM
    :NAME   SPATIAL-LOCATION-PHORICITY
    :INPUTS   SPATIAL-LOCATION
    :OUTPUTS   ((0.5 PHORIC-PLACE
               (PRESELECT SPACELOCATIVE SPATIAL-ADVERBIAL))
               (0.5 NONPHORIC-PLACE
               (PRESELECT SPACELOCATIVE SPATIAL-PROCESS)))
    :CHOOSER   SPATIAL-LOCATION-PHORICITY-CHOOSER
    :REGION   CIRCUMSTANTIAL
    :METAFUNCTION   EXPERIENTIAL
)
(CHOOSER
    :NAME   SPATIAL-LOCATION-PHORICITY-CHOOSER
    :DEFINITION   ((ASK (QUESTION-VARIABLE-Q SPACELOCATIVE)
           (VARIABLE
           (ASK (LOCATION-RELATION-SPECIFICITY-Q SPACELOCATIVE)
           (UNSPECIFIED (CHOOSE PHORIC-PLACE))
           (SPECIFIED (CHOOSE NONPHORIC-PLACE))))
           (NONVARIABLE
           (ASK (IDENTIFIABILITY-Q SPACELOCATIVE)
           (IDENTIFIABLE
           (ASK (LOCATION-RELATION-SPECIFICITY-Q SPACELOCATIVE)
           (UNSPECIFIED (CHOOSE PHORIC-PLACE))
           (SPECIFIED (CHOOSE NONPHORIC-PLACE))))
           (NOTIDENTIFIABLE (CHOOSE NONPHORIC-PLACE))))))

 (SYSTEM
    :NAME   LOCATION-STATE
    :INPUTS   NONPHORIC-PLACE
    :OUTPUTS   ((0.5 REST
               (PRESELECT SPACELOCATIVE REST-PROCESS))
               (0.5 MOTION
               (PRESELECT SPACELOCATIVE MOTION-PROCESS)))
    :CHOOSER   LOCATION-STATE-CHOOSER
    :REGION   CIRCUMSTANTIAL
    :METAFUNCTION   EXPERIENTIAL)

(CHOOSER
    :NAME   LOCATION-STATE-CHOOSER
    :DEFINITION   ((ASK (SOURCE-DESTINATION-Q SPACELOCATIVE PROCESS)
                  (NONSOURCEDESTINATION (CHOOSE REST))
                  (SOURCEDESTINATION (CHOOSE MOTION)))))

(SYSTEM
    :NAME   MOTION-DIRECTION
    :INPUTS   MOTION
    :OUTPUTS   ((0.5 TOWARDS-MOTION
               (PRESELECT SPACELOCATIVE TOWARDS))
               (0.5 AWAY-FROM-MOTION
               (PRESELECT SPACELOCATIVE AWAY-FROM)))
    :CHOOSER   MOTION-DIRECTION-CHOOSER
    :REGION   CIRCUMSTANTIAL
    :METAFUNCTION   EXPERIENTIAL
)

(CHOOSER
    :NAME   MOTION-DIRECTION-CHOOSER
    :DEFINITION   ((ASK (SOURCE-Q SPACELOCATIVE PROCESS)
                  (SOURCE (CHOOSE AWAY-FROM-MOTION))
                  (NOTSOURCE (CHOOSE TOWARDS-MOTION)))))
```

Figure 16: Classification of location dynamic processes (Cz, Bg, Ru)

2.2 Diathesis

This section documents the diathesis systems for the three Slavic languages. We provide a systemic functional account of the diathesis phenomena that occur in the present domain. The described networks draw upon the English Nigel grammar regions NONRELATIONALTRANSITIVITY and VOICE, exploiting cross-linguistic similarities as much as possible. Section 2.2.1 recapitulates the notions involved in diathesis. Section 2.2.2 sketches the types of diathesis needed for the intermediate demonstrator and Section 2.2.3 presents the actual implementation.

2.2.1 Notions involved in modeling Diathesis

There are various methods of modeling diathesis. For instance, in (Meljchuk, Kholodovich 1970) the notion of diathesis is defined as the accordance of deep cases to surface cases. Other models speak of relation changes. Following the SFG perspective, diathesis can be described as the relation between transitivity functions (participant roles), agency functions (Agent, Medium) and syntactic relations.

One kind of diathesis change that affects the mapping of transitivity, agency and syntactic functions is voice. For example, in an English material process that is in active voice, the Agent is conflated with the Actor and the Subject, the Goal is conflated with the Medium and the Direct Complement, and the verbal group is in the active voice from. In passive voice, the Medium is mapped onto the Subject and the Agent is realized with a prepositional phrase with *by*. The situation is similar in Slavic languages. The only difference is in the Agent realization in passive voice. In Russian and in Czech it is expressed by Instrumental case, in Bulgarian it is realized as in English by a prepositional phrase with the preposition *от*. The active – passive voice pairs for English and Slavic languages are illustrated in the examples below:

(11) *Active and passive voice pairs*

(a) En: The user opens the window - The window is opened by the user

(b) Bg: Potrebiteliat otvaria prozoreca.

The user opens the window

Prozorecat e otvoren (ot potrebitelia)

The window is opened (by the user)

(c) Cz: Uživatel otevře okno

The use-nom opens the window-acc

Okno-nom je otevřeno (uživatelem)

The window is opened (by the user)

(d) Ru: Пользователь откроет окно

The user-nom opens the window-acc

Окно открыто пользователем

The window-nom opened (by) the user-instr

In Czech and Bulgarian, the passive form with Agent is grammatically possible, but it is not used frequently. So, one kind of diathesis alternations is the active – passive alternation. Another kind of diathesis alternation for English is the active versus middle construction:

(12) *En: The user opens the window – The window opens.*

Because the transitivity organization of Slavic languages is different (cf. Section 2.1), voice alternations are affected. A voice alternation always also affects the verbal group. See examples (13a-c):

(13)

(a) Bg: Прозорецът се отваря

The window opens-refl

(b) Cz: Okno se otevře

The window opens-refl

(c) Ru: Окно откроется

The window opens-refl

The choice of voice therefore goes together tightly with that of agency. Voice type affects the mapping of syntactic functions, agency and transitivity roles, and the realization of the Process constituent.

2.2.2 Types of diathesis in the intermediate demonstrator

Recall that the transitivity system of Slavic languages is organized transitively rather than ergatively. While both patterns exists, the more productive and morpho-syntactically less complex alternation follows the transitive pattern. See example (14).

(14)

(a) En: The lion woke the tourist (active construction) – The tourist woke (middle construction).

(b) Bg: Лъвът събуди /разбуди/ туриста. - Туристът се събуди.

The lion woke the tourist - The tourist woke

(c) Cz: Lev probudil turistu. - Turista se probudil.

The lion woke the tourist - The tourist woke-refl

(d) Ru: Лев разбудил туриста – Турист проснулся.

The lion woke the tourist - The tourist woke-refl

Traditionally, verbs in the Slavic languages are classified into **transitive** and **nontransitive.** In example (14) above the first sentence of each pair has active voice, which is realized by a transitive construction with the verb being in active form. This is the same for all four languages. In English, the middle alternation is simply created by putting the second argument of the transitive variant into the Subject position in the middle variant, whereas in the three Slavic languages the change is also reflected on the verb, which takes on the "**medio-passive**" voice. This is formally realized in Russian morphologically by means of a reflexive verb, and in Czech and Bulgarian by insertion of a reflexive particle (a clitic).[3]

Another difference to English in voice formation is that in our three Slavic languages only the syntactic function of Directcomplement is involved in voice transformations: in English also Oblique Complements can become Subjects in passive voice (e.g., *I gave a book to Mary – Mary was given a book*), which is not possible in Bulgarian, Czech or Russian.

We can observe three voice meanings and voice forms (active-process, passive-process and medio-passive-process) for Czech, Bulgarian and Russian which interact with four transitivity features:

- Middle (one participant) -> **operative voice, active-process** (Agent = Subject)

One-role, middle processes are always in active voice. [4]

- Effective (two participants) -> **operative-voice, active-process** (Agent = Subject + Medium = Directcomplement)

  See example (15).

[3] To represent "reflexive" participants, Slavic languages use reflexive pronouns – Bg: „sebe si", Cz: "sebe" and Ru: „себя".

[4] We do not discuss here the specifically Slavic forms as "Я гуляю – Мне (хорошо) гуляется" (I walk – It is walking (well) for me) – they are not relevant to our register.

(15)

(a) En: Specify the endpoint.

(b) Bg: Prozorecat e otvoren (ot potrebitelia)

The window is opened (by the user).

(c) Cz: Určete koncový bod

Specify-active the end point-acc

(d) Ru: Укажите конечную точку

Specify-active the end point-acc

- Effective (two participants) -> **receptive-voice**, **passive-process**, (Medium = Subject + Agent in Instrumental Case or with Agentmarker „от" for Bulgarian)

  See example (16).

(16)

(a) En: The window is opened by the user.

(b) Bg: Прозорецът се отваря от потребителя.

The window is opened by the user

(c) Cz: Okno je otevřeno (uživatelem)

The window-nom is opened the user-instr

(d) Ru: Окно открыто пользователем

The window-nom is opened the user-instr

- Middle-transitive (one participant) **-> medio-passive-voice, reflexive-process** (Medium = Subject; no Agent possible)
  See example (17).

(17)

(a) En: The window opens.

(b) Bg: Прозорецът се отваря.

The window opens-refl

(c) Cz: Okno se otevře.

The window opens-refl

(d) Ru: Окно откроется

The window will open-refl

Since the transitivity and voice organization of Bulgarian, Czech and Russian is rather different from that of English, the Nigel modeling can only be re-used to a restricted degree. The implementation is described in the next section.

2.2.3 Modeling diathesis for Bulgarian, Czech and Russian

As stated above, in modeling diathesis, two grammar regions interact – NONRELATIONALTRANSITIVITY and VOICE. We take as point of departure for choices in voice (diathesis) the AGENCY system, which classifies processes into three types: middle, middle-transitive, effective (cf. Section 2.1).

For intransitive processes (middle, in the present terminology), voice can be defaulted to **operative** (i.e., active). Intransitive processes have the feature noncomplemented which leads to the gates ACTIVE-PROCESS and VOICE-LEXVERB, which are shown in Figure 17. These gates are identical to the English ones.

```
(GATE
    :NAME   ACTIVE-PROCESS
    :INPUTS   (OR (AND RELATIONAL
              (OR EXISTENTIAL ASCRIPTIVE))
        NONRANGED RANGE-OPERATIVE METEOROLOGICAL OPERATIVE IDENTIFYING
        BENEFACTIVEOPERATIVE BENEFACTIVE-NONMEDIATED-OPERATIVE
              (AND NONADDRESS REPORT))
    :OUTPUTS   ((1.0 ACTIVE-PROCESS))
    :CHOOSER   ACTIVE-PROCESS-CHOOSER
    :REGION   VOICE
    :METAFUNCTION   EXPERIENTIAL
)

(GATE
    :NAME   VOICE-LEXVERB
    :INPUTS   (OR (AND ACTIVE-PROCESS NOT-PHASE))
    :OUTPUTS  ((1.0 VOICE-LEXVERB
              (CONFLATE VOICE LEXVERB)))
    :CHOOSER   VOICE-LEXVERB-CHOOSER
    :REGION   VOICE
    :METAFUNCTION   EXPERIENTIAL
)
```

Figure 17: Selecting Slavic intransitive (En, Cz, Ru, Bg)

In Figure 18 we show a sample SPL for generating middle clauses (Ru: Okno ischeznet s ekrana – En: The window disappears from the screen).

```
(EXAMPLE
    :NAME   VOICE-2-4-8A-RU
    :GENERATEDFORM   "Okno ischeznet s ekrana."
    :TARGETFORM   "Okno ischeznet s ekrana."
    :LOGICALFORM
         (P / NONDIRECTED-ACTION :LEX ISCHEZNUTJ :TENSE PRESENT :ACTOR
         (D / OBJECT :LEX OKNO )
   :SOURCE
         (C1 / ONE-OR-TWO-D-LOCATION :LEX EKRAN))
    :SET-NAME   VOICE
)
```

Figure 18: Semantic form for the intransitive sentence (En, Cz, Ru, Bg)

Effective processes can occur in three types of diathesis: operative (active), receptive (passive) and medio-passive. The point of departure for them is the system AGENTIVITY, shown in Figure 19. It is identical for the Slavic languages and English.

```
(SYSTEM
    :NAME   AGENTIVITY
    :INPUTS   (OR EFFECTIVE-MATERIAL EFFECTIVE-MENTAL EFFECTIVE-VERBAL)
    :OUTPUTS  ((0.9 NONAGENTIVE)
              (0.1 AGENTIVE))
    :CHOOSER   AGENTIVITY-CHOOSER
    :REGION   VOICE
    :METAFUNCTION   EXPERIENTIAL
)

(CHOOSER
    :NAME   AGENTIVITY-CHOOSER
    :DEFINITION   ((ASK (PREFER-MENTION-AGENT-Q AGENT PROCESS)
                  (WITHHOLD (CHOOSE NONAGENTIVE))
                  (MENTION (CHOOSE AGENTIVE)))))
```

Figure 19: System and chooser for agentivity (En, Cz, Ru, Bg)

The feature **agentive** leads to the choice between **operative** (active) and **receptive** (passive) with Agent, the choice of **nonagentive** leads to passive without Agent or to **medio-passive**. The system EFFECTIVE VOICE is shown in Figure 20.

```
(SYSTEM
    :NAME   EFFECTIVE-VOICE
    :INPUTS   (OR (AND AGENTIVE
              (OR NONRECIPIENCY NONCLIENCY MENTAL
              (AND NONADDRESS REPORT-NAME)))
               EQUATIVE)
    :OUTPUTS  ((0.9 OPERATIVE
              (CONFLATE MEDIUM DIRECTCOMPLEMENT))
              (0.1 RECEPTIVE))
    :CHOOSER   EFFECTIVE-VOICE-CHOOSER
    :REGION   VOICE
    :METAFUNCTION   EXPERIENTIAL
)
```

Figure 20: System for active and passive voice (En, Cz, Ru, Bg)

The **operative** feature then leads to the realization of the Directcomplement and to the ACTIVE-PROCESS system shown in Figure 21. **Effective** and **active-process** features launch the gate MEDIUM-INSERT and then the gate GOAL-INSERT-CONFLATE (presented in Figure 22).

```
(GATE
    :NAME   ACTIVE-PROCESS
    :INPUTS   (OR (AND RELATIONAL
              (OR EXISTENTIAL ASCRIPTIVE))
        NONRANGED RANGE-OPERATIVE METEOROLOGICAL OPERATIVE IDENTIFYING
        BENEFACTIVEOPERATIVE BENEFACTIVE-NONMEDIATED-OPERATIVE
              (AND NONADDRESS REPORT))
    :OUTPUTS  ((1.0 ACTIVE-PROCESS))
    :CHOOSER   ACTIVE-PROCESS-CHOOSER
    :REGION   VOICE
    :METAFUNCTION   EXPERIENTIAL
)
```

Figure 21: System ACTIVE-PROCESS (En, Cz, Ru, Bg)

```
(GATE
    :NAME   MEDIUM-INSERT
    :INPUTS  (OR (AND EFFECTIVE
             (OR ACTIVE-PROCESS (AND PASSIVE-PROCESS SUBJECT-INSERTED))
             (OR MATERIAL MENTAL REPORT-NAME))
              EQUATIVE MIDDLE-TRANSITIVE
             (AND SUBJECT-INSERTED
             (OR NONRANGED MEDIATED RANGE-OPERATIVE)))
    :OUTPUTS (1.0 MEDIUM-INSERTED
             (INSERT MEDIUM)))
    :REGION   NONRELATIONALTRANSITIVITY
    :METAFUNCTION   EXPERIENTIAL
)
(GATE
    :NAME   GOAL-INSERT-CONFLATE
    :INPUTS   (AND EFFECTIVE-MATERIAL MEDIUM-INSERTED)
    :OUTPUTS   ((1.0 GOAL
               (INSERT GOAL)
               (CONFLATE GOAL MEDIUM)
               (PRESELECT GOAL NOMINAL-GROUP)))
    :CHOOSER   GOAL-INSERT-CONFLATE-CHOOSER
    :REGION   NONRELATIONALTRANSITIVITY
    :METAFUNCTION   EXPERIENTIAL
)
```

Figure 22: Medium and Goal insertion (Cz, Ru, Bg)

The inserted transitivity functions are conflated with Directcomplement in the EFFECTIVE-VOICE system for the active construction (**operative)** and with the Subject function for the passive construction (**receptive**) in the gate MEDIUM-SUBJECT-CONFLATE, shown on the Figure 23.

```
(GATE
    :NAME   MEDIUM-SUBJECT-CONFLATE
    :INPUTS   (AND MEDIUM-INSERTED
              (OR (AND NONRANGED MIDDLE)
              (AND MIDDLE-TRANSITIVE NONBENEFACTIVE)
         RANGE-OPERATIVE MEDIORECEPTIVE NONAGENTIVE-MEDIORECEPTIVE
              (AND (OR RECEPTIVE NONAGENTIVE)
              (OR NONBENEFACTIVE MENTAL RELATIONAL))))
    :OUTPUTS  ((1.0 MEDIUM-SUBJECT-CONFLATED
              (CONFLATE MEDIUM SUBJECT)))
    :REGION   VOICE
    :METAFUNCTION   EXPERIENTIAL
)
```

Figure 23: The updated system MEDIUM-SUBJECT-CONFLATE gate (Cz, Ru, Bg).

Transitive active voice and passive voice constructions with Agent have similar SPL representations, the latter including answers to the effective-voice-chooser inquiries, differentiating these two constructions. Examples of SPLs for active transitive and passive with Agent are shown in Figure 24.

```
(EXAMPLE
    :NAME   VOICE-1-RU
    :TARGETFORM "Poljzovatelj otkroet okno" - „The user opens the window‟
    :LOGICALFORM
          (P / DIRECTED-ACTION :LEX OTKRYTJ :ACTOR
          (P1 / PERSON :LEX POLJZOVATELJ)
    :ACTEE
          (D1 / OBJECT :LEX OKNO))
    :SET-NAME   VOICE
)

(EXAMPLE
    :NAME   VOICE-5-RU
    :TARGETFORM "okno otkryto poljzovatelem" - „The window is opened by
```

```
                 the user‟
    :LOGICALFORM
          (P / DIRECTED-ACTION :LEX OTKRYTJ
    ;;;;following questions from EFFECTIVE-VOICE CHOOSER
    :ACTUALIZATION-CONSTRAINER-Q   NONACTUALIZATIONCONSTRAINER
    :PARAGRAPH-THEME-EXIST-Q EXISTS
    :PARAGRAPH-THEME-ID D1
    :PATH-INCLUSION-Q CONTAINED
    :ACTOR
          (P1 / PERSON :LEX POLJZOVATELJ)
    :ACTEE
          (D1 / OBJECT :LEX OKNO))
    :SET-NAME   VOICE
)
```

Figure 24: Active and passive SPL representations (Ru)

For the passive construction, the form of the verb is described in the gate PASSIVE-VOICE, and the realization of the Agent is described in the gate ADJUNCT-AGENT. Both are shown in Figure 25.

```
(GATE
    :NAME   PASSIVE-PROCESS
    :INPUTS (OR RANGE-RECEPTIVE RECEPTIVE BENEFACTIVE-NONMEDIATED-
             RECEPTIVE
             MEDIORECEPTIVE BENERECEPTIVE
            (:ENGLISH NONAGENTIVE)
            (:RUSSIAN (AND EFFECTIVE NONAGENTIVE))
             NONMEDIATED)
    :OUTPUTS   ((1.0 PASSIVE-PROCESS
               (CLASSIFY VOICE BE-AUX)
               (INSERT VOICEDEPENDENT)
           (:ENGLISH (OUTCLASSIFY VOICE NEGATIVE-AUX)
               (INFLECTIFY VOICEDEPENDENT EDPARTICIPLE))
           (:RUSSIAN (INFLECTIFY VOICEDEPENDENT PAST-PARTICIPLE))))
    :REGION   VOICE
    :METAFUNCTION   EXPERIENTIAL
)

(GATE
    :NAME   ADJUNCT-AGENT
    :INPUTS   (OR RECEPTIVE MEDIORECEPTIVE BENERECEPTIVE
               BENEFACTIVE-NONMEDIATED-RECEPTIVE)
    :OUTPUTS  ((1.0 ADJUNCT-AGENT
      (:ENGLISH (INSERT AGENTMARKER)
                (LEXIFY AGENTMARKER BY)
                (ORDER AGENTMARKER AGENT))
      (:RUSSIAN (ORDER PROCESS AGENT)
                (PRESELECT AGENT INSTRUMENTAL))))
    :REGION   VOICE
    :METAFUNCTION   EXPERIENTIAL
)
```

Figure 25: Modeling passive with Agent (En, Cz, Ru, Bg) .

The medio-passive is accounted for by the gate MEDIO-PASSIVE-PROCESS, presented in Figure 26. This gate is relevant only for the three Slavic languages and not for English.

```
(GATE
    :NAME   MEDIO-PASSIVE-VOICE
    :INPUTS (AND MIDDLE-TRANSITIVE  NONAGENTIVE)
    :OUTPUTS   ((1.0 MEDIO-PASSIVE-PROCESS
               (CONFLATE VOICE LEXVERB)
               (CONFLATE FINITE VOICE)
               (INFLECTIFY LEXVERB REFLEXIVE-FORM)))
    :REGION   VOICE
    :METAFUNCTION   EXPERIENTIAL
)
```

Figure 26: The gate MEDIO-PASSIVE-PROCESS (Cz, Ru, Bg)

Figure 27 gives an example SPL for generating medio-passive.

```
(EXAMPLE
    :NAME   VOICE-4-RU
    :TARGETFORM   "Okno otkroetsja" - „The window opens‟
    :LOGICALFORM
          (P / DIRECTED-ACTION :LEX OTKRYTJ
    :CAUSATIVE-MENTION-Q WITHHOLD
    :PREFER-MENTION-AGENT-Q WITHHOLD
    :ACTOR
          (P1 / PERSON :LEX POLJZOVATELJ)
    :ACTEE
          (D1 / OBJECT :LEX OKNO))
    :SET-NAME   VOICE
)
```

Figure 27: The semantic representation for medio-passive (Ru)

For the Bulgarian and Czech grammars, the medio-passive is implemented as a subtype of passive-process. The gate PASSIVE-PROCESS and the new system PASSIVE-PROCESS-TYPE are shown in Figure 28.

```
(GATE
    :NAME   PASSIVE-PROCESS
    :INPUTS (OR RANGE-RECEPTIVE RECEPTIVE BENEFACTIVE-NONMEDIATED-
RECEPTIVE
    MEDIORECEPTIVE BENERECEPTIVE NONAGENTIVE NONMEDIATED)
    :OUTPUTS ((1.0 PASSIVE-PROCESS))
    :REGION   VOICE
)

(SYSTEM
    :NAME   PASSIVE-PROCESS-TYPE
    :INPUTS   PASSIVE-PROCESS
    :OUTPUTS   ((0.5 PARTICIPLE-PASSIVE
               (CLASSIFY VOICE SAM-AUX)
               (INSERT VOICEDEPENDENT)
               (INFLECTIFY VOICEDEPENDENT PASTPARTICIPLE))
               (0.5 MEDIO-PASSIVE
               (CONFLATE FINITE PROCESS)
               (INSERT REFLEXIVEPARTICLE)
               (LEXIFY REFLEXIVEPARTICLE SE)))
    :CHOOSER   PASSIVE-PROCESS-TYPE-CHOOSER
    :REGION   VOICE
)
```

Figure 28: The types of passive-processes in Czech and Bulgarian

In Czech and Bulgarian, the medio-passive is realized by insertion of a reflexive particle (a clitic). This clitic can be ordered in different places and there are different possible orderings for Czech and Bulgarian. Bulgarian orders the reflexive particle after the Finite as default, and Czech orders the reflexive particle after the first constituent in a clause. Two systems account for this (see Figure 29).

```
(SYSTEM
    :NAME   MEDIO-PASSIVE-ORDER (:CZECH)
    :INPUTS   (AND MEDIO-PASSIVE THEME-ORDER)
    :OUTPUTS  ((0.5 THEME-REFLEXIVEPARTICLE
              (ORDER THEME REFLEXIVEPARTICLE))
              (0.5 REFLEXIVEPARTICLE-THEME
              (ORDER REFLEXIVEPARTICLE THEME)))
    :CHOOSER   MEDIO-PASSIVE-ORDER-CHOOSER
    :REGION   voice
)

(SYSTEM
    :NAME   MEDIO-PASSIVE-ORDER (:BULGARIAN)
    :INPUTS   (MEDIO-PASSIVE)
    :OUTPUTS   ((0.5 finite-REFLEXIVEPARTICLE
               (ORDER finite REFLEXIVEPARTICLE))
               (0.5 REFLEXIVEPARTICLE-finite
               (ORDER REFLEXIVEPARTICLE finite)))
    :CHOOSER   MEDIO-PASSIVE-ORDER-CHOOSER
    :REGION   voice
)
```

Figure 29: Ordering of the reflexive clitic in Czech and Bulgarian

2.3 Minor transitivity

This section documents the system network for the prepositional phrase (PP). An analysis of the current domain (SPEC2; Andonova *et al.,* 1999) has revealed that only a certain subset of prepositional phrases are used in the intermediate demonstrator texts and in the register of instructional texts more generally. The described network draws upon the English Nigel grammar's prepositional phrase rank, exploiting cross-linguistic similarities as much as possible. In this chapter we first present a short summary of the differences between the Slavic languages under investigation in this region of grammar (Section 2.3.1). Then we present the implementation of the PP network focusing on realizational details (Section 2.3.2).

2.3.1 The prepositional phrase in Bulgarian, Czech and Russian

Prepositional phrases are treated as grammatical realizations of the Upper Model semantic relation of circumstantial. They are semantically represented as two place relations between a domain (a process or an object) and a range (an object). Following SFG terminology, in the grammar they are presented as the functional elements of Minorprocess (realized by a preposition) and Minirange (typically realized as a nominal group).

A general discussion of prepositional phrases in Slavic languages has been presented in the SPEC2 deliverable (Andonova *et al.* 1999). In Czech and Russian, prepositions are a case-governing word class - so the selection of a preposition for realizing the Minorprocess structural element is directly followed by a case inflection constraint on the Minirange of the PP. Also, for Russian and Czech the same preposition can select different cases. For example, the Russain preposition *v* may select locative or accusative case, depending on whether the process denotes location (locative case) or destination (accusative case). See example (18a) which illustrates the destination reading and (18b) illustrating the location reading. Examples (18c) and (18d) illustrate the location reading (with two prepositions: v and na) in Czech, and (18e) illustrates the Czech destination reading

(18) *Spatial-locating vs. Destination (Russian, Czech)*

(a) Ru: Добавьте элементы в стиль S1

Add elements-acc into the style-acc S1.1.

(b) Ru: В стиле S1 добавьте элементы в подтип S1.1.

In the style-loc S1 add elements-acc into the subtype-acc S1.1.

(c) Cz: V dialogovém okně Style zadejte s.

In dialogue-adj-loc box-loc Style enter s

(d) Cz: Na příkazovém řádku zadejte s

At command-adj-loc line-loc enter s

(e) Cz: Přeměňte čtyři úsečky vlna obdélník

Change four lines into rectangle-acc

Bulgarian resembles English: Since it does not have case, prepositions can be ambigous as to whether they denote location or destination. In (19a) *в стила S* is interpreted as destination because the verb implies directionality, and *в стила S* in (19b) is interpreted as location, since a destination is already expressed by *към*.

(19) *Spatial-locating vs. Destination*

(a) Bg: Добавете елементи в стила S

Add elements to the style S1

(b) Bg: В стила S добавете елементи към стила S1

In the style S add elements to the subtype S1

Since the differences between Russian, Bulgarian and Czech only lie in realization, we can use the same functional classification, i.e., the same system network.

Problems arise when the same meaning is expressed at different ranks in different languages. A case in point is Czech and Russian using a genitive construction for expressing the meaning of 'part-of' and Bulgarian using a prepositional phrase (again like English):

(20) *PP vs. NG (genitive)*

(a) Cz: Zadá se koncový bod oblouku.

Specify the endpoint-nom arc–gen

(b) Ru: Задается конечная точка дуги

Specify the endpoint-nom arc–gen

(c) Bg: Задава се крайната точка на дъгата

Specify the endpoint of the arc

Since we only deal with the structural unit of the prepositional phrase here, we have to ignore this issue for the time being.

In the following sections we present the implementation of the prepositional phrase by first giving the most general types (Section 2.3.2), and then moving to the two regions the prepositional phrase network is composed of: PPSPATIOTEMPORAL (Section 2.3.2.1) and PPOTHER (Section 2.3.2.2).

2.3.2 Types of prepositional phrases

The most general system at prepositional phrase rank is the system MINOR-PROCESS-TYPE (see Figure 30).

```
(SYSTEM
    :NAME   MINOR-PROCESS-TYPE
    :INPUTS   PREPOSITIONAL-PHRASE
    :OUTPUTS   ((0.125 SPATIO-TEMPORAL-PROCESS)
               (0.125 INSTRUMENTAL-PROCESS)
               (0.125 COMPARATIVE-PROCESS)
               (0.125 CAUSAL-PROCESS)
               (0.125 ACCOMPANIMENT-PROCESS)
               (0.125 MATTER-PROCESS)
               (0.125 ROLE-PROCESS
               (CLASSIFY MINORPROCESS PREPOSITION)
               (CLASSIFY MINORPROCESS ROLE-VERB))
               (0.125 PORTION-PROCESS
          (:BULGARIAN (LEXIFY MINORPROCESS HA))
)
    :CHOOSER   MINOR-PROCESS-TYPE-CHOOSER
    :REGION   PPOTHER
    :METAFUNCTION   IDEATIONAL
)
```

Figure 30: Minor process type system for (En, Bg, Cz and Ru)

The feature **spatio-temporal-process** is the entry condition ot the systems of the PPSPATIOTEMPORAL region. The other features are entry conditions to PPOTHER systems.

2.3.2.1 The PPSPATIOTEMPORAL region

The top level distinction in the system network of spatiotemporal PPs is **spatial-process** and **temporal process**. In this section we only consider spatial processes (the most frequently occurring type of prepositional phrase in the ImD texts is spatial-process, temporal processes do not occur).

The other major distinction made in the PPSPATIOTEMPORAL region are extent/location and spatial/temporal. This cross-classification is reflected by the systems SPATIO-TEMPORAL, EXTENT-PROCESS-TYPE and SPATIO-TEMPORAL-TYPE. They are presented in Figure 31.

```
(SYSTEM    :NAME   SPATIO-TEMPORAL    :INPUTS   SPATIO-TEMPORAL-PROCESS
    :OUTPUTS   ((0.5 EXTENT-PROCESS)
               (0.5 LOCATION-PROCESS))
    :CHOOSER   SPATIO-TEMPORAL-CHOOSER
    :REGION   PPSPATIOTEMPORAL
    :METAFUNCTION   IDEATIONAL
)
(CHOOSER
    :NAME   SPATIO-TEMPORAL-CHOOSER
    :DEFINITION   ((ASK (EXTENT-Q MINORPROCESS)
                  (EXTENT (CHOOSE EXTENT-PROCESS))
                  (NONEXTENT (CHOOSE LOCATION-PROCESS))))
)

(SYSTEM
    :NAME   EXTENT-PROCESS-TYPE
    :INPUTS   EXTENT-PROCESS
    :OUTPUTS   ((0.5 ABSOLUTE-EXTENT-PROCESS)
               (0.5 RELATIVE-EXTENT-PROCESS))
    :CHOOSER   EXTENT-PROCESS-TYPE-CHOOSER
    :REGION   PPSPATIOTEMPORAL
    :METAFUNCTION   IDEATIONAL
)
(CHOOSER
    :NAME   EXTENT-PROCESS-TYPE-CHOOSER
    :DEFINITION   ((ASK (ABSOLUTE-EXTENT-Q MINORPROCESS MINIRANGE)
                  (ABSOLUTE (CHOOSE ABSOLUTE-EXTENT-PROCESS))
                  (NONABSOLUTE (CHOOSE RELATIVE-EXTENT-PROCESS)))))

(SYSTEM
    :NAME   SPATIO-TEMPORAL-TYPE
    :INPUTS   (OR LOCATION-PROCESS RELATIVE-EXTENT-PROCESS)
    :OUTPUTS  ((0.5 SPATIAL-PROCESS)
              (0.5 TEMPORAL-PROCESS))
    :CHOOSER   SPATIO-TEMPORAL-TYPE-CHOOSER
    :REGION   PPSPATIOTEMPORAL
    :METAFUNCTION   IDEATIONAL
)
(CHOOSER
    :NAME   SPATIO-TEMPORAL-CHOOSER
    :DEFINITION   ((ASK (EXTENT-Q MINORPROCESS)
                  (EXTENT (CHOOSE EXTENT-PROCESS))
                  (NONEXTENT (CHOOSE LOCATION-PROCESS)))))
```

Figure 31:The top distinctions in the spatiotemporal region, systems and choosers (Cz, Ru, Bg)

Relations expressing spatial locations are further classified with respect to two pairs of features: On the one hand they are subdivided into relations that indicate an object's position with respect to an axis (grammatical feature **orientation-axis**). This choice is covered by the system ORIENTATION-AXIS-TYPE presented in Figure 32 . On the other hand, spatial locating relations are split into relations that encode the idea of a movement and those that imply the idea of rest (features **motion-process** and **rest-process**). The relevant systems are shown in Figure 33 . They provide the features which form the entry conditions to three other areas of the PPSPATIOTEMPORAL network – **orientation rest processes**, **nonorientation rest processes** and **motion processes**. In this Section we consider only the latter two, since they are the only ones occurring in the ImD texts.

