AGILE

Automatic Generation of Instructions in Languages of Eastern Europe

---

Title ***Specification of elaborated text structures***

Authors Ivana Kruijff-Korbayová
Geert-Jan Kruijff
John Bateman
Danail Dochev
Nevena Gromova
Tony Hartley
Elke Teich
Serge Sharoff
Lena Sokolova
Kamenka Staykova

Deliverable *TEXS2-Bu, TEXS2-Cz, TEXS2-Ru*

Status *Final*

Availability *Public*

Date *30 April 1999*

INCO COPERNICUS PL961104

Abstract:

This report contains the TEXS2-Bu, TEXS2-Cz and TEXS2-Ru deliverables. The purpose of this set of deliverables is to present detailed descriptions of the text structures occurring in the AGILE corpora and in other sources used, and to specify the text structures and the style to be generated by the intermediate prototype. In addition, we present our initial ideas concerning the implementation of the text structuring module, which is to be described in detail in the deliverable TEXM2.

Our approach to text structuring is set within the broader framework of systemic functional linguistics which is employed in the KPML system we use for tactical generation.

The descriptions of text structures we present are based on an additional corpus study which extends the corpus study carried out in Work Package 3 and reported in [AGILE 3.1]. Our main aim in the corpus study has been to survey the variety of possible text structuring strategies and the corresponding repertoire of linguistic realisations used in each of the languages when expressing instructions. We concentrated on a selection of basic grammatical features and their deployment in the studied texts. In this report, we explain the choice of the analysed features and we present and discuss our observations.

On the basis of the corpus analysis, we have chosen the text structures to be generated by the intermediate prototype. We describe these in detail, using similar methods as we used in the deliverables of Task 5.1 concerned with the texts structuring for the initial demonstrator [AGILE 5.1] to describe the text structures for the initial demonstrator. We also present sample texts in the three languages in which these text structures occur, and we explain the components of the SPL expressions reflecting the text style variations.

Finally, we provide initial specifications of the text planning mechanisms that will be used to generate these text structures in the AGILE intermediate prototype. The implementation of the text structuring module of the intermediate prototype will be reported in detail in the next deliverable of Task 5.2, namely deliverable TEXM-2.

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

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

Table of Content

1. Introduction (CU) 1
   1.1 The Theoretical Framework 1
   1.2 The Approach Pursued in AGILE 5
   1.3 Corpus Investigation Objectives 6
   1.4 Text Structure Parameters 7
   1.5 Text Structuring Module 8
   1.6 Report Outline 9
2. Corpus Investigation 10
   2.1 Characterisation of Instructional Texts 10
   2.2 Description of Additional Corpora 13
      2.2.1 Additional Czech Corpus 13
      2.2.2 Additional Bulgarian Corpus 15
      2.2.3 Additional Russian Corpus 15
   2.3 Text Structuring Findings 16
      2.3.1 Content Distribution 16
      2.3.2 Genre and Lexico-grammatical Means in Czech 18
      2.3.3 Genre and Lexico-grammatical Means in Bulgarian 22
      2.3.4 Genre and Lexico-grammatical Means in Russian 23
      2.3.5 Summary 27
3. Variation of Text Style for the Intermediate Prototype 29
   3.1 Explicit vs. Implicit Side-Effects 30
      English: 31
      Czech: 32
      Russian: 33
      Bulgarian: 34
   3.2 Text Style Alternations in Instructions 34
      3.2.1 Alternative Instructions in Czech 34
      3.2.2 Alternative Instructions in Russian 38
      3.2.3 Alternative Instructions in Bulgarian 41
   3.3 SPL Components Reflecting Text Style Options 43
      3.3.1 Realisation of Headings 43
      3.3.2 Realisation of Instruction Steps 44

Indicative mood 44
Imperative mood 44
Infinitive mood 45
Active voice 45
Medio-passive (reflexive passive) voice 45
3.3.3 Form of Address 45
Politeness 45
Number 46
Person 46
4. Text Structuring for the Intermediate Prototype 46
4.1 Text Templates and Text Structure Elements 47
4.1.1 Text Structure Elements 48
4.1.2 Text Templates and General Conditions on Realisation 50
4.2 Mapping Between Text Structure Elements and T-Box Concepts 52
4.3 Elaborate Example of Generating a Text Structure 54
4.3.1 The A-Box 54
4.3.2 The Text Structure 57
4.3.3 Realisation and Layout 58
5. A Prelude to the Text Structuring Module (TSM) 60
5.1 General Architecture of the TSM 60
5.2 Systemic Networks for Text Structuring 60
5.3 From Text Structure to SPL 61
6. Conclusions and Further Work 63
Agile documentation: 66
Czech corpus: 67
Appendices: Intermediate Prototype Texts 69
A. English 70
IMD Text 1: pages 47-48 70
IMD Text 2: page 46 71
IMD Text 3: page 58 71
IMD Text 4: pages 48/9 72
IMD Text 5: page 75 73
B. Bulgarian 74
B.1. Personal + imperative 74
IMD Text 1 74

IMD Text 2 75
IMD Text 3 75
IMD Text 4 76
IMD Text 5 77
B.2. Non-personal + indicative 78
IMD Text 1 78
IMD Text 2 79
IMD Text 3 80
IMD Text 4 81
IMD Text 5 82
C. Czech 83
C.1. Personal + imperative 83
IMD Text 1, pages 47-48 83
IMD Text 2, p. 46 84
IMD Text 3, p. 58 84
IMD Text 4, p. 48/9 85
IMD Text 5, p. 73 86
C.2. Personal + indicative (1st person plural) 87
IMD Text 1 87
IMD Text 2 88
IMD Text 3 88
IMD Text 4 89
IMD Text 5 90
C.3. Non-personal + indicative in reflexive passive 91
IMD Text 1 91
IMD Text 2 92
IMD Text 3 92
IMD Text 4 93
IMD Text 5 94
D. Russian 95
IMD Tex 1: pages 47-48 95
IMD Tex 2: page 46 96
IMD Tex 3: page 58 97
IMD Tex 4: pages 48/9 98
IMD Tex 5: page 75 99

Table of Figures

Figure 1 Language stratification in SFG 2
Figure 2 Grammatical means in instructional texts in Czech, additional corpus 21
Figure 3 Grammatical means in instructional texts in Bulgarian, additional corpus 22
Figure 4 - Example of HTML-style specification of TSE layout 52
Figure 5 Example of generated text 54
Figure 6 A-box for example text extracted from IMD-Text1 57
Figure 7 From A-box to text structure 58
Figure 8 Example of text realisation 59
Figure 9- CADCAM-INSTRUCTIONS / Text Structure Elements network 62

1. Introduction (CU)

This report presents the results of Task 5.2 within the Agile project work package WP5 concerned with text structuring. It contains the TEXS2-Bu, TEXS2-Cz and TEXS2-Ru deliverables, in which we address text structuring in Bulgarian, Czech and Russian instructional texts, for the purpose of the intermediate prototype. In addition, we present our initial ideas concerning the implementation of the text structuring module, which is to be described in detail in the deliverable TEXM2.

The work described here has as input the work done in Task 5.1 (Text Structuring for the initial demonstrator), and serves as input to the Tasks 5.3 (Text structuring for final prototype) and 8.1 (Integration of intermediate prototype).

According to the Technical Annexe [Agile project 1997, p. 27], the purpose of the present set of deliverables is to present detailed descriptions of the text structures occurring in the AGILE corpora and in other sources used, and to specify the text structures and the style to be generated by the intermediate prototype.

In order to define the problem of text structuring in AGILE and to explain properly what our aims are, we need to describe what the issues of text-structuring are from the viewpoint of the theoretical framework within which we are working. The framework determines the kinds of choices that we are making and the class of problems we are tackling in the approach to text structuring in AGILE. This in turn determines the features of texts that we need to study in our corpora.

1.1 The Theoretical Framework

The approach to text generation in the AGILE project employs a *functional* model of the language system. The overall framework is based on the systemic functional linguistic theory (SFL, e.g. [Halliday 1970, 1985]). We try to incorporate also the relevant insights of the Functional Generative Description approach (FGD, e.g. [Sgall et al. 1986]) developed since the sixties by the research group at Charles University in Prague. In this section, we introduce the essential ideas that guide our approach to text structuring.

A central characteristic of SFL is its view of language as a resource for expressing meaning. In this perspective, a linguistic utterance is the result of a complex choice process which recursively selects among options provided by interconnected networks of semantic, grammatical and lexical choice systems associated with different levels of abstraction. These levels of abstraction correspond to a stratal organisation of the language system, an idea common in functional linguistics. The relations between the strata are those of *abstraction* (higher strata are abstractions of lower ones) and *realisation* (a stratum is realised by the next stratum directly below). The stratification model of language is illustrated in Figure 1.

The strata of the linguistic resources are organised into networks of choices, each choice resulting in a different meaning realised (expressed) by appropriate

structures. Choices made in each stratum constrain the choices available in the stratum beneath.

At the most abstract stratum, the systems of the context in which language is embedded and located are discerned. Context thus constrains language. For the application area considered in the AGILE project, we are dealing with written texts whose purpose is to instruct the user of a software system, specifically AutoCAD, how to perform certain tasks. Thus our texts are instructions in the software domain.