```
(SYSTEM   :NAME   ORIENTATION-AXIS
    :INPUTS   (AND LOCATION-PROCESS SPATIAL-PROCESS)
    :OUTPUTS   ((0.5 ORIENTATION-AXIS)  (0.5 NONORIENTATION-AXIS))
    :CHOOSER   ORIENTATION-AXIS-CHOOSER
    :REGION   PPSPATIOTEMPORAL
    :METAFUNCTION   IDEATIONAL
)
(CHOOSER
    :NAME   ORIENTATION-AXIS-CHOOSER
    :DEFINITION   ((ASK (ORIENTATION-Q MINORPROCESS MINIRANGE)
                  (ORIENTED (CHOOSE ORIENTATION-AXIS))
                  (NONORIENTED (CHOOSE NONORIENTATION-AXIS)))))
```

Figure 32: System and chooser orientation-axis (Bg, Cz, Ru)

```
(SYSTEM
    :NAME   LOCATION-PROCESS-TYPE
    :INPUTS   (AND LOCATION-PROCESS SPATIAL-PROCESS)
    :OUTPUTS   ((0.5 REST-PROCESS (PRESELECT MINIRANGE PREPOSITIONAL))
               (0.5 MOTION-PROCESS))
    :CHOOSER   LOCATION-PROCESS-TYPE-CHOOSER
    :REGION   PPSPATIOTEMPORAL
    :METAFUNCTION   IDEATIONAL
)
(CHOOSER
    :NAME   LOCATION-PROCESS-TYPE-CHOOSER
    :DEFINITION   ((ASK (SOURCE-DESTINATION-PROCESS-Q MINORPROCESS
                    MINIRANGE)
                  (SOURCEDESTINATION (CHOOSE MOTION-PROCESS))
                  (NONSOURCEDESTINATION (CHOOSE REST-PROCESS)))))
```

Figure 33: System and chooser location-process-type (Bg, Cz, Ru)

In the area of nonorientation rest processes, the choice of a preposition interacts with the dimensional properties of the object that realizes the Minirange. In the English model three types are distinguished: **(***at)* **zero-dimension, (***on)* **one-two-dimensions, (***in)* **three-dimensions.** In principle, the three Slavic languages only show a distinction between **three-dimensions** and **one-two-dimensions**. There is a preposition that is used with **zero-dimension** objects, but its use is more restricted than the English *at*. It is important that accordingly in terms of the frequency of object types in Russian, the more frequent type in Russian is **three-dimensions**. A sample semantic input representation is shown in Figure 34.

```
(EXAMPLE
    :NAME   D1-TEXT4-4-RU
    :TARGETFORM   "V okne vyberite punkt Display joints, chtoby
                   otobrazitj liniju u vershin muljtilinii."
    :LOGICALFORM
          (R / RST-PURPOSE :DOMAIN
          (P / DIRECTED-ACTION :LEX VYBRATJ
    :SPEECHACT IMPERATIVE
    :THEME D
    :ACTEE
          (A1 / OBJECT :LEX PUNKT :LABEL-ASCRIPTION
          (A2 / OBJECT :NAME DISPLAY-JOINTS))
    :SPATIAL-LOCATING
          (D / THREE-D-LOCATION :LEX OKNO ))
    :RANGE
          (E / DIRECTED-ACTION :LEX OTOBRAZITJ :ACTEE
          (L / OBJECT :LEX LINIJA :SPATIAL-LOCATING
          (V / ZERO-D-LOCATION :LEX VERSHINA :MULTIPLICITY-Q MULTIPLE
    :META-ACTANT
          (M1 / OBJECT :LEX MULJTILINIJA)))))
    :SET-NAME   D1-TEXT4
)
```

Figure 34: Semantic representation with zero-dimension (Ru)

A number of gates then serve the realization of the Minorprocess by particular lexical items (prepositions). See, for example, the gates AT, IN and ON presented in Figure 35.

```
(GATE
    :NAME   AT
    :INPUTS   (OR (AND ZERO-DIMENSION REST-PROCESS)
               NONINCLUSIVE)
    :OUTPUTS   ((1.0 AT
               (LEXIFY MINORPROCESS U)
               (PRESELECT MINIRANGE RGENITIVE)))
    :REGION   PPSPATIOTEMPORAL
    :METAFUNCTION   IDEATIONAL
)
 (GATE
    :NAME   IN
    :INPUTS   (OR CONTAINMENT-IMPLICIT STRONG-INCLUSIVE IN-EXTENT)
    :OUTPUTS  ((1.0 IN
              (LEXIFY MINORPROCESS V)
        (:CZECH (PRESELECT MINIRANGE LOCAL))
        (:RUSSIAN (PRESELECT MINIRANGE PREPOSITIONAL[5]))
))
    :REGION   PPSPATIOTEMPORAL
    :METAFUNCTION   IDEATIONAL
)
(GATE
    :NAME   ON
    :INPUTS   (OR WEAK-INCLUSIVE
              (AND ONE-TWO-DIMENSIONS REST-PROCESS))
    :OUTPUTS  ((1.0 ON
              (LEXIFY MINORPROCESS NA)
        (:CZECH (PRESELECT MINIRANGE LOCAL))
        (:RUSSIAN (PRESELECT MINIRANGE PREPOSITIONAL))
))
    :REGION   PPSPATIOTEMPORAL
    :METAFUNCTION   IDEATIONAL
)
```

Figure 35: Gates realizing location prepositions
according to the dimensions of the object (Bg, Cz, Ru)

Motion processes are further classified according to the opposition source vs. destination. The relevant prepositions describe a dynamic meaning, one reflecting movement. This classification is triggered by the feature **motion-process**. The two major subtypes are relations denoting a movement **towards** a place/object and those indicating a movement **away-from** it. The system is shown in Figure 36.

[5] PREPOSITIONAL is the label for a morphological case, which is the traditional name for what has been called ‘locative’ in the text.

```
(SYSTEM
    :NAME   MOTION-PROCESS-TYPE
    :INPUTS   MOTION-PROCESS
    :OUTPUTS   ((0.5 AWAY-FROM)  (0.5 TOWARDS))
    :CHOOSER   MOTION-PROCESS-TYPE-CHOOSER
    :REGION   PPSPATIOTEMPORAL
    :METAFUNCTION   IDEATIONAL
)
(CHOOSER    :NAME   MOTION-PROCESS-TYPE-CHOOSER
    :DEFINITION   ((ASK (SOURCE-PROCESS-Q MINORPROCESS MINIRANGE)
                  (SOURCE (CHOOSE AWAY-FROM))
                  (NONSOURCE (CHOOSE TOWARDS)))))
```

Figure 36: Motion-process-type system and chooser (Bg, Cz, Ru)

The pairs of gates, realizing the dynamic prepositions for the one-two-dimensions objects and three-dimensions objects are shown in Figures 37 and 38. Instances of dynamic meanings with zero- dimension source objects are not present in the ImD texts. The gate realizing zero-dimension destination is shown in Figure 39.

```
(GATE
    :NAME   ONTO
    :INPUTS   (AND ONE-TWO-DIMENSIONS TOWARDS)
    :OUTPUTS   ((1.0 ONTO      (LEXIFY MINORPROCESS NA)
       (:RUSSIAN :CZECH (PRESELECT MINIRANGE ACCUSATIVE))
))
    :REGION   PPSPATIOTEMPORAL
    :METAFUNCTION   IDEATIONAL
)

(GATE
    :NAME   OFF
    :INPUTS    (AND AWAY-FROM ONE-TWO-DIMENSIONS)
    :OUTPUTS   ((1.0 OFF
       (:BULGARIAN (LEXIFY MINORPROCESS OT))
       (:CZECH (LEXIFY MINORPROCESS Z)
               (PRESELECT MINIRANGE GENITIVE-CASE))
       (:RUSSIAN  (LEXIFY MINORPROCESS S)
               (PRESELECT MINIRANGE RGENITIVE))
))
    :REGION   PPOTHER
    :METAFUNCTION   IDEATIONAL
)
```

Figure 37: One-two-dimensional dynamic prepositions (Bg, Cz, Ru)

```
(GATE
    :NAME   INTO
    :INPUTS    (AND TOWARDS THREE-DIMENSIONS)
    :OUTPUTS   ((1.0 THREE-DIMENSIONAL-MOTION
               (LEXIFY MINORPROCESS V)
        (:CZECH :RUSSIAN (PRESELECT MINIRANGE ACCUSATIVE))
))
    :REGION   PPSPATIOTEMPORAL
    :METAFUNCTION   IDEATIONAL
)

(GATE
    :NAME   FROM
    :INPUTS    (AND AWAY-FROM THREE-DIMENSIONS)
    :OUTPUTS   ((1.0 FROM
        (:BULGARIAN (LEXIFY MINORPROCESS NA))
        (:CZECH (PRESELECT MINIRANGE GENITIVE-CASE)
               (LEXIFY MINORPROCESS Z))
        (:RUSSIAN (PRESELECT MINIRANGE RGENITIVE)
               (LEXIFY MINORPROCESS IZ))

))
    :REGION   PPSPATIOTEMPORAL
    :METAFUNCTION   IDEATIONAL
)
```

Figure 38: Three-dimensional dynamic prepositions (Bg, Cz, Ru)

```
(GATE
    :NAME   TO
    :INPUTS   (AND ZERO-DIMENSION TOWARDS NONORIENTATION-AXIS)
    :OUTPUTS   ((1.0 TO
      (:BULGARIAN (LEXIFY MINORPROCESS КЪМ)
      (:CZECH (LEXIFY MINORPROCESS K)
              (PRESELECT MINIRANGE DATIVE-CASE))
      (:RUSSIAN (LEXIFY MINORPROCESS K)
              (PRESELECT MINIRANGE DATIVE-CASE))
))
    :REGION   PPSPATIOTEMPORAL
    :METAFUNCTION   IDEATIONAL
)
```

Figure 39: Zero-dimensional dynamic destination preposition (Bg, Cz, Ru)

Extent processes form the third subregion of the PPSPATIOTEMPORAL region we deal with in the ImD. Extent process is triggered in the system SPATIO-TEMPORAL shown earlier and then elaborated in two systems shown in Figure 40.

```
(SYSTEM
    :NAME   EXTENT-PROCESS-TYPE
    :INPUTS   EXTENT-PROCESS
    :OUTPUTS   ((0.5 ABSOLUTE-EXTENT-PROCESS)  (0.5 RELATIVE-EXTENT-
                 PROCESS))
    :CHOOSER   EXTENT-PROCESS-TYPE-CHOOSER
    :REGION   PPSPATIOTEMPORAL
    :METAFUNCTION   IDEATIONAL
)
(CHOOSER
    :NAME   EXTENT-PROCESS-TYPE-CHOOSER
    :DEFINITION   ((ASK (ABSOLUTE-EXTENT-Q MINORPROCESS MINIRANGE)
                  (ABSOLUTE (CHOOSE ABSOLUTE-EXTENT-PROCESS))
                  (NONABSOLUTE (CHOOSE RELATIVE-EXTENT-PROCESS)))))

(SYSTEM
    :NAME   EXTENT-PARALLELISM
    :INPUTS   (AND RELATIVE-EXTENT-PROCESS SPATIAL-PROCESS)
    :OUTPUTS   ((0.5 RELATIVE-PARALLEL
        (:ENGLISH  (LEXIFY MINORPROCESS ALONG))
        (:RUSSIAN  (LEXIFY MINORPROCESS PO)
                   (PRESELECT MINIRANGE DATIVE)
        (:BULGARIAN  (LEXIFY MINORPROCESS ПО))
))
                   (0.5 RELATIVE-NONPARALLEL
        (:ENGLISH  (LEXIFY MINORPROCESS ACROSS))
        (:RUSSIAN  (LEXIFY MINORPROCESS CHEREZ)
                   (PRESELECT MINIRANGE ACCUSATIVE))))
    :CHOOSER   EXTENT-PARALLELISM-CHOOSER
    :REGION   PPSPATIOTEMPORAL
    :METAFUNCTION   IDEATIONAL
)
(CHOOSER
    :NAME   EXTENT-PARALLELISM-CHOOSER
    :DEFINITION   ((ASK (PARALLEL-Q MINORPROCESS MINIRANGE)
                  (PARALLEL (CHOOSE RELATIVE-PARALLEL))
                  (NONPARALLEL (CHOOSE RELATIVE-NONPARALLEL)))))
```

Figure 40: Systems and choosers realizing extent-processes (Bg and Ru)

2.3.2.2 The PPOTHER region

In the PPOTHER grammar region we have four types of processes relevant for the ImD. They are accompaniment, causal (purpose), instrumental and portion processes. Accompaniment and portion processes require some language-specific changes.

In the ImD text, there is a minor process type that is realized in English with the preposition *in relation to*. The type coming closest to this is accompaniment-process. We therefore add a new subtype to accompaniment-process called **relative-additive**. This modification is shown in Figure 41.

```
(SYSTEM
    :NAME   ACCOMPANIMENT-PROCESS-TYPE
    :INPUTS   ACCOMPANIMENT-PROCESS
    :OUTPUTS   ((0.25 INCLUDING-COMITATIVE)
               (0.25 EXCLUDING-COMITATIVE
               (LEXIFY MINORPROCESS WITHOUT))
               (0.25 CUMULATIVE-ADDITIVE
               (LEXIFY MINORPROCESS ASWELLAS))
               (0.25 ALTERNATIVE-ADDITIVE
               (LEXIFY MINORPROCESS INSTEADOF))
               (0.25 RELATIVE-ADDITIVE
               (LEXIFY MINORPROCESS INRELATIONTO)))
    :CHOOSER   ACCOMPANIMENT-PROCESS-TYPE-CHOOSER
    :REGION   PPOTHER
    :METAFUNCTION   IDEATIONAL
)
```

Figure 41: The updated accompaniment system and chooser for English

For Russian, the type cumulative-additive is not needed, however. The meaning reflected here is realized by coordination (see example (21)).

(21) *The meaning of cumulative-additive expressed as coordination*

(a) Ru: Он любит меня также как и тебя

He loves me-acc as well as you-acc

(b) Ru: Он взял сына вместо дочери

He took the son-acc instead of the daughter-gen

The examples show that the relation of cumulative-additive, realized in Nigel by a prepositional phrase with the preposition *as well as* is presented in Russian as syntactic coordination, reflected by the case marking: in (21a), меня and тебя have the same case (governed by the verb), which indicates coordination of the two in one complex nominal group, wheras in (21b) the preposition вместо governs its own case. Thus, for Russian we omit the cumulative-additive feature from the ACCOMPANIMENT-PROCESS-TYPE system. The system and chooser for Russian is shown in Figure 42.

```
(SYSTEM
    :NAME   ACCOMPANIMENT-PROCESS-TYPE
    :INPUTS   ACCOMPANIMENT-PROCESS
    :OUTPUTS    ((0.25 INCLUDING-COMITATIVE)
                 (0.25 EXCLUDING-COMITATIVE
                 (LEXIFY MINORPROCESS BEZ)
                (PRESELECT MINIRANGE RGENITIVE))
                (0.25 RELATIVE-ADDITIVE
           (:RUSSIAN (LEXIFY MINORPROCESS PO-OTNOSHENIJU-K)
                (PRESELECT MINIRANGE DATIVE))
           (:BULGARIAN (LEXIFY MINORPROCESS СПРЯМО))
           (:CZECH (LEXIFY MINORPROCESS VE-VZTAHU-K)
                (PRESELECT MINIRANGE DATIVE-CASE))
  )
                (0.25 ALTERNATIVE-ADDITIVE
                (LEXIFY MINORPROCESS VMESTO)
                (PRESELECT MINIRANGE RGENITIVE)))
    :CHOOSER   ACCOMPANIMENT-PROCESS-TYPE-CHOOSER
    :REGION   PPOTHER
    :METAFUNCTION   IDEATIONAL
)

(CHOOSER
    :NAME   ACCOMPANIMENT-PROCESS-TYPE-CHOOSER
    :DEFINITION   ((ASK (PARTICIPATION-Q MINORPROCESS MINIRANGE)
                    (PARTICIPATION (CHOOSE INCLUDING-COMITATIVE))
                    (NONPARTICIPATION
                    (ASK (ADDITION-Q MINORPROCESS MINIRANGE)
                    (ADDITION (CHOOSE RELATIVE-ADDITIVE))
                    (NONADDITION
                    (ASK (ALTERNATIVE-Q MINORPROCESS MINIRANGE)
                    (ALTERNATIVE (CHOOSE ALTERNATIVE-ADDITIVE))
                    (NONALTERNATIVE (CHOOSE EXCLUDING-COMITATIVE)))))))

)
```

Figure 42: The accompaniment process type system and chooser: (Cz, Ru, Bg)

In Figure 43 we show an SPL input expression generating an additive relation with the preposition ‘in relation to’(Ru: po otnosheniju k).

```
(EXAMPLE
    :NAME   D1-TEXT2-11-RU
    :GENERATEDFORM   "Ukazhite ugol linii po otnosheniju k konechnoj
                      tochke dugi."
    :TARGETFORM   "Ukazhite ugol linii po otnosheniju k konechnoj
                   tochke dugi."
    :LOGICALFORM
        (P / DIRECTED-ACTION :LEX UKAZATJ :SPEECHACT IMPERATIVE :ACTEE
        (O / OBJECT :LEX UGOL :META-ACTANT
        (M1 / OBJECT :LEX LINIJA))
    :ADDITIVE
        (D / OBJECT :LEX TOCHKA :PROPERTY-ASCRIPTION
        (Q1 / QUALITY :LEX KONECHNYJ)
    :META-ACTANT
        (M2 / OBJECT :LEX DUGA)))
    :SET-NAME   D1-TEXT2)
```

Figure 43: Semantic representation for the relative-additive function (Ru).

Causal processes are selected identically for the three Slavic languages and English. The system CAUSAL-PROCESS-TYPE is presented in Figure 44.

```
(SYSTEM
    :NAME   CAUSAL-PROCESS-TYPE
    :INPUTS   CAUSAL-PROCESS
    :OUTPUTS   ((0.25 REASON-PROCESS
               (CLASSIFY MINORPROCESS PREPOSITION)
               (CLASSIFY MINORPROCESS REASON-PREP))
               (0.25 PURPOSIVE-PROCESS)
               (0.25 CLIENT)
               (0.25 CONCESSIVE-PROCESS
               (LEXIFY MINORPROCESS DESPITE)))
    :CHOOSER   CAUSAL-PROCESS-TYPE-CHOOSER
    :REGION   PPOTHER
    :METAFUNCTION   IDEATIONAL
)
(CHOOSER
    :NAME   CAUSAL-PROCESS-TYPE-CHOOSER
    :DEFINITION   ((ASK (PURPOSE-Q MINORPROCESS)
                  (NONPURPOSE
                  (ASK (REASON-Q MINORPROCESS)
                  (NONREASON
                  (ASK (BEHALF-Q MINORPROCESS)
                  (NONBEHALF (CHOOSE CONCESSIVE-PROCESS))
                  (BEHALF (CHOOSE CLIENT))))
                  (REASON (CHOOSE REASON-PROCESS))))
                  (PURPOSE (CHOOSE PURPOSIVE-PROCESS)))))
```

Figure 44: The causal-process-type system and chooser for (Cz, Ru, Bg)

The realization of purposive-process is accounted for in the FOR gate (Figure 45). For Slavic languages it differs from English in the input conditions and in lexicalization statements. In the input slot, the conditional feature extent-process has been omitted, because the prepositions corresponding to English *for* in Czech, Bulgarian and Russian do not denote the meaning of extent:

(Ru) для меня – (En) for me (client)

(Ru) для начальной точки – (En) for the startpoint (purposive-process)

(Ru) *для на три дня – (En) for three days (temporal-extent)

The gate FOR is presented in Figure 45.

```
(GATE
    :NAME   FOR
    :INPUTS   ((:ENGLISH(OR CLIENT ABSOLUTE-EXTENT-PROCESS
                PURPOSIVE-PROCESS))
               (:BULGARIAN(OR CLIENT PURPOSIVE-PROCESS))
               (:CZECH(OR CLIENT PURPOSIVE-PROCESS))
               (:RUSSIAN(OR CLIENT PURPOSIVE-PROCESS)))
    :OUTPUTS   ((1.0 FOR
               (:BULGARIAN(LEXIFY MINORPROCESS ЗА))
               (:CZECH(LEXIFY MINORPROCESS PRO)
               (PRESELECT MINIRANGE ACCUSATIVE-CASE))
               (:RUSSIAN(LEXIFY MINORPROCESS DLJA)
               (PRESELECT MINIRANGE RGENITIVE)))
    :REGION   PPOTHER
    :METAFUNCTION   IDEATIONAL)
```

Figure 45: The gate FOR realizing purpose process (Cz, Ru, Bg)

Instrumental processes are further subclassified in the system INSTRUMENTAL-PROCESS-TYPE. In the intermediate demonstrator texts, the only type of instrumental process that occurs is **agentive-instrument**. The system is presented in Figure 46.

```
(SYSTEM
    :NAME   INSTRUMENTAL-PROCESS-TYPE
    :INPUTS   INSTRUMENTAL-PROCESS
    :OUTPUTS   ((0.33333334 MANIPULATING-INSTRUMENT)
               (0.33333334 ENABLING-INSTRUMENT)
               (0.33333334 AGENTIVE-INSTRUMENT))
    :CHOOSER   INSTRUMENTAL-PROCESS-TYPE-CHOOSER
    :REGION   PPOTHER
    :METAFUNCTION   IDEATIONAL
)
(CHOOSER
    :NAME   INSTRUMENTAL-PROCESS-TYPE-CHOOSER
    :DEFINITION   ((ASK (MANIPULATION-Q MINORPROCESS MINIRANGE)
                  (MANIPULATION (CHOOSE MANIPULATING-INSTRUMENT))
                  (NONMANIPULATION
                  (ASK (ENABLING-Q MINORPROCESS MINIRANGE)
                  (ENABLING (CHOOSE ENABLING-INSTRUMENT))
                  (NONENABLING
                  (ASK (AGENTIVE-Q MINORPROCESS MINIRANGE)
                  (AGENTIVE (CHOOSE AGENTIVE-INSTRUMENT))))))))
```

Figure 46: System and chooser for instrumental-process-type (Cz, Ru, Bg)

The realization of the agentive-instrument process for Russian and for English are shown in Figure 47. A sample SPL for generating an instrumental Circumstance is shown in Figure 48.

```
(GATE
    :NAME   BY
    :INPUTS   (OR NONEXPLICIT-ENABLING AGENTIVE-INSTRUMENT)
    :OUTPUTS   (1.0 BY
       (:ENGLISH   (LEXIFY MINORPROCESS BY))
       (:BULGARIAN (LEXIFY MINORPROCESS C))
       (:RUSSIAN   (LEXIFY MINORPROCESS ELLIPSISZERO)
                   (PRESELECT MINIRANGE INSTRUMENTAL))
)
    :REGION   PPOTHER
    :METAFUNCTION   IDEATIONAL
)
```

Figure 47: Gate for agentive-instrument realization (Bg, En and Ru)

```
(EXAMPLE
    :NAME   D1-TEXT2-2-RU
    :TARGETFORM   "ZAPUSTITE KOMANDU PLINE SLEDUJUSCHIM SPOSOBOM"
    :LOGICALFORM
        (P / DIRECTED-ACTION :LEX ZAPUSTITJ :SPEECHACT IMPERATIVE :ACTEE
        (D / OBJECT :LEX KOMANDA :LABEL-ASCRIPTION
        (L1 / OBJECT :NAME PLINE))
    :GENERALIZED-MEANS
        (M / ABSTRACTION :LEX SPOSOB :PROPERTY-ASCRIPTION
        (F / QUALITY :LEX SLEDUJUSCHIJ)))
    :SET-NAME   D1-TEXT2
 )
```

Figure 48: Semantic representation for agentive-instrument function.

Finally, we have to account for the portion-process type for Bulgarian. Recall that for Czech and Russian, the meaning of ‘part-of’ is realized by a nominal group in genitive case. The realization of the feature **portion-process** for Bulgarian is described in the system MINOR-PROCESS-TYPE shown in Figure 30.

2.4 Mood

The present chapter describes the implementation of the MOOD region for Bulgarian, Czech and Russian. The linguistic properties of mood in Czech, Bulgarian and Russian have been presented in the SPEC2 deliverable (Andonova *et al.*, 1999). On the basis of the Nigel grammar a number of modifications have been introduced in order to generate imperative and indicative clauses in Bulgarian, Czech and Russian for the purposes of the intermediate demonstrator. We describe the process of generation in the MOOD region and present the systems and choosers which have been changed or added.

The region’s topmost system is MOOD-TYPE which divides clauses into **indicative** and **imperative**. One main difference emerges in the comparison of the English with the Bulgarian/ Russian/ Czech imperative clause. In English imperative sentences, a nonfinite form of the verb (infinitive) is used, whereas in Slavic languages imperatives are expressed by a finite verb form (2nd person). In order to account for this, we insert Finite in the **imperative** output of the MOOD-TYPE system as follows:

MOOD-TYPE:
    (independent-clause-simplex) →
        [indicative],
        [imperative] (+Finite).

The corresponding system and chooser implementations are shown in Figure 49.

```
(SYSTEM
    :NAME   MOOD-TYPE
    :INPUTS   INDEPENDENT-CLAUSE-SIPLEX
    :OUTPUTS   ((0.5 IMPERATIVE
              (INSERT FINITE))
              (0.5 INDICATIVE))
    :CHOOSER   MOOD-TYPE-CHOOSER
    :REGION   MOOD
    :METAFUNCTION   INTERPERSONAL
)

(CHOOSER
    :NAME   MOOD-TYPE
    :DEFINITION   ((ASK (COMMAND-Q SPEECHACT)
                  (COMMAND (CHOOSE IMPERATIVE)
                  (IDENTIFY SUBJECT
                  (COMMAND-RESPONSIBLE-ID SPEECHACT)))
      (NOCOMMAND  (CHOOSE INDICATIVE)))
))
```

Figure 49: Mood type: system and chooser (Cz, Ru, Bg)

In the following sections, we shall discuss the details of **indicative** clause processing (Section 2.4.1) and then we present the implementations in the grammar which serve to realized imperative clauses in the three languages (Section 2.4.2).

2.4.1 Indicative Clauses

The three Slavic languages use the same system of differentiating indicative clauses into **interrogative** and **declarative** as presented in English:

INDICATIVE-TYPE:

(indicative) →
[declarative] , [interrogative]

In the genre of written software instructions, there are typicaly no **interrogative** clauses and there are no such occurrences in the AGILE corpus. Therefore, we do not consider the interrogative branch of the network any further here.

Declarative clauses occur most frequently in the texts when the text is in the so-called nonpersonal style (cf. TEXS2 deliverable; Kruijff-Korbayova *et al.*, 1999). In this style, all clauses are declarative and use the medio-passive construction, which is specific to Slavic languages (cf. Section 2.3). Example (22) below shows the correspondences to English imperative clauses.

(22) *Declarative clause in medio/reflexive passive (text 4)*

(a) En: Under Name, enter the name of the style.

(b) Cz: Pod Jméno se zadá název stylu.

Under Name, enter-refl-passive3sg the name of the style.

(c) Bg: В полето Name се въвежда име на стила.

Under Name, enter-refl-passive3sg the name of the style.

(d) Ru: В пункте Name задается имя стиля.

Under Name, enter-refl-passive3sg the name of the style.

Clauses expressing side effect are also in declarative. See for example (23):

(23) *Declarative clause Side Effect (text 1)*

(a) En: The Select Color dialog box appears.

(b) Cz: Objeví se dialogové okénko Select Color.

Appear-refl3sg dialog box Select Color.

(c) Bg: Диалоговият прозорец Select Color се появява на екрана.

The dialog box Select Color appear-refl3sg on the screen.

(d) Ru: На экране появится диалоговое окно Select Color.

on the screen appear-refl3sg the dialog box Select Color

A major system of the grammar network for indicative clauses is the FINITE-CLAUSE SYSTEM which introduces the Mood element. By means of the Expand realization statement Mood (Finite) in this system, the clause structure of indicative clauses is built up with a constituent labeled Mood which has a sub-constituent labeled Finite. The system is shown below. Its implementation is given in Figure 50:

FINITE-CLAUSE:

(indirect-indicative; finite-enhancing; finite-elaborating;
finite-extending; finite-rankshift; indicative) →
[Finite-clause]
(+Mood, Mood(Finite))

```
 (GATE
    :NAME   FINITE-CLAUSE
    :INPUTS   (OR INDIRECT-INDICATIVE FINITE-ENHANCING FINITE-ELABORATING
               FINITE-EXTENDING INDICATIVE FINITE-RANKSHIFT)
    :OUTPUTS  ((1.0 FINITE-CLAUSE
              (INSERT MOOD)
              (EXPAND MOOD FINITE)))
    :REGION   DEPENDENCY
    :METAFUNCTION   LOGICAL
)
```

Figure 50: Finite clause: system (Cz, Ru, Bg)

Further elaboration of the Mood element requires the insertion of the Subject and the Finite.

FINITE-INSERT:

(finite-clause; finite-in-imperative) →
[Finite-inserted]
(+Finite)

```
(GATE
    :NAME   FINITE-INSERT
    :INPUTS   (OR FINITE-CLAUSE FINITE-IN-IMPERATIVE)
    :OUTPUTS  ((1.0 FINITE-INSERTED
              (INSERT FINITE)))
    :CHOOSER   FINITE-INSERT-CHOOSER
    :REGION   MOOD
    :METAFUNCTION   INTERPERSONAL
)

(CHOOSER
    :NAME   FINITE-INSERT-CHOOSER
    :DEFINITION   ((CHOOSE FINITE-INSERTED))
)
```

Figure 51: Finite insert: system and chooser (Cz, Ru, Bg)

By passing through the INDICATIVE-MOOD-PERSON system the person of the Finite is chosen. Figure 52 below presents the system and its chooser.

INDICATIVE-MOOD-PERSON:

(finite-clause,

[mental, verbal, creative, dispositive, identifying, relational-other,

(material, nonmeteorological)]) →

[indicative-interactant]

[indicative-noninteractant]

(Finite:::Thirdperson-form)

```
(SYSTEM
    :NAME      INDICATIVE-MOOD-PERSON
    :INPUTS    (AND (OR MENTAL VERBAL CREATIVE DISPOSITIVE
               (AND MATERIAL NONMETEOROLOGICAL)
                IDENTIFYING RELATIONAL-OTHER)
                FINITE-CLAUSE)
    :OUTPUTS   ((0.5 INDICATIVE-INTERACTANT)
               (0.5 INDICATIVE-NONINTERACTANT
               (INFLECTIFY FINITE THIRDPERSON-FORM)))
    :CHOOSER    INDICATIVE-MOOD-PERSON-CHOOSER
    :REGION   MOOD
    :METAFUNCTION   INTERPERSONAL
)

(CHOOSER
    :NAME   INDICATIVE-MOOD-PERSON-CHOOSER
    :DEFINITION   ((ASK (MEMBER-SET-Q SPEAKER SUBJECT)
                  (NIL)
                  (INCLUDED (CHOOSE INDICATIVE-INTERACTANT))
                  (NOTINCLUDED
                  (ASK (MEMBER-SET-QQ HEARER SUBJECT)
                  (INCLUDED (CHOOSE INDICATIVE-INTERACTANT))
                  (NOTINCLUDED (CHOOSE INDICATIVE-NONINTERACTANT))))))
)
```

Figure 52: Person in Indicative Mood: system and chooser (Cz, Ru, Bg)

The Finite's further elaboration leads to other regions such as VOICE (cf. Section 2.2) and TENSE (cf. Section 2.5). The presence or absence of the Subject is accounted for by the systems INDICATIVE-SUBJECT-PRESUMPTION, EXPLICIT-DECLARATIVE-SUBJECT and SUBJECT-INSERT. They are identical with those in the Nigel grammar. We only present the definitions here:

INDICATIVE-SUBJECT-PRESUMPTION:

([(declarative, speaker-subject)

(interrogative, addressee-subject)]) →

[subject-explicit]

[subject-implicit]

All declarative sentences of the target texts have the **subject-explicit** feature.

EXPLICIT-DECLARATIVE-SUBJECT:

(declarative,

[addressee-subject; indicative-noninterractant;

metathing-subject; speaker-plus-subject]) →

[explicit-declarative-subject]

SUBJECT-INSERT:

([wh-subject; meteorological; existential;

indicative-noninterractant; metathing-subject;

subject-specified; explicit-subject-elaborating;

indirect-indicative; finite-enhancing; finite-extending;

finite-rankshift; explicit-subject;

explicit-declarative-subject; explicit-interrogative-subject;

imperative-other; imperative-subject-explicit; subject-explicit;

(imperative [oblative; suggestive])])→

[subject-inserted]

(+Subject)

In Bulgarian Russian and Czech Subject and Finite agree in number and person, and, depending on tense selection, they may also agree in gender. Since we do not have any tense variation in the ImD texts and tense is present, Subject-Finite agreement in gender is not our current issue, but we have to see to persona and number agreement. Figure 53 shows two new systems which account for Subject-Finite agreement in number. Figure 54 shows three systems which provide the Finite with the appropriate person.

```
(GATE
      :NAME   FINITE-SG
      :INPUTS (OR SPEAKER-SUBJECT IT-SUBJECT
               NONPLURAL-SUBJECT)
      :OUTPUTS
              ((1.0 FINITE-SG
              (INFLECTIFY FINITE SINGULAR-FORM)))
      :REGION   MOOD
)

(GATE
      :NAME   FINITE-PL
      :INPUTS (OR SPEAKER-PLUS-SUBJECT ADDRESSEE-SUBJECT PLURAL-SUBJECT )
      :OUTPUTS
              ((1.0 FINITE-PL
              (INFLECTIFY FINITE PLURAL-FORM)))
      :REGION   MOOD
)
```

Figure 53: Subject-Finite agreement in plurality/singularity: gates (Bg, Cz, Ru)

```
(Gate
      :NAME   FINITE-FIRST
      :INPUTS (OR SPEAKER-SUBJECT SPEAKER-PLUS-SUBJECT)
      :OUTPUTS
              ((1.0 FINITE-FIRST
              (INFLECTIFY FINITE FIRSTPERSON-FORM)))
      :REGION   MOOD
)

 (GATE
      :NAME   Finite-Second
      :INPUTS ADDRESSEE-SUBJECT
      :OUTPUTS
              ((1.0 FINITE-SECOND
              (INFLECTIFY FINITE SECONDPERSON-FORM)))
      :REGION   MOOD
)

(GATE
      :NAME   FINITE-THIRD
      :INPUTS     (OR INDICATIVE-NONINTERACTANT IT-SUBJECT)
      :OUTPUTS
                  ((1.0 FINITE-THIRD
                  (INFLECTIFY FINITE THIRDPERSON-FORM)))
      :REGION   MOOD
)
```

Figure 54: Subject-Finite agreement in person: gates (Bg, Cz, Ru)

Employing these systems, we can generate sentences such as those in (22) and (23). For an example of an input SPL expression see below:

```
 (EXAMPLE
    :NAME   PERS-TEXT1-9-a1
    :TARGETFORM "Диалоговият прозорец Select Color се появява на екрана."
    :LOGICALFORM
    (C / NONDIRECTED-ACTION :LEX POIAVIAVAM-SE
        :ACTOR (C1 / OBJECT
        :PROPERTY-ASCRIPTION (Q1 / QUALITY :LEX DIALOGOV)
                                     :LEX PROZOREC
        :CLASS-ASCRIPTION (L2 / SOFTWARE-COMMAND
                      :NAME SELCOL)
        :IDENTIFIABILITY-Q IDENTIFIABLE )
        :SPATIAL-LOCATING
            (P / ONE-OR-TWO-D-LOCATION :LEX EKRAN
            :IDENTIFIABILITY-Q IDENTIFIABLE))
    :SET-NAME   PERS-TEXT1)
```

Choice of declarative is the default here, so this information is not included in the SPL. The grammatical structure of the generated sentence is shown in Figure 55. Note that Subject and Finite carry the same person features (singular-form).

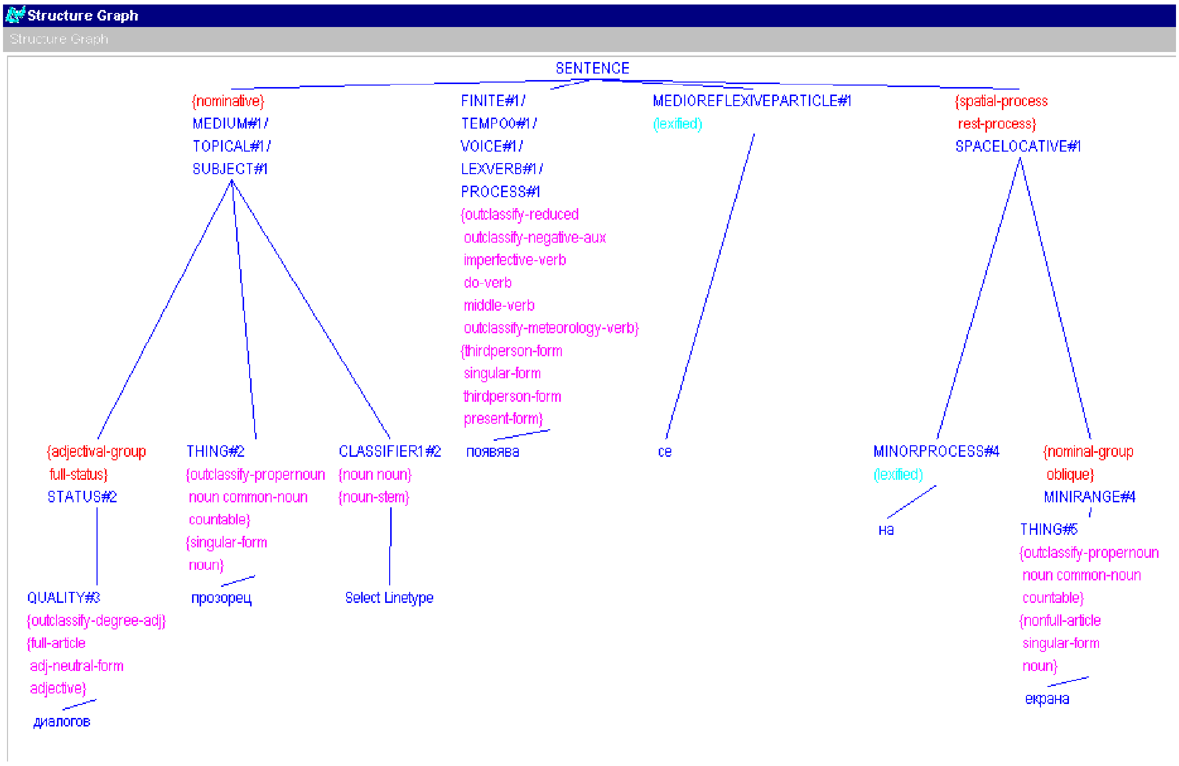


Figure 55: Generated structure for „Диалоговият прозорец Select Color се появява на екрана“.

2.4.2 Imperative Clauses

The following is an example of an imperative clause as it occurs in the ImD texts:

(24) *Imperative clause (text 2)*

(a) En: Specify the start point of the line segment.

(b) Cz: Určete počáteční bod rovného segmentu.

Specify-imp-2pl the start point-acc the line segment-gen

(c) Bg: Задайте началната точка на отсечката.

Specify-imp-2pl the start point of the line segment

(d) Ru: Укажите начальную точку сегмента линии.

Specify-imp-2pl the start point-acc the line segment-gen

All occurrences of **imperative** clauses are **jussive** (rather than **suggestive** or **oblative**). Therefore we only consider the jussive branch. This choice is specified in the system IMPERATIVE-INTERACTANT-SUBJECT given below. Its chooser implementation is shown in Figure 56.

Common in the three languages (and different from English) for jussive imperative clauses is that the Finite is realized in the **imperative-form** which means that the verb is put in **secondperson-form**.

IMPERATIVE-INTERACTANT-SUBJECT:
(imperative-interactant)→

[jussive]
  (Finite:::secondperson-form,Finite:::imperative-form),
[suggestive]
:Bu (+Conjunct, Conjunct ! Haide,
    Finite:::plural-form, Finite:::firstperson-form,
    Finite:::indicative-form)
:Cz (Finite:::plural-form, Finite:::firstperson-form,
    Finite:::imperative-form)
:Ru (Finite:::plural-form, Finite:::firstperson-form,
    +AuxDavatj:::imperative-form)
[oblative]
:Bu (+Conjunct, Conjunct ! Neka, Finite:::indicative-form,
    Finite:::firstperson-form)
:Cz (+Conjunct, Conjunct ! At3, Finite::: indicative-form)
:Ru (+Conjunct, Conjunct ! Pust,
    Finite::: indicative-form)

```
(CHOOSER
    :NAME   IMPERATIVE-INTERACTANT-SUBJECT-CHOOSER
    :DEFINITION   ((ASK (MEMBER-SET-Q SPEAKER SUBJECT)
                  (NOTINCLUDED (CHOOSE JUSSIVE))
                  (INCLUDED
                  (ASK (SAME-AS-Q SPEAKER SUBJECT)
                  (SAME (CHOOSE OBLATIVE))
                  (DIFFERENT (CHOOSE SUGGESTIVE))))))
)
```

Figure 56: Imperative Interactant Subject: chooser (Cz, Ru, Bg)

To generate the appropriate verb form in imperative clauses, we need an additional system: imperatives are further distinguished as to the relationship of speaker and addressee. There are different realizations depending on whether the imperative is expressed politely or nonpolitely. The system given below thus distinguishes between **polite-imperative** and **personal-imperative.** The realization statements specify the form of the Finite – **plural-form** for **polite-imperative** and **singular-form** for **personal-imperative**. The new system IMPERATIVE-POLITENESS-TYPE is shown in Figure 57.

IMPERATIVE-POLITENESS-TYPE

(jussive) →

[polite-imperative] (Finite:::plural-form)

[personal-imperative] (Finite:::singular-form)

```
 (SYSTEM
    :NAME   IMPERATIVE-POLITENESS-TYPE
    :INPUTS   JUSSIVE
    :OUTPUTS   ((0.5 POLITE-IMPERATIVE
                (INFLECTIFY FINITE PLURAL-FORM))
                (0.5 PERSONAL-IMPERATIVE
                (INFLECTIFY FINITE SINGULAR-FORM)))
    :CHOOSER   IMPERATIVE-POLITENESS-TYPE-CHOOSER
    :REGION   MOOD
)

(CHOOSER
    :NAME   IMPERATIVE-POLITENESS-TYPE-CHOOSER
    :DEFINITION   ((ASK (IMPERATIVE-TYPE-Q PROCESS)
                   (POLITE (CHOOSE POLITE-IMPERATIVE))
                   (PERSONAL (CHOOSE PERSONAL-IMPERATIVE))))
 )

(ASKOPERATOR
    :NAME   IMPERATIVE-TYPE-Q
    :DOMAIN   TP
    :PARAMETERS   (PROCESS)
    :ENGLISH   ("Is the"
                 PROCESS
                "polite imperative?"
          )
    :OPERATORCODE   KPML::TRIVIALDEFAULTCODE
    :PARAMETERASSOCIATIONTYPES   (CONCEPT)
    :ANSWERSET   (POLITE PERSONAL)
    :TRIVIALDEFAULT   POLITE
)
```

Figure 57: Imperative Politeness Type: system, chooser and inquiry (Cz,Ru,Bg)

Although the Subject (semantically the addressee) can potentially be expressed explicitly in imperatives in Slavic languages, the imperative clauses in the ImD texts have no explicit Subject. The optionality of the Subject is accounted for in the system of IMPERATIVE-SUBJECT-PRESUMPTION. The implementation of the system is given in Figure 58:

IMPERATIVE-SUBJECT-PRESUMPTION

(imperative) →

[imperative-subject-implicit]

[imperative-subject-explicit] (Subject : Addressee)

```
(SYSTEM
    :NAME   IMPERATIVE-SUBJECT-PRESUMPTION
    :INPUTS   JUSSIVE
    :OUTPUTS   ((0.9 IMPERATIVE-SUBJECT-IMPLICIT)
               (0.1 IMPERATIVE-SUBJECT-EXPLICIT
               (PRESELECT SUBJECT ADDRESSEE)))
    :CHOOSER   IMPERATIVE-SUBJECT-PRESUMPTION-CHOOSER
    :REGION   MOOD
    :METAFUNCTION   INTERPERSONAL
)

(CHOOSER
    :NAME   IMPERATIVE-SUBJECT-PRESUMPTION-CHOOSER
    :DEFINITION   ((ASK (EXPRESS-HEARER-Q HEARER ONUS)
                  (WITHHOLDHEARER (CHOOSE IMPERATIVE-SUBJECT-IMPLICIT))
                  (EXPRESSHEARER (CHOOSE IMPERATIVE-SUBJECT-EXPLICIT)))
                  (* CHRISTIAN " 6-NOV-84 10:35:50"))
    :DATE   "8-4-1992 15:51:28 at IPSI."
)

(ASKOPERATOR
    :NAME   EXPRESS-HEARER-Q
    :DOMAIN   TP
    :PARAMETERS   (HEARER1 SPEC)
    :ENGLISH   (
                "Is the hearer"
                 HEARER1
                "specified in"
                 SPEC
                "to be expressed?"
                )
    :OPERATORCODE   KPML::EXPRESS-HEARER-Q-CODE
    :PARAMETERASSOCIATIONTYPES   (CONCEPT MODIFICATIONSET)
    :ANSWERSET   (EXPRESSHEARER WITHHOLDHEARER)
    :TRIVIALDEFAULT   EXPRESSHEARER
)
```

Figure 58: Imperative Subject Presumption :

system, chooser and inquiry Express-Hearer-Q (Cz, Ru, Bg)

Employing this implementation, we can generate imperative sentences such as those in (24). The polite imperative form is defaulted according to the style of texts generated and is not explicitly given in the SPL. For a sample SPL see below (generating the clause „Specify a point.“).

```
(EXAMPLE
    :SET-NAME PERS-TEXT2-BG
    :NAME PERS-TEXT2-8
    :TARGETFORM   "Задайте точка."
    :LOGICALFORM
         (S1 / DIRECTED-ACTION :LEX ZADAM
         :SPEECHACT IMPERATIVE
    :ASPECT-Q PERFECTIVE-ASPECT
    :ACTEE (D / OBJECT :LEX TOCHKA)
))
```

The grammatical structure of the generated sentence is shown in Figure 59.

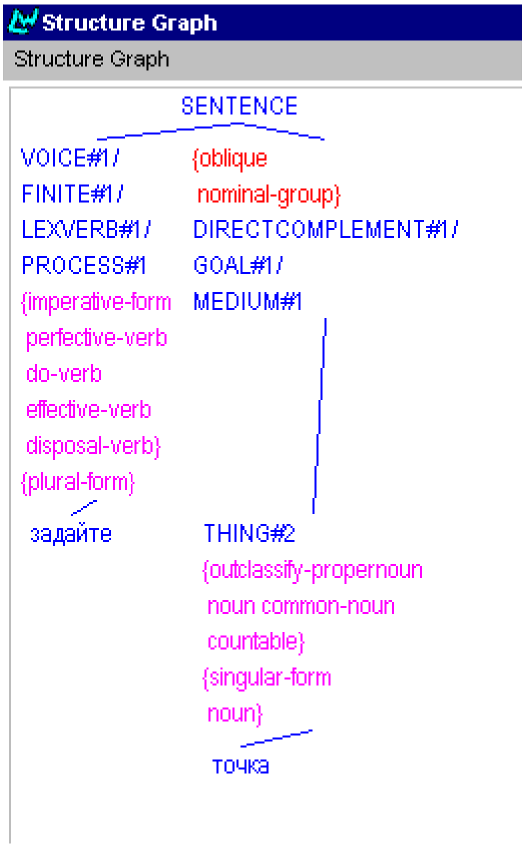


Figure 59: Generated structure for "Задайте точка."

2.5 Tense

In this chapter we describe the basic systems in the region of TENSE. The general discussion of tenses in Bulgarian, Czech and Russian languages has been presented in the SPEC2 deliverable (Andonova *et al.*, 1999). Here, we present the implementations for our three languages, as relevant for the texts generated in the intermediate demonstrator.

The entry condition to the TENSE region is the feature **finite-clause**. The types of finite clauses we deal with are exemplified in (25)-(27):

(25) *Independent Clause Simplex- Side Effect (text 4)*

(a) En: The Multiline Style dialog box appears.

(b) Cz: Objeví se dialogové okénko Multiline Style.

Appear-refl-3sg dialog box Multiline Style.

(c) Bg: Появява се диалоговият прозорец Multiline Style.

Appear-refl-3sg dialog box Multiline Style.

(d) Ru: (На экране) появится диалоговое окноMultiline Style.

(On the screen) appear-refl-3sg dialog box Multiline Style.

In the Bulgarian version, side effects as illsutrated above are expressed by means of the present-tense verb forms of *imperfective aspect* whereas in Russian and Czech they are realized by present-tense forms of *perfective verbs* with the meaning of futurity.

Unlike Czech and Russian, the Bulgarian texts for the ImD include examples of finite clauses which correspond to English imperatives in the nonpersonal style. An example is given below.

(26) *Independent Clause Simplex in nonpersonal style (text 2)*

(a) En: Specify the endpoint of the arc.

(b) Bg: Задава се крайната точка на дъгата.

Specify-refl-passive3sg the endpoint of the arc.

Here, the Bulgarian version requires a finite verb form in the appropriate tense, in this case, present tense. The Russian and Czech versions do not contain stylistic variation which employ this type of finite clause for expressing a 'command'.

Another type of finite clauses are dependent clauses in some clause complexes, as in the example given below.

(27) *Finite Dependent Clause (text 3)*

(a) En: Start the ARC command using one of these methods:

(b) Bg: *Personal style*

Стартирайте командата ARC,

като използвате един от следните методи:

by use-2pl one of these methods:

Nonpersonal style

Стартира се командата ARC,

като се използва един от следните методи:

by use-refl.passive,3sg one of these methods:

In Bulgarian, the dependent clause is finite whereas the corresponding English dependent clause is nonfinite. In Czech and Russian, the corresponding texts do not use combinations of main and dependent clauses. Instead, they render the intended meaning by means of a single clause incorporating a nominal phrase, i.e., by means of nominalization. That is why specification of tense in dependent clauses needs to be applied for the realization of Bulgarian only. The tense in such clauses in Bulgarian is present.

Thus, in general, the only tense used in the texts chosen for the ImD is present tense. As noted above, the use of present tense perfective verb forms in the Russian and Czech texts carries the meaning of future. This does not require, however, a change in the specification of tense in the grammar. The use of present in the TENSE system is sufficient for the implementation in the ImD and does not require carrying out an exhaustive theoretical analysis and comparison of the tense systems of the three languages which exhibit considerable differences otherwise.

There are no occurrences of modal verbs, so we will not consider further the branch **modal** of the system DEICTICITY- the main system of the region of TENSE as presented in Figure 60 together with its chooser.

```
 (SYSTEM
    :NAME   DEICTICITY
    :INPUTS   FINITE-CLAUSE
    :OUTPUTS   ((0.5 MODAL
               (CLASSIFY FINITE MODAL-AUX))
               (0.5 TEMPORAL
               (INSERT TEMPO0)
               (CONFLATE TEMPO0 FINITE))
    :CHOOSER   DEICTICITY-CHOOSER
    :REGION   TENSE
    :METAFUNCTION   LOGICAL
)

(CHOOSER
    :NAME   DEICTICITY-CHOOSER
    :DEFINITION   ((ASK (MODALITY-Q PROCESS)
                  (MODAL (IDENTIFY FINITE
                  (MODALITY-ID PROCESS))
                  (IDENTIFY FINITE
                  (CONCEPTUAL-CORRELATE-ID FINITE))
                  (COPYHUB SPEAKINGTIME TEMPO1)
                  (CHOOSE MODAL))
                  (NONMODAL (COPYHUB SPEAKINGTIME TEMPO0)
                  (CHOOSE TEMPORAL)))
```

Figure 60: Deicticity: system and chooser (Cz, Ru, Bg)

Following the **temporal** branch, we get to the system PRIMARY-TENSE with the features **future**, **present** and **past**. The logic of the choice is provided by the inquiries TIME-IN-RELATION-TO-SPEAKING-TIME-ID, COUNTERFACTUALITY-Q, EXTENSIONALITY-Q, LOGICO-TEMPORAL-CONDITION-Q and PRECEDE-Q. They are shown in Figure 61 together with the system and the chooser PRIMARY-TENSE.

```
(SYSTEM
    :NAME   PRIMARY-TENSE
    :INPUTS   TEMPORAL
    :OUTPUTS   ((0.33333334 FUTURE
               (CLASSIFY FINITE FUTURE-AUX)
               (CONFLATE TEMPO0 FINITE))
               (0.33333334 PRESENT
               (CONFLATE FINITE TEMPO0)
               (INFLECTIFY FINITE PRESENT-FORM))
               (0.33333334 PAST
               (CONFLATE FINITE TEMPO0)
               (INFLECTIFY FINITE PAST-FORM)))
    :CHOOSER   PRIMARY-TENSE-CHOOSER
    :REGION   TENSE
    :METAFUNCTION   LOGICAL
)

(CHOOSER
    :NAME   PRIMARY-TENSE-CHOOSER
    :DEFINITION   ((IDENTIFY TEMPO1
           (TIME-IN-RELATION-TO-SPEAKING-TIME-ID TEMPO0 SPEECHACT))
           (ASK (COUNTERFACTUALITY-Q ONUS TEMPO1)
           (COUNTERFACTUAL (CHOOSE PAST))
           (NONCOUNTERFACTUAL
           (ASK (EXTENSIONALITY-Q PROCESS)
           (EXTENSIONAL
           (ASK (LOGICO-TEMPORAL-CONDITION-Q ONUS)
           (LOGICOTEMPORALCONDITION
           (ASK (PRECEDE-Q TEMPO1 TEMPO0)
           (PRECEDES (CHOOSE PAST))
           (NOTPRECEDES (CHOOSE PRESENT))))
           (NOTLOGICOTEMPORALCONDITION
           (ASK (PRECEDE-Q TEMPO1 TEMPO0)
           (NOTPRECEDE)
           (PRECEDES (CHOOSE PAST))
           (NOTPRECEDES
           (ASK (PRECEDE-Q TEMPO0 TEMPO1)
           (PRECEDES (CHOOSE FUTURE))
           (NOTPRECEDES (CHOOSE
                        PRESENT)))))))
           (INTENSIONAL (CHOOSE PRESENT)))))
))

(ASKOPERATOR
    :NAME   COUNTERFACTUALITY-Q
    :DOMAIN   KB
    :PARAMETERS   (POTENTIALITY ASSESSMENTTIME)
    :ENGLISH ("Is the occurrence, performance, or"
              "actualization of"
               POTENTIALITY
              "at the time specified by"
               ASSESSMENTTIME
              "a hypothesis inconsistent with the"
              "facts, i.e. counter to them,
               at that time or judged to"
              "be highly unlikely?"
              )
    :OPERATORCODE   KPML::TRIVIALDEFAULTCODE
    :PARAMETERASSOCIATIONTYPES   (CONCEPT CONCEPT)
    :ANSWERSET  (COUNTERFACTUAL NONCOUNTERFACTUAL)
    :TRIVIALDEFAULT   NONCOUNTERFACTUAL
)

(ASKOPERATOR
    :NAME   EXTENSIONALITY-Q
    :DOMAIN   KB
    :PARAMETERS   (CONCEPT)
    :ENGLISH   ("Is"
                CONCEPT
                "extensional i.e. concrete rather than"
                "intensional i.e. generic?"
                )
    :OPERATORCODE   KPML::TRIVIALDEFAULTCODE
    :PARAMETERASSOCIATIONTYPES   (CONCEPT)
    :ANSWERSET   (EXTENSIONAL INTENSIONAL)
    :PRESELECTIONGUIDANCE   (
          (SINGULAR . EXTENSIONAL)
          (NOUN . EXTENSIONAL)
          )
    :TRIVIALDEFAULT   EXTENSIONAL
)

(ASKOPERATOR
    :NAME   LOGICO-TEMPORAL-CONDITION-Q
    :DOMAIN   KB
    :PARAMETERS   (ITEM)
    :ENGLISH   ("Does the state of affairs,
                 i.e. event or"
                "situation, specified for expression by"
                 ITEM
                "constitute a logical or temporal"
                "condition, i.e. restriction,
                 on some process i.e. does"
                "it set up, logically or temporally,
                 the possible world"
                "in which or in relation to which
                 this process is"
                "performed?"
                )
    :OPERATORCODE   KPML::TRIVIALDEFAULTCODE
    :PARAMETERASSOCIATIONTYPES   (CONCEPT)
    :ANSWERSET (LOGICOTEMPORALCONDITION
                NOTLOGICOTEMPORALCONDITION)
    :TRIVIALDEFAULT   NOTLOGICOTEMPORALCONDITION
)

(ASKOPERATOR
    :NAME   PRECEDE-Q
    :DOMAIN   KB
    :PARAMETERS   (FIRSTTIME SECONDTIME)
    :ENGLISH ("Does the moment
               or interval of time"
               FIRSTTIME
              "strictly precede the moment
               or interval"
               SECONDTIME
              "?"
              )
    :OPERATORCODE   KPML::PRECEDE-Q-CODE
    :PARAMETERASSOCIATIONTYPES   (CONCEPT CONCEPT)
    :ANSWERSET   (PRECEDES NOTPRECEDES)
    :TRIVIALDEFAULT   NOTPRECEDES
)
```

Figure 61: Primary Tense: system and chooser, relevant inquiries

(Cz, Bg, Ru)

In our ImD texts, the default tense is present. The logic of tense choices that is employed in the NIGEL implementation is described in detail in (Matthiessen, 1984).

In Figure 62 we show a grammatical structure which includes the tense information in the constituent Finite.

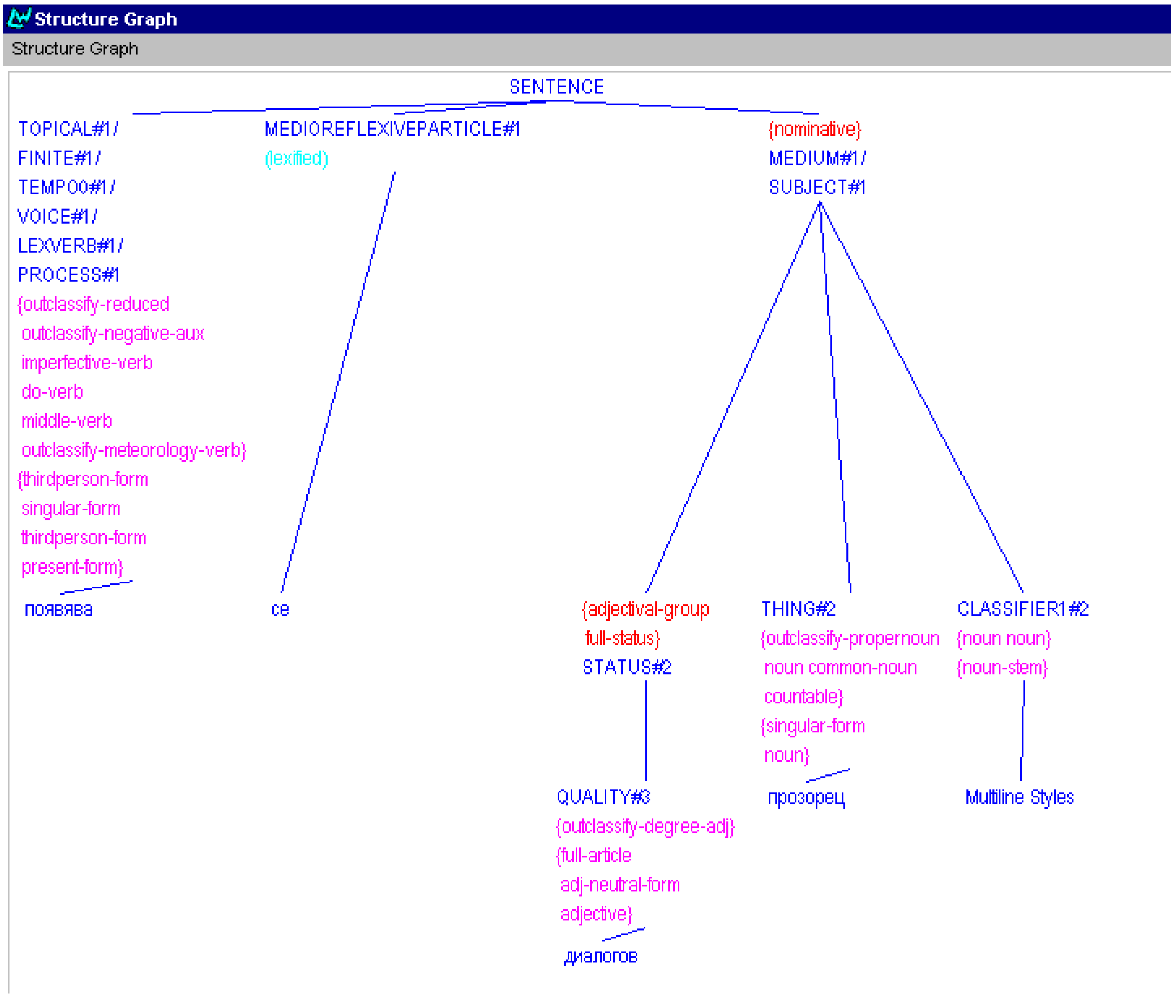


Figure 62: Generated structure for „Появява се диалоговият прозорец Multiline Styles.“

2.6 Aspect

A general discussion of aspect as a property of Slavic verbs has been presented in the SPEC2 deliverable (Andonova *et al.* 1999). On this basis, we specify a system of ASPECT as follows:

ASPECT

clause-simplex→

[perfective] (Process::perfective-verb)

[imperfective] (Process::imperfective-verb)

An analysis of the texts for the ImD provides us with the motivations for a choice between perfective and imperfective, which we can use to formulate a chooser:

- All processes in imperative clauses are realized by perfective verbs, so as the first branching inquiry in the chooser we can use COMMAND-Q – with the answer COMMAND, the choice is **perfective**.
- Processes in indicative clauses can be perfective or imperfective. The motivation for choosing one or the other lies roughly in ‘repeatability’ or ‘habituality’: repeated, habitual actions are typically realized in imperfective and actions that happen only once are realized in perfective. To cover this motivation we add the inquiry REPEATABLE-Q.
- Most of the finite clauses that are not imperative use imperfective verbs, so we can default the answer to REPEATABLE-Q to REPEATABLE. This leads to the choice of **imperfective-verb**. If the choice of **perfective** is intended, REPEATABLE-Q has to be included in the SPL input specification.

The implementations of the ASPECT system, the chooser and the new inquiry are shown in Figure 63.

```
(SYSTEM
    :NAME   ASPECT
    :INPUTS   CLAUSE-SIMPLEX
    :OUTPUTS   ((0.5 PERFECTIVE
               (CLASSIFY PROCESS PERFECTIVE-VERB))
               (0.5 IMPERFECTIVE
               (CLASSIFY PROCESS IMPERFECTIVE-VERB)))
    :CHOOSER   ASPECT-CHOOSER
    :REGION   TENSE
)

(CHOOSER
    :NAME   ASPECT-CHOOSER
    :DEFINITION   ((ASK (COMMAND-Q SPEECHACT)
                  (COMMAND (CHOOSE PERFECTIVE))
                  (NOCOMMAND
                  (ASK (REPEATABLE-Q PROCESS)
                  (NONREPEATABLE (CHOOSE PERFECTIVE))
                  (REPEATABLE (CHOOSE IMPERFECTIVE))))))
)

(ASKOPERATOR
    :NAME   COMMAND-Q
    :DOMAIN   TP
    :PARAMETERS   (ACT1)
    :ENGLISH   (
          "Is the illocutionary point of the surface"
          "level speech act represented by"
           ACT1
          "a command, i.e. a request of an action"
          "by the hearer?"
          )
    :OPERATORCODE   KPML::COMMAND-Q-CODE
    :PARAMETERASSOCIATIONTYPES   (CONCEPT)
    :ANSWERSET   (COMMAND NOCOMMAND)
)

(ASKOPERATOR
    :NAME   REPEATABLE-Q
    :DOMAIN   KB
    :PARAMETERS   (PROCESS)
    :ENGLISH   (
          "Is the process "
           PROCESS
          "semanticaly repeatable ? "
          )
    :OPERATORCODE   KPML::TRIVIALDEFAULTCODE
    :PARAMETERASSOCIATIONTYPES   (CONCEPT)
    :ANSWERSET   (REPEATABLE NONREPEATABLE)
    :TRIVIALDEFAULT   REPEATABLE
 )
```

Figure 63: Aspect: system, chooser and inquiries (Cz, Ru, Bg)

Using this implementation we are able to generate the appropriate aspect forms occuring in the texts for the ImD. In (28) we show an example from text 1:

(28) *Clause complex (text 1)*

(a) En: Choose Element Properties to add elements to the style.

(b) Cz: Vyberte Vlastnosti prvků pro přidání elementů ke stylu.

Choose-imp.perf-asp. Element Properties to add-nominal elements to the style.

(c) Bg: Изберете Element Properties, за да добавите елементи към стила.

Choose-imp.perf-asp. Element Properties to add-ind.perf-asp elements to the style.

(d) Ru: Нажмите кнопку Element Properties, чтобы добавить элементы в стиль.

Choose-imp.perf-asp. the button Element Properties to add-infinitive elements to the style.

The SPL used in generating the Bulgarian variant is given below. The choice of **perfective** in the main clause is the consequence of the clause being in imperative mood. For the dependent clause, the inquiry REPEATABLE-Q effects the choice of aspect.

A grammatical structure including aspect information in the Finite constituent is given in Figure 64.

```
(EXAMPLE
    :NAME   PERS-TEXT1-6
    :TARGETFORM   "Изберете Element Properties, за да добавите елементи
                   към стила."
    :LOGICALFORM
           (R / RST-PURPOSE
    :DOMAIN
           (D / DIRECTED-ACTION :LEX IZBERA
           :SPEECHACT IMPERATIVE
           :ACTEE (E / SOFTWARE-COMMAND :NAME "Element Properties"))
    :RANGE
           (C / DIRECTED-ACTION :LEX DOBAVIA
           :REPEATABLE-Q NONREPEATABLE
           :ACTEE (C1 / OBJECT :LEX ELEMENT
                               :NUMBER PLURAL)
           :DESTINATION (P / OBJECT :LEX STIL
                        :IDENTIFIABILITY-Q IDENTIFIABLE))
)
```

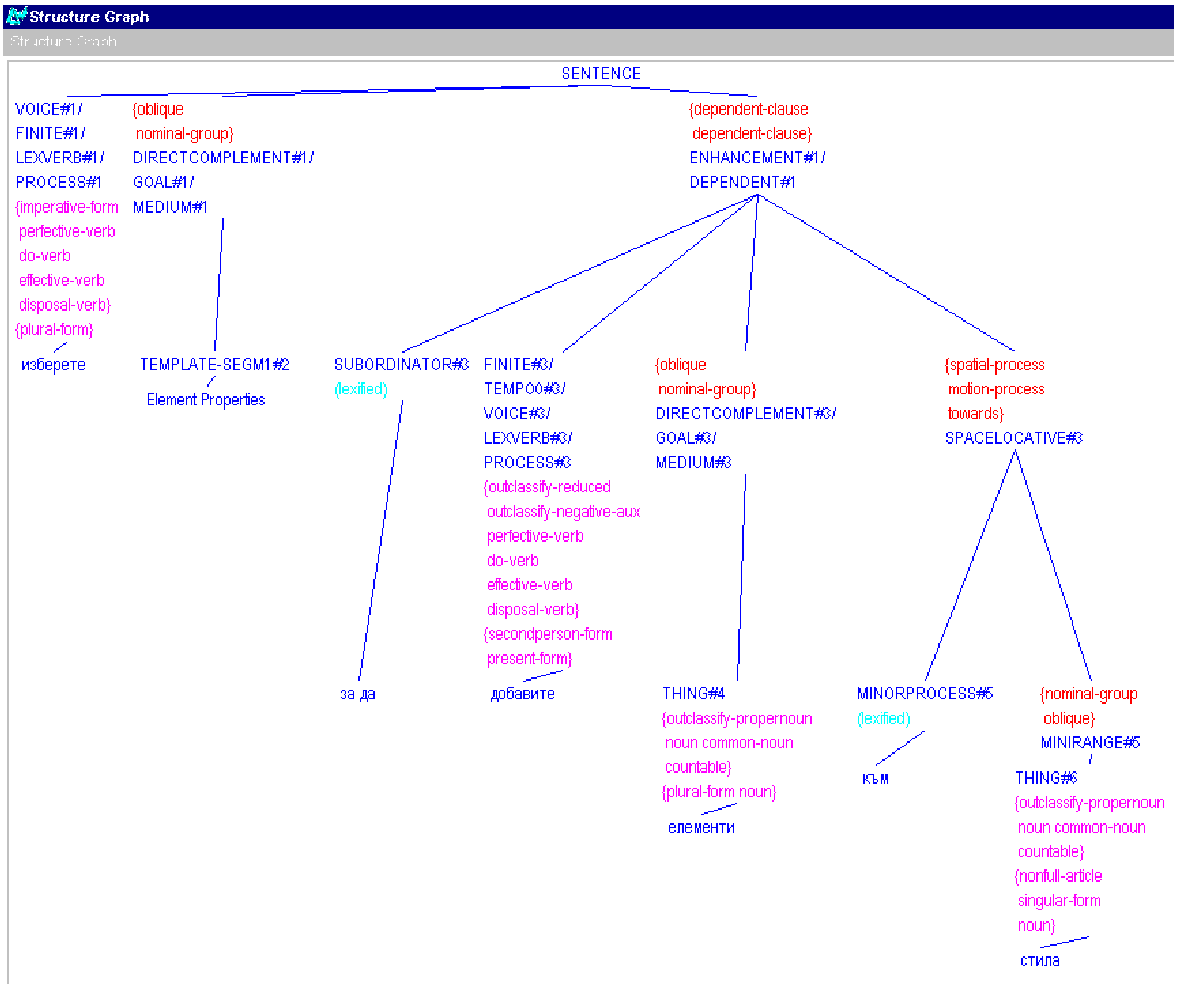


Figure 64: Generated structure for “Изберете Element Properties, за да добавите елементи към стила”

2.7 Clause Complexity

In this chapter, we describe the implementations in the CLAUSECOMPLEX region, as relevant for the texts generated in the AGILE intermediate demonstrator. Our implementation efforts did not start from scratch, but have built upon the Nigel grammar for English. Therefore, the way clause complexity is implemented in the AGILE grammars mirrors the treatment in Halliday's SFG (Halliday, 1985) to a large degree. A general discussion of clause complexity in SFG with respect to the AGILE project has been presented in the SPEC2 deliverable (Andonova *et al.*, 1999). In the SPEC2 deliverable, we also discussed how Nigel implements SFG’s notion of clause complexity within the KPML framework, and how far the approach taken in the Nigel grammar for English can be adopted for generating sentences in Czech, Bulgarian, or Russian. The present chapter presents a self-contained account of the current implementations for our three languages, focusing on systems and choosers.

The region that is at the heart of generating clause complexes is CLAUSECOMPLEX. In this section we describe the core of this region’s network. First, let us show an example of a clause complex taken from the ImD text 1 (for more examples of various types of clause complexity see the deliverable SPEC2 (Andonova *et al.*, 1999)). The simplex clauses of which the clause complex consists are printed on separate lines numbered (i) through (iii):

(29) *Clause complex*

(a) Cz:

i) Vyberte OK,
Choose-imp-2pl OK

ii) abyste uložili vlastnosti elementu multičáry
would-2pl save-pastparticiple properties-acc element-gen multiline-gen

iii) a opustili dialogový panel Element Properties.
and leave-pastparticiple dialogue box-acc Element Properties

(b) Bg:

i) Изберете OK,
Choose-imp-2pl OK

ii) за да запишете характеристиките на елемента на мултилинията
so that save-2pl properties of element of multiline-the

iii) и да излезете от диалоговия прозорец Element Properties.
and that leave-2pl from dialogue window Element Properties

(c) Ru:

i) Нажмите кнопку OK,
Choose-imp-2pl button-acc OK

ii) чтобы сохранить стиль элементов мультилинии
in-order-to save-inf style element-gen multiline-gen

iii) и закрыть диалоговое окно Element Properties.
and close-inf dialogue window-acc Element Properties

(d) En:

i) Choose OK

ii) to save the style of the multiline element

iii) and to exit the Element Properties dialog box.

The topmost system of the CLAUSECOMPLEX region is CLAUSECOMPLEXITY. This system has as outputs the grammatical features **clausecomplex** and **clausesimplex**. The feature **clausecomplex** serves as input for the system TYPE-OF-INTERDEPENDENCE, which is formally specified as follows:

TYPE-OF-INTERDEPENDENCE:

```
      (clausecomplex) →
            [expansion], [projection].
```

The corresponding system and chooser implementations are shown in Figure 60. They are the same for our three languages as well as for English.