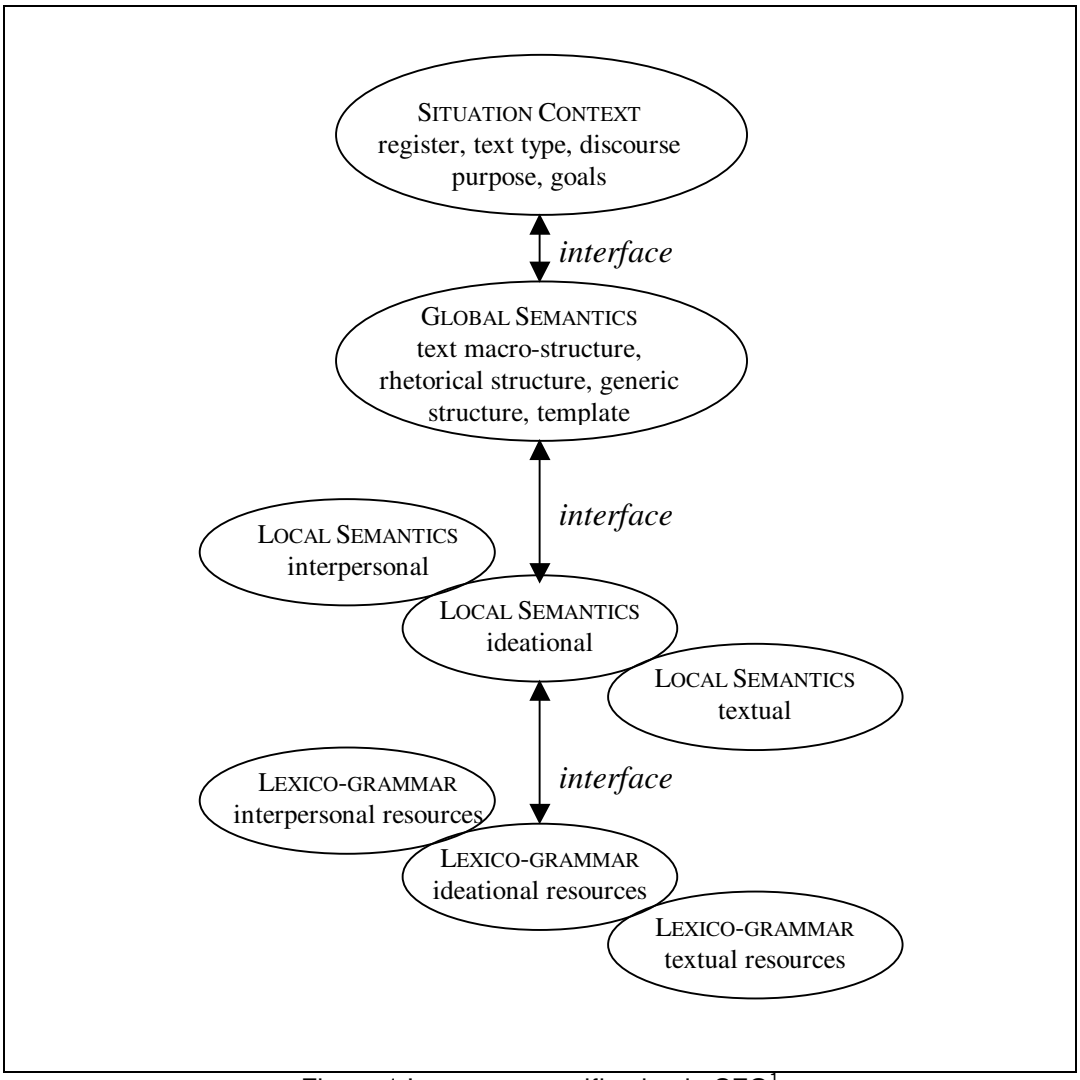


Figure 1 Language stratification in SFG[1]

In the model illustrated in the figure, an intermediate level of **global semantics** is assumed, which covers the realisations at the borderline between text and sentences. It can be characterised as the level of text organisation corresponding to the given context and the content to be communicated.

[1] The picture in Figure 1 is adopted from [Ramm and Villiger 1995], where it appeared as a modification of a figure presented in [Matthiessen and Bateman 1991].

Below the global semantics stratum is **local-semantics stratum**, which is viewed as the resources for (sentence) *meaning* comprising the linguistic potential of discourse-related phenomena, whereas the lexico-grammatical level provides the resources for *wording* these meanings [Ramm and Villiger 1995, Matthiessen and Bateman 1991].

In addition to the vertical organisation into different strata, SFL theory works with a metafunctional diversification that cuts across the strata horizontally. Three metafunctions are distinguished, which reflect the simultaneous functional layers corresponding to different perspectives of the linguistic system: *ideational, interpersonal and textual metafunction*.

As described in [Matthiessen and Bateman 1991], the *ideational metafunction*, further subdivided into *experiential* and *logical*, is concerned with propositional content. The *interpersonal metafunction* covers the resources for creating and maintaining social relationships between the writer and the reader. The *textual metafunction* provides the resources for contextualising the other two types of information.

In [Ramm and Villiger 1995] it is argued that taking such a stratified model as the underlying organisation enables a better account of the complexity of discourse organisation than is possible in approaches that distinguish only two levels, semantic and syntactic. The advantage should be that there in fact two interfaces on the way between the global semantics and the lexico-grammar, thanks to the stratum of local semantics. Therefore, more diversified linguistic information can be modelled. Their study then concentrates on the interaction between global discourse parameters such as text type and subject matter and the selection of local features responsible for realisation. Such an approach promises to help overcoming the notorious problem in text generation known as the *generation gap*, i.e. the gap between global level text planning (strategic generation) and lexico-grammatical expression (tactical generation). The problem lies in the text planner's lack of control over the fine-grained distinctions available in the grammar. The stratification proposed in the model just discussed decomposes the interface between the text planner's output (representation of global discourse structure) and the input to lexico-grammatical realisation (a representation of clause-level interpretation of global discourse structure) into smaller modules that should be easier to handle than one complex interface. This is the reason why we also employ such a stratified approach.

The investigation of global discourse parameters such as text function, text structure and subject matter is the domain of *text linguistics*, which classifies texts into different types. Texts of different *text type* differ in the lexico-grammatical options chosen to bring out the intended textual meaning [Ramm and Villiger 1995]. The local, lexico-grammatical options which are usually considered to be under discourse-level control are the following:

- lexical choice: the decision which content words to choose from the lexicon in order to express the intended ideational meaning
- grammatical means: the decision concerning, among others, the predicate-argument configuration (ideational), grammatical realisation of mood (interpersonal), word order (textual).

Another line of global text structure parameters study is *register theory* [Halliday and Hasan 1985] which concentrates on the context of use.

[Ramm and Villiger 1995] argue that the text type has a decisive influence on which type of information has priority in structuring the text. There are said to be text types whose structure is more *domain-driven*, i.e. the text structure reflects some inherent organisation of the domain. But even in these text types, for instance *descriptive, expository, instructive*, etc., it appears that there are important differences regarding how the information is realised [Ramm et al. 1995].

The view that the domain-specific organisation of information is not the only source of control in the generation process is corroborated by the research reported by Hartley and Paris [1996], which also employs an SFL approach. They adopt an approach to generation common in technical domains which employs a *task structure*, i.e., a goal and its associated plan, as an underlying representation. Generating an instruction for performing a task means to chose some *task elements* to be expressed and linearise them so that they form a coherent set for a given goal the user might have.

Hartley and Paris analyse a corpus of software instructions in French and English, in particular Macintosh manuals, for the purpose of their automatic generation from an underlying semantic model of the task. Their objective is to establish whether the *task model* alone is sufficient to control the linguistic output of a text generation system, or whether additional control is required. The corpus study shows that task structure does not suffice to control text generation, and thus another source of information is needed in order to constrain the choices available to the tactical generator. They propose that for the instructional texts an obvious option is to explore the *communicative purpose* as this additional source of control.

The communicative purpose does not need to be constant throughout a manual, as Hartley and Paris observe. They discern three different *functional sections* in the Macintosh manuals:

- a *tutorial* containing exercises for new users
- a *series of step-by-step instructions* for the major tasks to be accomplished
- a *ready-reference summary* of the commands

Moreover, they suggest that it is useful to distinguish between two more specific communication purposes in the step-by-step section:

- *procedure*, where the purpose is to enable the reader to perform a task
- *elaboration*, where the purpose is to increase the reader's knowledge about the task, the way to achieve it, or the properties of the system as a whole.

The functional sections differentiated by communicative purposes according to [Hartley and Paris 1996] can be considered to constitute different *genres* in terms of [Martin 1992]. Genres are distinguished by their communicative purposes, and they control text structure and realisation. In Martin's view, genre is defined as a

staged, goal-oriented social process realised through register, the context of situation, which in turn is realised in language to achieve the goals of a text. According to Hartley and Paris, genre is responsible for the selection of a text structure in terms of task elements. As part of the realisation process, generic choices preselect a register associated with particular elements of text structure, which in turn preselect lexico-grammatical features.

The comparative analysis of the lexico-grammatical features found in step-by-step instructions and ready-reference as reported in [Hartley and Paris 1996] reveals that

- ready-reference and procedure, one "sub-genre" within step-by-step, are in sharp contrast,
- while elaboration, the other "sub-genre" within step-by-step, shares features with both procedure and ready-reference.

The study shows that genre, like task structure, provides some control over the linguistic resources during generation. While neither of them alone suffices in this process, together they offer a prospect of adequate control over the output of a text generator.

1.2 The Approach Pursued in AGILE

In order to handle text planning in the process of text generation in the AGILE project, we build on the research findings discussed above. We adopt a strategy which incorporates genre and task structure as two sources of control as proposed by Hartley and Paris [1996] into the stratificational view introduced by Matthiessen and Bateman [1991]. Moreover, following the findings of our corpus analysis, we allow for different *text styles* to be used within a genre, i.e. to fulfil a given communicative purpose.

We employ *text templates* and *text structure elements* to control the output of the text generation process. The task structure and task elements discussed above are reflected by *text structure elements*. The genre and text style are reflected by text templates.

Text templates correspond to the text styles used in our instructional texts, with regard to a specific genre. We conceive of a text template as, first of all, a coherent set of choices regarding certain aspects of the realisation, for example options related to mood, voice, person and number, etc. Rather than to have to specify each of these aspects individually, the user can choose one template or another. The advantage of this is that we are able to capture interdependencies between the options for the three languages that concern us. For instance, person and number are only relevant in the active voice in indicative mood, or in the imperative. It is also the case that a particular way of realising the heading of an instructional text may be incompatible with a certain way of realising the instruction-steps. It is also the case that text style parameters need to be used consistently throughout an instructional text which pertains to one genre, and, if genres are varied in a manual, also in all parts of the manual that pertain to that genre. In addition, each text template can define a particular layout of the

generated text. The text templates mediate between the text type, purpose and goal on the one hand and the local semantics on the other hand.

Besides text templates, we use *text structure elements*. These are directly related to particular parts of the definition of the text content (A-box), and orthogonal to the text templates. For each text structure element, the typical ways of expressing the content mapped to it are specified for each language.

The text template chosen by the user and the text structure elements to which the content defined by the user is mapped thus co-determine the actual realisation of the text content.

1.3 Corpus Investigation Objectives

The objectives of the corpus investigation we report on in the present set of deliverables have been as follows:

- to study the variety of genres and text styles as global discourse parameters employed in instructional texts in Czech, Bulgarian and Russian
- to account for the interaction of these global discourse parameters with the selection of local features responsible for the appearance of the individual linguistic units that make up the texts we are considering
- to specify which lexico-grammatical features are influenced by the global parameters

On the basis of this study, we develop a proposal which enables us to control the relevant features in the course of text generation in AGILE. Let us now briefly present the particular objectives of the corpus investigation and the features on which we concentrated our study.