```
 (SYSTEM
    :NAME   TYPE-OF-INTERDEPENDENCE
    :INPUTS   CLAUSE-COMPLEX
    :OUTPUTS   ((0.5 PROJECTION
               (* INSERT PROJECTOR)
               (INSERT PROJECTED))
               (0.5 EXPANSION))
    :CHOOSER   TYPE-OF-INTERDEPENDENCE-CHOOSER
    :REGION   CLAUSECOMPLEX
    :METAFUNCTION   LOGICAL
)

(CHOOSER
    :NAME   TYPE-OF-INTERDEPENDENCE-CHOOSER
    :DEFINITION   ((ASK (PROJECTION-Q ONUS)
                  (PROJECTED (CHOOSE PROJECTION)
                  (IDENTIFY PROJECTOR
                  (PROJECTOR-ID ONUS))
                  (PLEDGE PROJECTOR ONUS)
                  (IDENTIFY PROJECTED
                  (PROJECTED-ID ONUS))
                  (PLEDGE PROJECTED ONUS))
                  (NOTPROJECTED (IDENTIFY PART1
                  (DUAL-PROCESS-PART1-ID ONUS))
                  (IDENTIFY PART2
                  (DUAL-PROCESS-PART2-ID ONUS PART1))
                  (IDENTIFY PART1
                  (CONCEPTUAL-CORRELATE-ID PART1))
                  (PLEDGE PART1 ONUS)
                  (IDENTIFY PART2
                  (CONCEPTUAL-CORRELATE-ID PART2))
                  (PLEDGE PART2 ONUS)
                  (CHOOSE EXPANSION)))
))
```

Figure 65: Type of interdependence: system and chooser (Cz, Ru, Bg)

The ImD texts do not show any occurrence of projection, and neither does the rest of the AGILE corpus. Therefore, we do not consider that branch of the network any further, and we concentrate on the expansion branch.

Through **expansion**, two systems are triggered, namely EXPANSION-TYPE and EXPANSION-TAXIS, which are specified as follows (the corresponding implementations are shown in Figure 66; they are the same for our three languages as well as for English):

EXPANSION-TYPE:

```
      (expansion) →
            [enhancement]     (+Enhancement),
            [extension]       (+Extension),
            [elaboration]     (+Elaboration).
```

EXPANSION-TAXIS

```
      (expansion) →
            [general-hypotactic-expansion],
            [paratactic-expansion].
```

```
(SYSTEM
    :NAME   EXPANSION-TAXIS
    :INPUTS   EXPANSION
    :OUTPUTS   ((0.5 PARATACTIC-EXPANSION)
               (0.5 GENERAL-HYPOTACTIC-EXPANSION))
    :CHOOSER   EXPANSION-TAXIS-CHOOSER
    :REGION   CLAUSECOMPLEX
    :METAFUNCTION   LOGICAL
)

(SYSTEM
    :NAME   EXPANSION-TYPE
    :INPUTS   EXPANSION
    :OUTPUTS   ((0.33333334 ELABORATION
               (INSERT ELABORATION))
               (0.33333334 EXTENSION
               (INSERT EXTENSION))
               (0.33333334 ENHANCEMENT
               (INSERT ENHANCEMENT)))
    :CHOOSER   EXPANSION-TYPE-CHOOSER
    :REGION   CLAUSECOMPLEX
    :METAFUNCTION   LOGICAL
)

(CHOOSER
    :NAME   EXPANSION-TAXIS-CHOOSER
    :DEFINITION   ((ASK (TAXIS-PROMINENCE-Q PART1 PART2)
                  (NOTLESS
                  (ASK (TAXIS-PROMINENCE-Q PART2 PART1)
                  (NOTLESS (CHOOSE PARATACTIC-EXPANSION))
                  (LESS (COPYHUB PART2 DEPENDENT)
                  (COPYHUB PART1 TERMINANT)
                  (CHOOSE GENERAL-HYPOTACTIC-EXPANSION))))
                  (LESS (COPYHUB PART1 DEPENDENT)
                  (COPYHUB PART2 TERMINANT)
                  (CHOOSE GENERAL-HYPOTACTIC-EXPANSION)))
                  (* LP " 5-Jun-87 16:55:50"))
    :DATE   "8-4-1992 15:49:13 at IPSI."
)

(CHOOSER
    :NAME   EXPANSION-TYPE-CHOOSER
    :DEFINITION   ((ASK (ELABORATION-Q PART1 PART2)
                  (SAME (CHOOSE ELABORATION))
                  (DISTINCT
                  (ASK (CONDITIONING-Q PART1 PART2)
                  (NONCONDITIONING (CHOOSE EXTENSION))
                  (CONDITIONING (CHOOSE ENHANCEMENT))))))
```

Figure 66: Expansion type and expansion taxis: systems and choosers (Cz, Bg, Ru)

The system EXPANSION-TAXIS accounts for the interaction of taxis and logico-semantic relations. The tactic dimension has its reflection in the grammatical structure through the following systems (see Figure 67 for the corresponding implementations, which again are the same for our three languages and for English):

PARATAXIS:

```
{paratactic-expansion,quoting} →
      [parataxis]
            (+Initiating,
             +Continuing,
             Initiating … Continuing).
```

HYPOTAXIS-ALPHA-COMPLEXITY:

```
      (general-hypotactic-expansion) →
            [complex-alpha-hypotactic-expansion]
                  (+Dependent,
                   +Terminant,
                   Terminant : full),
            [hypotactic-expansion]
```

```
 (GATE
    :NAME   PARATAXIS
    :INPUTS   (OR QUOTING PARATACTIC-EXPANSION)
    :OUTPUTS  ((1.0 PARATAXIS
              (INSERT INITIATING)
              (INSERT CONTINUING)
              (PARTITION INITIATING CONTINUING)))
    :REGION   CLAUSECOMPLEX
    :METAFUNCTION   LOGICAL
)

(SYSTEM
    :NAME   HYPOTAXIS-ALPHA-COMPLEXITY
    :INPUTS   GENERAL-HYPOTACTIC-EXPANSION
    :OUTPUTS   ((0.5 HYPOTACTIC-EXPANSION)
               (0.5 COMPLEX-ALPHA-HYPOTACTIC-EXPANSION
               (INSERT DEPENDENT)
               (INSERT TERMINANT)
               (PRESELECT TERMINANT FULL)))
    :CHOOSER   HYPOTAXIS-ALPHA-COMPLEXITY-CHOOSER
    :REGION   CLAUSECOMPLEX
    :METAFUNCTION   LOGICAL
)

(CHOOSER
    :NAME   HYPOTAXIS-ALPHA-COMPLEXITY-CHOOSER
    :DEFINITION   ((ASK (MULTIPLE-PROCESS-Q PART1)
                  (SINGLE (IDENTIFY PROCESS
                  (PROCESS-ID PART1))
                  (IDENTIFY WHERE-AM-I
                  (WHERE-AM-I-ID))
                  (IDENTIFY PROCESS
                  (MODIFICATION-SPECIFICATION-ID WHERE-AM-I PROCESS))
                  (* IDENTIFY PROCESS
                  (TERM-SPECIFICATION-ID PROCESS VERB))
                  (IDENTIFY EVENTTIME
                  (REFERENCE-TIME-ID PROCESS))
                  (IDENTIFY SPEAKINGTIME
                  (SPEAKING-TIME-ID SPEECHACT))
                  (CHOOSE HYPOTACTIC-EXPANSION))
                  (MULTIPLE
                  (CHOOSE COMPLEX-ALPHA-HYPOTACTIC-EXPANSION)))))
```

Figure 67: Parataxis and hypotaxis: systems and chooser (Cz, Ru, Bg)

The above mentioned interaction between taxis and expansion types can be observed in example (29) above: a paratactic additive extension relation holds between (ii) and (iii), and a hypotactic causal-conditional enhancement relation holds between the (ii)-(iii) sub-complex and the simplex clause (i). For further examples of various types of paratactic and hypotactic expansion see the deliverable SPEC2 (Andonova *et al.*, 1999).

The generation of a clause complex proceeds as follows. On the first traversal through these systems, the basic type of expansion relation and the type of interdependence are determined. 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 then determines 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 the complexity of nominal groups (NOMINALGROUPCOMPLEXITY region) or prepositional phrases (CIRCUMSTANCE region). What is important about this observation is that the CLAUSECOMPLEX region is thus primarily concerned with *classifying* clauses, which to some extent influences the grammatical categories of the clauses involved, but leaving the more detailed realization to other regions.

As for variation in style as discussed in the TEXS2 deliverable (Kruijff-Korbayová *et al.* 1999), note that all clauses within a clause complex need to realize the content using the same style parameters. For instance in a hypotactic clause complex, if non-personal style in reflexive passive is chosen, then both the main clause and the dependent have to be realized in reflexive passive, as demonstrated in example (30) for Czech (a variant of (29a)):

(30)

i)	Vybere	se	OK,
	Choose-ind-1sg	refl	OK

ii)	aby	se	uložily	vlastnosti	elementu	multičáry
	so	refl	save-pastparticiple-3pl	properties-nom	element-gen	multiline-gen

iii)	a	opustil	se	dialogový	panel	Element	Properties.
	and	leave-pastparticiple-3sg	refl	dialogue	box-acc	Element	Properties

This is the same for other styles. It should be noted that in Czech, in the non-personal style using infinitive, one has to use reflexive passive in the dependent clause, because the dependent clause has to be finite. See the infinitive variant of (30) in (31) below:

(31)

i)	Vybrat	OK,
	Choose-inf	OK

ii)	aby	se	uložily	vlastnosti	elementu	multičáry
	so	refl	save-pastparticiple-3pl	properties-nom	element-gen	multiline-gen

iii)	a	opustil	se	dialogový	panel	Element	Properties.
	and	leave-pastparticiple-3sg	refl	dialogue	box-acc	Element	Properties

Even though a variety of types of expansion in combination with parataxis or hypotaxis have been encountered in the AGILE corpus, in the ImD, we have a narrower focus. The coverage in this phase includes the following clause complexity types:

- paratactic extension (addition type)[6]
- paratactic enhancement (reason circumstantials)
- hypotactic enhancement (manner and cause circumstantials)

We shall describe the implementations for these types of clause complexity in detail in the following sections. The other types of clause complexity are also present in our grammars, since we have inherited them from the Nigel grammar for English; and since the region of clause complexity is rather general, the implementations therefore apply to our three languages as well. Differences are encountered only at the level of realizing particular types of clauses.

2.7.1 Paratactic Extension

Parataxis is a relation between two elements in which neither is dependent on the other. In **extension,** the secondary clause *adds* something new to the meaning expressed in the primary. What is added may either be an **addition (positive, negative or adversative)**, or a **variation (replacive, subtractive or alternative)**. The type we cover in the ImD is **positive addition**. The following example shows the relevant sentences from one of the ImD texts in the personal style in imperative mood.

(32) *positive addition (text 5)*

(a) En: Specify the internal point and press Return.

(b) Cz: Určete vnitřní bod a stiskněte Enter.
Specify-2pl internal-acc point-acc and press-2pl Return

(c) Bg: Въведете вътрешна точка и натиснете Return.
Specify-2pl internal point and press-2pl Return

(d) Ru: Укажите внутреннюю точку и нажмите Return.
Specify-2pl internal-acc point-acc and press-2pl Return

The relevant system is specified below. Its implementation is shown in Figure 68. Note that the respective connectives are inserted 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. We use language-specific insertions in each grammar. In the following specification, we collapse the realizations (in the order En, Cz, Bg, Ru), typeset them in bold and separate them by "|".

[6] Variation is present only at the level of groups.

EXTENDING-COORDINATION-TYPE:
(extending-coordination) →
[additive-coordination]
(+Coordinator,
Coordinator^Extension,
Coordinator ! **and** | **a** | и | и),
[alternative-coordination]
(+Correlator,
Correlator^Extended,
Correlator ! **either** | **bud’** | или | или,
+Coordinator,
Coordinator^Extension,
Coordinator ! **or** | **nebo** | или | или),
[contrastive-coordination]
(+Coordinator,
Coordinator ! **but** | **ale** | но | но,
Coordinator^Extension).

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*.

```
(SYSTEM
    :NAME   EXTENDING-COORDINATION-TYPE
    :INPUTS   EXTENDING-COORDINATION
    :OUTPUTS ((0.33333334 ADDITIVE-COORDINATION
              (INSERT COORDINATOR)
              (ORDER COORDINATOR EXTENSION)
              (LEXIFY COORDINATOR and | a | i | i))
              (0.33333334 ALTERNATIVE-COORDINATION
              (INSERT CORRELATOR)
              (ORDER CORRELATOR EXTENDED)
              (LEXIFY CORRELATOR either | bud3 | ili | ili)
              (INSERT COORDINATOR)
              (ORDER COORDINATOR EXTENSION)
              (LEXIFY COORDINATOR or | nebo | ili | ili))
              (0.33333334 CONTRASTIVE-COORDINATION
              (INSERT COORDINATOR)
              (LEXIFY COORDINATOR but | ale | no | no)
              (ORDER COORDINATOR EXTENSION)))
    :CHOOSER   EXTENDING-COORDINATION-TYPE-CHOOSER
    :REGION   CLAUSECOMPLEX
)

(CHOOSER
    :NAME   EXTENDING-CLAUSE-TYPE-CHOOSER
    :DEFINITION   ((IDENTIFY SUBORDINATOR
                   (CONJUNCTIVE-RELATION-ID ONUS))
                   (IDENTIFY SUBORDINATOR
                   (CONCEPTUAL-CORRELATE-ID SUBORDINATOR))
                   (ASK (CONTRASTIVE-EXTENSION-Q SUBORDINATOR ONUS)
                   (NONCONTRASTIVE
                   (ASK (CONJUNCTIVE-EXTENSION-Q SUBORDINATOR ONUS)
                   (NONCONJUNCTIVE
                   (ASK (DISJUNCTIVE-EXTENSION-Q SUBORDINATOR ONUS)
                   (DISJUNCTIVE (CHOOSE ALTERNATIVE-EXTENDING))))
                   (CONJUNCTIVE (CHOOSE ADDITIVE-EXTENDING))))
                   (CONTRASTIVE (CHOOSE VARYING-EXTENDING)))))
```

Figure 68: Extending coordination type: system and chooser (Cz, Ru, Bg)

Employing this implementation, we can generate sentences such as those in (32). The relevant concept which reflects the semantics of positive addition is conjunction. It is a logical relation, which can be used in an SPL as follows:

```
(EXAMPLE
  :SET-NAME IMDP-T5-Cz
  :NAME   IMD-T5-imp-14
  :TARGETFORM   "URČETE VNITŘNÍ BOD A STISKNĚTE RETURN "
  :LOGICALFORM
  (C / CONJUNCTION
    :DOMAIN
    (S1 / CREATIVE-MATERIAL-ACTION :LEX urc3it
          :SPEECHACT IMPERATIVE
          :ACTEE (P / OBJECT :LEX bod
          :IDENTIFIABILITY-Q notIDENTIFIABLE
          :PROPERTY-ASCRIPTION (P / QUALITY :LEX vnitr3ni2)   ))
    :RANGE
    (S2 / CREATIVE-MATERIAL-ACTION :LEX stisknout
          :SPEECHACT IMPERATIVE
          :ACTEE (A / OBJECT :NAME return
          :IDENTIFIABILITY-Q NOTIDENTIFIABLE) ) ) )
```

The grammatical structure of the generated sentence is shown in Figure 69.

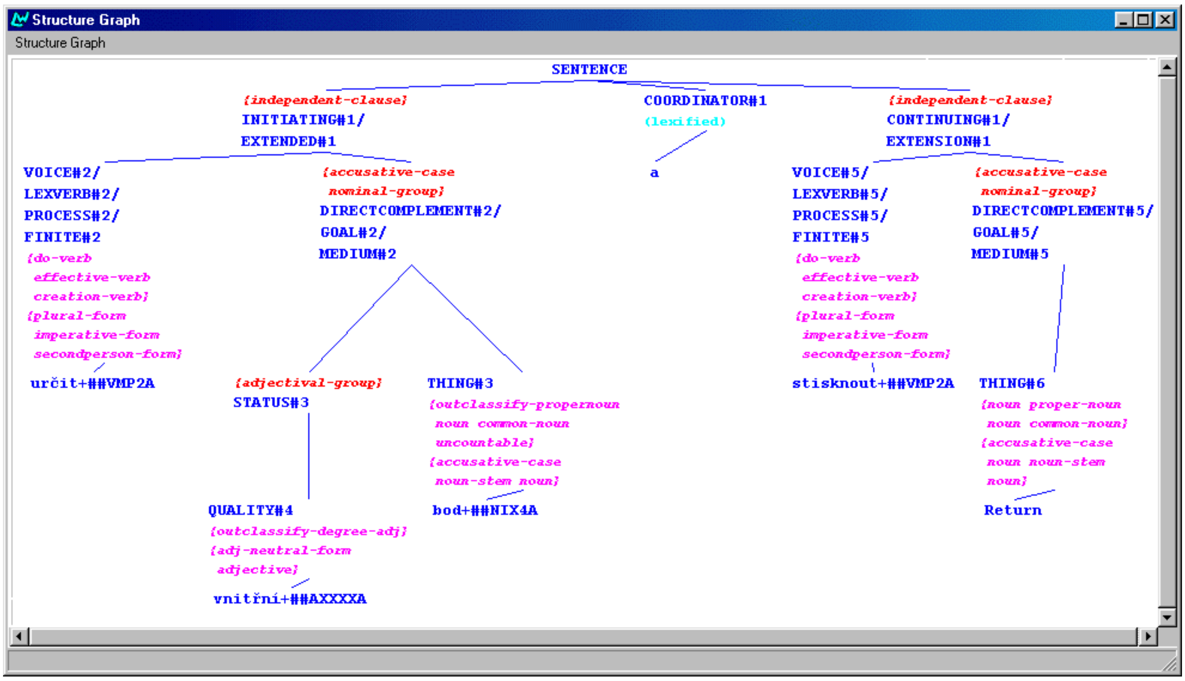


Figure 69: Generated structure for the sentence "Určete vnitřní bod a stiskněte Return" (example (32)).

2.7.2 Paratactic Enhancement

In the case of **enhancement**, one clause enhances the meaning of another by qualifying it by reference to time, place, manner, cause or condition (i.e., by giving circumstantial information). The type of paratactic enhancement included in the ImD is **causal-coordination**[7] of type reason (meaning "because P so result Q", where P is an action and Q is a goal to be achieved by action P), as exemplified in the following examples from the ImD texts.

(33) *Reason causal-coordination (text 4)*

(a) En: Choose OK and close the dialog box.

(b) Cz:. Vyberte OK a uzavřete dialogový panel.
Choose-2pl OK and close-2pl dialog-acc box-acc

(c) Bg: Изберете ОК и затворете диалоговия прозорец
Choose-2pl OK and close-2pl dialog box

(d) Ru: Нажмите кнопку OK и закройте диалоговое окно
Choose-2pl OK and close-2pl dialog-acc box-acc

The relevant system is specified below. We have to ensure that the action part is realized as the Enhancement and thus as initiating, and the Goal as the Enhanced and thus as

[7] What is called causal-coordination here corresponds to what we referred to as causal-conditional in the SPEC2 deliverable (Andonova *et al.*, 1999), where we closely followed Halliday's terminology (Halliday, 1985).

continuing. The corresponding implementations are shown in Figure 70. Note that here, also the respective connectives are inserted. We again collapse the realizations (in the order En, Cz, Bg, Ru), typeset them in bold and separate them by "|".

QUALIFYING-COORDINATION-TYPE:

```
      (qualifying-coordination) →
            [causal-coordination]
                  (Enhancement/Initiating,
                   Enhanced/Continuing,
                   Coordinator ! and | a | и | и),
            [temporal-coordination]
                  (Coordinator ! then | pak | след това | затем,
                   Enhanced/Initiating,
                   Enchancement/Continuing).
```

```
(SYSTEM
    :NAME   QUALIFYING-COORDINATION-TYPE
    :INPUTS   QUALIFYING-COORDINATION
    :OUTPUTS   ((0.5 TEMPORAL-COORDINATION
               (LEXIFY COORDINATOR TEMP-SUCC-COOR)
               (CONFLATE ENHANCED INITIATING)
               (CONFLATE ENHANCEMENT CONTINUING))
               (0.5 CAUSAL-COORDINATION
               (CONFLATE ENHANCEMENT INITIATING)
               (CONFLATE ENHANCED CONTINUING)
               (LEXIFY COORDINATOR CAUS-COOR)))
    :CHOOSER   QUALIFYING-COORDINATION-TYPE-CHOOSER
    :REGION   CLAUSECOMPLEX
)

(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 70: Qualifying-coordination type: system and chooser (Cz, Ru, Bg)

However, this implementation is not yet fully operational in the current versions of our grammars. We have not yet succeeded in the development of a proper linking between the Upper Model concepts and causal coordination. Therefore, even though the Upper Model contains the necessary concepts, and the grammar contains the necessary features, we have to modify the inquiry codes such that the concepts get properly realized.

For the time being, we generate sentences such as those in (33) by explicitly specifying conjunction in the SPL as follows:

```
(EXAMPLE
  :SET-NAME IMDP-T4-Cz
  :NAME   IMD-T4-imp-2-5
  :TARGETFORM   " Vyberte OK a uzavřete dialogový panel"
  :LOGICALFORM
  (c / conjunction
    :Domain
    (S2 / DISPOSITIVE -MATERIAL-ACTION :LEX zavr3i2t
          :SPEECHACT IMPERATIVE
          :ACTEE (A / OBJECT :lex panel
          :IDENTIFIABILITY-Q NOTIDENTIFIABLE

          :Property-aScription (P / Quality :lex dialogovy2)   )  )
    :Range
    (S1 / DISPOSITIVE-MATERIAL-ACTION :LEX vybrat
          :caused-process-q caused
          :SPEECHACT IMPERATIVE
          :ACTEE (P / OBJECT :NAME gui-ok ) )   )  )
```

The grammatical structure of the generated sentence is shown in Figure 71.

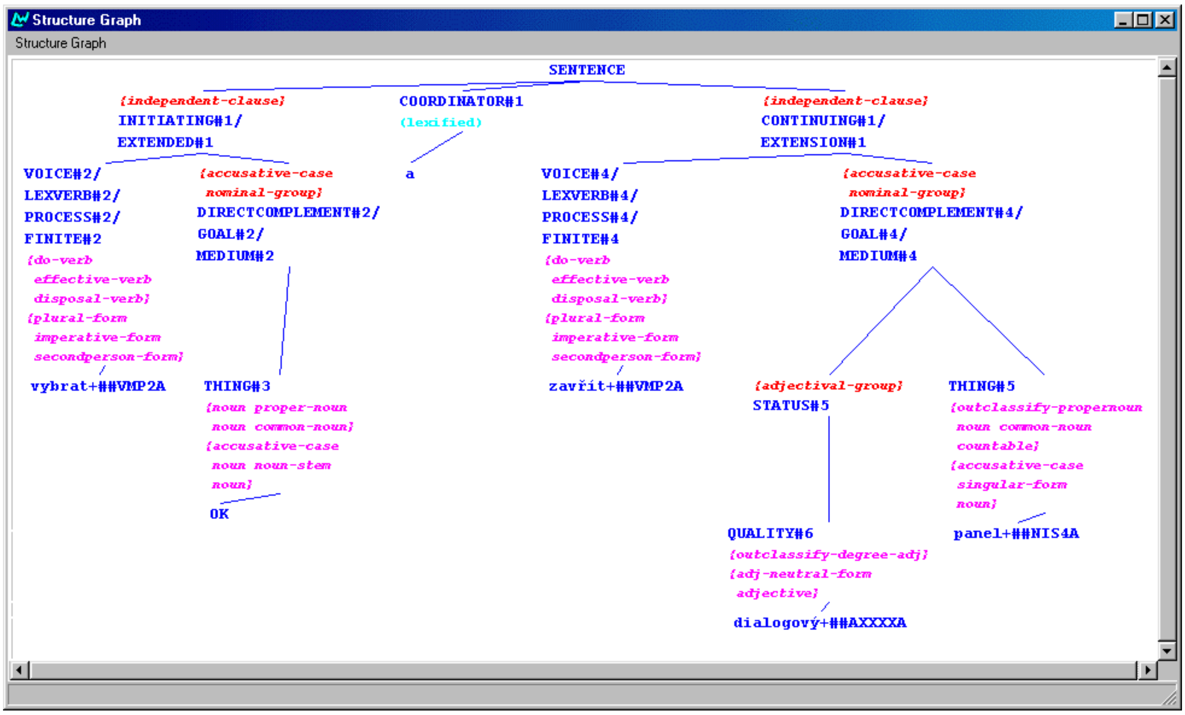


Figure 71: Generated structure

for “Vyberte OK a zavřete dialogový panel” (example (33))

2.7.3 Hypotactic Enhancement

For the ImD, we consider the types of hypotactic enhancement listed in the table in Figure 72 below.

Category		*Meaning*
Manner	means	N by means of M
Causal-coordination	cause: purpose	because intention Q so action P

Figure 72: Types of hypotactic enhancement covered in intermediate prototype

The following examples from the ImD texts illustrate these types. We show examples where the dependent part is realized either by a finite or non-finite clause. Means and Purpose Circumstances can also be realized by nominalization, in which case it is a circumstantial complementation of the main verb rather than a dependent clause. See examples (34) below.

(34) *Purpose causal-coordination Circumstance (text 2)*

(a) En: Press Return to end the polyline.

(b) Cz:
(pers. imp., dep. finite)
Stiskněte Return, abyste křivku ukončili
Press-2pl Return would-2pl polyline-acc end-2pl-pastparticiple

(impers. refl., dep. finite)
Stiskne se Return, aby se křivka ukončila
Press-3sg refl Return would-3sg refl polyline-nom end-3sg-pastparticiple

(pers. imper., dep. finite)
Stiskněte Return, abyste křivku ukončili
Press-2pl Return, would-2pl polyline-acc end-2pl-pastparticiple
Press Return so that you would end the polyline.

(impers. refl., dep. finite)
Stiskne se Return, aby se křivka ukončila
Press-3sg refl Return, would-3sg refl polyline-nom end-3sg-pastparticiple
Return is pressed so that the polyline would be ended.

(c) Bg:
(pers. imper., dep. finite)
Натиснете Return, за да завършите полилинията.
Press-2pl Return so that end-2pl polyline

(d) Ru:
(pers. imper., dep. nonfinite)
Нажмите клавишу Return, чтобы завершить рисование полилинии.
Press-2pl key Return, in-order-to end drawing polyline-gen

(35) *Means (Manner) Circumstance (text 2)*[8]

(a) En: Start the PLINE command using one of the following methods

(b) Bg:
(pers. imper., dep. finite)
Стартирайте командата PLINE, като използвате един от следните методи
Start-2pl command-the PLINE, by use-2pl one of following methods

(impers. refl., dep. finite)
Стартира се командата PLINE, като се използва един от следните методи
Start-3sg refl command-the PLINE, by refl use-3sg one of following methods

[8] Neither the intermediate prototype texts nor the rest of the AGILE corpus contain occurrences of Manner Circumstance realized by a dependent clause in Czech.

(c) Ru:
(pers. imper., dep. nonfinite)
Запустите команду PLINE, воспользовавшись одним из следующих способов
Start-2pl command-acc PLINE using-gerund one-instr of following-gen methods-gen

Besides the realization of the Means or Purpose Circumstance by a dependent clause, it is also possible to realize it by a nominalization in some cases. This is demonstrated in the following examples:

(36) *Purpose nominalized Circumstance (text 2)*

(a) En: Press Return to end the polyline.

(b) Cz:
(pers. imp., rank shifted circumstance)
Stiskněte Return pro ukončení křivky
Press-2pl Return for ending polyline-gen

(impers. refl., rank shifted circumstance)
Stiskne se Return pro ukončení křivky
Press-3sg refl Return for ending polyline-gen

(c) Bg:
(impers. refl., rank shifted circumstance)
Натиска се Return за завършване на полилинията
Press-3sg refl Return for ending of polyline

(37) *Means manner circumstance (text 2)*

(a) En: Start the PLINE command using one of the following methods

(b) Cz:
(pers. imper., rank shifted circumstance)
Spusťte příkaz KŘIVKA použitím jednoho z následujících způsobů
Start-2pl command PLINE using-instr one-gen of following-gen methods-gen

(impers. refl., rank shifted circumstance)
Spustí se příkaz KŘIVKA použitím jednoho z následujících způsobů
Start-3sg refl command PLINE using-instr one-gen of following-gen methods-gen

Besides the realization of means by a dependent clause or by a rank shifted circumstance there is another possibility in Czech and Russian, similarly to English. One can realize the same content in the following way:

(38) *Means circumstantial complementation (text 2)*

(a) En: Start the PLINE command by one of these methods

(b) Cz:

(pers. imper.)
Spusťte příkaz KŘIVKA jedním z následujících způsobů
Start-2pl command PLINE one-instr of following-gen methods-gen

(impers. refl.)
Spustí se příkaz KŘIVKA jedním z následujících způsobů
Start-3sg refl command PLINE one-instr of following-gen methods-gen

(c) Ru:

(pers. imper.)
Запустите команду PLINE одним из следующих способов
Start-2pl command PLINE one-instr of following-gen methods-gen

This possibility is not available in Bulgarian, however. The realization of Means or Purpose by a circumstantial complementation rather than by a dependent clause needs to be accounted for in the transitivity region, which is not the focus of the present chapter.

Now we concentrate on the cases of realization by clause complexes. The relevant system is specified below, and its implementation is shown in Figure 73. They are the same for English and our three languages. What we have to ensure is the appropriate realization of the dependent clauses as finite or non-finite. This is handled in the region of DEPENDENCY.

QUALIFYING-CONDITION-TYPE:

```
      (qualifying-condition) →
            [concessive-condition]
                  (*Enhancement : concession-dependent)
            [conditional-condition] →
                  (*Enhancement : conditional-dependent)
            [purposive-condition]
                  (*Enhancement : purpose-dependent)
            [manner-condition]
                  (*Enhancement : manner-dependent)
            [spatial-condition]
                  (*Enhancement : spatial-dependent)
            [causal-condition]
                  (*Enhancement : causal-dependent)
            [temporal-condition]
                  (*Enhancement : temporal-dependent)
```

```
(SYSTEM
    :NAME   QUALIFYING-CONDITION-TYPE
    :INPUTS   QUALIFYING-CONDITION
    :OUTPUTS   ((0.14285715 TEMPORAL-CONDITION
               (* PRESELECT ENHANCEMENT TEMPORAL-DEPENDENT))
               (0.14285715 CAUSAL-CONDITION
               (* PRESELECT ENHANCEMENT CAUSE-DEPENDENT))
               (0.14285715 SPATIAL-CONDITION
               (* PRESELECT ENHANCEMENT SPATIAL-DEPENDENT))
               (0.14285715 MANNER-CONDITION
               (* PRESELECT ENHANCEMENT MANNER-DEPENDENT))
               (0.14285715 PURPOSIVE-CONDITION
               (* PRESELECT ENHANCEMENT PURPOSE-DEPENDENT))
               (0.14285715 CONDITIONAL-CONDITION
               (* PRESELECT ENHANCEMENT CONDITIONAL-DEPENDENT))
               (0.14285715 CONCESSIVE-CONDITION
               (* PRESELECT ENHANCEMENT CONCESSION-DEPENDENT)))
    :CHOOSER   QUALIFYING-CONDITION-TYPE-CHOOSER
    :REGION   CLAUSECOMPLEX
    :METAFUNCTION   LOGICAL
)

(CHOOSER
    :NAME   QUALIFYING-CONDITION-TYPE-CHOOSER
    :DEFINITION ((ASK (MANNER-CONDITION-Q ENHANCED ENHANCEMENT)
                (MANNERCONDITION (CHOOSE MANNER-CONDITION))
                (NONMANNERCONDITION
                (ASK (CAUSE-CONDITION-Q ENHANCED ENHANCEMENT)
                (CAUSECONDITION (CHOOSE CAUSAL-CONDITION))
                (NONCAUSECONDITION
                (ASK (LOGICAL-CONDITION-Q ENHANCED ENHANCEMENT)
                (LOGICALCONDITION (CHOOSE CONDITIONAL-CONDITION))
                (NONLOGICALCONDITION (ASK (CONCESSIVE-CONDITION-Q
                 ENHANCED ENHANCEMENT)
                (CONCESSIVE (CHOOSE CONCESSIVE-CONDITION))
                (NOTCONCESSIVE
                (ASK (PURPOSE-CONDITION-Q ENHANCED
                 ENHANCEMENT)
                (PURPOSECONDITION
                (CHOOSE PURPOSIVE-CONDITION))
                (NONPURPOSECONDITION
                (ASK (SPACE-CONDITION-Q ENHANCED
                 ENHANCEMENT)
                (SPACECONDITION (CHOOSE SPATIAL-CONDITION))
                (NONSPACECONDITION (ASK (TIME-CONDITION-Q
                 ENHANCED ENHANCEMENT)
                (TIMECONDITION
                (CHOOSE TEMPORAL-CONDITION))
                (NONTIMECONDITION))))))))))))) ))
```

Figure 73: Qualifying-condition type: system and chooser (Cz, Ru, Bg)

With the current implementations, we are able to generate clause complexes like those in examples (34) and (35) above. We will now demonstrate the generation of purpose hypotactic enhancement clauses in Russian and in Bulgarian. As shown above, Russian realizes the purpose dependent clauses as non-finite. We use the following SPL to specify the semantics of the sentence in (34d):

```
(EXAMPLE
    :NAME   D1-TEXT2-12-RU
    :GENERATEDFORM "Nazhmite klavishu Return, chtoby zavershitj risovanie
                   polilinii."
```

```
 :TARGETFORM    "Nazhmite klavishu Return, chtoby zavershitj risovanie
                 polilinii."
 :LOGICALFORM (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))))
 :SET-NAME   D1-TEXT2)
```

The generated grammatical structure for (34d) is depicted in Figure 74.

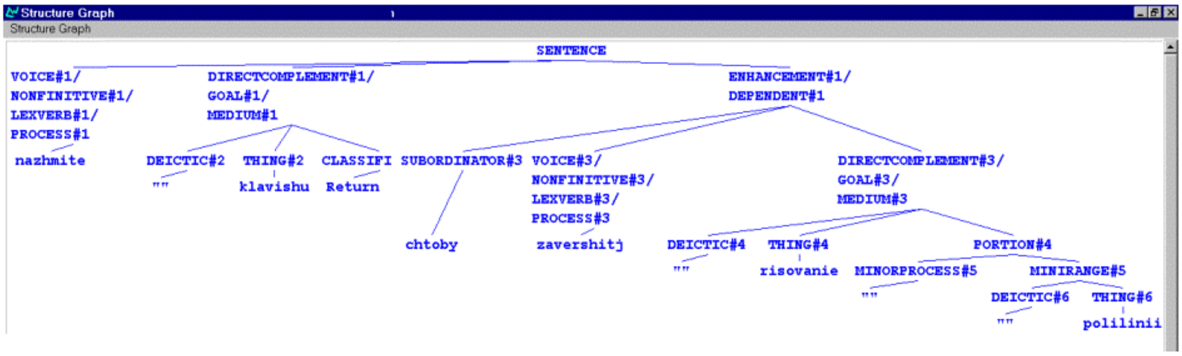


Figure 74: Generated structure for the sentence from “Нажмите клавишу Return, чтобы завершить рисование полилинии.” (example (34d))

In Bulgarian and in Czech, unlike in Russian, purpose dependent clauses are realized as finite. We use the following SPL to specify the semantics of the Bulgarian sentence in (34c):

```
(EXAMPLE
 :NAME   PERS-TEXT2-14
 :TARGETFORM   " Натиснете Return, за да завършите полилинията."
 :LOGICALFORM
   (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)))
 :SET-NAME   PERS-TEXT2)
```

The generated grammatical structure for (34c) is depicted in Figure 75.

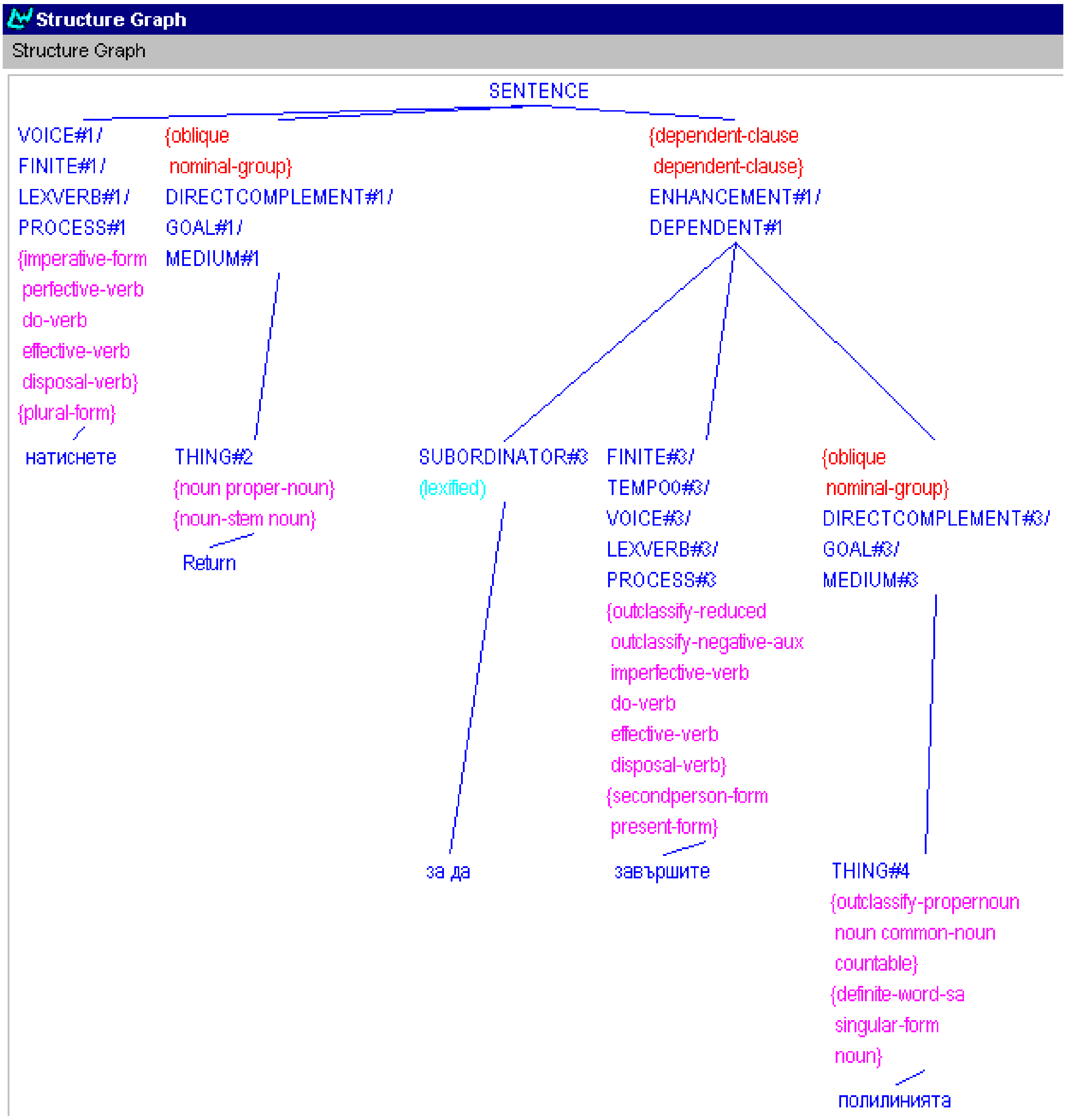


Figure 75: Generated structure for the sentence from “Натиснете Return, за да завършите полилинията." (example (34c))

2.8 Determination

In this chapter, we describe the implementations in the region of DETERMINATION, as relevant for the ImD. In the SPEC2 deliverable (Andonova *et al.*, 1999), we discussed the essential types of determination in Czech, Russian and Bulgarian in comparison to English, and the various possibilities of realization of the Deictic element in these languages in general, as well as with respect to instructional texts. The work reported on in the present chapter has a narrower focus: We concentrate on the realization of the Deictic element insofar it is encountered in the AGILE ImD texts.

As with other regions of our grammars, we build upon the Nigel grammar for English. We also discussed in the LSPEC2 deliverable (Andonova *et al.*, 1990) how Nigel implements SFG's notion of determination within the KPML framework, and how far the approach taken in the Nigel grammar for English can be adopted for generating sentences in Czech, Bulgarian, or Russian. Here, we present a self-contained account of the current implementations for our three languages.

Determination is one the aspects involved in nominal reference. The notion of specific determination means that a particular subset (class of things) is denoted and that it is identifiable both for the speaker and the hearer. Non-specific determination can either span over the entire set of denoted entities (total) or a subset thereof (partial).

In English, the function of determination is carried by the Deictic element, which indicates whether the class of things referred to by the nominal group is specific (unique) or non-specific (non-unique). The situation is similar in Slavic languages, as we discussed in the LSPEC2 deliverable (Andonova *et al.,* 1999). However, there are differences. First of all, Czech and Russian do not have a definite or indefinite article. Bulgarian does not have an indefinite article, while it has a definiteness marker, which is realized as a suffix. Another characteristic of Slavic languages, especially Czech and Russian, is that specificity vs. non-specificity can be reflected by word order, which in turn reflects information structure (see the chapter on word order in this deliverable (Section 2.9), as well as LSPEC2 (Andonova *et al.*, 1999) for detailed discussions).

The present chapter is structured as follows. We first discuss the issue of explicit vs. implicit Deictic element in Slavic languages (Section 2.8.1). Then we address explicit specific determination (Section 2.8.2). A particular issue discussed separately is the generation of the definite suffix in Bulgarian (Section 2.8.2.3). Then we turn to explicit non-specific determination (Section 2.8.3). We also briefly mention the generation of the Post-deictic element (Section 2.8.4). As we proceed, we provide examples of generating the various kinds of determination our grammars currently cover.

2.8.1 Explicit vs. Implicit Deictic Element

There is an important difference between English and Slavic languages concerning the realization of the Deictic element: Czech, Russian and Bulgarian differ from English in that they do not have articles, neither definite nor indefinite; Bulgarian, unlike Czech and Russian, has a counterpart to the English definite article. It is a morpheme, which marks specificity. It is attached as a suffix to the first element of a nominal group. In LSPEC2 (Andonova *et al.*, 1999), we introduced the common term "applicator" to refer to the means of marking specificity and non-specificity, be they separate words or morphemes. We said that the presence of any applicator is not syntactically obligatory in Slavic languages. The following examples taken from the ImD text 1 illustrate this:

(39)

(a) Cz: Zvolte barvu elementu
Select color-acc element-gen

(b) Ru: Выберите цвет элемента.
Select color-acc element-gen

(c) En: Select the element's color

In this example, color is unspecific and element is specific. None of the sentences in Russian or Czech contains any explicit realization of any Deictic element. The following example illustrates the use of a suffix to mark specificity in Bulgarian:

(40) *Bg: Посочете цвета на елемента.*
Select color-def of element-def
Select the color of the element.

Both nominal groups in (42) include a specific nonselective Deictic element, which is realized by the short form of the definite suffix, i.e. *a*.

The question that emerges is how to capture the cases where no explicit realization in the form of a Deictic element is present. One possible view is that similarly to English, all nominal groups contain a Deictic element, but it can be realized as an empty string. Under this view, a Deictic element is inserted whenever a nominal group is being generated. This is the view that is currently adopted in the implementations of the grammars for Russian and for Bulgarian.

An alternative view is that nominal groups do not have to contain a Deictic element, unless an explicit Deictic element is specified by the semantics. In the current implementation of the AGILE grammar for Czech, we decided to adopt this latter view. It enables us concentrate the choice between explicit vs. implicit Deictic element in one system, no matter how complex the decision criteria may be. This solution is modular and therefore convenient from the point of view of grammar development and text planning.

So, in the Czech grammar implementation, we have created the following new system called DEICTIC-EXPLICITNESS in the DETERMINATION region in which the decision about inserting or not inserting the Deictic element is made (see Figure 76 for the system and chooser implementations):

```
DEICTIC-EXPLICITNESS
{NOMINAL-SPECIFIC; NOMINAL-NONSPECIFIC;
      NOMINAL-RELATIVE; NOMINAL-INTERROGATIVE} →
      [EXPLICIT-DEICTIC] (+DEICTIC)
      [IMPLICIT-DEICTIC]
```

The input features of this system are generated in the NOUNTYPE region. The explicit Deictic element gets realized throughout a number of systems (see Sections 2.8.2 and 2.8.3).

The question when to insert an explicit Deictic element and when not to insert one is not trivial at all in the Slavic languages. Nominal groups in Czech and Russian and also non-specific nominal groups in Bulgarian do not have to contain an explicit Deictic element when specificity or non-specificity can be deduced by the hearer from the context or from general knowledge. Also, it is possible that the speaker does not make deicticity explicit because it is not important for the content she is communicating. When discussing determination in the SPEC2 deliverable (Andonova *et al.*, 1999) we also pointed to the correlation of information structure and determination. In particular, in Czech and Russian, there is a tendency to interpret contextually bound nominal groups as specific.[9] However, this is not more than a tendency.

```
(SYSTEM
   :NAME   DEICTIC-EXPLICITNESS
   :INPUTS   (OR NOMINAL-SPECIFIC NOMINAL-NONSPECIFIC
              NOMINAL-RELATIVE NOMINAL-INTERROGATIVE)
   :OUTPUTS  ((0.3 EXPLICIT-DEICTIC
             (INSERT DEICTIC))
             (0.7 IMPLICIT-DEICTIC))
   :CHOOSER   DEICTIC-EXPLICITNESS-CHOOSER
   :REGION   DETERMINATION )

(CHOOSER
   :NAME   DEICTIC-EXPLICITNESS-CHOOSER
   :DEFINITION   ((ASK (IDENTIFIABILITY-Q THING)
                 (IDENTIFIABLE (CHOOSE EXPLICIT -DEICTIC))
                 (NOTIDENTIFIABLE
                 (CHOOSE IMPLICIT-DEICTIC)))) )
```

Figure 76 Deictic expliciteness: system and provisional chooser (Cz)

At this point we provide a provisional implementation which is very simplified in regard to the decision whether or not to insert an explicit Deictic element. For the time being, we link this decision to identifiability in the following way: When an entity is considered identifiable, we insert an explicit Deictic element, when it is not identifiable, no Deictic element is inserted. Identifiability is usually specified in the SPL. If it is not specified, the default choice is not-identifiable. Even though this approach is simplified for Czech and Russian, it seems to be the right one for Bulgarian, where the definite suffix indeed does reflect identifiability.

The current implementation of the Czech grammar thus enables us to generate nominal groups without a Deictic element. We demonstrate the grammatical structure generated for the sentence in example (39) in Figure 77. It is obtained using the following SPL:

```
(EXAMPLE
  :NAME      no-deictic
  :SET-NAME   Pokus
  :TARGETFORM   "Zvolte barvu elementu"
  :LOGICALFORM (S / dispositive-material-action  :LEX zvolit
       :SPEECHACT Imperative
       :ACTEE
             (P / Object       :LEX barva
             :Process-modification-q noprocess
             :generalized-possession-inverse
             (M / Object       :LEX multic3a2ra))))
```

[9] See LSPEC2 (Andonova *et al.,* 1999) for the definition of contextual boundness.

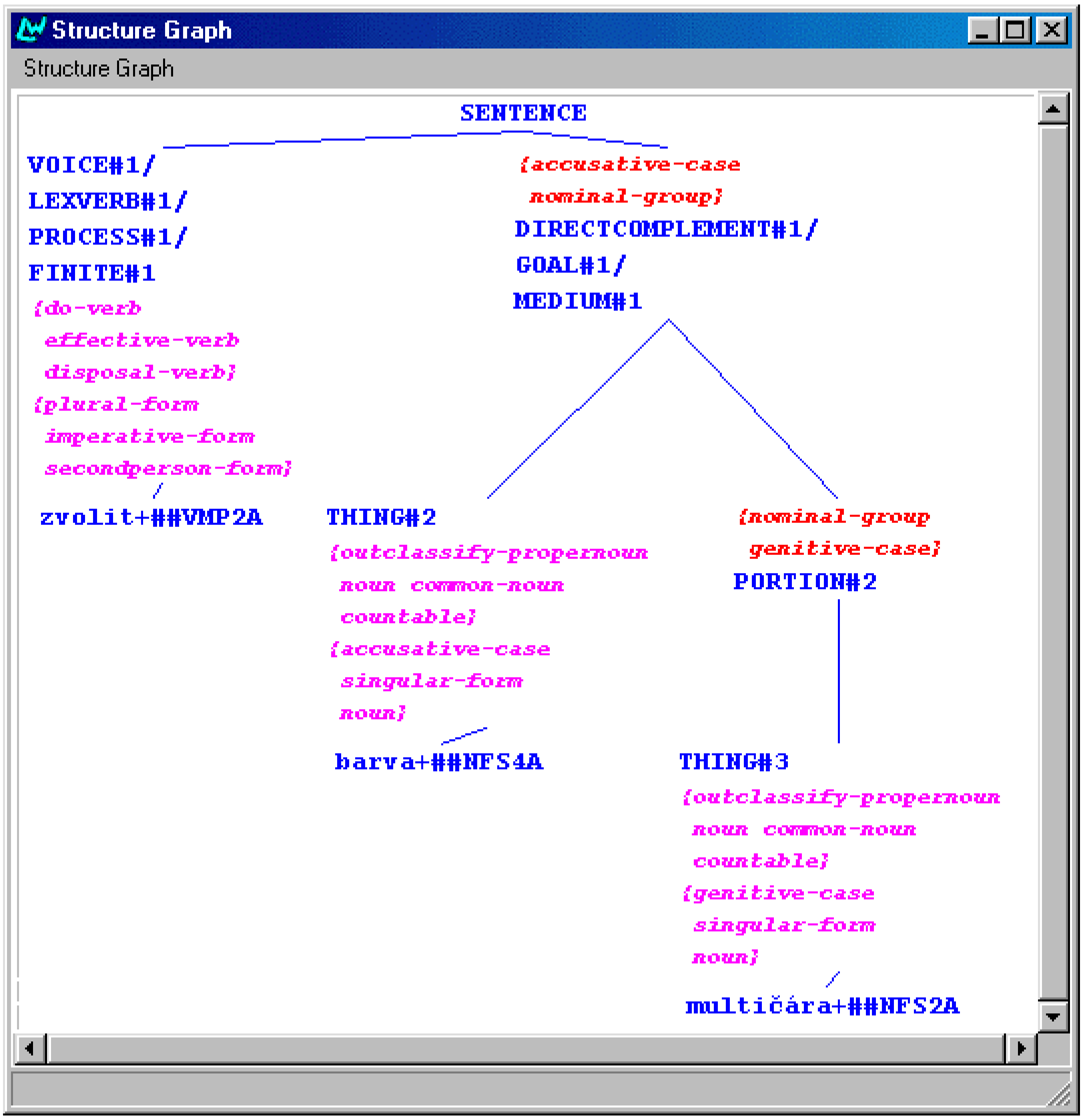


Figure 77: Generated structure for the Czech sentence “Zvolte barvu elementu” (example (39a))

As mentioned above, we are also able to generate an explicit Deictic element. Let us now discuss the relevant implementations in more detail. We first dicuss specific determination and than non-specific determination.

2.8.2 Specific Determination

The generation of a nominal group involving specific determination is entered through the feature **nominal-specific**. The first system on this path through the network is called SPECIFIC-TYPE. It is defined as follows (see Figure 78 for the corresponding implementations):

```
SPECIFIC-TYPE
(nominal-specific)→
[nonselective] Deictic ! Cz:Ten | Ru:Etot | Bg:
[possessive-selection]
                  Deictic : nominal-group
                  Deictic : genitive
[demonstrative-selection]
[time-specific] Deictic / Thing
```

```
 (SYSTEM
    :NAME   SPECIFIC-TYPE
    :INPUTS   NOMINAL-SPECIFIC
    :OUTPUTS   ((0.25 NONSELECTIVE
                   (Czech : LEXIFY DEICTIC TEN)
                   (Russian : LEXIFY DEICTIC ETOT)
                   (Bulgarian :))))
                   (0.25 POSSESSIVE-SELECTION
                   (PRESELECT DEICTIC NOMINAL-GROUP)
                   (PRESELECT DEICTIC GENITIVE))
                   (0.25 DEMONSTRATIVE-SELECTION)
                   (0.25 TIME-SPECIFIC
                   (CONFLATE DEICTIC THING)))
    :CHOOSER   SPECIFIC-TYPE-CHOOSER
    :REGION   DETERMINATION
 )

(CHOOSER
    :NAME   SPECIFIC-TYPE-CHOOSER
    :DEFINITION
        ((ASK (PROXIMITY-MODIFICATION-Q THING)
        (PROXIMITY (IDENTIFY DEICTIC (PROXIMITY-MOD-ID THING))
        (IDENTIFY DEICTIC (CONCEPTUAL-CORRELATE-ID DEICTIC))
        (CHOOSE DEMONSTRATIVE-SELECTION))
        (NOPROXIMITY
        (ASK (POSSESSOR-MODIFICATION-Q THING)
        (POSSESSOR (IDENTIFY DEICTIC(POSSESSOR-MOD-ID THING))
        (IDENTIFY DEICTIC
        (CONCEPTUAL-CORRELATE-ID DEICTIC))
        (PLEDGE DEICTIC ONUS)
        (CHOOSE POSSESSIVE-SELECTION))
        (NOPOSSESSOR (ASK (TIME-Q THING)
        (TIME (ASK (CALENDER-TERM-Q THING)
        (NOTCALENDRIC
        (ASK (PERIOD-MODIFICATION-Q THING)
        (PERIOD (ASK (CALENDER-TERM-Q PERIOD)
        (NOTCALENDRIC
        (CHOOSE DEMONSTRATIVE-SELECTION))
        (CALENDRIC (CHOOSE TIME-SPECIFIC))))
        (NOPERIOD (CHOOSE NONSELECTIVE))))
         CALENDRIC (CHOOSE TIME-SPECIFIC))))
        (NOTIME (CHOOSE NONSELECTIVE))))))))
```

Figure 78 Specific type: system and chooser (Cz, Bg, Ru)

We are currently interested only in the features **nonselective** and **demonstrative-selection**, because the others are not needed for the ImD. In Czech and Russian, the specific Deictic element is realized by a demonstrative pronoun. In Bulgarian, specific non-selective determination is reflected by the definite suffix. Let us now first describe the implementations for Czech and Russian (Section 2.8.2.1), and then turn to those for Bulgarian (Section 2.8.2.2).

2.8.2.1 Nonselective Specific Determination in Czech and Russian

The **nonselective** feature leads to a specific Deictic realized as follows: in Czech, it is realized by the demonstrative pronoun of neutral proximity, namely *ten*; in Russian, it is realized by the demonstrative pronoun of near proximity, namely *этот*. A modification of the examples shown above illustrates this:

(41)

(a) Cz: Zvolte barvu toho elementu
Select color-acc the-gen element-gen

(b) Ru: Выберите цвет этого элемента.
Select color-acc the-gen element-gen

(c) En: Select the element's color

2.8.2.2 Nonselective Specific Determination in Bulgarian

The **nonselective** feature leads to a specific Deictic realized by the definite suffix in Bulgarian. We use our external morphological module, in order to get the right word form. When a particular word in the generated structure has the feature **definite–word**, the morphological module returns the word with the definite suffix attached to it.

Therefore, the element of the nominal group which is to carry the definiteness marker must get the grammatical feature **definite–word**. One of the specifics of Bulgarian is the existence of a full and a short form of the article for the words having masculine gender. So, the final scheme is as follows:

- Nominal group in nominative case → definite-word plus full-suffix
- Nominal group in oblique case → definite-word plus short-suffix

In order to generate a determined noun in a nominal group we use the systems in Figure 79 Figure 76.

```
(GATE
    :NAME  NOMINATIVE-NONSELECTIVE-NOUN
    :INPUTS   (AND NONSELECTIVE NOMINATIVE
                   NOT-STATUS-MODIFIED NOT-AGE-MODIFIED
                   NOT-COLOUR-MODIFIED NOT-SIZE-MODIFIED
                   NO-POST-DEICTIC)
    :OUTPUTS   ((1.0 FULL-ARTICLE
              (INFLECTIFY THING DEFINITE-WORD-FA)))
    :REGION   DETERMINATION
)
(GATE
    :NAME  OBLIQUE-NONSELECTIVE-NOUN
    :INPUTS   (AND NONSELECTIVE OBLIQUE
                   NOT-STATUS-MODIFIED NOT-AGE-MODIFIED
                   NOT-COLOUR-MODIFIED NOT-SIZE-MODIFIED
                   NO-POST-DEICTIC)
    :OUTPUTS   ((1.0 SHORT-ARTICLE
              (INFLECTIFY THING DEFINITE-WORD-SA)))
    :REGION   DETERMINATION
)
```

Figure 79: Nominative-nonselective and Oblique-nonselective gates (Bg)

The input condition (AND OBLIQUE NONSELECTIVE) of the systems in Figure 79 above leads to the insertion of the full and short suffix form according to the the rules of the Bulgarian grammar.

In order to obtain a determined noun (head of the nominal group) only for the cases when the noun has the definite suffix, the input conditions of the systems NOMINATIVE-NONSELECTIVE-NOUN and OBLIQUE-NONSELECTIVE-NOUN are (nonselective nominative not-status-modified not-age-modified not-colour-modified not-size-modified no-post-deictic) and (nonselective oblique not-status-modified not-age-modified not-colour-modified not-size-modified no-post-deictic), respectively. The mentioned input conditions are sufficient for the generation of the nominal groups in our texts, although they may not cover all the cases in the Bulgarian language in general.

The definite suffix is attached to the first element of a nominal group. This is achieved by using the systems shown in Figure 80. By means of the system ADJECTIVAL-GR-DETERMINATION-FA, we can generate the full suffix, and trough the system ADJECTIVAL-GR-DETERMINATION-SA, we are able to generate the short suffix. Both systems use the chooser ADJECTIVAL-GR-DETERMINATION-CHOOSER.

We demonstrate the generation of the nonselective specific Deictic elements realized by definite suffixes in Bulgarian by showing the structure generated for the following sentence (see Figure 81):

(42)

Bg: Диалоговият прозорец Select Linetype се появява на екрана
Dialogue-def window Select Linetype refl appears on screen-def
The dialogue box Select Linetype appears on the screen.

We use the following SPL to generate this sentence:

```
(EXAMPLE
    :NAME   NONPERS-TEXT1-11-1
    :LOGICALFORM
    (C / NONDIRECTED-ACTION :LEX POIAVIAVAM-SE
      :ASPECT-Q PERFECTIVE-ASPECT
      :ACTOR (C1 / OBJECT
            :property-ascription (q1 / quality :lex DIALOGOV)
            :LEX prozorec
            :CLASS-ASCRIPTION (L2 / SOFTWARE-COMMAND :NAME SELLINT)
      :IDENTIFIABILITY-Q IDENTIFIABLE )
      :SPATIAL-LOCATING
      (P / ONE-OR-TWO-D-LOCATION :LEX EKRAN
      :IDENTIFIABILITY-Q IDENTIFIABLE))
    :SET-NAME   NONPERS-TEXT1)
```

The important characteristics in the generated grammatical structure in

Figure 81 are:

definite-word-FA for Quality;

absence of any characteristics for the Thing / Subject

definite-word-SA for the Minirange / Thing

```
 (SYSTEM
    :NAME   ADJECTIVAL-GR-DETERMINATION-FA
    :INPUTS   (AND NOMINATIVE NONSELECTIVE
              (OR STATUS-MODIFIED AGE-MODIFIED
               COLOUR-MODIFIED SIZE-MODIFIED
               POST-DEICTIC))
    :OUTPUTS  ((0.2 STATUS-FULL-ARTICLE
              (PRESELECT STATUS DEFINITE-WORD-FA))
              (0.2 AGE-DETERMINATION
              (PRESELECT AGE DEFINITE-WORD-FA))
              (0.2 COLOUR-DETERMINATION
              (PRESELECT COLOUR DEFINITE-WORD-FA))
              (0.2 SIZE-DETERMINATION
              (PRESELECT SIZE DEFINITE-WORD-FA))
              (0.2 POST-DEICTIC-DETERMINATION
              (PRESELECT POST-DEICTIC DEFINITE-WORD-FA)))
    :CHOOSER   ADJECTIVAL-GR-DETERMINATION-CHOOSER
    :REGION   DETERMINATION
)
(SYSTEM
    :NAME   ADJECTIVAL-GR-DETERMINATION-SA
    :INPUTS   (AND OBLIQUE NONSELECTIVE
              (OR STATUS-MODIFIED AGE-MODIFIED
               COLOUR-MODIFIED SIZE-MODIFIED
               POST-DEICTIC))
    :OUTPUTS  ((0.2 STATUS-DETERMINATION
              (PRESELECT STATUS DEFINITE-WORD-SA))
              (0.2 AGE-DETERMINATION
              (PRESELECT AGE DEFINITE-WORD-SA))
              (0.2 COLOUR-DETERMINATION
              (PRESELECT COLOUR DEFINITE-WORD-SA))
              (0.2 SIZE-DETERMINATION
              (PRESELECT SIZE DEFINITE-WORD-SA))
              (0.2 POST-DEICTIC-DETERMINATION
              (PRESELECT POST-DEICTIC DEFINITE-WORD-SA)))
    :CHOOSER   ADJECTIVAL-GR-DETERMINATION-CHOOSER
    :REGION    DETERMINATION
)
(CHOOSER
    :NAME ADJECTIVAL-GR-DETERMINATION-CHOOSER
    :DEFINITION    ((ASK (STATUS-MODIFICATION-Q THING)
                   (STATUS (CHOOSE STATUS-DETERMINATION))
                   (NONSTATUS (ASK (SIZE-MODIFICATION-Q THING)
                   (SIZE (CHOOSE SIZE-DETERMINATION))
                   (NONSIZE (ASK (COLOUR-MODIFICATION-Q THING)
                   (COLOUR (CHOOSE COLOUR-DETERMINATION))
                   (NONCOLOUR (ASK (AGE-MODIFICATION-Q THING))
                   (AGE (CHOOSE AGE-DETERMINATION))
                   (NONAGE (ASK (SELECTION-MODE-SPECIFIED-Q THING)
                   (SPECIFIED (CHOOSE POST-DEICTIC-DETERMINATION))
                   (UNSPECIFIED
                   (CHOOSE POST-DEICTIC-DETERMINATION)))))))))))
```

Figure 80: Adjectival group determination: systems and chooser (Bg)

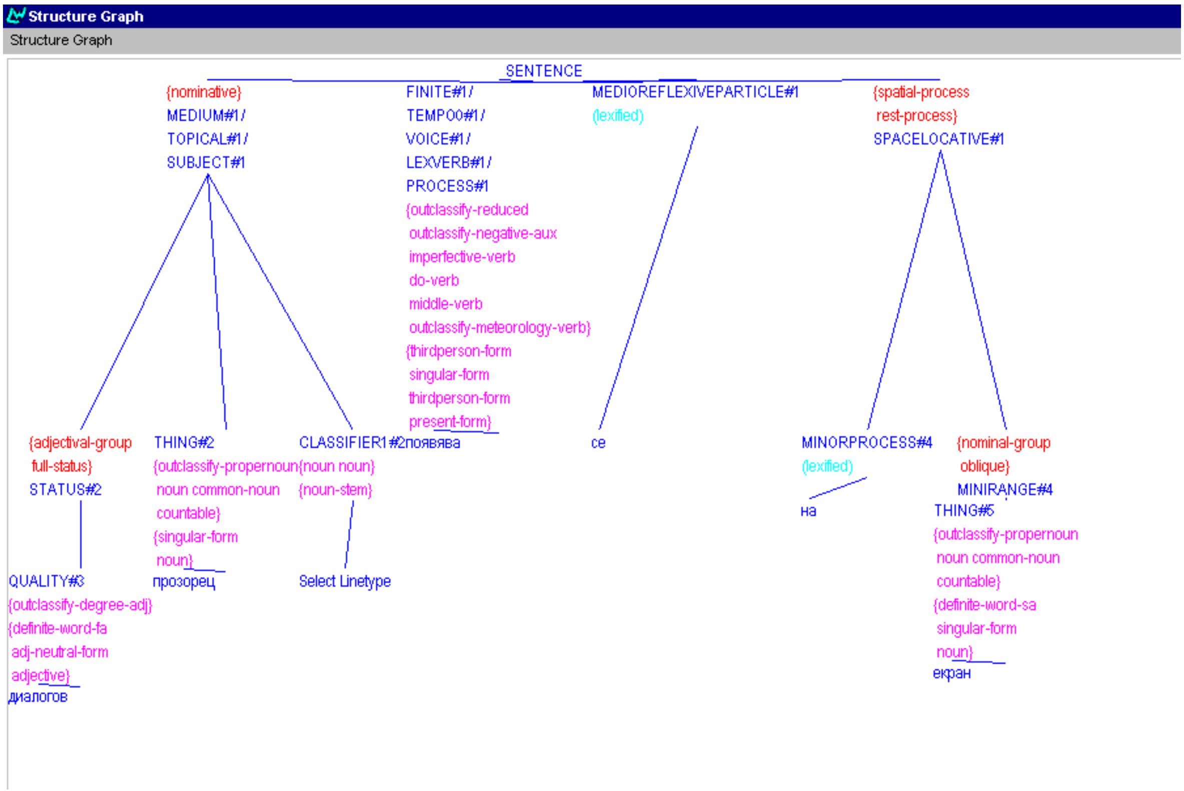


Figure 81: Generated structure for the Bulgarian sentence “Диалоговият прозорец Select Linetype се появява на екрана” (example (42))

2.8.2.3 *Demonstrative Specific Determination*

The **demonstrative-selection** feature in the SPECIFIC-TYPE system leads further to the DEMONSTRATIVE-NONQUESTIONING system, where we distinguish on the basis of proximity in all three languages. We should ideally distinguish between **unmarked, near** and **far**. However, for the purpose of the ImD the **near** and **far** features suffice (see Figure 82 for the implementations):

```
DEMONSTRATIVE-NONQUESTIONING
({demonstrative-specific-pronoun;
     demonstrative-selection})→
[proximity-near]
       [proximity-far]
```

```
(SYSTEM
    :NAME   DEMONSTRATIVE-NONQUESTIONING
    :INPUTS   (OR DEMONSTRATIVE-SPECIFIC-PRONOUN DEMONSTRATIVE-SELECTION)
    :OUTPUTS  ((0.5 NEAR)
              (0.5 FAR))
    :CHOOSER   DEMONSTRATIVE-NONQUESTIONING-CHOOSER
    :REGION   DETERMINATION
    :METAFUNCTION   TEXTUAL
 )

(CHOOSER
    :NAME   DEMONSTRATIVE-NONQUESTIONING-CHOOSER
    :DEFINITION   ((ASK (DISTANCE-Q DEICTIC)
                   (DISTANT (CHOOSE FAR))
                   (NONDISTANT (CHOOSE NEAR)))))
```

Figure 82 Demonstrative non-questioning: system and chooser (Cz, Bg, Ru)

In the current versions of the grammars, the Deictic is lexified in the gates called THIS and THESE for **near**, and THAT and THOSE for **far** (see Figure 85 for the implementations). The following examples taken from the ImD text 1 illustrate a realization of the **near** feature:

(43)

(a) Cz: Pro vytvoření dalšího elementu tyto kroky opakujte
For creating another element-gen these steps-acc repeat

(b) Bg: Повторете тези стъпки, за да дефинирате друг елемент
Repeat these steps in-order to define another element

(c) Ru: Повторите эти шаги, чтобы задать еще один элемент.
Repeat these steps in-order-to specify more one element

(d) En: Repeat these steps to define another element.