- Similarly to the initial demonstrator, the texts considered for generation by the intermediate prototype are simplified versions of routine passages occurring frequently in the user guide of the AutoCAD system, namely instructions to users how to perform a task. However, the scope of the intermediate prototype is broader; we take into account alternative ways of expressing instructions.
- In order to determine the range of useful alternations, we assembled additional corpora of instructional texts in our three languages, and we carried out additional corpus analyses. This enabled us to provide descriptions of a wider range of text structuring choices than is present in our initial corpora drawn from the AutoCAD manual. First, we extended the analysis of the AGILE CAD/CAM corpus studied in WP3 by taking into account also other parts of the CAD/CAM manual than the step-by-step lists of instructions studied in the previous phase. Second, we assembled and studied some additional corpora of instructional texts from various sources for each of the three languages.

We collected samples of other instructional texts in the software domain in Bulgarian, Czech and Russian. Where available, we preferred original texts to translations. However, if original texts were very scarce, we chose some translated material that has been judged as good text by native speakers. In order

to achieve a broader scope with respect to the linguistic means employed in instructions, we also included a few samples of instructional texts from other domains.

The corpus analysis carried out in this phase of the AGILE project has been rather qualitative than quantitative. Unlike in WP3, we did not fully annotate the corpora this time. Our main aim has been to overview the variety of possible text structuring strategies and the corresponding repertoire of linguistic realisations used in each of the languages when expressing instructions.

1.4 Text Structure Parameters

The comparison of various instructional texts confirmed our initial hypothesis that instructions can be expressed in alternative ways within each of the three languages. Some of the realisations are common across the three languages, some are not.

As stated above, it is common to consider the lexical choice and the choice of grammatical means of realisation as the local parameters under discourse-level control. We did not investigate variations in lexical choice in the sense of alternative ways of referring to the same concept. But rather, besides variation in grammatical means we concentrated on variation in the distribution of content and the corresponding choice of lexical realisation. The alternations we studied can thus be classified into two main groups as follows:

(i) *choice of grammatical means*: whether the choices available to the tactical generator are predetermined in certain ways by the text style; if so, whether it is a distinguishable subset of options

(ii) *distribution of content and lexical realisation*:
– whether information is realised explicitly in the most direct way corresponding straightforwardly to the task model as specified in an A-box, so that the inference load required in interpretation is minimal, or
– whether some information is left more or less implicit; what is the realisation of the explicitly conveyed information

In summary, the most obvious alternations under (i) are concerned with the ways of addressing the readership, expressing or hiding the intentional agent of an action, and the mode of realising instructions; under (ii), we consider the complexity of linguistic expressions, the order of presenting various bits of information, and the level of detail in the explicit expression of the content.

As an example of (i), consider the variation of using indicative versus imperative mood, or passive versus active voice. For illustration, compare the four versions of expressing the same instruction in (1) below.

(1)

(a) *First start the* PLINE *command by choosing* PLINE *from the* Draw *menu.*

(b) *We first start the* PLINE *command by choosing* PLINE *from the* Draw *menu.*

(c) *One first starts the* PLINE *command by choosing* PLINE *from the* Draw *menu.*

(d) *First, the* PLINE *command is started by choosing* PLINE *from the* Draw *menu.*

An example of the variation reflected in (ii) above is the explicit mentioning of side-effects versus their implicit assumption, as illustrated in (2) below, where (a) illustrates the version with explicit side-effect and (b) with the side-effect being left implicit.

(2)

(a) *Choose* Color. *The* Select Color *dialog box appears.*
Select the element's color.

(b) *Choose* Color.
Then select the element's color from the Select Color *dialog box.*

While the alternations under (i) are really alternations concerning lexico-grammatical options, the alternations under (ii) involve a more complex interplay between semantics and the lexico-grammar, because they do not only involve differences in *how a given content is presented*, but also in *what content is presented explicitly*.

It should be stressed that the alternations concern the realisation of a given, constant, content which is provided as the input, in the form of an output text. The user of the text generation system developed in AGILE can influence the generated output by choosing a particular template which sets the relevant parameters related to the choices under (i) and (ii) above. The behaviour of the text generation system as a whole is thus parametrised for text style.

On the basis of the observations gained by analyzing our corpora, we have chosen the text structures to be generated by the intermediate prototype. We decided to concentrate on the choices of basic grammatical features under (i), and from (ii), we decided to cover alternations related to whether side effects of actions are explicitly expressed or not.

1.5 Text Structuring Module

Let us remind that the application scenario of the system developed in AGILE is such that the user provides a definition of a text content in the form of an A(ssertion)-box (cf. the report of Task 2.1 concerned with the domain modelling for the initial demonstrator [AGILE 2.1], and the report of Task 5.1 concerned with the text structuring for the initial demonstrator). This input is handled by a 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. Succinctly put, a text template defines a text style by conditioning the way text structure elements are to be realised. Text structure elements themselves can be mapped to layout rules. We specify the text structure elements common to the languages under consideration in AGILE, and given the target texts for the intermediate prototype In order to account for the text style alternations reflecting the parameter setting provided by the user, the TSM has to produce the appropriate SPL components.

Given the SPLs produced by the TSM, the tactical generators for the three languages take the SPLs as input, and generate the corresponding sentences (cf. the report of Task 7.1 concerned with the implementation of tactical generators for the initial demonstrator [AGILE 7.1]).

Given this processing flow, the alternations described above are to be reflected in the SPLs obtained as the output of the TSM. The two classes of alternations, (i) and (ii), are reflected slightly differently by the TSM. The changes of parameters corresponding to (i) result quite straightforwardly in changes of the corresponding conditions in the SPLs. In most cases the content conveyed by individual sentences of the text is not affected (with the exception of expressing or not the agent of an action). The settings of the parameters under (ii), on the other hand, affect the distribution of content over the text. When a side effect of an action is to be expressed explicitly, this means the presence of one clause to convey the corresponding content, for example that a dialog box appears on the screen. Subsequent clauses than can exploit the context provided by this clause, for example, a subsequent action of entering some data in that dialog box, does not need to express the location explicitly, but can assume it to be known from the context. When the side effect is not expressed, such action needs to have an explicitly expressed location.

1.6 Report Outline

The report is organised as follows. Section 2 presents the results of corpus investigation. For each of the three languages under consideration we first describe the corpora used and, second, we discuss and exemplify text style alternations in instructional texts on the basis of our corpus investigation. In Section 3, we describe the text structure alternations for realising instructions chosen for the intermediate prototype in detail and the components of SPL formulas corresponding to the linguistic realisations. In Section 4, we discuss our approach to text structuring using similar methods as we used in the previous Work Package 5 report [AGILE 5.1] to describe the text structures for the initial demonstrator. In Section 5, we provide an initial discussion of the text planning mechanisms that will be used to obtain these text structures in the process of automatic text generation in the implemented texts structuring module. In the final Section 6, we conclude by a brief summary of the approach to text structuring presented in this report and its assessment.

2. Corpus Investigation

In this section, we report on the corpus investigation carried out in order to choose and specify the text structures to be covered in the intermediate prototype.

Similarly to the initial demonstrator, the texts considered for generation at this stage are simplified versions of descriptions of step-by-step procedures for performing tasks as found in the user guide of the AutoCAD system. They satisfy the characteristics of texts feasible for automatic generation as discussed in [Power et al. 1994, Power and Scott 1997] by being routine passages occurring frequently in the manual, which have a quite simple semantic content and a regular style, terminology and structure. The texts to be covered by the intermediate prototype are provided in the Appendix.

The scope of the intermediate prototype is broader than that of the initial demonstrator. Not only do we enable the generation of a wider range of texts covering a larger part of the CAD/CAM domain. We also cover a wider range of alternative ways of expressing instructions. The alternations are obtained through a more complex approach to text structuring. They concern ways of addressing the readership, expressing or hiding the intentional agent of an action, the mode of realising instructions and the level of detail of expressing the content explicitly.

In order to determine the range of useful alternations, we performed a corpus study extending the observations attained in Task 3.1 and reported in [AGILE 3.1]. For the sake of descriptions covering a wider range of text structuring choices than is present in our initial corpora drawn from the AutoCAD manual, we needed an extended corpus study. This in turn required us to include additional texts in the investigation.

The need for additional corpora stems from the fact that the text structuring encountered in our basic corpus extracted from the AutoCAD user guide does not vary. This is of course natural, since the essential parameters of text structuring should be kept constant in a coherent text. Moreover, the texts are not original Bulgarian, Czech and Russian ones, but are translated from English, and the translation is rather literal. Therefore, we needed to investigate also original instructional texts in our languages in order to determine the range of possible text structures.

This section is organised as follows. We first discuss some general characteristics of instructional texts. Then we provide the details about the corpus of instructional texts collected for each of the three languages under consideration. Finally, we discuss the interactions between global and local discourse parameters and lexico-grammatical features encountered in the instructional texts we studied.

2.1 Characterisation of Instructional Texts

First of all, we had to ask ourselves what type of texts to include in the class of *instructional texts.* Available classifications of text genres, e.g. Longacre's work as cited in [Martin 1992], distinguish between texts which prescribe actions and

texts that do not. However, it is not easy to fit "instructional texts" into such classification.

An initial screening of some texts revealed that an instance of what we intuitively consider an instructional text may contain parts that are actually **instructing** as well as **expository** or **descriptive** parts (in [Martin 1992] terms).

Our observation that a text does not have to be of the same genre throughout is in line with the findings reported in [Hartley and Paris 1996] we discussed in Section 1.1. Let us remind that they distinguished between two more specific communication purposes, i.e. genres or "sub-genres", within the step-by-step section:

- *procedure*, where the purpose is to enable the reader to perform a task
- *elaboration*, where the purpose is to increase the reader's knowledge about the task, the way to achieve it, or the properties of the system as a whole.

An example of instructive (procedure) and descriptive (elaboration) text appearing together is the explicit realisation of a side effects amidst step-wise instructions, as in the above example. Or, another example taken from the AutoCAD corpus with elaboration amidst procedure is the following:

(3)

(a) *English*: You can enter these relative values in the form *@distance<angle* (in this case, you would enter @3<100).

(b) *Czech*: Můžete zadat tyto relativní hodnoty ve formě *@vzdálenost<úhel* (v tomto případě, raději zadejte @3<100).

(c) *Bulgarian*: Можете да въведете тези относителни стойности под формата на *@разстояние<ъгъл* (в дадения случай: @3<100)

(d) *Russian:* Можно указать эти данные в следующей форме: *@расстояние<угол* (в нашем случае, *@3<100*).

This also illustrates the observation made by Hartley and Paris that unlike in the procedure texts, the elaboration is often realised with an explicit modality element, namely expressing possibility rather than necessity or obligation.

So we can say that we also encounter such a distinction in our texts. This is of course not very surprising given the fact that we too analysed primarily software manuals. Moreover, we found such a distinction between "sub-genres" also in our texts from other domains. In other than AutoCAD texts, there are often descriptions of the context of the actions, or the results of actions, which would be classified as elaboration genre as defined above.