We demonstrate the grammatical structure including the Deictic realized by a demonstrative pronoun in Czech below, focusing on the following relevant portion of the sentence in (44):

(44)

Cz: Opakujte tyto kroky
Repeat these steps-acc
Repeat these steps.

The generated structure is depicted in Figure 83. It is obtained using the following SPL:

```
(EXAMPLE
    :NAME   deictic-near
    :SET-NAME   Pokus
    :TARGETFORM   "Opakujte tyto kroky."
    :LOGICALFORM
    (P / DIRECTED-ACTION :LEX opakovat
    :SPEECHACT IMPERATIVE
    :ACTEE (S / OBJECT :LEX krok
           :MULTIPLICITY-Q MULTIPLE
           :identifiability-q  identifiable
           :proximity-modification-q proximity
           :proximity-mod-id (speaker / person :distance-q nondistant)
))
```

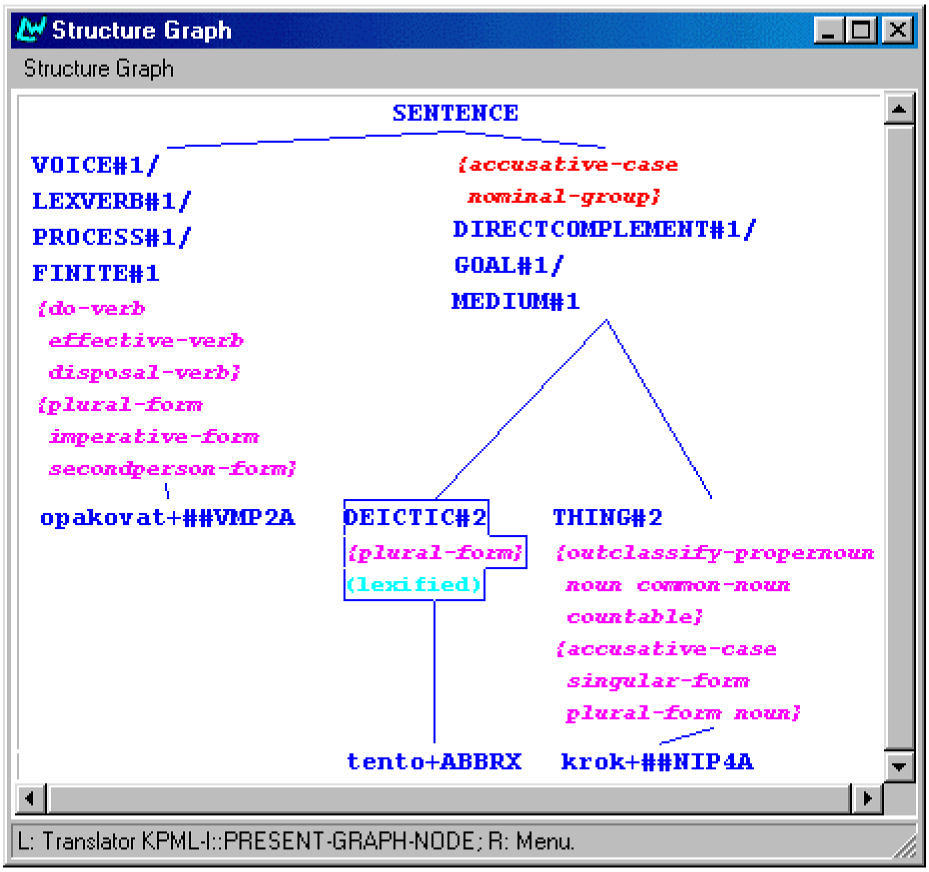
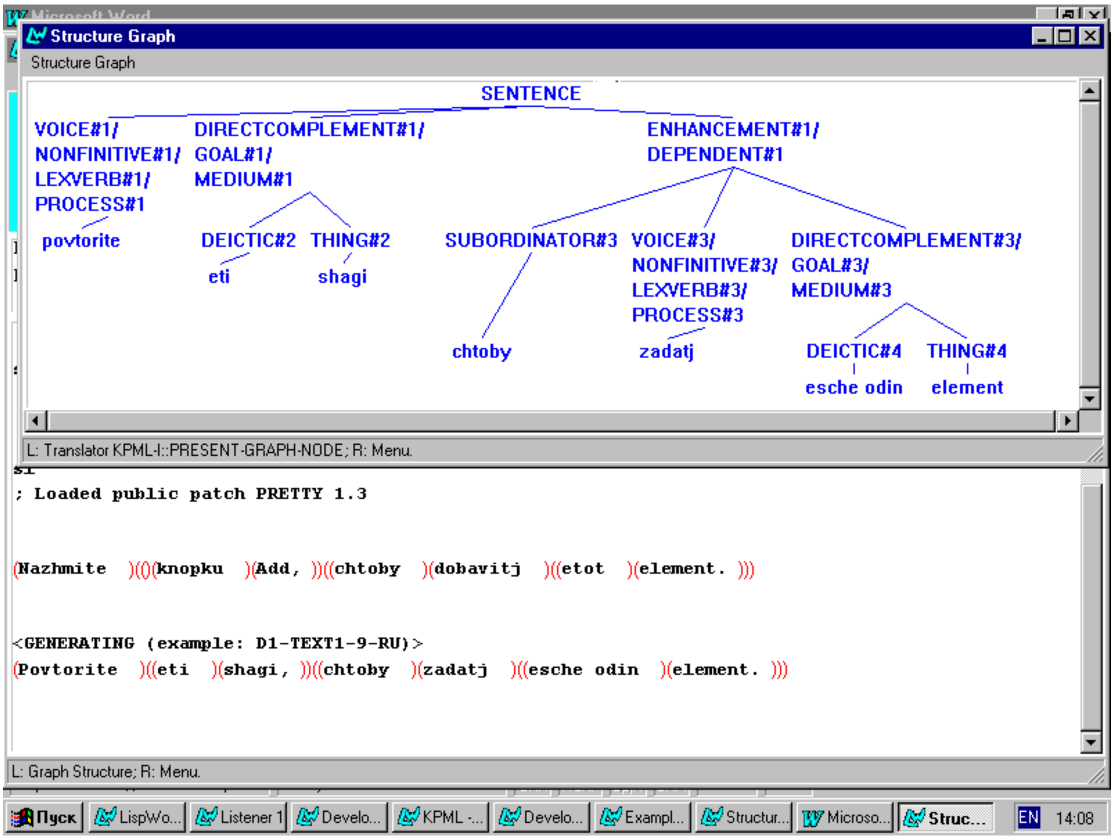


Figure 83: Generated structure for the Czech sentence “Opakujte tyto kroky” (example (44))

The structure generated for the Russian sentence in (43c) is shown in Figure 84.



L: Translator KPML-I::PRESENT-GRAPH-NODE; R: Menu.

```
s1
; Loaded public patch PRETTY 1.3

(Nazhmite )(()(knopku )(Add, ))((chtoby )(dobavitj )((etot )(element. )))

<GENERATING (example: D1-TEXT1-9-RU)>
(Povtorite )((eti )(shagi, ))((chtoby )(zadatj )((esche odin )(element. )))
```

L: Graph Structure; R: Menu.

Figure 84: Generated structure for the Russian sentence “Повторите эти шаги, чтобы задать еще один элемент” (example (43c))

When the Deictic element is realized by an adjective or a pronoun, it has to agree with the head noun in number and gender, and in Czech and Russian also in case.

```
(GATE
    :NAME   THIS
    :INPUTS       (AND NEAR NONPLURAL)
    :OUTPUTS      ((1.0 THIS
                  (Czech: LEXIFY DEICTIC TENTO)
                  (Russian: LEXIFY DEICTIC Etot)
                  (Bulgarian: LEXIFY DEICTIC tozi)
                  (INFLECTIFY DEICTIC SINGULAR-FORM)))
    :REGION   DETERMINATION )

(GATE
    :NAME   THESE
    :INPUTS    (AND NEAR PLURAL)
    :OUTPUTS   ((1.0 THESE
               (Czech: LEXIFY DEICTIC TENTO)
               (Russian: LEXIFY DEICTIC ETOT)
               (Bulgarian: LEXIFY DEICTIC tezi
               (INFLECTIFY DEICTIC PLURAL-FORM)))
    :REGION   DETERMINATION )

(GATE
    :NAME   THAT
    :INPUTS    (AND FAR NONPLURAL)
    :OUTPUTS   ((1.0 THAT
               (Czech: LEXIFY DEICTIC TAMTEN)
               (Russian: LEXIFY DEICTIC Tot)
               (Bulgarian: LEXIFY DEICTIC onzi)
               (INFLECTIFY DEICTIC SINGULAR-FORM)))
    :REGION   DETERMINATION )

(GATE
    :NAME   THOSE
    :INPUTS    (AND FAR PLURAL)
    :OUTPUTS   ((1.0 THOSE
               (Czech: LEXIFY DEICTIC TAMTEN)
               (Russian: LEXIFY DEICTIC Tot)
               (Bulgarian: LEXIFY DEICTIC onezi)
               (INFLECTIFY DEICTIC PLURAL-FORM)))
    :REGION   DETERMINATION )
```

Figure 85 This, these, that, those: gate (Cz, Ru, Bg)

2.8.3 Non-specific determination

The ImD texts also include instances of explicit realization of non-specific Deictic elements. In particular, Cz *jeden*, Bg *един*, and Ru *один* (En *one*) as used in the following examples extracted from the ImD text 2 are of this kind (for a discussion of this kind of determination, see the LSPEC2 deliverable (Andonova *et al.*, 1999):

(45)

(a) Cz: Spusťte příkaz PLINE jedním z následujících způsobů
Start command PLINE one-ins of following-gen methods-gen

(b) Bg: Стартирайте командата PLINE,
Start command-the PLINE
като използвате един от следните методи
while use one of following methods

(c) Ru: Запустите команду PLINE одним из следующих способов
Start command PLINE one of following methods

(d) En: Start the PLINE command by/using one of the following methods

We demonstrate the generation of this kind of non-specific determination using the following simplified example (for the generated structure see Figure 86):

(46)

Cz: Vyberte jeden způsob.
Select-imp one method-acc
Select one of the methods.

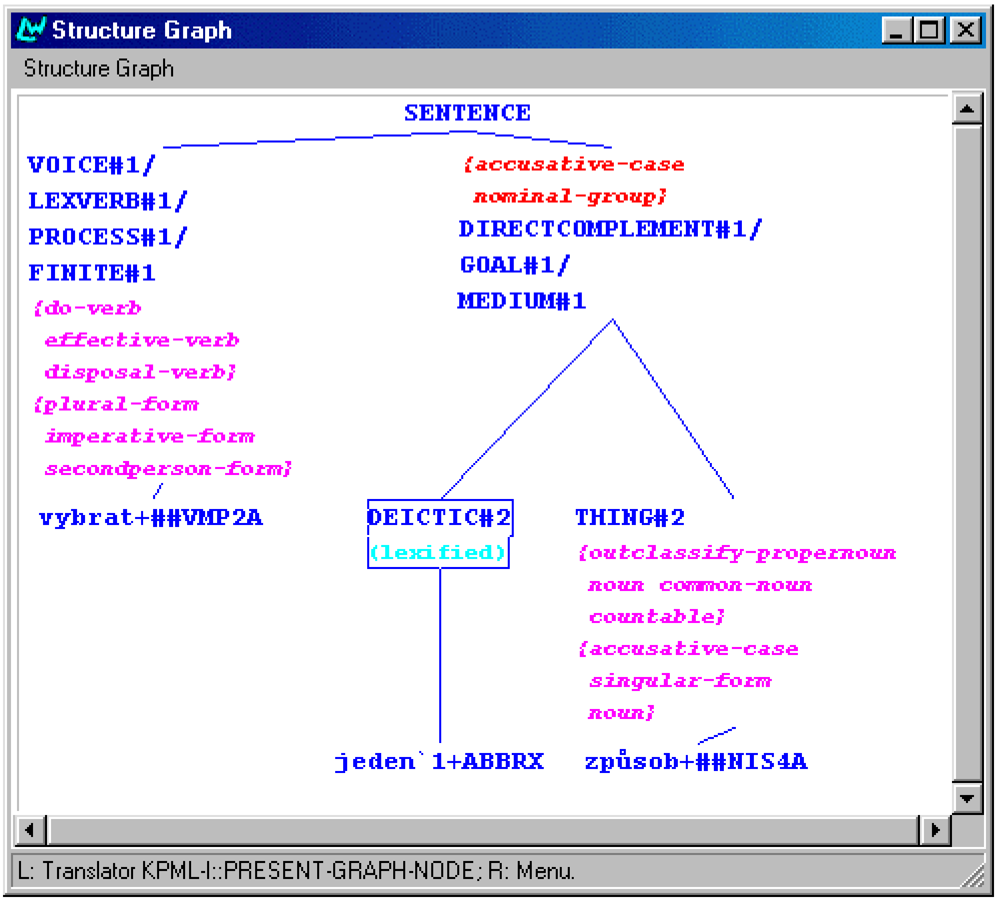


Figure 86: Generated structure for the Czech sentence “Vyberte jeden způsob” with a non-specific Deictic element (example (46))

This sentence can be generated using the following SPL:

```
(EXAMPLE
    :NAME   deictic-one
    :SET-NAME   Pokus
    :TARGETFORM   "Vyberte jeden zpusob"
    :LOGICALFORM (
      S / dispositive-material-action
           :LEX vybrat
           :SPEECHACT Imperative
           :ACTEE
                 (P / Object
                 :LEX zpu4sob
                 :determiner one-of )))
```

The last line in the SPL makes use of the following SPL macro:

```
(defspl-macro :determiner
      ((one-of
             :amount-attention-q nonminimalattention
             :current-representative-id ?s1
             :potential-representative-id ?s2
             :presuppose-existence-q presupposed
             :selection-particularity-q particular
             :set-totality-q (?s1 ?s2) partial)))
```

The feature that is responsible for the realization of the non-specific Deictic element is **particular-restricted** in the SINGULAR-RESTRICTED-TYPE system. The path to this system leads through the systems SELECTIVE-PARTIAL-TYPE, PARTIAL-TYPE and NONSPECIFIC-TYPE in region DETERMINATION, where the features **restricted**, **selective-partial** and **partial** are chosen, respectively. The input feature of the NONSPECIFIC-TYPE system is **nominal-nonspecific**, which is generated in the NOUNTYPE region. The implementations of the relevant systems and choosers in the DETERMINATION region are shown in Figure 87.

In the current implemementation of the Czech grammar, there is a problem with the simplification concerning the generation of an explicit Deictic element only for identifiable elements, as described in Section 2.8.1. In order to handle the generation of the explicit non-specific Deictic elements, we have to improve the implementation of the DEICTIC-EXPLICITNESS-CHOOSER beyond that shown in Figure 76 in the future.

```
(SYSTEM
    :NAME   SINGULAR-RESTRICTED-TYPE
    :INPUTS    (AND RESTRICTED SINGULAR)
    :OUTPUTS   ((0.5 PARTICULAR-RESTRICTED
               (LEXIFY DEICTIC JEDEN))
               (0.5 NONPARTICULAR-RESTRICTED))
    :CHOOSER   SINGULAR-RESTRICTED-CHOOSER
    :REGION   DETERMINATION )
(SYSTEM
    :NAME   SELECTIVE-PARTIAL-TYPE
    :INPUTS   SELECTIVE-PARTIAL
    :OUTPUTS   ((0.5 RESTRICTED) (0.5 UNRESTRICTED))
    :CHOOSER   SELECTIVE-PARTIAL-TYPE-CHOOSER
    :REGION   DETERMINATION
    :METAFUNCTION   IDEATIONAL )
(SYSTEM
    :NAME   PARTIAL-TYPE
    :INPUTS   PARTIAL
    :OUTPUTS   ((0.5 SELECTIVE-PARTIAL)
               (0.5 NONSELECTIVE-PARTIAL))
    :CHOOSER   PARTIAL-TYPE-CHOOSER
    :REGION   DETERMINATION
    :METAFUNCTION   IDEATIONAL)
(SYSTEM
    :NAME   NONSPECIFIC-TYPE
    :INPUTS
               (OR NOMINAL-NONSPECIFIC NONDEICTIC-VERBAL-AGENT)
    :OUTPUTS   ((0.5 TOTAL)  (0.5 PARTIAL))
    :CHOOSER   NONSPECIFIC-TYPE-ALTERNATE-CHOOSER
    :REGION   DETERMINATION
    :METAFUNCTION   IDEATIONAL)
(CHOOSER
    :NAME   SINGULAR-RESTRICTED-CHOOSER
    :DEFINITION   ((ASK (SELECTION-PARTICULARITY-Q
                    CURRENTREPRESENTATIVESET POTENTIALREPRESENTATIVESET)
                  (PARTICULAR (CHOOSE PARTICULAR-RESTRICTED))
                  (NONPARTICULAR (CHOOSE NONPARTICULAR-RESTRICTED)))))
(CHOOSER
    :NAME   SELECTIVE-PARTIAL-TYPE-CHOOSER
    :DEFINITION   ((ASK (PRESUPPOSE-EXISTENCE-Q THING)
                  (NOTPRESUPPOSED (CHOOSE UNRESTRICTED))
                  (PRESUPPOSED (CHOOSE RESTRICTED)))))
(CHOOSER
    :NAME   PARTIAL-TYPE-CHOOSER
    :DEFINITION   ((ASK (AMOUNT-ATTENTION-Q THING)
                  (NONMINIMALATTENTION (CHOOSE SELECTIVE-PARTIAL))
                  (MINIMALATTENTION (CHOOSE NONSELECTIVE-PARTIAL)))))
(CHOOSER
    :NAME   NONSPECIFIC-TYPE-ALTERNATE-CHOOSER
    :DEFINITION ((ASK (SET-TOTALITY-Q
     CURRENTREPRESENTATIVESET POTENTIALREPRESENTATIVESET)
                (TOTAL (CHOOSE TOTAL))(PARTIAL (CHOOSE PARTIAL)))))
```

Figure 87: Non-specific determination (one of): systems and choosers (Cz, Bg, Ru)

We have discussed the generation of the ‘one-of’ construction based on the assumption that its semantics can be represented as involving a non-specific Deictic element. Alternatively, one can generate these sentences as including a Numerative element rather than a Deictic one. We demonstrate this possibility of representation using the following Czech sentence:

(47)

Cz: Vyberte jeden z těchto způsobů.
Select-imp one-acc of these methods-gen
Select one of the methods.

The SPL is as follows:

```
(EXAMPLE
    :NAME   numerative-one
    :SET-NAME   Pokus
    :TARGETFORM   "Vyberte jeden z techto zpusobu"
    :LOGICALFORM (
      S / dispositive-material-action
            :LEX vybrat
            :SPEECHACT Imperative
            :ACTEE
                  (P / Object
                  :LEX zpu4sob
                  :quantity-selection-q quantity
                  :quantity-selection-id 1
                  :number plural
                  :determiner this
                  )))
```

The last line in the SPL makes use of the following SPL macro:

```
(defspl-macro :determiner
      (this
             :identifiability-q  identifiable
             :proximity-modification-q proximity
             :proximity-mod-id (speaker / person
                                     :distance-q nondistant)))
```

The generated structure is shown in Figure 88.

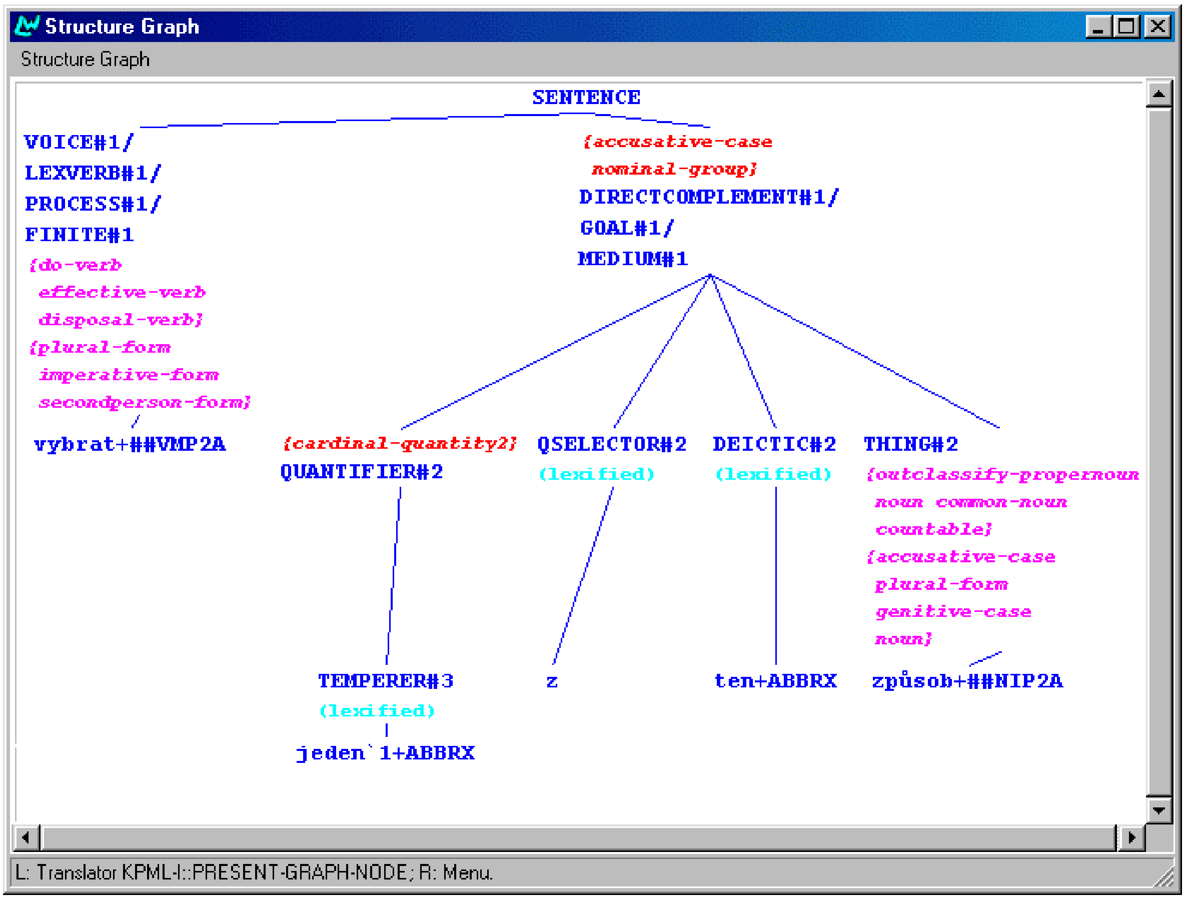


Figure 88: Generated structure for the Czech sentence "Vyberte jeden z těchto způsobů" with a Numerative element (example (47))

2.8.4 Post-deictic Element

Besides the Deictic element, we also encounter the equivalents to the English Post-deictic elements in all three languages (see "following" in (45a-d)). These are realized by adjectives which have to agree with the head noun. These should be implemented in the system POST-DEICTICITY in the region POST-DEICTICITY.

However, we have not yet implemented the generation of Post-deictic elements for Slavic languages. Instead, we generate the counterparts of "following" as Qualities. In order to demonstrate the generation, we use the following Czech sentence:

(48) *Cz: Vyberte jeden z těch následujících způsobů.*
Choose one-acc of these-gen- following-gen methods-gen
Choose the following method.

This sentence is generated using the following SPL:

```
(EXAMPLE
    :NAME   following
    :SET-NAME   Pokus
    :TARGETFORM " Vyberte jeden z těch následujících způsobů"
    :LOGICALFORM (
      S / dispositive-material-action
:LEX vybrat
:SPEECHACT Imperative
:ACTEE
      (P / Object
      :LEX zpu4sob
      :identifiability-q notidentifiable
            :quantity-selection-q quantity
            :quantity-selection-id 1
            :number plural
      :Property-aScription
      (F / quality :LEX na2sleduji2ci2))))
```

The generated structure is shown in Figure 89.

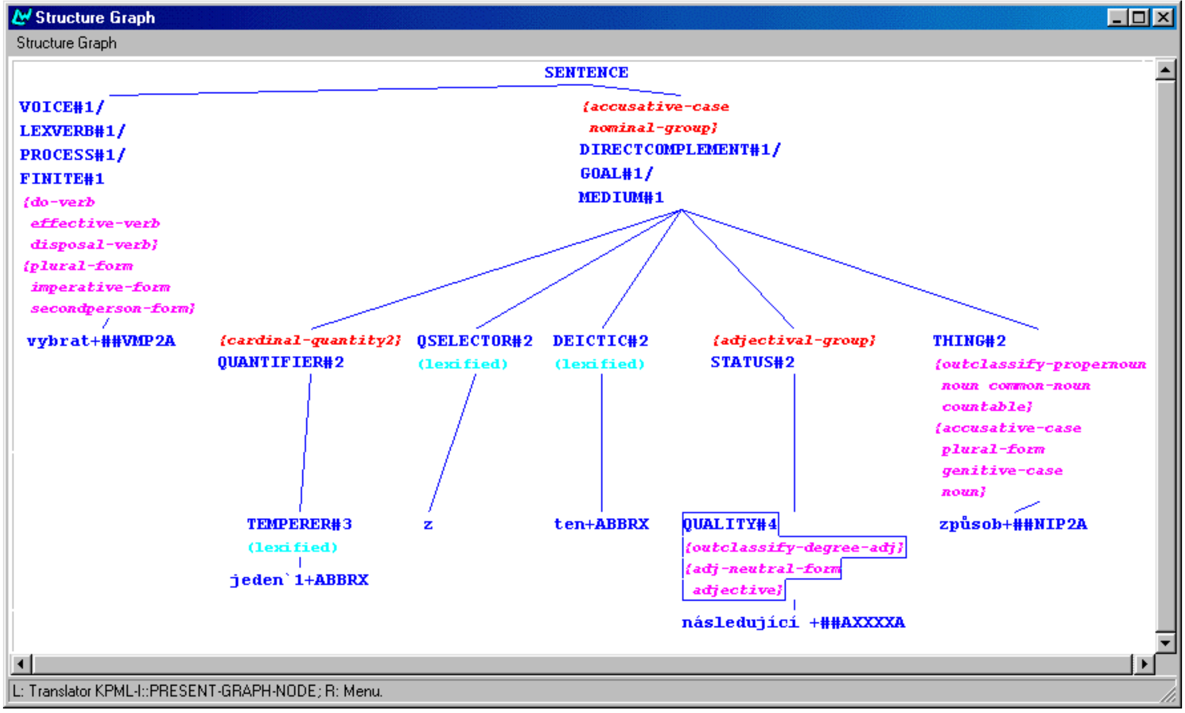


Figure 89: Generated structure for the Czech sentence “Vyberte jeden z těchto následujících způsobů” with a Numerative element (example (48))

2.9 Word Order

In this section, we describe how we handle word order in the current versions of the AGILE text generation system. It is known that Slavic languages exhibit a relatively high degree of word order freedom. Compared to languages like English or French, where clause constituents cannot be “moved around"” with the same relative freedom without simultaneous changes in syntactic structure. However, different word order variants of a sentence, even though they are grammatically well-formed, do not necessarily have the same meaning and are generally not interchangeable in a given context. This means that in the process of automatic generation of continuous texts from an underlying representation of the content, we have to ensure that a semantically and contextually appropriate word order is chosen.

Various factors can be discerned in the language system in general that play an important role in expressing a given content in a linear form. For instance, English is an example of a language where word order is strongly constrained by grammatical structure. In such a language with a rather fixed word order, differences in information structure are often reflected by varying the intonation pattern of a sentence or by the choice of definite vs. indefinite article with a nominal group. In Slavic languages, these effects are often achieved by varying word order in accordance to **information structure.**[10]

The discussion in this section follows up on the discussion of word order in the SPEC2 deliverable (Andonova *et al.*, 1999), where we presented a number of examples demonstrating the relevance of word ordering phenomena, and pointed out some similarities and differences among Czech, Russian and Bulgarian.

In order to account for the word ordering phenomena within the AGILE project, we decided to build upon the insights of existing linguistic theories. Currently we restrict ourselves to combining the following two approaches, where the first provides the basis and the second is used for a more elaborate treatment of word order phenomena in the Slavic languages:

- **Halliday's thematic structure** (Halliday, 1985), as developed in the Systemic Functional Grammar (SFG) framework, is chosen as the **basis** because SFG is the framework adopted in the Penman system (Bateman *et al.*, 1990) on which our AGILE grammars, developed in the KPML environment (Bateman , 1997), are based;
- the **topic-focus articulation** approach developed within the framework of Functional Generative Description (FGD, Sgall *et al.*, 1986) serves the **elaboration** of the SFG approach towards a more flexible treatment required for languages with a higher degree of free word order than English, especially because Halliday's approach is not sufficiently specific with respect to the ordering of non-thematic constituents.

We discussed these two approaches in detail in SPEC2. On the basis of that discussion, we also proposed a way in which to combine the insights of SFG and FGD concerning word order, and sketched the corresponding ordering algorithm in abstract terms. In the present report, we describe the algorithm in more detail, especially its input, its application and its implementation.

The motivation underlying the approach that we propose is to develop a rather general treatment of word ordering phenomena in Slavic languages. We would like to account for the fact that word ordering in Slavic languages is governed to a much larger extent by information structure than by constraints derived from grammatical structure. Our effort is thus aimed at a flexible linguistically appropriate treatment of word order in Slavic languages.

Considering the texts generated in the ImD of the AGILE project, it may perhaps seem that the approach we advocate is unnecessarily complex. It would be possible to say that the word order in the sentences that we are currently generating is the same as in English in most cases. For the cases where there is a difference we could include a few specific rules which would yield the desired results. Such an approach would be easy and straightforward. However, we do not want to adopt this purely engineering strategy. We prefer to take a step back and re-consider the way word order is handled in the tactical generators with the KPML grammars, even if it means that the development is more difficult and slower. What we try to obtain is a linguistically well-grounded approach that can be extended beyond the scope of the sentences in the ImD texts.

[10] We use the term *information structure* as a general term embracing notions, such as Theme-Rheme, Given-New, information distribution and the like.

In addition, if we do develop the treatment of word order that uses information structure as the main guidance, we make it possible to put the various theories of information structure and word order to a practical test. This will obviously provide feedback for the improvement of the linguistic theories. In this way the work carried out within the AGILE project can have the extra value of contributing to a better understanding of word ordering principles involved in natural languages in general.

The remainder of the present chapter is structured as follows. We first briefly recapitulate the essential terminology that is used in our approach (Section 2.9.1). Then we turn to the details of the approach adopted in AGILE (Section 2.9.2). To provide the reader with and overall picture, we present the abstract word ordering algorithm (Section 2.9.3) and point out its placement within the entire process of text generation (Section 2.9.4). In Section 2.9.5 we discuss the placement of the reflexive particle in Bulgarian and in Czech. Finally, we summarize the issues we tackle in the implementation phase (Section 2.9.6)

2.9.1 Essential Terminology

Some of the considerations related to word order have been dealt with in the SFG framework, which we take as the starting point for developing the linguistic specifications in the AGILE project. We present the main word order-related SFG notions first.

2.9.1.1 Thematic and Information Structure in SFG

According to (Halliday, 1985), a clause as a message consists of a **Theme** combined with a **Rheme**, and in this configuration, the Theme is the ground from which the clause is taking off. As noted earlier, Halliday distinguishes between the **thematic structure** of a clause and the **information structure**. The latter is the distinction between **Given** and **New** within an information unit: the speaker presents information to the listener as recoverable (Given) or not recoverable (New). The thematic structure and information structure are closely related but not the same. Whereas the Theme contains the experiential item the speaker chooses to take as the point of departure, the Given is what the speaker believes the listener already knows or has accessible.[11]

The notion of Theme tells us a number of things about “the first” position in the clause, but it does not tell us much about the word order of “the rest” of the clause. Presumably, Halliday leaves this to be decided by the grammatical structure, which is an approach sufficient for English. However, in languages with a high degree of free word order the grammar is not very strict about the placement of the elements of structure after the Theme. The examples we discussed in SPEC2 showed that ordering in our languages is to a great extent determined by what is presumed to be salient in the context. This means that ordering depends on information structure. These issues have been studied in detail in the Praguian FGD framework (Sgall *et al.*, 1986). We incorporate the most essential ideas into the AGILE account of word order in Slavic languages.

[11] Note that Theme in the Hallidayan description can embrace more than just the experiential element. The experiential element is referred as “topical”, and it is in fact the last element in “multiple themes”. For instance, in *Well, then we don't join in*, the Theme consists of *well, then we*, where *we* is the topical (experiential) element.

2.9.1.2 Topic-Focus Articulation in FGD

FGD works with a notion of information structure as a dichotomy called **topic-focus articulation** (TFA). TFA is defined on the basis of a distinction between contextually bound (CB) and non-bound (NB) items in a sentence (cf. (Sgall *et al.*, 1986), or (Kruijff-Korbayová, 1998) for an overview). The motivation behind this distinction corresponds to that underlying the Given/New dichotomy in SFG. A CB item is assumed to convey some content that is accessible from the discourse context. Such an item may refer to an entity already explicitly referred to in the discourse, or an 'implicitly evoked' entity (cf. (Hajičová, 1993) for a summarizing discussion).

The ordering of NB items in a sentence follows the so-called **systemic ordering** (SO). SO is a language specific ordering of complementations, i.e. 'arguments' and 'adjuncts', of verbs, nouns, adjectives or adverbs which corresponds to neutral word order. It may differ from one language to another, but is considered constant within a given language. SO in Czech has been studied in detail (Sgall *et al.*, 1986). The SOs of Russian and Bulgarian have not yet been studied in general. We expect the SOs for the main types of complementations in Russian and Bulgarian to be similar to the Czech one, though there can be slight differences (Andonova *et al.*, 1999).

The FGD claims concerning word order can be summarized as follows:

- The main principle of word order in Czech is that the Topic precedes the Focus. Since the Topic may be empty (esp. in discourse-initial sentences) or may be deleted on the surface due to ellipsis, it is possible that the surface form of some sentences only consists of the realizations of elements belonging to the Focus.
- In the primary cases when the Topic consists of the CB elements, and the Focus of the NB ones, one can say that the CB elements precede the NB elements.
  A more general formulation of this principle uses the degrees of the so-called **communicative dynamism** (CD, Sgall *et al.*, 1986): in primary cases, CD and the surface word order correspond to each other quite closely, at least within clauses. So, the ordering from left to right in the surface realization corresponds to the increasing degrees of CD.

There are the following exceptions to this principle in Czech:

- Clitics: they have to be placed in the so-called Wackernagel's position, characterized roughly as the position between the first and the second element in a clause.[12]
- The main verb: its preferred default (unmarked) placement is after, but not necessarily immediately following, the surface Subject, if there is one.

In the next section we show how the SFG and FGD ideas concerning information structure can be used in an integrated approach to generate contextually appropriate ordering of clause elements.

2.9.2 The Approach in AGILE

As the starting point for specifying the principles of word ordering in the context of AGILE, we combine the FGD-based strategy which reflects information structure with the possibility of thematization in the SFG spirit. For Czech and Russian, we need to allow for more

[12] Naturally, this leaves to be defined what 'first element' means. It is easy to show that 'first element' does not equate to 'first constituent', since the element can be of arbitrary complexity. We use the notion of Theme for this purpose (see below).

freedom in word order (i.e., a looser relation between ordering and grammatical structure) than in Bulgarian (see the LSPEC2 deliverable, Andonova *et al.* ,1999).

We propose to preserve the SFG notion of Theme (as well as the possibility of multiple theme within which the experiential element is called topical). This conception appears useful in any language in order to account for text structuring concerns across sentences within connected spans of texts. For instance, the decision what to chose as a 'point of departure' can be motivated by a particular style chosen for the text, in which case it is not necessary to look for a motivation for a particular ordering based on information structure.[13] The following Czech examples adapted from the AGILE ImD text 1 illustrate ordering motivated by text organization. The order in (49a) and (50a) can be attributed to a marked Theme, whereas in (49b) and (50b) the unmarked Theme corresponds to the Process (since Subject is not realized in the imperative):

(49) *Thematic spatial-location (Czech)*

(a) Z dialogového panelu *Color Selection* zvolte barvu elementu
From dialogue box-gen *Color Selection* choose-imp color-acc element-gen
"From the *Color Selection* dialogue box, choose the element's color."

(b) Zvolte barvu elementu z dialogového panelu *Color Selection*.
Choose-imp color-acc element-gen from dialogue box-gen *Color Selection*
"Choose the element's color from the *Color Selection* dialogue box."

(50) *Thematic purpose circumstance (Czech)*

(a) Pro přidání elementů ke stylu vyberte *Element Properties*.
For add-nom elements to style-dat select-imp *Element Properties*
"In order to add elements to the style select *Element Properties*."

(b) Vyberte *Element Properties* pro přidání elementů ke stylu.
Select-imp *Element Properties* for add-nom elements to style-dat
"Select *Element Properties* in order to add elements to the style."

If the information structure in terms of contextual boundness or non-boundness of the structure elements is considered the same in both the (a) and (b) variants, the difference in word order can be attributed to a different text organization strategy. The order in (49a) can be achieved by thematization of the spatial-location of the action performed by the user, and the order in (50a) can be achieved by thematization of the purpose of an action performed by the user. The decisions about text organization and the consequent thematization are to be made in the text planning phase, i.e. by the text structuring module in the AGILE system (see the TEXM2 deliverable, Kruijff *et al.*, 1999).

For the ordering of non-thematic constituents within a clause, which is not determined by the syntactic structure, we use notions adopted from FGD, namely the distinction between contextual boundness (CN) and non-boundness (NB) in combination with the so-called systemic ordering (SO).

Contextual boundness is treated as a local feature, i.e., a complex CB constituent can contain 'locally' contextually NB constituents, and vice versa. For instance, a complex sentence can contain one CB and one NB clause, and within each of them, elements are discerned as locally CB vs. NB. Also within a complex nominal group which is, e.g., NB as

[13] The relation between text style and the choice of Theme is discussed also in (Teich *et al.*, 1996).

a whole, some parts can be CB. As an example of this, consider the following sequence of sentences in Czech adapted from the ImD text 1:

(51)

Přidejte element. Nejdříve zvolte barvu tohoto elementu | jeho barvu
Add-imp element-acc. First select color-acc this element-gen | its color-acc.
"Add an element. First select the element's | its color."

The nominal groups *barva tohoto elementu* (*color of this element*) or *jeho barva* (*its color*) consist of a Thing (*barva*) related by "generalized ownership" relation to another Thing (*element* or *it*. The entire nominal groups as such are NB, however, the modifiers corresponding to the owner, i.e. *tento element* (*this element*) or *jeho* (*its*), are (locally) CB.

2.9.3 Word Ordering Algorithm

On the basis of the above mentioned insights, we formulate an abstract ordering algorithm, as shown in Figure 90. The Theme is determined by text organization. In the AGILE system, this means it is determined by the Text Structuring Module system (see the TEXM2 deliverable, Kruijff *et al.*, 1999) and the element that is to be thematized is explicitly marked as such in the SPL. If no element is explicitly chosen as Theme in the SPL or by the grammar, the thematic position is filled by the first CB element. For the ordering of the non-thematic constituents within a clause, which is not determined by the syntactic structure, we use 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.

```
Given a list $\Gamma$ of ordering constraints imposed by the grammar,
Given a list L1 of constituents that need to be ordered,
Given a list $\Delta$ giving communicative dynamism of cb constituents,

      create empty lists LC and LN

      % LC is going to contain cb constituents ($\gamma$),
      % LN nb constituents ($\eta$)

      repeat for each element E in L1
                  if E is CB,
                        then add E into LC,
                        else add E into LN.
      if the verb is CB,
            then  Order the verb at the end of LC
                  Order the remainder according to $\Delta$
            else  Order all elements in LC according to $\Delta$
      % Thus, if $e_i < e_j$ in $\Delta$, then $e_i < e_j$ in LC except for the verb.
      if $\Gamma \neq \varnothing$ then
            Order elements in L1 using ordering constraints in $\Gamma$
      % These constraints may, for example, involve ordering
      %  "heavy" constituents at the end of the clause.
```

Figure 90: Abstract algorithm for word order

The proposed ordering algorithm as such is the same for all the three languages under consideration. What may differ is first of all the systemic ordering, and therefore the default ordering of non-bound elements. 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.

In order to show the role of this algorithm within the entire process of text generation in AGILE, we recapitulate our text planning strategy and the interface between the text planner and the sentence generator in Section 2.9.5.

One particular problem concerns the placement of clitics in Slavic languages. In the sentences we have been analyzing and generating in AGILE so far, we have been considering the placement of the reflexive particle, i.e. *se* in Czech and *ce* in Bulgarian.[14] The reflexive particle is used either with reflexive tantum verbs, or in the reflexive passive construction used for the non-personal style of conveying instructions in Czech and Bulgarian. In both languages, clitics are supposed to occur in the so-called Wackernagel's position (Avgustinova and Oliva, 1995). But there are some differences between these two languages in this respect, which deserve further study. We describe our current implementations concerning the placement of the reflexive particle in Bulgarian and in Czech in the next section.

2.9.4 Placement of Particles

2.9.4.1 Placement of the reflexive particle in Bulgarian

In Bulgarian, clitics should either directly follow or directly precede the Finite element in a clause. Since we do not consider any clitics other than the reflexive particle *ce* at this point, we do not have to deal with their mutual ordering and can simply say that *ce* either directly follows or directly precedes the Finite element.

The ordering of *ce* after the Finite occurs in the cases where Finite appears clause-initial, i.e. it is conflated with Theme. The ordering of *ce* before Finite occurs in the cases where Finite is not conflated with Theme, that is, some other element occupies the thematic position. The following examples extracted from the ImD text 1 (non-personal style) illustrate the placement of *ce* in Bulgarian in a few cases of reflexive passive constructions. Namely, in a simple indicative clause with thematized verb (52), in a clause with a thematized spatial-locative element expressed by a complex prepositional phrase (53), in a main clause in a clause complex beginning with a temporal Conjunct (54), and in a dependent clause, also beginning with a Conjunct (55):

(52)

Избира се Color.
Selects refl Color.
"One selects Color."

(53)

От функционалния ред Properties или менюто Data се избира Multiline Styles.
From tool bar Properties or menu Data refl choses Multiline

[14] In Russian, the reflexive particle is attached as a suffix to the verb, so no specific ordering issues arise.

Styles.
"From the Properties toolbar or Data menu one choses Multiline Styles."

(54)

Първо се отваря диалоговият прозорец Multiline Styles, …
First refl opens dialogue window Multiline Styles, …
"First one opens the dialogue box Multiline Styles, …"

(55)

като се използва един от следните методи.
by refl uses one of following methods.
" by using one of the following methods."

When we disregard the placement of other clitics, the placement of the reflexive particle *ce* in Bulgarian after or before the Finite element can be captured as follows (the corresponding implementations are shown in Figure 91):

```
MEDIO-PASSIVE-ORDER:
      (MEDIO-PASSIVE) →
            [finite-reflexiveparticiple]  Finite^Reflexiveparticiple
            [reflexiveparticiple-finite]  Reflexiveparticiple^Finite
```

```
(SYSTEM
    :NAME   MEDIO-PASSIVE-ORDER
    :INPUTS    (MEDIO-PASSIVE)
    :OUTPUTS   ((0.5 FINITE-REFLEXIVEPARTICLE
               (ORDER FINITE REFLEXIVEPARTICLE))
               (0.5 REFLEXIVEPARTICLE-FINITE
               (ORDER REFLEXIVEPARTICLE FINITE)))
    :CHOOSER   MEDIO-PASSIVE-ORDER-CHOOSER
    :REGION   VOICE
)
(CHOOSER
    :NAME   MEDIO-PASSIVE-ORDER-CHOOSER
    :DEFINITION   ((DEFAULTCHOOSE FINITE-REFLEXIVEPARTICLE))
)
```

Figure 91: Medio-passive order: system and chooser for Bulgarian

2.9.4.2 Placement of the reflexive particle in Czech

In Czech, the situation is more complicated than in Bulgarian, because the position of clitics is not fixed directly next to the Finite element. They are placed at the Wachernagel's position. In order to describe the placement of clitics in our grammars, we use the notion of Theme. We propose to place clitics after the Theme. If there is more than one clitic, their mutual order is determined by the grammar. In our current discussion, we concentrate on the placement of the reflexive particle *se*.

We illustrate the various possible placements of *se* in Czech by examples. (56) through (58) correspond to the Bulgarian examples Bulgarian in (52) through (55) above, and show *se* in a simple indicative clause with thematized verb (56), in a clause with a thematized spatial-locative element expressed by a complex prepositional phrase (57) and a clause beginning by a temporal conjunct (58):

(56)

Vybere se Color.
Selects refl Color.
“One selects Color.”

(57)

Z nástrojového panelu Properties nebo menu Data se vybere Multiline Styles.
From tool bar Properties or menu Data refl choses Multiline Styles.
“From the Properties toolbar or Data menu one choses Multiline Styles.”

(58)

Nejdříve se otevře okno Multiline Styles jednou z následujících metod.
First refl opens window Multiline Styles by-one-instr of following
methods.
“First one opens the dialogue box Multiline Styles using one of the following
methods.”

Incidentally, the reflexive particle is placed next to the Finite element in these Czech sentences, i.e., exactly the same as in Bulgarian. The reflexive particle is always going to be placed directly after the Finite element whenever the Finite element is included in the Theme. However, when the Finite element is not included in the Theme, the reflexive particle does not need to appear immediately preceding it. The following modifications of the above examples demonstrate such cases when other elements appear between the reflexive particle and the Finite element in Czech:

(59)

Potom se v dialogovém panelu Properties vybere Color.
Then refl in dialogue box Properties selects Color.
“Then one selects Color in the Properties dialogue box.”

(60)

Z menu Data se potom vybere Multiline Styles
From menu Data refl then choses Multiline Styles
“Then one choses Multiline Styles from the Data menu.”

(61)

Potom se soubor jednou z následujících metod uloží.
Then refl file by-one-instr of following methods saves
“Then one saves the file using one of the following methods.”

In (59), the Theme consists of a temporal Conjunct. The reflexive particle follows the Theme. It is itself followed by a spatial-locative Circumstance, after which comes the Finite element and finally the Subject. In (60), the Theme consists of a spatial-locative Circumstance. It is followed by the reflexive particle, which is itself followed by a temporal Circumstance, after which there is the Finite element followed by the Subject. In (61), the Theme again consists of a temporal Conjunct, which is followed by the reflexive particle. After it come the Subject then an Instrument Circumstance. The Finite element is placed at the end of the sentence.

Our proposal to place clitics, and in particular the reflexive particle *se*, after the Theme, is captured in the following formal specification (the corresponding implementations are shown in Figure 92):

MEDIO-PASSIVE-ORDER:
(AND THEME-ORDER (OR MEDIO-PASSIVE REFLEXIVE-TANTUM)) →
[theme-reflexiveparticiple]   theme^reflexiveparticiple

The input feature **theme-order** ensures that this system only applies when there actually is a Theme, the other input features capture the cases of a reflexive passive construction or a reflexive-tantum verb.

```
(SYSTEM
    :NAME   MEDIO-PASSIVE-ORDER
    :INPUTS    (AND THEME-ORDER
               (OR MEDIO-PASSIVE REFLEXIVE-TANTUM))
    :OUTPUTS   ((1.0 THEME-REFLEXIVEPARTICLE
               (ORDER THEME REFLEXIVEPARTICLE)))
    :REGION   VOICE
)
```

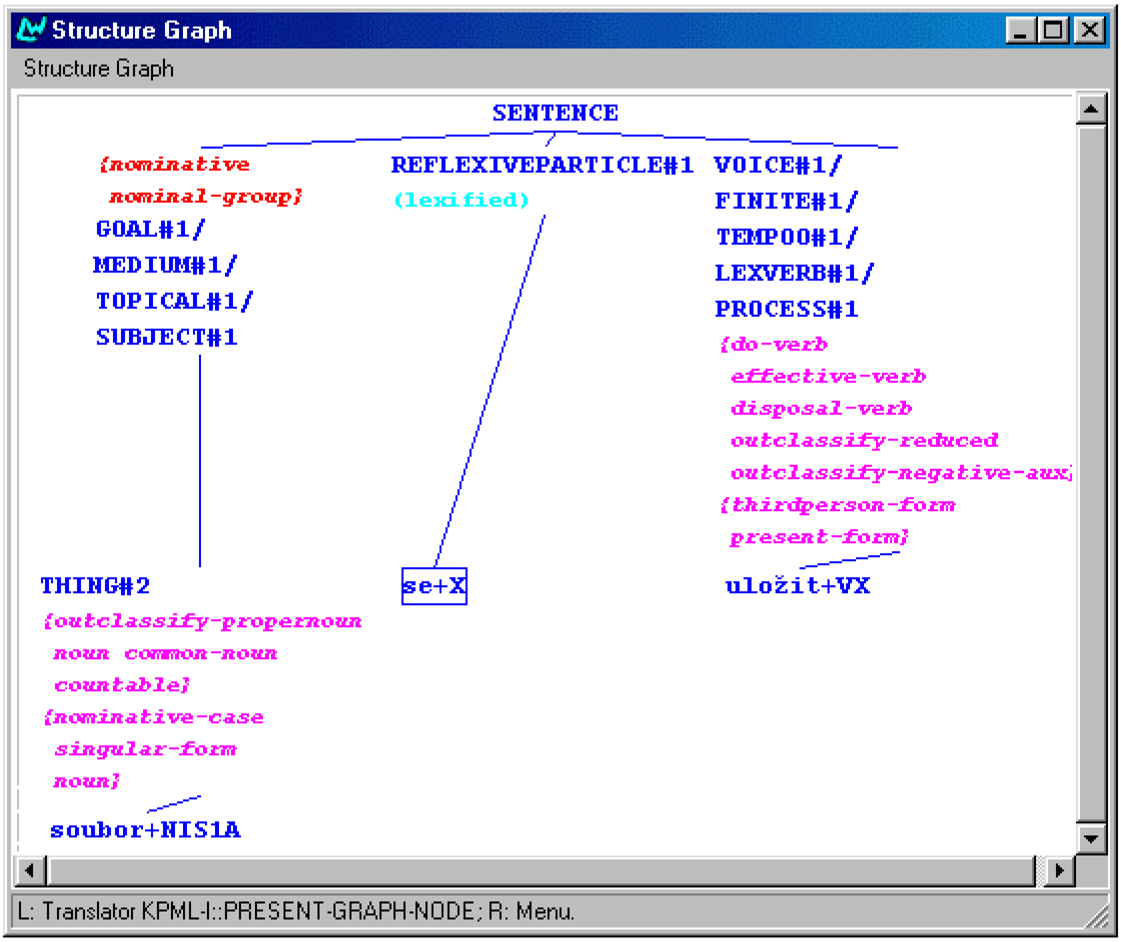
Figure 92: Medio-passive order: gate for Czech

Let us close this discussion with an example of a generated sentence in Czech. The SPL from which we generate is shown in Figure 93, and the generated grammatical structure is shown in Figure 94.

```
   (EXAMPLE
       :NAME ex1
       :SET-NAME   test-refl-Cz
       :TARGETFORM   "Soubor se uloží"
       :LOGICALFORM
             (Act / DIRECTED-ACTION
                   :LEX uloz3it
                   :PREFER-MENTION-AGENT-Q WITHHOLD
                   :ACTEE
                         (Obj1 / OBJECT :lex soubor
                         :identifiability-q notidentifiable )
      ))
```

Figure 93: SPL for the Czech sentence "Soubor se uloží"

Figure 94: Grammatical structure generated for the sentence "Soubor se uloží"



2.9.5 The Role of Word Ordering Within Text Generation

The text generation scenario in the AGILE system is as follows: using an authoring interface, a user of the system specifies as input an A-box, 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 convey the given content, in a particular 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 detail in the TEXS2 (Kruijff-Korbayová *et al.*, 1999) and TEXM2 (Kruijff *et al.*, 1999) deliverables. The essential ideas can be summarized as follows. The major component is formed by the systemic networks for text structuring. Following (Bateman, 1997), we construct 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 the lexico-grammar builds up grammatical structures. Using KPML to implement the means for text structuring facilitates the interaction between global level text generation (strategic generation) and lexico-grammatical expression (tactical generation).[15]

The organization of the region reflects the viewpoint that text templates and text structure elements are essentially orthogonal ideas. Therefore it consists of two parts. One part of the region 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 realization 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 made by traversing the systemic networks, we divide the A-box into smaller A-boxes that can be associated with the text structure elements of which the text structure is composed. In KPML, this can easily be done using so-called ID-inquiries. Finally, SPLs are generated, using these “smallest-size” A-boxes and the realization constraints imposed by text templates.

The SPLs then serve as input to the language-specific tactical generators that generate sentences. It is here, in the specification of the semantics of individual sentences, that the issue of generating contextually appropriate word order arises.

The idea that we implement is that the SPLs contain information about contextual boundness vs. non-boundness of each clause element. When the corresponding syntactic constituents are generated, the ordering algorithm sketched above applies. So for instance, once all the constituents of a clause have been created, their ordering is determined. As we stated above, grammatical constraints on the ordering are taken into account first. The ordering according to information structure, i.e. contextual boundness vs. non-boundness, is applied next. Finally, if the ordering of some elements is still undecided, defaults are applied. Such ordering process takes place at every level of grammatical structure. 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.

[15] 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, in which text planning resources are represented at one stratum and sentence planning resources at another.

The current approach thus handles ordering after the determination of syntactic structure and after the completion of generation of syntactic constituents at a given level, e.g. the clause. Word order is currently the only reflection of the information structure of the sentences we generate.

2.9.6 Summary

As stated earlier, there are essentially three sources of ordering constraints that we need to work with:

- ordering imposed by the grammatical structure; examples of strict constraints are that a preposition precedes the accompanying nominal group, the placement of clause connectives, the positioning of clitics with respect to other clause constituents but also their mutual ordering, etc.
- ordering derived on the basis of information structure; using '<' for linear precedence, it can be schematized as follows: *Theme* < *Clitics* < *Rest-CB* < *Verb* < *Rest-NB*; this ordering sometimes needs to be reconciled with general ordering preferences, for instance the preference for "verb secondness"
- ordering defaults which apply on elements unordered on the basis of the above principles; systemic ordering can be incorporated through defaults

Grammatical constraints are handled in the grammar by using ordering statements in the relevant systems. What we do in comparison to English in order to enable information structure to take its share in word order is to use less of such grammar-based ordering constraints. For example, in Czech and Russian, we relax the placement of Subject before Finite, or the placement of a Direct Complement after Finite in an indicative clause in active voice. We also allow the ordering of the verb's arguments and of the circumstances to follow information structure.

The text structuring module determines which elements are CB, and this information is encoded explicitly in the SPL. Without an explicit CB statement, the default is NB. On the basis of the CB/NB distinction in the SPL, we make sure that the status of a syntactic element generated through the grammar is determined as either CB or NB.

This information can than be used by the ordering algorithm to order those elements which have not been ordered by grammatical constraints. This ordering takes place when one traversal through the network has produced a single level of structure.

Finally, the ordering defaults can be applied, which basically encode systemic ordering for each of the languages. In addition, some basic defaults are present to ensure that there is at most one ordering for every generated grammatical structure.

3. Conclusions and future work

We have presented implementations of the grammars for Bulgarian, Czech and Russian with which it is possible to generate texts with stylistic variation for the ImD. Analyses of these texts and of instruction manuals more generally revealed the main areas of grammar that had to be treated in this round of implementation: **transitivity** and **minor transitivity**, **diathesis** (voice), **mood**, **tense**, **aspect**, **clause-complexity**, **determination** and **word order**. These are the core areas a computational grammar must cover, if it is to be a resource that can be used to generate natural text.

The methodology we have adopted has been once again that of **resource sharing**. To support resource sharing, not only with English, but among the three languages under investigation in the AGILE project, we have adopted a style of work in which language-specific biases are avoided. The work has been distributed across sites according to phenomena (BAS: mood, tense, aspect; CU: clause complexity, determination, word order; RRIAI: transitivity, minor transitivity, diathesis) rather than according to individual languages. We have thus worked in a truly contrastive-linguistic fashion, trying to make use of cross-linguistic commonalities among Bulgarian, Czech and Russian as much as possible and at the same time identifying areas of divergence and treating them in a principled way. Without this style of carrying out the implementational work in task 7.2 of the project and using a suitable platform to do so, a fast prototyping of tactical generators for these three languages would not have been possible. We now have available basic coverage grammars for all three languages with which texts of the complexity and variation given in the appendix can be generated.

The challenge for the particular implementation platform we use for grammar development - the KPML system - has been to cope with a set of languages typologically different from the languages treated within the system before. The experience in AGILE feeds back to the KPML system a number of insights about gaps in the system that become particularly obvious because we deal with typologically different languages. One area that has been dealt with in the present deliverable is **word order** – a notorious problem for implementations of languages with flexible word order, such as Czech and Russian, and to a somewhat lesser degree Bulgarian. Also, some areas, such as e.g., **aspect**, had not been treated at all before in any grammar implementation using KPML. Another area, which is on the agenda for the next round of implementation, is **agreement** - a rather complex phenomenon in all of the three languages we deal with here.

Future work will be guided by the goal of making the current grammar implementations more robust. This includes:

- Fleshing out the existing grammar systems and developing test suites (sets of SPLs) for each area in focus.
- Elaborating solutions to other problems encountered in dealing with highly inflectional languages (e.g., agreement).

Also, given that the goal of the next phase in AGILE is to generate a hyper-document in which different genres (instruction manual, ready-reference, short descriptions of each functionality of a CAD/CAM system) are interlinked, additional types of linguistic variation will have to be covered.

References

[Andonova *et al.*, 1999] Andonova E., J. Bateman, N. Gromova, A. Hartley, G.-J. Kruijff, I. Kruijff-Korbayová, S. Sharoff, H. Skoumalová, L. Sokolova, K. Staykova, and E. Teich, 1999. Formal specification of extended grammar models. AGILE deliverable SPEC2-BAS, SPEC2-CU, SPEC2-RU.

[Avgustoniva & Oliva, 1995] Avgustinova T. and K.Oliva, 1995. The Position of Sentential Clitics in the Czech Clause. CLAUS-Report Nr. 68, University of Saarbrücken, December 1995.

[Bateman *et al.*, 1998] Bateman J., E. Teich, D. Dochev, N. Gromova, J. Hana, I. Kruijff-Korbayova, H. Skoumalova, S. Sharoff, E. Sokolova , 1998. Grammatical resource implementation for Bulgarian, Czech and Russian. AGILE deliverable IMPL1-BAS, IMPL1-CU, IMPL1-RU.

[Bateman *et al.*, 1997] Bateman J. A, 1997. Enabling technology for multilingual natural language generation: the KPML environment. Natural Language Engineering, 1(1).

[Bateman *et al.*, 1990] Bateman J., R. Kasper, J. D. Moore, and R. Whitney, 1990. A general organization of knowledge for natural language processing: the Penman Upper Model. Technical report, USC/ISI, Marina del Rey, California.

[Hajičová, 1995] Hajičová E., 1995. Surface and underlying word order. In: Prague Linguistic Circle Papers, Vol.1, Amsterdam: John Benjamins, 113-124.

[Hajičová & Sgall, 1987] Hajičová E. and P. Sgall, 1987. The Ordering Principle. Journal of Pragmatics. 11(4): pp. 435-454.

[Hajičová, 1993] Hajičová E., 1993. Issues of sentence structure and discourse patterns. Theoretical and computational linguistics, Vol. 2. Prague: Charles University.

[Halliday, 1967, 1968] Halliday M. A. K., 1967, 1968. Notes on transitivity and theme in English. Journal of Linguistics 1967: 3, 37-81, 199-244; 1968: 4, 179-215.

[Halliday, 1985] Halliday M. A. K., 1985. An Introduction to Functional Grammar. Arnold.

[Halliday, 1970] Halliday M. A. K., 1970. Language structure and language function. In: New Horizons in Linguistics. John Lyons (ed.) Penguin Books Ltd. pp. 140-165.

[Kruijff *et al.*, 1999] Kruijff G.-J., I. Kruijff-Korbayová, J. Bateman, 1999. *The Text Structuring Module for the Intermediate Prototype.* AGILE project, AGILE deliverable TEXM2, July 1999.

[Kruijff-Korbayová & Kruijff, 1999] Kruijff-Korbayová I., G.-J. Kruijff, 1999. Contextually Appropriate Ordering of Nominal Expressions. To appear in: Proceedings of the ESSLLLI¨'99 Workshop on Generating Nominal Expressions, Utrecht, August 1999.

[Kruijff-Korbayová & Kruijff, 1999] Kruijff-Korbayová I., G.-J. Kruijff, 1999. Text Structuring in a Multilingual System for Generation of Instructions. To appear in: Proceedings of the Second Workshop on Text, Speech and Dialogue, Mariánské Lázně. Springer and Verlag, September 1999.

[Kruijff-Korbayová *et al.*, 1999] Kruijff-Korbayová I., G.-J. Kruijff, J. Bateman, D. Dochev, N. Gromova, A. Hartley, E. Teich, S. Sharoff, E. Sokolova, K. Staykova, 1999. Specification of elaborated text structures. AGILE deliverable TEXS2.

[Kruijff-Korbayová, 1998] Kruijff-Korbayová I., 1998. The Dynamic Potential of Topic and Focus: An Approach to Discourse Representation Theory. PhD Thesis. Charles University, Prague.

[Matthiessen, 1984] Matthiessen C.M.I.M., 1984. Choosing Tense in English. USC/ISI, Marina del Rey, CA, ISI/RR-84-143.

[Meljchuk & Kholodovich, 1970] Meljchuk I.A., A. A. Kholodovich., 1970. K teorii grammaticheskogo zaloga. In: Narody Azii i Afriki, 1970, No. 4. (In Russian).

[Sgall *et al.*, 1986] Sgall P., E. Hajičová and J. Panevová , 1986. The meaning of the sentence in its pragmatic aspects. Mey, J.L. (ed.), Reidel.

[Teich *et al.*, 1996] Teich E., Bateman J. and Degand L., 1996. Multilingual textuality: experiences from multilingual text generation. In Zock M & G. Adorni, eds., *Trends in Natural Language Generation. An Artificial Intelligence Perspective*, pp. 331-349, Springer, Berlin and New York.

Appendices: Intermediate Prototype Texts

We present first the English base texts extracted from the AutoCAD manual and modified so that we would be able to generate them in the intermendiate prototype phase, and so that they exhibit the intended variation of +/- side effect: (a) versions with side-effects, (b) without.. Sometimes (i) and (ii) versions are presented where the expressed content differs.

The Bulgarian, Czech and Russian texts corresponding to the English base ons are shown in text style alternations.

In Bulgarian, we have included two possible styles: personal in imperative mood in 2nd person plural (polite form), and non-personal in indicative mood using medio-passive voice (with a reflexive particle).

In Czech, we have included three styles: personal in imperative mood as in Bulgarian, personal in indicative mood in 1st person plural and non-personal in indicative mood using reflexive passive voice.

In Russian, we have included the personal style in imperative mood as in Bulgarian and Czech. The non-personal style in indicative mood is realized in thr same way as in Bulgarian, using medio-passive voice.

In all the styles, we present the (a) and (b) alternatives for +/- side effects, and the occasssional (i) and (ii) versions differring in content.

- **English**

IMD Text 1: pages 47-48

To create a multiline style

First open the Multiline Styles dialog box using one of these methods:

Windows: From the Object Properties toolbar or the Data menu, choose Multiline Style.

DOS and UNIX: From the Data menu, choose Multiline style.

1. Choose Element Properties to add elements to the style.
2. In the Element Properties dialog box, enter the offset of the multiline element.
3. Select Add to add the element.
4. Choose Color.
   (a) The Select Color dialog box appears. Select the element's color.
   (b) Then select the element's color from the Select Color dialog box.
5. Choose Linetype.
   (a) The Select Linetype dialog box appears. Select the element's linetype.
   (b) Then select the element's linetype from the Select Linetype dialog box.
6. Repeat these steps to define another element.
7. Choose OK to save the style of the multiline element and to exit the Element Properties dialog box.

IMD Text 2: page 46

To draw a line and arc combination polyline

First draw the 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.

4. Enter a to switch to Arc mode.
(a) The Arc mode confirmation dialog box appears. Select OK.
(b) Then select OK in the Arc mode confirmation dialog box.

5. Specify the endpoint of the arc.

6. Enter l to return to Line mode.
(a) The Line mode confirmation dialog box appears. Select OK.
(b) Then select OK in the Line mode confirmation dialog box.

7. (i) Enter the distance of the line in relation to the endpoint of the arc.
Enter the angle of the line in relation to the endpoint of the arc.
(ii) Enter the distance and angle of the line in relation to the endpoint of the arc.

8. Press Return to end the polyline.

IMD Text 3: page 58

To draw an arc by specifying three points.

Start the ARC command using one of these methods:

Windows: From the Arc flyout on the Draw toolbar, choose 3 Points.

DOS and UNIX: From the draw menu choose Arc. Then choose 3 Points.

1. Specify the start point by entering endp and selecting the line so the arcs snaps to the endpoint of the line.

2. Specify the second point by entering poi and selecting a point to snap to.

3. Specify the endpoint.

IMD Text 4: pages 48/9

To specify the properties of a multiline and save the style.

(a) From the Data menu, choose Multiline Style. The Multiline Style dialog box appears.

(b) From the Data menu, choose Multiline Style.

First specify the properties of the multiline.

1. (a) Choose Multiline Properties. The Multiline Properties dialog box appears.
   (b) In the Multiline Styles dialog box, choose Multiline Properties.
2. (i) In the Multiline Properties dialog box, select Display Joints to display a line at the vertices of the multiline.
   (ii) Select Display Joints to display a line at the vertices of the multiline.
3. Under Caps, select a line or an arc for the startpoint of the multiline. Then select a line or an arc for the endpoint of the multiline. Lastly, enter an angle.
4. Under Fill, select On to display a background color.
5. Choose Color. Then select the background fill color from the Select Color dialog box.
6. (a) Choose OK to return to the Multiline Styles dialog box. The Multiline Properties dialog box disappears.
   (b) Choose OK to return to the Multiline Styles dialog box.

Now save the style.

1. Under Name, enter the name of the style.
2. Under Description, enter a description.
3. Select Add to add the style to the drawing.
4. Select Save to save the style to a file.
5. Choose OK and close the dialog box.

IMD Text 5: page 75

To define a boundary set in a complex drawing

1. Open the Boundary Hatch dialog box using one of these methods:

Windows: From the Hatch flyout on the Draw toolbar, choose Hatch.