But even when we concentrate on the instructive parts of texts as such. i.e. procedure genre according to [Hartley and Paris 1996], they appear to be of two different kinds:

- those that tell the reader to perform certain tasks in a certain order if he wants achieve a particular goal (let us call them **procedural** henceforth

- those that tell the user what to do, or what could be done under certain circumstances (let us call them **informative** henceforth)

It seems that the text in the introductory sections in the AutoCAD manual, as well as the ones we annotated as "Further possibilities", are more of the "informative" type, while the step-wise sections we annotated as "Procedure" in the earlier corpus analysis within Work Package WP3 [AGILE 3.1] are of the procedural type.

The procedural text type is what we encounter not only in instructional texts in the software domain or other technical domains, but also in cookery books etc. The time sequence is important there, hence the organisation into ordered steps, often also accompanied by discourse markers indicating sequentiality (e.g. adverbials like 'first', 'now', 'then').

The informative text type seems to be at the borderline between instructive and descriptive text type in [Martin 1992] terms, sharing features of both, similarly to the elaboration sub-genre according to [Hartley and Paris 1996]. It is often encountered in software manuals. It does tell the user what to do (that is why we include it in the instructional type), but the principal organisation method appears to be the situation in which the user is, for instance a particular mode or window of the interface, much less so the sequential ordering (although it may also be present and indicated by discourse markers). In this type of text, we get modals much more often, e.g. you can do this or that. Also, various possibilities for achieving the goal are often mentioned.

Within the context of the AGILE project, the target of generation are instructions of the type that appear software manuals. We label such texts *software instructional texts*. They appear to be a species within the more general class of *technical instructional texts*, i.e., instructional texts in technical domains. Software instructional texts and technical instructionl text have been of prime interest in our present corpus study. We agreed to view technical instructional texts as texts characterised by the following pragmatic features:

- **Explicit goal orientation** The goal of an instructive text is to prepare the reader to work with an artefact –e.g., a technical system, a home appliance device, or a program. Therefore it has to be clear and well-structured presentation of operational information about the functions, structure, modes of operation and ways of use of the artefact.
- **Implicit model of the target audience** The instructional texts are build upon a set of authors' assumptions about the intended readers –their conceptual system (implying the reader's domain model), overall background knowledge of the field, their habits and preferences to acquire such kind of information. Often the authors of software technical instructions apply a rather rigid picture of the reader's characteristics (e.g. adults, using more or less professionally a programming system).

The main target of the overall text is to give the reader operational information. Therefore, our study focussed on the more standard procedural parts of the instructional texts.

The texts we collected for the purpose of the present corpus investigation fulfil the characteristics of instructional texts discussed above. They contain procedural as well as informative parts in order to enable us to compare the realisations in these two sub-genres.

2.2 Description of Additional Corpora

First, we extended the analysis of the AGILE CAD/CAM corpus studied in WP3 by taking into account also other parts of the CAD/CAM manual than the step-wise lists of instructions studied in the previous phase, namely the passages which are informative rather than procedural.

Second, we assembled and studied some additional corpora of instructional texts extracted from various software manuals and other sources for each of the three languages. Where available, we preferred original texts to translations. However, if original texts were very scarce, we chose some translated material that has been judged as good text by native speakers. In order to extend the scope of investigation with respect to the linguistic means employed in instructions beyond software instructional texts, we included a few samples of technical instructional texts from other domains, e.g., technical device manuals, household appliances user guides, car repair manuals, and "non-technical" instructional texts, e.g., painting instructions, cookery books.

2.2.1 Additional Czech Corpus

As mentioned above, we included some more passages from the AutoCAD manual. In particular, these were instructions from the introductory passages which precede the numbered steps in each section of chapter 2 of the manual. For illustration, see examples (3) and (4) below.

(4) p. 44

Úsečky používejte tehdy, pokud budete potřebovat editovat jednotlivé, segmenty. Jestliže budete chtít nakreslit sadu úseček jako jeden objekt, pak použijte křivku.
(Use lines when you need to edit individual segments. If you want to draw a set of lines as one object, use polyline.)

(5) p. 47

Aby se vám lépe ovládal počet elementů a vlastnosti každého elementu, můžete si vytvořit pojmenované styly multičar. Styl také ovládá vyplnění pozadí a koncové tvary. Můžete také editovat multičáry, ovládat průsečíky, rohové uzly a počet vrcholů.
(In order to have better control over the number of elements and the properties of each element, you can create named multiline styles. The style controls the background fill and the end shapes. You can also edit multilines, control the joints, corner nodes and the number of corners.)

These texts instruct the user what to do if s/he wants to achieve a particular effect ("procedural"), or which command(s) to use under certain conditions ("informative"). They exhibit a more varied repertoire of lexico-grammatical features than the instructions in the numbered steps, e.g., explicit modality elements and conditionals.

Besides the AutoCAD manual texts, we collected other instructional texts of various types. Some of the additional texts are, like the AutoCAD manual, translations from English, others are original, written by Czech authors. None of the entire texts of this additional corpus were available in electronic form, only in hard copy. We selected some passages of various lengths either randomly or because they included some interesting phenomenon and typed those in, in order to have electronically available samples of this additional corpus. The size of this electronically available part of the additional corpus is rather small (altogether around 60 KB).[2]

Primarily, we looked for software instructional texts. We included samples from the user guides of the following software products:

- a drawing editor (original: SGP Baltazar manual [Baltazar manual])
- text editors (translation: MS Word manual [Word manual], original: MS Word 6 guide [Word guide], and text editor T602 manual [T602 manual])
- user interfaces (original: MS Works 4 guide [Works guide], Windows for Workgroups guide [Win guide])

Then, we included samples from technical instructional texts in other domains, namely the user guides describing the installation of the following:

- a personal computer (probably translation: Autocont manual [PC manual])
- a network adapter (translation: EtherLink manual [Ether. manual])
- a satellite receiver (original: [Tesla manual])

Finally, we included samples from the following instruction texts:

- the Trabant car repair guide (original: [Trabant guide])
- painting guidelines (original: [painting guide])
- four cooking books (original: [general recipes], [cheese recipes], making tea and coffee [drink recipes], food conservation [conservation])
- two travel guides (original: Italy [Italy guide], mountain tourist guide “Krušné hory” [mountain guide])

These instructional texts are characterised by a descreasing level of technicality, and orientation on operational information in the sense of utilising some technical device. Our motivation behind including them was to be able to compare, eventually, the text structures encountered in texts where the main communicative purpose is

- to familiarize the reader with the tools provided by a software system or another technical device, and to provide instructions for using these tools to perform particular tasks

[2] Due to copyright restrictions, we cannot make the texts publicly available.

- to provide the reader with instructions for achieving particular goals without an explicit concern for the tools that are being used

The current corpus analysis has not fully explored the differences between these two types of instructional texts, but we would like to address them at some later stage in the project if it proves to be relevant.

2.2.2 Additional Bulgarian Corpus

We included the following passages from the AutoCAD manual:

- the first chapter of the AutoCAD manual
- the introductory passages preceding the procedures in chapter 2, which were used in the corpus study in Work Package 3 [AGILE 3.1]

These texts are translations from English [AutoCAD 1996].

Besides the AutoCAD manual texts, we analysed other software manuals (mainly translations from English, but also an original Bulgarian manual). We inspected samples from the user guides of the following software products:

- CAD system for DOS (PLOT 3 user guide, written by Bulgarian authors) [ПЛОТ3 1989],
- Popular operating system for personal computers (Windows 95 authorised manual, translated from English) [Windows95 1998].
- Internet programming through Java Script (a book, translated from English), [JavaScript 1996].

The additional texts for analysis were available only in hard copy form.

2.2.3 Additional Russian Corpus

Besides the texts from the AutoCAD manual chosen for the intermediate prototype, we collected other technical instructional texts of various types. Some them additional texts are translations from English, like the AutoCAD manual. Others are original Russian texts. We analysed samples from software instructional text, namely the user guides of the following software products:

- CAD systems for DOS (original: SPRUT user guide, PICAD operations manual),
- text editors (translation: MS Word manual),
- specifications for software applications (translation: software for a Doppler effect navigation equipment).

In addition, we analysed the following technical instructional texts:

- a manual for the hydraulic system of the TU-204 aircraft
- a car repair manual
- some instructions for home appliance devices (Kettle Jet Classic cordless, Cordless telephone, Mounting ski boots and so on –usually translations)

2.3 Text Structuring Findings

The repertoire of aspects that could be considered in a corpus study like ours is very rich. However, we had to narrow the scope of the corpus analysis down in order to keep it accomplishable within the current task of the text structuring work package. We chose to concentrate on:[3]

- expressing or hiding the intentional agent of an action (agency, voice)
- ways of addressing the readership (person, number, voice)
- mode of realising instructions (mood, voice)
- the level of detail of expressing the content explicitly, in particular regarding side-effects of actions

The first three issues have all to do directly with lexico-grammatical features, and are obviously interleaved. For instance, if the agent is unidentified ("hidden"), it is impossible to realise the instruction in imperative, because imperative involves reference to the user, even though there is no overt expression in the sentence establishing the reference.

The corpus analysis carried out in this phase of the AGILE project has been rather qualitative than quantitative. Unlike in the Work Package 3 [AGILE 3.1], we did not fully annotate the corpora this time. Our main aim has been to overview the variety of possible text structuring strategies and the corresponding repertoire of linguistic realisations used in each of the languages when expressing instructions.

The investigation of the issues of wording has been much easier and it has been clearer what should be taken into account, than in the case of the issues of content. We discuss our findings in detail later in this section, but first we turn our attention to content distribution.

2.3.1 Content Distribution

It was more difficult to collect corpus evidence concerning the explicit presentation of information. The main reason for this is that in order to study these issues properly we would have to have an idea about the modelling of the tasks in the domains from which our additional corpora were extracted. However, this was of course impossible for such a wide variety of instructional texts. Our investigation in this respect has therefore been rather informal.

There are texts in each of the languages under consideration with no or little non-instructional text interleaved with the instructions, as well as texts where the amount of other text than pure instructions is greater. For instance, in the Czech [painting guide] the structure is such that each paragraph begins by an instruction what to do and then the rest of the paragraph provides detailed descriptions

[3] We indicate in brackets the grammatical features that are mainly concerned in each case, i.e. agency, mood, voice, person, number. For their detailed discussion and linguistic specifications with respect to the three languages under consideration, see [AGILE 6.2].

concerning the reasons why, the result state etc. In the Czech car repair guide [Trabant manual], it is often the case that subtasks are detailed out in running text within one item in the numbered list.

In the software instructional texts, some mixture of instructions and other types of text is very common. It often concerns side effects, but also various information what can or should not be done, formats of entering information, descriptions of objects, relations to other tasks and commands, etc.