DOS and UNIX: From the Draw menu, choose Hatch

2. Under Boundary choose Advanced.
   (a) The Advanced Options dialog box appears. Choose Make New Boundary Set.
   (b) In the Advanced Options dialog box, choose Make New Boundary Set.

3. At the Select Objects prompt, specify the corner points for the boundary set and press Return.

4. In the Advanced Options dialog box, choose OK.

5. In the Boundary Hatch dialog box, choose Pick Points.

6. Specify the internal point and press return.

7. In the Boundary Hatch dialog box, choose Apply to apply the hatch.

- **Bulgarian**

B.1. Personal + imperative

IMD Text 1

Създаване стил на мултилиния

Първо отворете диалоговия прозорец Multiline Styles, като използвате един от следните методи:

Windows: От функционалния ред Object Properties или менюто Data изберете Multiline Style.

DOS и UNIX: От менюто Data изберете Multiline Style.

1. Изберете Element Properties, за да прибавите елементи към стила.

2.В диалоговия прозорец Element Properties въведете отместването на елемента на мултилинията.

3. Изберете Add, за да добавите елемента.

4. Изберете Color.
(a) Диалоговият прозорец Select Color се появява на екрана. Посочете цвета на елемента.
(b) След това, в диалоговия прозорец Select Color посочете цвета на елемента.

5. Изберете Linetype.
(a) Диалоговият прозорец Select Linetype се появява на екрана. Посочете вида на линията на елемента.
(b) След това, в диалоговия прозорец Select Linetype посочете вида на линията на елемента.

6. Повторете тези стъпки, за да дефинирате друг елемент.

7. Изберете OK, за да запишете характеристиките на елемента на мултилинията и да излезете от диалоговия прозорец Element Properties.

IMD Text 2

Чертаене на полилиния, съставена от отсечки и дъги

Първо начертайте отсечката.

1. Стартирайте командата PLINE, като използвате един от следните методи:

Windows: От плаващото меню Polyline на функционалния ред Draw изберете Polyline.

DOS и UNIX От менюто Draw изберете Polyline.

2. Задайте началната точка на отсечката.

3. Задайте крайната точка на отсечката.

4. Въведете **a**, за да превключите на режим
   (a) Появява се диалоговият прозорец на режима Arc. Изберете OK.
   (b) След това изберете OK в диалоговия прозорец на режима Arc.

5. Задайте крайната точка на дъгата.

6. Въведете **l**, за да се върнете в режим Line.
   (a) Появява се диалоговият прозорец на режима Line. Изберете OK.
   (b) След това изберете OK в диалоговия прозорец на режима Line.

7. (i) Въведете дължината на отсечката от крайната точка на дъгата. Въведете ъгъла на отсечката спрямо крайната точка на дъгата.
   (ii) Въведете дължината и ъгъла на отсечката спрямо крайната точка на дъгата.

8. Натиснете Return, за да завършите полилинията.

IMD Text 3

Чертаене на дъга по три точки

Стартирайте командата ARC, като използвате един от следните методи:

Windows От плаващото меню Arc на функционалния ред Draw изберете 3_Points.

DOS и UNIX От менюто Draw изберете Arc. След това изберете 3_Points.

1. Задайте началната точка, като въведете **endp** и посочите линията, така че дъгата да се захване за крайната точка на линията.

2. Задайте втората точка, като въведете **poi** и посочите точка на захващане.

3. Задайте крайната точка.

IMD Text 4

Задаване характеристиките на мултилиния и записване на стила

(a) От менюто Data изберете Multiline Style. Появява се диалоговият прозорец Multiline Style.

(b) От менюто Data изберете Multiline Style.

Първо задайте характеристиките на мултилинията

1. (a) Изберете Multiline Properties. Появява се диалоговият прозорец Multiline Properties.
   (b) От диалоговия прозорец Multiline Styles изберете Multiline Properties.

2. (i) От диалоговия прозорец Multiline Properties изберете Display joints, за да се появи линия във върховете на мултилинията.
   (ii) Изберете Display joints, за да се появи линия във върховете на мултилинията.

3. От подменюто Caps изберете линия или дъга за началото на мултилинията. След това изберете линия или дъга за края на мултилинията. Накрая въведете ъгъл.

4. Изберете On от подменюто Fill, за да видите основния цвят.

5. Изберете Color. След това посочете основен запълващ цвят от диалоговия прозорец Select Color.

6. (a) Изберете OK, за да се върнете в диалоговия прозорец Multiline Styles. Диалоговият прозорец Multiline Properties се затваря.
   (b) Изберете OK, за да се върнете в диалоговия прозорец Multiline Styles.

Сега запишете стила.

1. В полето Name въведете име на стила.

2. В полето Description въведете описание.

3. Изберете Add, за да добавите стила към чертежа.

4. Изберете Save, за да запишете стила във файл.

5. Изберете OK и затворете диалоговия прозорец.

IMD Text 5

Дефиниране на област за щриховане в сложен чертеж

1. Отворете диалоговия прозорец Boundary Hatch, като използвате един от следните методи:

Windows От плаващото меню Hatch на функционалния ред Draw изберете Hatch.

DOS и UNIX От менюто Draw изберете Hatch.

2. От подменюто Boundary изберете Advanced.
(a) Появява се диалоговият прозорец Advanced Options. Изберете Make New Boundary Set.
(b) От диалоговия прозорец Advanced Options изберете Make New Boundary Set.

3. След подсказващото съобщение Select Objects задайте ъгловите точки на областта за щриховане и натиснете Return.

6. Изберете OK в диалоговия прозорец Advanced Options.

7. От диалоговия прозорец Boundary Hatch изберете Pick Points

8. Посочете вътрешна точка и натиснете Return.

9. От диалоговия прозорец Boundary Hatch изберете Apply, за да получите щриховката.

B.2. Non-personal + indicative

IMD Text 1

Създаване стил на мултилиния

Първо се отваря диалоговият прозорец Multiline Styles, като се използва един от следните методи:

Windows: От функционалния ред Object Properties или менюто Data се избира Multiline Style.

DOS и UNIX: От менюто Data се избира Multiline Style.

1. За да се прибавят елементи към стила се избира Element Properties.
2. В диалоговия прозорец Element Properties се въвежда отместването на елемента на мултилинията.
3. Избира се Add за добавяне на елемента.
4. Избира се Color.
   (a) Диалоговият прозорец Select Color се появява на екрана.
   В него се посочва цвета на елемента.
   (b) След това, в диалоговия прозорец Select Color се посочва цветът на елемента.
5. Избира се Linetype.
   (a) Диалоговият прозорец Select Linetype се появява на екрана.
   В него се посочва видът на линията на елемента.
   (b) След това, в диалоговия прозорец Select Linetype се посочва видът на линията на елемента.
6. Повтарят се тези стъпки, за да се дефинира друг елемент.
7. Избира се OK за записване характеристиките на елемента на мултилинията и за излизане от диалоговия прозорец Element Properties.

IMD Text 2

Чертаене на полилиния, съставена от отсечки и дъги

Първо се чертае отсечката.

1. Стартира се командата PLINE, като се използва един от следните методи:

Windows: От плаващото меню Polyline на функционалния ред Draw се избира Polyline.

DOS и UNIX: От менюто Draw се избира Polyline.

2. Задава се началната точка на отсечката.

3. Задава се крайната точка на отсечката.

4. Въвежда се **a** за превключване на режим Arc.
   (a) Появява се диалоговият прозорец на режима Arc. От него се избира OK.
   (b) След това в диалоговия прозорец на режима Arc се избира OK.

5. Задава се крайната точка на дъгата.

6. Въвежда се **l** за връщане в режим Line.
   (a) Появява се диалоговият прозорец на режима Line. От него се избира OK.
   (b) След това в диалоговия прозорец на режима Line се избира OK.

7. (i) Въвежда се дължината на отсечката от крайната точка на дъгата. Въвежда се ъгълът на отсечката спрямо крайната точка на дъгата.
   (ii) Въвеждат се дължината и ъгълът на отсечката спрямо крайната точка на дъгата.

8. Натиска се Return за завършване на полилинията.

IMD Text 3

Чертаене на дъга по три точки

Стартира се командата ARC, като се използва един от следните методи:

Windows: От плаващото меню Arc на функционалния ред Draw се избира 3_Points.

DOS и UNIX: От менюто Draw се избира Arc. След това се избира 3_Points.

1. Задава се началната точка, като се въвежда **endp** и се посочва линията, така че дъгата да се захване за крайната точка на линията.
2. Задава се втората точка, като се въвежда **poi** и се посочва точка на захващане.
3. Задава се крайната точка.

IMD Text 4

Задаване характеристиките на мултилиния и записване на стила

(a) От менюто Data се избира Multiline Style. Появява се диалоговият прозорец Multiline Style.

(b) От менюто Data се избира Multiline Style.

Първо се задават характеристиките на мултилинията

1. (a) Избира се Multiline Properties. Появява се диалоговият прозорец Multiline Properties.
   (b) От диалоговия прозорец Multiline Styles се избира Multiline Properties.

2. (i) От диалоговия прозорец Multiline Properties се избира Display joints, за да се появи линия във върховете на мултилинията.
   (ii) Избира се Display joints, за да се появи линия във върховете на мултилинията.

3. От подменюто Caps се избира линия или дъга за началото на мултилинията. След това се избира линия или дъга за края на мултилинията. Накрая се въвежда ъгъл.

4. Избира се On от подменюто Fill, за да се появи основният цвят.

5. Избира се Color. След това се посочва основен запълващ цвят от диалоговия прозорец Select Color.

6. (a) Избира се ОК за връщане в диалоговия прозорец Multiline Styles. При това диалоговият прозорец Multiline Properties се затваря.
   (b) Избира се ОК за връщане в диалоговия прозорец Multiline Styles.

Сега се записва стилът.

1. В полето Name се въвежда име на стила.

2. В полето Description се въвежда описание.

3. Избира се Add за добавяне на стила към чертежа.

4. Избира се Save за записване на стила във файл.

5. Избира се ОК за затваряне на диалоговия прозорец.

IMD Text 5

Дефиниране на област за щриховане в сложен чертеж

1. Отваря се диалоговият прозорец Boundary Hatch, като се използва един от следните методи:

Windows От плаващото меню Hatch на функционалния ред Draw се избира Hatch.

DOS и UNIX От менюто Draw се избира Hatch.

2. От подменюто Boundary се избира Advanced.
(a) Появява се диалоговият прозорец Advanced Options. От него се избира Make New Boundary Set.
(b) От диалоговия прозорец Advanced Options се избира Make New Boundary Set.

4. След подсказващото съобщение Select Objects се задават ъгловите точки на областта за щриховане и се натиска Return.

5. В диалоговия прозорец Advanced Options се избира OK.

6. От диалоговия прозорец Boundary Hatch се избира Pick Points.

7. Посочва се вътрешна точка и се натиска Return.

8. От диалоговия прозорец Boundary Hatch се избира Apply за получаване на щриховката.

- **Czech**

C.1. Personal + imperative

IMD Text 1, pages 47-48

Vytvoření stylu multičáry

Nejdříve otevřete dialogový panel Styly multičár jednou z následujících metod:

Windows: Z nástrojového panelu Vlastnosti objektů nebo z menu Data vyberte *Styl mutičáry*.

DOS a UNIX: Z menu Data vyberte *Styl multičáry*.

1. Vyberte *Vlastnosti prvků* pro přidání elementů ke stylu.
2. V dialogovém panelu Vlastnosti prvků zadejte rozměr posunutí multičáry.
3. Vyberte *Přidat* pro přidání elementu.
4. Vyberte *Barva*.
   (a) Poté zvolte barvu elementu z dialogového panelu Výběr barvy.
   (b) Objeví se dialogový panel Výběr barvy. Zvolte barvu elementu.
5. Vyberte *Typ čáry*.
   (a) Poté zvolte typ čáry daného elementu z dialogového panelu Výběr typů čar.
   (b) Objeví se dialogový panel Výběr typů čar. Zvolte typ čáry daného elementu.
6. Pro vytvoření dalšího elementu tyto kroky opakujte.
7. Vyberte OK pro uložení vlastností elementu multičáry a pro opuštění dialogového panelu Vlastnosti multičáry.

IMD Text 2, p. 46

Nakreslení křivky kombinované z přímek a oblouků

Nejdříve nakreslíme rovný segment.

1. Spusťte příkaz KŘIVKA jedním z následujících způsobů:

Windows: Z plovoucího ikonového menu Křivka na nástrojovém panelu Kresli vyberte *Křivka.*

DOS a UNIX: Z menu Kresli vyberte *Křivka.*

2. Určete počáteční bod rovného segmentu.

3. Určete koncový bod rovného segmentu.

4. Pro přepnutí do režimu kreslení oblouků zadejte **o**.
(a) Objeví se dialogový panel Potvrzení režimu kreslení oblouků. Vyberte OK.
(b) Poté vyberte OK v dialogovém panelu Potvrzení režimu kreslení oblouků.

5. Určete koncový bod oblouku.

6. Zadejte e pro návrat do režimu kreslení úseček.
(a) Objeví se dialogový panel Potvrzení režimu kreslení úseček. Vyberte OK.
(b) Poté vyberte OK v dialogovém panelu Potvrzení režimu kreslení úseček.

7.
(i) Zadejte vzdálenost úsečky ve vztahu ke koncovému bodu oblouku. Zadejte úhel úsečky ve vztahu ke koncovému bodu oblouku.
(ii) Zadejte vzdálenost a úhel úsečky ve vztahu ke koncovému bodu oblouku.

8. Stiskněte ENTER pro ukončení křivky.

IMD Text 3, p. 58

Nakreslení oblouku určením tří bodů

Spusťte příkaz Oblouk jedním z následujících způsobů:

Windows: Z plovoucího ikonového menu Oblouk na nástrojovém panelu Kresli vyberte 3 body.

DOS a Unix: Z menu Kresli vyberte Oblouk. Pak vyberte 3 body.

1. Určete počáteční bod zadáním kon a vybráním čáry, takže oblouk se přichytí ke koncovému bodu.

2. Určete druhý bod zadáním bod a vybráním bodu pro přichycení.

3. Určete koncový bod.

IMD Text 4, p. 48/9

Určení vlastností multičáry a uložení stylu

(a) Z menu Data vyberte *Styl multičáry.* Objeví se dialogový panel Styly multičar.

(b) Z menu Data vyberte *Styl multičáry.*

Nejdříve určete vlastnosti multičáry.

1. (a) Vyberte *Vlastnosti multičáry.* Objeví se dialogový panel Vlastnosti multičáry.
   (b) V dialogovém panelu Styly multičar vyberte *Vlastnosti multičáry.*
2. (i) V dialogovém panelu *Vlastnosti multičáry* vyberte *Zobraz klouby* pro zobrazení čáry ve vrcholech multičáry.
   (ii) Vyberte *Zobraz klouby* pro zobrazení čáry ve vrcholech multičáry.
3. Pod *Zakončení* zvolte úsečku nebo oblouk pro počáteční bod multičáry. Poté vyberte úsečku nebo oblouk pro koncový bod multičáry. Nakonec zadejte úhel.
4. Pod *Vyplnění* vyberte *Ano* pro zobrazení barvy pozadí.
5. Vyberte *Barva*. Pak z dialogového panelu Výběr barvy zvolte barvu pro vyplnění pozadí.
6. (a) Vyberte *OK* pro návrat do dialogového panelu Styly multičár. Dialogový panel Vlastnosti multičáry zmizí.
   (b) Vyberte *OK* pro návrat do dialogového panelu Styly multičár.

Nyní styl uložte.

1. Pod *Jméno* zadejte název stylu.
2. Pod *Popis* zadejte popis.
3. Vyberte Přidat k přidání vytvořeného stylu k výkresu.
4. Vyberte *Uložit* pro uložení stylu do souboru.
5. Vyberte *OK* a uzavřete dialogový panel.

IMD Text 5, p. 73

Definování hraniční množiny v kkomplexním výkrese

1. Otevřete dialogový panel Hraniční šrafování jedním z následujících způsobů:

Windows: Z plovoucího ikonového menu Šrafy na nástrojovém panelu Kresli vyberte Šrafy.

DOS a Unix: Z menu Kresli vyberte Šrafy.

2. Pod Hranice šrafování vyberte Pokročilé.
(a) Objeví se dialogový panel Pokročilé možnosti. Vyberte Tvořit novou hraniční množinu.
(b) V dialogovém panelu Pokročilé možnosti vyberte Tvořit novou hraniční množinu.

3. Při výzvě Výběr objektů určete rohové body hraniční množiny a stiskněte Enter.

4. V dialogovém panelu Pokročilé možnosti vyberte OK.

5. V dialogovém panelu Hraniční šrafování vyberte Výběr objektů.

6. Určete vnitřní bod a stiskněte Enter.

7. V dialogovém panelu Hraniční šrafování vyberte Aplikovat pro vyšrafování plochy.[16]

[16] The Czech manual says *V dialogovém panel Hraniční šrafování vyberte Šrafuj pro vyšrafování plochy*, however, such a specific verb does not correspond to the English "apply" which can be used in varied contexts. We therefore modified the text towards this context-independent realization. Moreover, "šrafuj" is a 2nd person singular form, i.e., a realization of a familiar address which does not fit into the contexts in non-presonal style and also not into personal style where we are consistently using the polite form of address.

C.2. Personal + indicative (1st person plural)

IMD Text 1

Vytvoření stylu multičáry

Nejdříve otevřeme dialogový panel Styly multičár jednou z následujících metod:

Windows: Z nástrojového panelu Vlastnosti objektů nebo z menu Data vybereme *Styl mutičáry*.

DOS a UNIX: Z menu Data vybereme *Styl multičáry.*

1. Vybereme *Vlastnosti prvků* pro přidání elementů ke stylu.
2. V dialogovém panelu Vlastnosti prvků zadáme rozměr posunutí multičáry.
3. Vybereme *Přidat* pro přidání elementu.
4. Vybereme *Barva*.
   (a) Poté zvolíme barvu elementu z dialogového panelu Výběr barvy.
   (b) Objeví se dialogový panel Výběr barvy. Zvolíme barvu elementu.
5. Vybereme *Typ čáry*.
   (a) Poté zvolíme typ čáry daného elementu z dialogového panelu Výběr typů čar.
   (b) Objeví se dialogový panel Výběr typů čar. Zvolíme typ čáry daného elementu.
6. Pro vytvoření dalšího elementu tyto kroky opakujeme.
7. Vybereme OK pro uložení vlastností elementu multičáry a pro opuštění dialogového panelu Vlastnosti multičáry.

IMD Text 2

Nakreslení křivky kombinované z přímek a oblouků

Nejdříve nakreslíme rovný segment.

1. Spustíme příkaz KŘIVKA jedním z následujících způsobů:

Windows: Z plovoucího ikonového menu Křivka na nástrojovém panelu Kresli vybereme *Křivka*.

DOS a UNIX: Z menu Kresli vybereme *Křivka.*

2. Určíme počáteční bod rovného segmentu.

3. Určíme koncový bod rovného segmentu.

4. Pro přepnutí do režimu kreslení oblouků zadáme **o**.
(a) Objeví se dialogový panel Potvrzení režimu kreslení oblouků. Vybereme OK.
(b) Poté vybereme OK v dialogovém panelu Potvrzení režimu kreslení oblouků.

5. Určíme koncový bod oblouku.

6. Zadáme e pro návrat do režimu kreslení úseček.
(a) Objeví se dialogový panel Potvrzení režimu kreslení úseček. Vybereme OK.
(b) Poté vybereme OK v dialogovém panelu Potvrzení režimu kreslení úseček.

7.
(i) Zadáme vzdálenost úsečky ve vztahu ke koncovému bodu oblouku. Zadáme úhel úsečky ve vztahu ke koncovému bodu oblouku.
(ii) Zadáme vzdálenost a úhel úsečky ve vztahu ke koncovému bodu oblouku.

8. Stiskneme ENTER pro ukončení křivky.

IMD Text 3

Nakreslení oblouku určením tří bodů

Spustíme příkaz Oblouk jedním z následujících způsobů:

Windows: Z plovoucího ikonového menu Oblouk na nástrojovém panelu Kresli vybereme 3 body.

DOS a Unix: Z menu Kresli vybereme Oblouk. Pak vybereme 3 body.

1. Určíme počáteční bod zadáním kon a vybráním čáry, takže oblouk se přichytí ke koncovému bodu.

2. Určíme druhý bod zadáním bod a vybráním bodu pro přichycení.

3. Určíme koncový bod.

IMD Text 4

Určení vlastností multičáry a uložení stylu

(a) Z menu Data vybereme *Styl multičáry.* Objeví se dialogový panel Styly multičar.

(b) Z menu Data vybereme *Styl multičáry.*

Nejdříve určíme vlastnosti multičáry.

1. (a) Vybereme *Vlastnosti multičáry.* Objeví se dialogový panel Vlastnosti multičáry.
   (b) V dialogovém panelu Styly multičar vybereme *Vlastnosti multičáry.*

2. (i) V dialogovém panelu *Vlastnosti multičáry* vybereme *Zobraz klouby* pro zobrazení čáry ve vrcholech multičáry.
   (ii) Vybereme *Zobraz klouby* pro zobrazení čáry ve vrcholech multičáry.

3. Pod *Zakončení* zvolíme úsečku nebo oblouk pro počáteční bod multičáry. Poté vybereme úsečku nebo oblouk pro koncový bod multičáry. Nakonec zadáme úhel.

4. Pod *Vyplnění* vybereme *Ano* pro zobrazení barvy pozadí.

5. Vybereme *Barva*. Pak z dialogového panelu Výběr barvy zvolíme barvu pro vyplnění pozadí.

6. (a) Vybereme *OK* pro návrat do dialogového panelu Styly multičár. Dialogový panel Vlastnosti multičáry zmizí.
   (b) Vybereme *OK* pro návrat do dialogového panelu Styly multičár.

Nyní styl uložíme.

1. Pod *Jméno* zadáme název stylu.
2. Pod *Popis* zadáme popis.
3. Vybereme Přidat k přidání vytvořeného stylu k výkresu.
4. Vybereme *Uložit* pro uložení stylu do souboru.
5. Vybereme *OK* a uzavřeme dialogový panel.

IMD Text 5

Definování hraniční množiny v kkomplexním výkrese

1. Otevřeme dialogový panel Hraniční šrafování jedním z následujících způsobů:

Windows: Z plovoucího ikonového menu Šrafy na nástrojovém panelu Kresli vybereme Šrafy.

DOS a Unix: Z menu Kresli vybereme Šrafy.

2. Pod Hranice šrafování vybereme Pokročilé.
(a) Objeví se dialogový panel Pokročilé možnosti. Vybereme Tvořit novou hraniční množinu.
(b) V dialogovém panelu Pokročilé možnosti vybereme Tvořit novou hraniční množinu.

3. Při výzvě Výběr objektů určíme rohové body hraniční množiny a stiskneme Enter.

4. V dialogovém panelu Pokročilé možnosti vybereme OK.

5. V dialogovém panelu Hraniční šrafování vybereme Výběr objektů.

6. Určíme vnitřní bod a stiskneme Enter.

7. V dialogovém panelu Hraniční šrafování vybereme Aplikovat pro vyšrafování plochy.

C.3. Non-personal + indicative in reflexive passive

IMD Text 1

Vytvoření stylu multičáry

Nejdříve se otevře dialogový panel Styly multičár jednou z následujících metod:

Windows: Z nástrojového panelu Vlastnosti objektů nebo z menu Data se vybere *Styl mutičáry*.

DOS a UNIX: Z menu Data se vybere *Styl multičáry*.

1. Vybere se *Vlastnosti prvků* pro přidání elementů ke stylu.
2. V dialogovém panelu Vlastnosti prvků se zadá rozměr posunutí multičáry.
3. Vybere se *Přidat* pro přidání elementu.
4. Vybere se *Barva*.
   (a) Poté se zvolí barva elementu z dialogového panelu Výběr barvy.
   (b) Objeví se dialogový panel Výběr barvy. Zvolí se barva elementu.
5. Vybere se *Typ čáry*.
   (a) Poté se zvolí typ čáry daného elementu z dialogového panelu Výběr typů čar.
   (b) Objeví se dialogový panel Výběr typů čar. Zvolí se typ čáry daného elementu.
6. Pro vytvoření dalšího elementu se tyto kroky opakují.
7. Vybere se OK pro uložení vlastností elementu multičáry a pro opuštění dialogového panelu Vlastnosti multičáry.

IMD Text 2

Nakreslení křivky kombinované z přímek a oblouků

Nejdříve se nakreslí rovný segment.

1. Spustí se příkaz KŘIVKA jedním z následujících způsobů:

Windows: Z plovoucího ikonového menu Křivka na nástrojovém panelu Kresli se vybere *Křivka.*

DOS a UNIX: Z menu Kresli se vybere *Křivka.*

2. Určí se počáteční bod rovného segmentu.

3. Určí se koncový bod rovného segmentu.

4. Pro přepnutí do režimu kreslení oblouků se zadá **o**.
(a) Objeví se dialogový panel Potvrzení režimu kreslení oblouků. Vybere se OK.
(b) Poté se vybere OK v dialogovém panelu Potvrzení režimu kreslení oblouků.

5. Určí se koncový bod oblouku.

6. Zadá se **e** pro návrat do režimu kreslení úseček.
(a) Objeví se dialogový panel Potvrzení režimu kreslení úseček. Vybere se OK.
(b) Poté se vybere OK v dialogovém panelu Potvrzení režimu kreslení úseček.

7.
(i) Zadá se vzdálenost úsečky ve vztahu ke koncovému bodu oblouku. Zadá se úhel úsečky ve vztahu ke koncovému bodu oblouku.
(ii) Zadá se vzdálenost a úhel úsečky ve vztahu ke koncovému bodu oblouku.

8. Stiskne se ENTER pro ukončení křivky.

IMD Text 3

Nakreslení oblouku určením tří bodů

Spustí se příkaz Oblouk jedním z následujících způsobů:

Windows: Z plovoucího ikonového menu Oblouk na nástrojovém panelu Kresli se vybere 3 body.

DOS a Unix: Z menu Kresli se vybere Oblouk. Pak se vybere 3 body.

1. Určí se počáteční bod zadáním kon a vybráním čáry, takže oblouk se přichytí ke koncovému bodu.

2. Určí se druhý bod zadáním bod a vybráním bodu pro přichycení.

3. Určí se koncový bod.

IMD Text 4

Určení vlastností multičáry a uložení stylu

(a) Z menu Data se vybere *Styl multičáry.* Objeví se dialogový panel Styly multičar.

(b) Z menu Data se vybere *Styl multičáry.*

Nejdříve se určí vlastnosti multičáry.

1. (a) Vybere se *Vlastnosti multičáry.* Objeví se dialogový panel Vlastnosti multičáry.
   (b) V dialogovém panelu Styly multičar se vybere *Vlastnosti multičáry.*
2. (i) V dialogovém panelu *Vlastnosti multičáry* se vybere *Zobraz klouby* pro zobrazení čáry ve vrcholech multičáry.
   (ii) Vybere se *Zobraz klouby* pro zobrazení čáry ve vrcholech multičáry.
3. Pod *Zakončení* se zvolí úsečka nebo oblouk pro počáteční bod multičáry. Poté se vybere úsečka nebo oblouk pro koncový bod multičáry. Nakonec se zadá úhel.
4. Pod *Vyplnění* se vybere *Ano* pro zobrazení barvy pozadí.
5. Vybere se *Barva*. Pak se z dialogového panelu Výběr barvy zvolí barva pro vyplnění pozadí.
6. (a) Vybere se *OK* pro návrat do dialogového panelu Styly multičár. Dialogový panel Vlastnosti multičáry zmizí.
   (b) Vybere se *OK* pro návrat do dialogového panelu Styly multičár.

Nyní se styl uloží.

1. Pod *Jméno* se zadá název stylu.
2. Pod *Popis* se zadá popis.
3. Vybere se Přidat k přidání vytvořeného stylu k výkresu.
4. Vybere se *Uložit* pro uložení stylu do souboru.
5. Vybere se *OK* a uzavře se dialogový panel.

IMD Text 5

Definování hraniční množiny v kkomplexním výkrese

1. Otevře se dialogový panel Hraniční šrafování jedním z následujících způsobů:

Windows: Z plovoucího ikonového menu Šrafy na nástrojovém panelu Kresli se vybere Šrafy.

DOS a Unix: Z menu Kresli se vybere Šrafy.

2. Pod Hranice šrafování se vybere Pokročilé.
   (a) Objeví se dialogový panel Pokročilé možnosti. Vybere se Tvořit novou hraniční množinu.
   (b) V dialogovém panelu Pokročilé možnosti se vybere Tvořit novou hraniční množinu.

3. Při výzvě Výběr objektů se určí rohové body hraniční množiny a stiskne se Enter.

4. V dialogovém panelu Pokročilé možnosti se vybere OK.

5. V dialogovém panelu Hraniční šrafování se vybere Výběr objektů.

6. Určí se vnitřní bod a stiskne se Enter.

7. V dialogovém panelu Hraniční šrafování se vybere Aplikovat pro vyšrafování plochy.

- **Russian**

IMD Text 1: pages 47-48

Чтобы создать стиль мультилинии

Сначала откройте диалоговое окно Multiline Styles одним из следующих способов:

Windows: В панели инструментов Object Properties или в меню Data выберите пункт Multiline Style..

DOS & UNIX: В меню Data выберите пункт Multiline Style.

1. Нажмите кнопку Element Properties, чтобы добавить элементы в стиль.

2. В диалоговом окне Element Properties введите смещение первого элемента линии.

3. Нажмите кнопку Add, чтобы добавить этот элемент.

4. Выберите пункт Color. Затем выберите цвет элемента в диалоговом окне Select Color.

Выберите пункт Color. На экране появится диалоговое окно Select Color. Выберите в нем цвет элемента.

5.

   (a) Выберите пункт Linetype. На экране появится диалоговое окно Select Linetype. Выберите в нем тип линии элемента.
   (b) Выберите пункт Linetype. Затем выберите тип линии элемента в диалоговом окне Select Linetype.

6. Повторите эти шаги, чтобы задать еще один элемент.

7. Нажмите кнопку OK, чтобы сохранить стиль элементов мультилинии и закрыть диалоговое окно Element Properties.

IMD Text 2: page 46

Чтобы нарисовать полилинию, состоящую из отрезков прямых и дуг

Сначала нарисуйте отрезок прямой.

1. Запустите команду PLINE одним из следующих способов:

Windows: В палитре Polyline на панели инструментов Draw выберите пункт Polyline.

DOS & UNIX: В меню Draw выберите пункт Polyline.

2. Укажите начальную точку отрезка прямой.

3. Укажите конечную точку отрезка прямой.

4.
(a) Нажмите клавишу **a**, чтобы перейти в режим Arc. На экране появится диалоговое окно Arc mode. Нажмите OK.
(b) Нажмите клавишу **a**, чтобы перейти в режим Arc. Затем нажмите OK в диалоговом окне Arc mode.

5. Укажите конечную точку дуги.

6.
(a) Нажмите клавишу **l**, чтобы вернуться в режим Line. На экране появится диалоговое окно Line mode. Нажмите OK.
(b) Нажмите клавишу **l**, чтобы вернуться в режим Line. Затем нажмите OK в диалоговом окне Line mode.

7. (a) Укажите расстояние линии по отношению к конечной точке дуги. Укажите угол линии по отношению к конечной точке дуги.
(b) Укажите расстояние и угол линии по отношению к конечной точке дуги.

8. Нажмите клавишу Return, чтобы завершить рисование полилинии.

IMD Text 3: page 58

Чтобы нарисовать дугу по трем заданным точкам

Запустите команду ARC одним из следующих способов:

Windows: В палитре ARC на панели инструментов Draw выберите пункт 3 Points.

DOS & UNIX: В меню Draw выберите пункт ARC. Затем выберите пункт 3 Points.

1. Чтобы указать начальную точку дуги, введите endp и задайте линию, к конечной точке которой привязана дуга.

2. Чтобы указать вторую точку, введите poi и задайте точку для привязки дуги.

3. Укажите конечную точку.

IMD Text 4: pages 48/9

Чтобы определить свойства мультилинии и сохранить ее стиль

(a) В меню Data выберите пункт Multiline Style. На экране появится диалоговое окно Multiline Style.

(b) В меню Data выберите пункт Multiline Style.

Сначала определите свойства мультилинии.

1. (a) Выберите пункт Multiline Properties. На экране появится диалоговое окно Multiline Properties.
   (b) В диалоговом окне Multiline Styles, выберите пункт Multiline Properties.
2. (i) В диалоговом окне Multiline Properties выберите пункт Display joints, чтобы отобразить линию у вершин мультилинии.
   (ii) Выберите пункт Display joints, чтобы отобразить линию у вершин.
3. В окне Caps задайте прямую или дугу для начальной точки мультилинии. Затем задайте прямую или дугу для конечной точки мультилинии. Наконец задайте угол.
4. В окне Fill нажмите On, чтобы показать цвет фона.
5. Нажмите кнопку Color. Затем в диалоговом окне Select Color укажите цвет фона.
6. (a) Нажмите кнопку OK, чтобы вернуться в диалоговое окно Multiline Styles. Диалоговое окно Multiline Properties исчезнет с экрана.
   (b) Нажмите кнопку OK, чтобы вернуться в диалоговое окно Multiline Styles.

Теперь сохраните стиль.

1. В пункте Name задайте имя стиля.
2. В пункте Description задайте описание стиля.
3. Нажмите кнопку Add, чтобы добавить стиль мультилинии к рисунку.
4. Нажмите кнопку Save, чтобы сохранить стиль в файл.
5. Нажмите кнопку OK и закройте диалоговое окно.

IMD Text 5: page 75

Чтобы определить набор границ в сложном рисунке

1. Откройте диалоговое окно Boundary Hatch одним из следующих способов:

Windows: В палитре Hatch на панели инструментов Draw выберите пункт Hatch.

DOS & UNIX: В меню Draw выберите пункт Hatch.

2. (a) В пункте Boundary нажмите кнопку Advanced. На экране появится диалоговое окно Advanced Options. Нажмите кнопку Make New Boundary Set.
(b) В пункте Boundary нажмите кнопку Advanced.

3. В диалоговом окне Advanced Options нажмите кнопку Make New Boundary Set.

4. В окне запроса Select Objects укажите граничные точки набора границ и нажмите Return.

5. В диалоговом окне Advanced Options нажмите OK.

6. В диалоговом окне Boundary Hatch нажмите кнопку Pick Points.

7. Укажите внутреннюю точку и нажмите Return.

8. В диалоговом окне Boundary Hatch нажмите кнопку Apply, чтобы применить штриховку.