An interesting issue regarding content distribution is the level of detail to which the actions are explained. What we mean is how much recursion of goal-method is involved. One can perfectly well say *Save the file* in which case there are no details of the methods at all. Or, one can say *Save the file using the command Save* or *Save the file using the command Save which you start by clicking on the Save icon* etc.[4]

How much detail is provided depends on, or should be determined by, the level of knowledge assumed on the reader's (user's) side, exploiting some model of the reader (user). This is easily illustrated by the following common sense example: an experienced cook knows what "some salt" means, what tools to use for stirring and how exactly to stir, etc. Similarly, an experienced Windows user knows where to find commands and how to trigger them, for instance.

Unfortunately, none of the manuals for other software products than AutoCAD will shed any light on how much needs to be explicitly presented to the reader of the AutoCAD manual. All that thinking has been compiled into the manual by the technical authors who wrote it. What we can get from looking at the other manuals is more instances of goal-method recursion and so a wider range of realisations. What the intermediate prototype will show is how a CAD/CAM instruction might look if the author decides that more information should be made explicit, using the parameter +/- side effect to illustrate the differences.

The +/- side effect parameterisation is a plausible choice: for instance, this alternation could correspond to two situations in which the user is consulting the manual – (i) while performing the task (or attempting to) at the computer, in which case mention of the side effect in the text is made redundant by the appearance of the dialog box on the screen, or (ii) off-line, with no visual feedback from the software itself.

In the following three sections, we present our observations of the interactions between global discourse parameters and lexico-grammatical features for each of the three languages under consideration.

---

[4] Similarly, descriptions of the objects being referred to can be more or less detailed, for instance with respect to their location, shape, colour, etc. However, the issues of planning referring expression has not been an object of the present study.

2.3.2 Genre and Lexico-grammatical Means in Czech

We present our findings concerning the lexico-grammatical features and in one integrated table in Figure 2 below. First of all, let us overview the range of possible realisations of instructions in Czech.

- **Agent identified, imperative mood**
  - **speaker included or excluded** (this means whether the speaker is or is not included among the possible agents); with speaker included we get first person (plural), (6a), but this possibility for imperative is not encountered in our corpus, with speaker excluded we get second person, cf. ex. (6b,c)
  – **polite (formal) or familiar (informal) form** of the imperative can be used for the second person, the former yielding second person plural (6b), the latter second person singular (6c); instructions in books for adults use always the polite form of imperative in Czech
  - **the kind of imperative** encountered in instructions is the jussive; the optative kind of imperative also exists in Czech, but it is not used in instructional texts;
  - **actor can be either implicit or explicitly realised**, however, in instructional texts the latter is not encountered

(6) Imperative mood

(a) *Zadejme OK* (*Enter*-1pl *OK*)

(b) *Zadejte OK* (*Enter*-2pl *OK*)

(c) *Zadej OK* (*Enter*-2sg *OK*)

- **Agent identified, indicative mood**
  – again, **speaker included or excluded**; the speaker included, first person plural version is encountered in our corpus, cf. ex. (7a), as well as the speaker excluded, second person version, cf. ex. (7b,c)
  – again, **polite (formal) or familiar (informal) form** of the indicative can be used for the second person, and instructions in books for adults use always the polite form of indicative (7b) rather than the familiar one (7c)
  - **perfective or imperfective aspect** can be used depending on the processual characteristics of the action
  – identified agent implies **active voice**, actor is thus conflated with Subject and can be either implicit or explicitly realised;
  – again, **actor can be either implicit or explicitly realised**; in those cases where actor is the user (or a general, unspecified agent), it is most common to pronominalise it; since it is conflated with the Subject, such pronoun is normally dropped in Czech, because Czech is a pro-drop language

(7) Indicative mood, active voice, Subject/Actor=User is dropped

(a) *Zadáme OK* (*Enter*-1pl *OK*)

(b) *Zadáte OK* (*Enter*-2pl *OK*)

(c) *Zadáš OK* (*Enter*-2sg *OK*)

- **Agent unidentified, indicative mood**
  – **Agent unidentified** implies **passive voice** in the indicative mood, the agent

responsible for the action is suppressed; by the agent we normally understand the user, but is could also be the software system;
– the **complex passive or the reflexive passive** can be used, but we only encountered the latter in our corpus
– again, **perfective or imperfective aspect** can be used depending on the processual characteristics of the action

(8) *Zadá se OK* (*Enter*-3sg refl *OK*)

- **Agent unidentified, infinitive**
  – the agent responsible for the action is again suppressed
  – again, **perfective or imperfective aspect** can be used depending on the processual characteristics of the action

(9) *Zadat OK* (*Enter*-inf *OK*)

The possibilities of realising instructions in Czech overviewed above differ in agency, mood, voice, person and number. We therefore concentrated on these in our corpus analysis (see the table in Figure 2 for a summary).

First of all, the corpus analysis confirmed our expectation that Czech instructional texts commonly employ the indicative mood rather than imperative or the infinitive, even though these three modes are in general viable alternatives. Within the indicative mood, active voice is much more common than the passive. So we can summarise that in the Czech instructional texts, both translated and original it is much more frequent that the agent is identified (even though possibly general, unspecific) then that the agent is unidentified.

In the original Czech instructions, imperative mood is rather rare. It appears sometimes in instructions presented in the style of numbered lists ([Baltazar manual], [Works guide], [Win guide], [Tesla manual]), but in the original texts we looked at it never appeared in running text (unlike in the translated [AutoCAD manual] where imperative is used also in running text). It also was the case that all the passages that used imperative were classified as procedural text rather than informative, in both original Czech texts and translations (again with the exception of the [AutoCAD manual]).

The use of infinitive in the texts we studied is very rare, in fact we only found a couple of occurrences in the [PC manual]. Our conjecture is that the use of infinitive to express instructions in Czech is confined to informal texts or telegraphic-style texts. Thus, not the genres appearing in edited manuals for general non-expert public. We include infinitive as a mode for expressing instructions in the scope of our investigation and experiments in AGILE for the sake of completeness and comparison.

Regarding person and number (in both indicative and imperative form), we only encountered first or second person and the plural number. First person (only in indicative mood) is most naturally seen as realising the general, unspecified agent, rather than a specified set of agents including the writer. Whatever interpretation we choose, it is clear that first person plural form in indicative mood is a very common realisation in original Czech instructional texts.

As for second person, we only encountered the polite form, and therefore plural number in our corpus. Actually, we did have one set of instruction texts at our disposal which employed the familiar imperative form, and thus singular number. These were translated texts of instructions for children how to conduct simple “scientific-experiments”. Since the targeted readership for these texts was different from the one assumed for the texts we generate in AGILE, we decided not to take these texts into consideration in the present corpus study.

As we said above, texts with identified agent prevailed in our corpus. Complex passive appears entirely absent in instruction texts. The only source in which agent was unidentified, and which employed reflexive passive, was the [Trabant guide]. This fact may suggest that passive voice is really not very common in modern Czech instruction texts. Nevertheless, we include reflexive passive as a mode for expressing instructions in the scope of our investigation and experiments in AGILE for the sake of completeness and comparison.

Figure 2 presents a summary of our corpus analysis in tabular form in. The first two columns identify the source (with respect to the list provided above) and whether the texts were original or translated. The next two columns concern genre and layout: The third column identifies the sub-genre of instructive texts, i.e. procedural or informative (cf. the introduction to Section 2). The fourth column identifies the layout of the text. We distinguished between running text and listing (bulleted or numbered lists). As we noted above, original Czech instructions never used imperatives within running text, no matter whether procedural or informative.

The contents of the columns for mood, person and number, and voice should be self-explanatory. In cases where we encountered diversity within one source, multiple lines are included.

Corpus	Source	Sub-Genre	Format	Mood	Person, number	Voice
AutoCAD intro	tran	info	running	imper	2nd, pl	active
Baltazar manual	orig	proc	running list	indic indic, imper	1st, pl. 1st, 2nd, pl	active
Word manual[5]	tran	proc info	list running	imper indic	2nd, pl	active
Word guide	orig	info	running	indic	2nd, pl	active
T602 manual	orig	info	running	indic	2nd, pl	active
Works guide[6]	orig	info, proc	running list	indic imper	2nd, pl	active
Win guide[7]	orig	info, proc	running list	indic imper	2nd, pl	active
PC manual	tran	proc	list	indic imper infin	2nd, pl	active
Ether manual	tran	proc	list	imper	2nd, pl	active[8]
Tesla manual	orig	proc	list	imper indic	2nd, pl 1st, pl	active
painting guide	orig	proc	running	indic	1st, pl	active
Trabant guide	orig	proc	list	indic	3rd, sg, pl 1st, pl	reflexive passive active[9]
general recipes	orig	proc	running	indic	1st, pl	active
cheese recipes	orig	proc	running	indic	1st, pl	active
drink recipes	orig	proc	running	indic	2nd, pl	active
conservation	orig	proc	running	indic	1st, pl	active
Italy guide[10]	orig	proc	running	indic	1st, pl	active
mountain guide[11]	orig	proc	running	indic	1st, pl	active

Figure 2 Grammatical means in instructional texts in Czech, additional corpus

[5] Headings (goals) as: *Jak uložit soubor (how to save a file.)*

[6] Few procedural text units.

[7] Few procedural text units.

[8] But: *Připojení se provádí upevněním konektorů do odpovídajících zdířek*. (refl.)

[9] Very few instructions in active voice in these texts. Mostly reflexive passive.

[10] Mostly descriptive text, instructions rare.

[11] Mostly descriptive text, instructions rare.

2.3.3 Genre and Lexico-grammatical Means in Bulgarian

When analysing the corpus we found that as with the Czech texts, standard patterns were commonly used. The informative passages of texts introduce the main concepts and objects used in the further description as well as the basic principles of the interface system in case of manuals for software systems. As was expected, they are written with the use of more varied syntactic and rhetorical structures.

A summary of our findings concerning the lexico-grammatical features is presented in tabular form in Figure 3 below. The meaning of the columns and the abbreviations is the same as explained for Figure 2 containing the corpus analysis summary for Czech.

Corpus	Source	Sub-genre	Mood	Person, number	Voice
AutoCAD	tran	info	indic	2nd p. pl	active
AutoCAD	tran	proc	imper indic	2nd p. pl. 3rd p. sg	active reflexive
Java Script	tran	info	indic imper	2nd p. pl. 3rd p. sg, pl 1st p sg, pl	active reflexive
Windows 95	tran	info	indic	2nd p. pl	active
Windows 95	tran	proc	imper	2nd p. pl. 3rd p. sg	active reflexive
PLOT 3	orig	proc	indic imper	2nd p. pl 3rd p. sg., pl. 3rd p. sg.	active passive reflexive

Figure 3 Grammatical means in instructional texts in Bulgarian, additional corpus

The most frequent verb form in the corpus as a whole is 2nd person plural, because of the imperative mood, widespread in the procedure texts. Occurrences of verbs in 1st person singular or 1st person plural were met only in the Java Script manual (informative text). Even there they were rare, appearing in sentences explicitly concerning the author, or mentioning both the author and the reader. The 3rd person plural appears only in passive constructions. The voice is predominantly active, although in original Bulgarian text the passive voice is far more frequent than the active.

We have thus noted a significant difference between Bulgarian translated texts and original ones. The texts translated from English tend to use active, personal verb forms, while in the original instructional texts impersonal forms predominate. However, we have to consider that the analysed original text is ten years old. The majority of software manuals used in Bulgaria nowadays are translations from English and they influence seriously the style of current Bulgarian instructional texts towards the use of personal imperative forms.

2.3.4 Genre and Lexico-grammatical Means in Russian

In this section we consider some problems arising as evidences from the Russian corpus analysis of instructional texts in different technical areas. We consider some specifics of discourse structure in software instructions reflected in the style of Russian instructional texts. We begin with some original instructional texts in more traditional areas (aircraft manuals, car-repair instructions, home appliances instructions), and finish with a study of an extended corpus of instructions for using software.

2.3.4.1 Rhetorical Functions for Representation of Text Structure

An interesting feature of original Russian manuals is that they do not address the reader directly, for example, impersonal constructions are actively used, even direct warnings are written in an impersonal form [Sharoff and Sokolova, 1995]. So the main style of the Russian instructional texts is impersonal. It seems that Russian technical manuals often describe not instructions for use but laws of some artificial world. As the result, these manuals often provide no explicit cues to the rhetorical structure, so placing a greater burden of interpretation on the reader, as this extract taken from the manual for the hydraulic system of the aircraft TU-204 exemplifies:

(10) В начале работы после запуска двигателя на земле при охлажденной жидкости до температуры ниже -35°С, необходимо произвести разогрев жидкости гидросистемы до температуры выше -35°С, включив кран кольцевания выключателем КК на панели наземной подготовки на щитке ГИДРОСИСТЕМА. Электрическая схема включения крана кольцевания представлена на рисунке.
(Upon the beginning of operations after lighting the jet of the aircraft on the ground, cooling the liquid in hydraulic system down to the temperature below -35°C should result in its heating to the temperature above -35°C by turning on the circulation mode switch CM which is placed on the “ground operations” board of the HYDRAULIC SYSTEM panel. The electric circuit for turning the circulation mode on is shown on the drawing.)

In the context of our project of multilingual text generation, two dimensions of style can be considered: style as a multilingual phenomenon and as a phenomenon inside a concrete language. In [DiMarco and Hirst 1993] four parameters of a text style are distinguished: lexical choice (using words of different generalising or emotional or other special characteristics), syntax (using simple or complex constructions), theme (different notions in focus) and semantics (aspects of the situation description, for example, more estimative or more technical one). In a multilingual perspective, all these parameters are involved. For example, lexical choice differences are evident from our study of the sub-register of car-repair manuals. English (as well as German and French) tends to use much more abstract lexical meanings in comparison to Russian. This naturally influences the different forms of SPL (see Section 3) for these languages. For example, coordination reduction which is necessary for these Western European languages using one word for the two processes and impossible for Russian having two different words and obviously two propositions:

(11)

(G) [Motoreinfuelldeckel und Ablassschraube] entfernen.
(Engine oil filler cap and drain bolt remove);
(E) Remove [the engine oil filler cap and drain bolt];
(F) Retirer [le bouchon de remplissage d'huile ainsi que le boulon de vidange].
(Remove the engine oil filler cap and drain bolt);
(R) [Снять крышку маслозаливной горловины] и [отвернуть сливную пробку].
(Take away oil filler cap and screw-off drain bolt).

Another influence on text style are syntactic variations, when some meanings are realised by different grammatical means. In particular, the meanings of Russian cases can be expressed by predicates in English, for example,

(12)

(E) using one of the following methods:
(R) одним из следующих методов
(one-Instr from following methods-Gen);
(E) by specifying three points
(R) по трем точкам
("along" three points-Dat).

Theme variations are common in discussions of English–Russian translation. The thematic and the focused part are very often ordered inversely in this pair of languages, for example,

(13)

(E) You can use the ENTER key to create blank lines.
(R) Для создания пустых строк используется клавиша ENTER
(For creation [of] empty lines-gen it-is-used key ENTER)

Multilingual semantic variations are illustrated, for example, by texts of instructions for choosing the appropriate shoe size in English, French, German and Italian [Vanderlinden and Scott, 1995].

In contrast to multilingual styles variations in instructional texts, the variations inside one language are very limited with respect to lexical choice, syntax and theme parameters. The obvious variations concern mostly the semantic aspect, in particular, different reflection of text structure in instructional text. For example, in contrast to the impersonal style of the fragments of Russian manuals presented above, the instructional texts for home appliances which are translated from English use mostly the imperative style.

2.3.4.2 Software Instructions

We start here with some general remarks concerning the register. The general characteristics of instructions is the presence of anonymous AUTHOR and indefinite number of ADDRESSEE(s). The AUTHOR influences the ADDRESSEE to perform some actions using the given software product. But in difference to traditional technical instructions, the software product behaves like

another conscious participant. It responds to commands, answers, suggests some actions. It "speaks" a specified natural language (language of the dialog with the user) which can be chosen or changed (in our case usually English or Russian). So we have another embedded communicative structure: ACTOR (the former ADDRESSEE), and ADDRESSEE2 (the product). So the software instructions demonstrate a new functional style, which differs from the instructional texts from traditional areas mentioned above. So we can distinguish three discourse types in the register of software instructional texts:

1. the first type reflects the inner domain-driven organisation (usually described in specifications for a product systems),
2. user –computer communicative discourse, including user, his/her actions and computer reactions as replays to these actions, and
3. embracing communicative structure –the author of the text and the user as addressee.

All these types appear in Russian software instructional texts which are always a combination of different styles. Besides texts from the AutoCAD manual chosen for the intermediate prototype, we collected other instructional texts of various types. Some of the additional texts are translations from English, like the AutoCAD manual. Others are original Russian texts. As mentioned above (Section 2.2.3), we analysed samples from user guides of CAD systems for DOS (SPRUT user guide, PICAD operations manual), MS Word manual, and specifications for a software application (Doppler effect navigation equipment).

SPRUT is an original Russian software and belongs to the register under consideration in AGILE. So its style is of special interest for us. Its largest part is written in an impersonal descriptive style as is traditional for Russian instructions. But it has specific parts where the author describes the embedded communicative situation addressee –addressee2 (product). The latter is presented as a person and included in the semantic structure as a third participant. The description deploys in present or future tense form, perfective aspect; for example:

(14) Редактор не позволит Вам создать строки большей длины.
(Editor will not allow you to create lines of bigger length.)

(15) При завершении работы система SVN проверит были ли внесены изменения d текущий проект.
(Upon completion of operations, SVN system will check whether there were any alternations in the current project.)

(16) Если текущий проект был изменен, то система предупредит Вас об этом.
(If the current project has been changed, the systems warns you about this.)

In these examples, the user is presented as the addressee in an embraced communication plan, that is in relation to the author (evidence from the personal pronoun, 2nd person, polite form –*Вам, Вас*). The other perspective of the user–computer communication strategy is presented by the user-oriented semantic

structure. In this case, the user is also presented in 3rd person, and functions as subject:

(17) Пользователь может вести несколько баз проектов.
(User can develop several bases of projects.)

But (s)he also can be presented as the addressee of the embedded communication:

(18) Система предоставляет пользователю ряд дополнительных средств.
(System gives user some additional opportunities.)

The same strategy is possible in impersonal instructions without explicit reference to a user. In this case propositions are introduced by modal operators: *можно* (it is possible), *достаточно* (it is enough), *следует* (you should), *необходимо* (it is necessary), for example,

(19) При выходе из этой подсистемы необходимо указать контур (или контуры), которые необходимо вернуть при возврате в систему SVN.
(By exiting from this system it is necessary to specify contour (or contours), which it-is-necessary to return when returning to SVN system.)

Goals are expressed by headings that are either nominalisations

(20) Управление выводом геометрических объектов в графическое окно
(Controlling output of geometrical objects in graphic window)

(21) Выбор контура (Selection of a contour)

or infinitival clauses

(22) Создать новый проект (To create a new project)

The operation manual for PC-CAPS, a sub-application of PICAD, is also originally written in Russian (though the product it describes is English). It mostly consists of descriptive sentences, in which separate propositions are usually realised in medio-passive, for example,

(23) Состояние слоя и его цвета изменяются по команде VLYR;
The state of the layer and its colors are altered by the VLYR command

This style reflects the semantic strategy related to the data-driven part of the instructional discourse structure. It describes functions of commands, keys and so on, author, user and computer system are omitted from the semantic representation. This manual also demonstrates an additional style, realising recommendations. They are realised by a special indefinite-personal construction with the third person plural verb form without a subject, for example,

(24) Часть слоев делают невидимыми, чтобы не перегружать чертеж.
(Some layers-Acc they-make invisible, in order to not overload drawing.)

or by the modal operator *удобно (it is convenient),* for example,

(25) При создании символов микросхем удобно выбрать шаг координатной сетки 5 мм.
Upon creation of the chips symbols it-is-convenient to choose coordinate grid step of 5 mm.

Descriptions, which are related to the embracing situation, rather than to the embedded interaction between the user and the application, sometimes use suggestive imperative forms which otherwise are impossible in this register:

(26) Приведем структуру слоев графического редактора.
Let's describe the structure of layers in the graphical editor.

This form is usual for mathematical texts. It presupposes that the author is the actor. In the instructional texts we have an embedded situation involved, so the user is the actor, not author.

The MS Word user guide is translated from English. It contains both descriptive passages as they are found in traditional Russian instructions and imperative clauses which follow the English tradition and are close to the style of text under consideration in AGILE. In this style, the only possible imperative realisation is the 2nd person plural (polite). Singular form, suggestive form (see the previous example) or occasional means to express the imperative situation are not possible.

The above research shows that Russian manuals are written in a mixed style. The obvious variations are related to the semantic aspect, in particular, differently reflecting discourse structure of instructional text. Some parts realise step-by-step structure, but some others present descriptions, often influenced by the data organisation, not the user-instructive perspective. [Sokolova, to appear] discusses in detail the discourse structure of instructional texts and its reflection in text styles in Russian and, in particular, in imperative semantic structure.

2.3.5 Summary

We studied the lexico-grammatical features in relation to genre in additional corpora in order to determine what alternations in text style should be covered in the phase of the intermediate prototype. The range of variations in software instructional texts within each particular language appear rather limited with respect to the features we studied. The main parameter turns out to be the presentation of the Agent responsible for the actions addressed in the instructions. This is, in turn, reflected in the choice of mood, agency and voice features. Our observations concerning the range of realisations encountered the most in the procedural and informative sub-genre in the texts we studied can be summarised as follows.

- **Agent identified, imperative mood**
  Bu, Cz, Ru: realisations corresponding to excluded speaker in polite form of address, and therefore in second person plural; the kind of imperative is jussive, actor implicit
- **Agent identified, indicative mood**
  Bu, Cz: realisations corresponding to speaker excluded in polite form of address, and therefore second person plural; active voice; implicit actor

Cz: realisations corresponding to speaker included, general actor, and thus in first person plural; active voice, actor implicit

- **Agent unidentified, indicative mood**
  Bu, Cz, Ru: realisations corresponding to suppressed agent, medio-passive or reflexive passive
- **Agent unidentified, infinitive**
  Ru: realisations corresponding to suppressed agent

On the basis of the observations gained by analyzing our corpora, we have chosen the text style alternations to be generated by the intermediate prototype, which are discussed in detail in the next section.

3. Variation of Text Style for the Intermediate Prototype

In this section, we present the alternations of style to be covered in the intermediate prototype, and how the text style alternations are reflected by components of SPL formulas. We consider alternative realisations which concern the following two aspects (cf. (i) and (ii) in Section 1.4):

- different lexico-grammatical realisations of instructions
- explicit vs. implicit side-effects

When side-effects are not included, this means that all the sentences in the text are *instructions*. When side-effects are explicitly realised, it means that instructions are interleaved with other style of text, namely *descriptions*.

Instructions expressing processes which have the user as the agent can be realised in the indicative mood in active or in passive voice. Passive voice is the case when reference to the user (or agent in general) is being "avoided". We introduce the distinction between *non-personal* and *personal* style to reflect this choice of hiding or expressing the intentional agent of an action.

Non-personal style means realising instructions in such a way that the addressee (user in our case) is omitted, any reference to him/her whatsoever is avoided. This means using indicative mood in passive voice or an infinitive (in Czech and Russian; not available in Bulgarian).

Personal style means realising instructions in such a way that the addressee (user in our case) is either explicitly mentioned, or implicit but directly implied. This means using indicative mood in active voice (including the cases of dropped Subjects) or imperative mood.

In personal style, i.e. when realising processes which have the user as the actor in indicative mood in active voice or in imperative mood, the person and number have to be selected. They can be seen as conditioned by two factors:

- whether the speaker is included among the addressees
- the social relationship between the speaker and the addressee(s)
- the actual number of addressees

If the speaker is included, the first person is chosen, otherwise the second person. In our kind of instructional texts, it does not make sense that the writer would address the text only to herself, so the number in case of speaker-included is always plural. In case of speaker excluded, we have a choice between singular and plural. Qua the actual number of addressees, it is natural to assume that there is only one reader at a time, one user of the system. However, the resulting grammatical number does not have to be, and mostly is not, singular in the AGILE languages. The reason is the distinction between polite (formal) and familiar (informal) form of address. The polite second person coincides with the

plural number, the familiar one coincides with singular. It seems that in book-style for adults addressing is always polite.[12]

Other interesting text structuring variations –e.g. the complexity of linguistic expressions, the order of presenting various bits of information, the level of detail of expressing other parts of the content than side-effects, the generation of various anaphoric forms, and different ways of expressing rhetoric relations— are not covered systematically in the intermediate prototype.

For the purposes of presenting and explaining the text style alternations we are going to cover in the intermediate prototype, we created an abbreviated text by extraction from Text 1 for the intermediate prototype (abbreviated as IMD-Text1). We shall illustrate the text style variations using this shorter text in the next sections.

3.1 Explicit vs. Implicit Side-Effects

First, we present the two essential versions, with and without explicit side-effect, below. The text pieces where there is a difference are highlighted.

Let us remind that the purpose of including this variation is primarily in demonstrating linguistic flexibility. The ultimate responsibility for content determination lies with the author in building the A-box. The system can only express information present in the A-box or inferable from it.

[12] Instructional texts intended for children or for peers in an informal group usually employ the familiar form of address in second person singular.

English:

EnA. *English, version with explicit side-effect:*

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. The Element Properties dialog box appears.
2. Enter the offset of the multiline element.

EnB. *English, version without explicit side-effect:*

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. Enter the offset of the multiline element in the Element Properties dialog box.

The “direct” translations of these two source versions for each of our three languages, showing the explicit vs. implicit side-effects variation, are presented below. We use the polite form of the imperative in all cases.

Czech:

CzA. *Czech, version with explicit side-effect:*

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 mlutičáry.

DOS a UNIX Z menu Data vyberte Styl multičáry.

1. Vyberte Vlastnosti prvků pro přidání elementů ke stylu.
   Objeví se dialogový panel Vlastnosti prvků.
2. Zadejte posunutí elementu multičáry.

CzB. *Czech, version without explicit side-effect:*

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 multičáry.

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 posunutí elementu multičáry.

Russian:

RuA. *Russian, version with explicit side-effect:*

Чтобы создать стиль мультилинии

Сначала откройте диалоговое окно Multiline Styles odnim iz sledujushchich metodov:

Windows в панели инструментов Object Properties или в меню Data выберите пункт Multiline Style.

DOS ili Unix в меню Data выберите пункт Multiline Style.

1. Нажмите кнопку Element Properties, чтобы добавить элементы в стиль.
   Na ekrane pojavitsja dialogovoje okno Element Properties.
2. введите смещение первого элемента линии.

RuB. *Russian, version without explicit side-effect:*

Чтобы создать стиль мультилинии

Сначала откройте диалоговое окно Multiline Styles odnim iz sledujushchich metodov:

Windows в панели инструментов Object Properties или в меню Data выберите пункт Multiline Style.

DOS ili 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.

Bulgarian:

BuA. *Bulgarian version with explicit side-effect:*

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. Диалоговият прозорец Element Properties се появява на екрана.
2. Въведете отместването на мултилинията.

BuB. *Bulgarian version without explicit side-effect:*

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 мултилинията.

3.2 Text Style Alternations in Instructions

In the three sections to follow, we discuss these alternations for each language:

1. What alternative ways if any can be used to express the heading? Is there any interdependency with the style in which the instruction-steps are expressed?
2. What alternative ways can be used to express the instruction-steps in personal and non-personal style? The style needs to be the same throughout the text. It emerges that the fact of expressing or not expressing the side-effects does not affect the realisations, so it suffices to show the alternations for one version. Let us choose the version without explicit side-effects, that is version B.

3.2.1 Alternative Instructions in Czech

In Czech, the alternations available for expressing instruction-steps can be summarised as follows:

a. Personal + active, indicative

b. Personal + imperative

c. Non-personal + reflexive passive, indicative

d. Non-personal + infinitive

3.2.1.1 Text Style Alternations in Headings

As for the headings, the nominal group realisation is the most common and natural. It can be either a nominalisation (rank shift) or a "true" nominal group. This way of realising headings is universal, it does not matter what style is used to express the instruction steps.

In the MS Word manual, we encountered an alternative realisation of headings in the form of "how-questions", i.e. *Jak vytvoříme styl multičáry (How we create a multiline style)*. In this way of realising the headings, it is possible to reflect the text style options used in the instruction-steps to some extent. We enumerate the features for heading realisation appropriate for the individual text styles below, and illustrate each of them by an example:

a. Personal + active indicative: active interrogative, same person and number
Jak vytvoříte styl multičáry (How you create a multiline style).

b. Personal + imperative: infinitive interrogative
Jak vytvořit styl multičáry (How to create a multiline style).

c. Non-personal + reflexive passive indicative: reflexive passive interrogative
Jak se vytvoří styl multičáry (How is a multiline style created).

d. Non-personal + infinitive: infinitive interrogative
Jak vytvořit styl multičáry (How to create a multiline style). (same as b)

It would in principle also be possible to use a subordinate purpose clause to express the headings, .e.g. *Abychom vytvořili styl multičáry (For us to create a multiline style)* Unlike in English, this clause is a conditional. It can again distinguish active and passive voice. Imperative or infinitive are not possible. Again, the features used in the heading should correspond to those used in the instruction-steps, as shown below:

a. Personal + active indicative: active interrogative, same person and number
Abyste vytvořili styl multičáry (To create a multiline style).

b. Personal + imperative: no corresponding heading realisation

c. Non-personal + reflexive passive indicative: reflexive passive interrogative
Aby se vytvořil styl multičáry (For a multiline style to be created).

d. Non-personal + infinitive: no corresponding heading realisation

This means that such heading realisation is not universal. It can only be used for the styles (a) and (c). This way of realising headings is not common anyway in Czech. This is probably because we prefer headings which can stand on their own, like nominal groups or independent clauses. A dependent purpose clause cannot stand alone.

For the purpose of the intermediate prototype we only consider one heading style, namely the realisation by a **nominal group**, as the most common and universal style.

3.2.1.2 Text Style Alternations in Instruction Steps

Besides the alternations in mood and voice listed above, we have also the possibility of varying person. We will not show all the alternations. To avoid blowing the number of illustrated possibilities, we choose the **second person plural** (corresponding to **polite form**) in all cases. It could always be other persons instead, as discussed elsewhere.

Also, we could in principle vary the realisation of reference to the user, but in our kind of text it does not make much sense. So, wherever the Subject corresponds to the user, we consider it as pronominal, and we drop it.

Below we shown the text style alternations for the short text extracted from IMD-Text1, all in the version without explicit side-effects.

CzB-a. Personal + active indicative

Vytvoření stylu multičáry

Nejdříve otevřete dialogový panel Styly multičár jednou z následujících metod:

Windows Z nástrojového panelu Vlastnosti objektů nebo z menu Data vyberete Styl multičáry.

DOS a UNIX Z menu Data vyberete Styl multičáry.

1. Vyberete Vlastnosti prvků pro přidání elementů ke stylu.
2. V dialogovém panelu Vlastnosti prvků zadáte posunutí elementu multičáry.

CzB-b. Personal + imperative

Vytvoření stylu multičáry

Nejdříve otevřete dialogový panel Styly multičár jednou z následujících metod:[13]

Windows Z nástrojového panelu Vlastnosti objektů nebo z menu Data vyberte Styl multičáry.

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 posunutí elementu multičáry.

CzB-c. Non-personal + reflexive passive, indicative

Vytvoření stylu multičáry

Nejdříve se otevře dialogový panel Styly multičár jednou z následujících metod:[14]

Windows Z nástrojového panelu Vlastnosti objektů nebo z menu Data se vybere Styl multičáry.

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 posunutí elementu multičáry.

[13] Incidentally, the indicative and imperative second person plural forms of the verb otevřít (open) are the same, homonymous.

[14] Incidentally, the indicative and imperative second person plural forms of the verb otevřít (open) are the same, homonymous.

CzB-d. Non-personal + infinitive

Vytvoření stylu multičáry

Nejdříve se otevřít dialogový panel Styly multičár jednou z následujících metod:[15]

Windows Z nástrojového panelu Vlastnosti objektů nebo z menu Data se vybrat Styl multičáry.

DOS a UNIX Z menu Data se vybrat Styl multičáry.

1. Vybrat Vlastnosti prvků pro přidání elementů ke stylu.
2. V dialogovém panelu Vlastnosti prvků zadat posunutí elementu multičáry.

3.2.2 Alternative Instructions in Russian

3.2.2.1 Text Style Alternations in Headings

Similarly to Czech, headings in Russian software instructions can be expressed in variety of ways. Translated manuals (our AutoCAD excerpts belong to this category) tend to follow the original English headings:

(27) Чтобы создать стиль мультилинии (To create a multiline style)

A subtle difference between English and Russian consists in explicitness of the purpose in the Russian heading, while the English heading describes a proposal.

Another common pattern evidenced by the English usage is “how-statements”:

(28) Как создать стиль мультилинии (How to create a multiline style)

Other options involve nominalisation of the process in order to express the heading as a label:

(29) Создание стиля мультилинии (Creating a multiline style)

or as a purpose:

(30) Для создания стиля мультилинии (For creation of a multiline style)

3.2.2.2 Text Style Alterations in Instruction Steps

Besides the alterations in mood and voice listed above, we have also the possibility of varying person. The space of possibilities is similar to that discussed above for Czech. We will not show all the alternations. We choose the **second person plural** (corresponding to **polite form**) in all cases. It could always be other persons instead, as discussed elsewhere.

[15] Incidentally, the indicative and imperative second person plural forms of the verb otevřít (open) are the same, homonymous.

Another option involves changes in the rhetorical organisation of instruction. Personal instructions, which are realised by imperative clauses, express commands to be followed by the user, so the text is developed as a sequence of operations to be performed to achieve a task. Non-personal statements express options available in a particular screen object of software, so the text is developed as a sequence of descriptions for screen objects (menu items, controls in dialog boxes, etc). Though in some cases, both personal and impersonal styles combine, for example, when a sequence of commands is interrupted by a reference to a side effect. Any of these options are subjected to a minor modulation caused by difference between a dichotomy of complex procedures vs. simple actions, as reflected at the level of language, not only in the DM. The complete procedure (and its first step) or a reference to repetition of steps are considered complex, while mouse clicking or key pressing are simpler actions. The sixth step in the IMD-Text1 (p.47):

(31)

(a) Repeat these steps to define another element.

can be rendered in Russian with the clauses in the reverse order:

(b) Чтобы определить еще один элемент повторите эти шаги,

while generally a sequence of simple-action, its purpose, is routinely copied in Russian text. Another example of this difference concerns commands expressed in infinitival mode. For complex procedures it requires a predicative modal adverbial (*нужно,* it is necessary) or an impersonal verb form (*следует*, you should):

(32)

(a) Сначала нужно открыть диалоговое окно Multiline Styles одним из следующих способов:
(First open the Multiline Styles dialog box using one of these methods)

(b) чтобы задать еще один элемент, следует повторить эти шаги.
(Repeat these steps to define another element.)

So, we can model two styles - imperative and infinitive modal impersonal, which are similar. Other style variations are simple vs. complex sentences and implicit vs. explicit realisation of the side-effects. Other style variations are variations inside the sentence: word order (differentiating, in particular, Russian and English styles), nominalisations.

In the following two boxes we show the two alternative available in Russian. The numbering of the alternatives follows the one in the Czech version.

RuB-b. Personal + imperative

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.

RuB-c. Non-personal + middle passive, indicative [16]

Создание стиля мультилинии

Элементы стиля вводятся в диалоговом окне Multiline Styles, которое открывается нажатием кнопки Multiline Style в панели инструментов Object Properties или в меню Data.

Сначала вводится смещение первого элемента линии и этот элемент добавляется нажатием кнопки Add.

Цвет элемента выбирается в диалоговом окне Select Color, которое открывается нажатием кнопки Color.

Тип линии элемента выбирается в диалоговом окне Select Linetype, которое открывается нажатием кнопки Linetype.

Нажатием кнопки OK сохраняется стиль всех заданных элементов мультилинии и закрывается диалоговое окно Element Properties.

RuB-d. Non-personal + infinitive

Чтобы создать стиль мультилинии следует (необходимо) выполнить следующие действия:

Открыть диалоговое окно Multiline Styles. Для этого выбрать выбрать пункт Multiline Style в панели инструментов Object Properties или в меню Data.

Затем нажать кнопку Element Properties, чтобы добавить элементы в стиль.

Для этого

[16] Though this style is more traditional for original Russian manuals, but it differs significantly from the two former styles, so we can't model it as a style variation in our system now. It is more coherent and generalising, and has a more complicated communicative organisation

(2) в диалоговом окне Element Properties ввести смещение первого элемента линии и нажать кнопку Add, чтобы добавить этот элемент.

(4) Выбрать пункт Color. Затем выбрать цвет элемента в диалоговом окне Select Color.

(5) Выбрать пункт Linetype. Затем выбрать тип линии элемента в диалоговом окне Select Linetype.

Повторить эти шаги, чтобы задать еще один элемент.

Нажать кнопку OK, чтобы сохранить стиль элементов мультилинии и закрыть диалоговое окно Element Properties.

3.2.3 Alternative Instructions in Bulgarian

The corpus analysis leads to the conclusion that two basic styles comparable to those discussed in detail for Czech are used in Bulgarian instructional texts:

- *Personal + imperative:* Изберете Color. (Choose Color).
- *Non-personal + indicative in passive voice:*
  Въвежда се отместването на мултилинията
  (The multiline offset is entered).

In Bulgarian the reflexive particle *ce (se)* has two functions: (i) it can express reflexivity; (ii) it is a formal marker of non-personal passive constructions. This leads to certain formal confusion. The following examples illustrate the related phenomena:

(33) **Reflexive verb**:
Диалоговият прозорец се появява на екрана.
(The dialog box appears on the screen).

The verb появявам се is a reflexive verb. Semantically the emphasis is on the process, not on the agent.

(34) **Non-personal, passive with reflexive particle**
Бутонът се натиска, за да завърши въвеждането.
(The button is pressed to end the input process.)

where the agent is not relevant, we emphasise on the process. Such realisations are usual for non-personal procedural texts.

(35) **Personal, passive (not reflexive)**
Бутонът е натиснат (от потребителя)
(The button is pressed (by the user).)

Such realisations appear also in informative texts. This example is included here to illustrate the use of the passive voice in personal style, opposite to the non-personal use of passive illustrated in (34) [Popov 1962]. In this case the agent is the user, but it is implicit. It is possible to mark the agent explicitly as in *Бутонът е натиснат от потребителя*, but this is not necessary as the explicit mentioning the agent doesn't give us more information. In allowing including an

explicit agent in a passive construction, Bulgarian and Russian are alike, while Czech does not allow such realisation (cf. [AGILE 6.2, Section 2.2]).

The non-personal style with active verb form usually occurs in the side-effect of the procedures. The non-personal style with passive is very typical for original instructional texts in Bulgarian. The personal passive does not occur in instructional texts.

Personal style in indicative mood as described in Section 3.2.1 for Czech is also potentially possible in Bulgarian, but it may be applied only in oral lecture-style communications, never in written manuals.

3.2.3.1 Text Style Alternations in Headings

Practically all the headings in the analysed instructional texts are represented by nominal groups. Only in isolated cases in the translations, question phrases ("how-questions") are used for headings.

3.2.3.2 Text Style Alternations in Instruction Steps

Besides the alternations in mood and voice discussed above, we have also the possibility of varying person. The **second person plural** (corresponding to **polite form**) is used practically in all cases. Another possibility is to apply second person singular, corresponding to more informal use. It is considered appropriate for oral communication, but not for instructional texts (apart from some special child-oriented texts).

The following two short examples extracted from IMD-Text1 illustrate the alternations considered appropriate.

BuB-b. Personal + imperative

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 мултилинията.

BuB-c. Non-personal, indicative

Създаване стил на мултилиния

Първо се отваря диалоговия 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 мултилинията.

3.3 SPL Components Reflecting Text Style Options

In the preceding section, we explained and illustrated a set of alternative realisations for personal vs. non-personal style of headings and instruction steps to be covered in the intermediate prototype. We also hinted at the related mood and voice options which can ensure the generation of the intended realisations. Now we would like to show how these distinctions can be reflected in SPLs, so that our tactical generators can produce the appropriate output.

3.3.1 Realisation of Headings

3.3.1.1 Headings in Czech and Bulgarian

The realisation of headings by a **nominal group** as the most common and universal style. For Czech and Bulgarian, we consider only this one heading style for the purpose of the intermediate prototype. For Russian, we consider also the realisation by a partial purpose clause.

The nominal group headings in our texts are most frequently constructed as nominalisation of a verb and its direct complement. In KPML terms, the result of such nominalisation is “nominalisation-noun” The SPL component leading to the heading realised by a nominal group is the specification that the required group does not express a speech act.

```
:EXIST-SPEECH-ACT-Q  NOSPEECHACT
```

It is arranged by the grammar networks nominalisation to be generated in this case. The SPL component, which links the noun feature with its grammatical and semantic role explicitly is:

```
:PROCESSUAL-Q            PROCESSUAL
```

The nominalised noun is usually used together with an object, which corresponds to the direct complement of the verb before nominalisation, which is derived from the actee of the process corresponding to the nominalised verb.

3.3.1.2 Headings in Russian

Russian, like Czech and Bulgarian, can use heading realised by a nominal group. However, the headings in our AutoCAD texts follow the English pattern, and are realised as partial purpose clauses. We can achieve this using the following SPL statements (this semantics does not express purpose of this action, but still the lexico-grammar generates the correct output):

Чтобы создать мультилинию

```
(S / DISPOSITIVE-MATERIAL-ACTION :LEX SOZDATJ
   :PROPOSAL-Q PROPOSAL
   :REPORT-Q REPORT :ENTIRENESS-Q PARTIAL :ACTEE
   (D / OBJECT :LEX MULJTILINIJA))
```

3.3.2 Realisation of Instruction Steps

The first distinction we make for the text style of the instruction-steps is between **personal** and **non-personal**, each of which may further subdivide into two possibilities, depending on the language, as discussed above (Sections 2.3 and 3.2). This determines the choice of mood, voice and form of address. We now present the relevant SPL bundles. In the implementation of the text structuring module, we will restrict express the restrictions on the available combinations of choices in general and for each individual language.

3.3.2.1 Mood

Indicative mood

Indicative mood is the default. We can nevertheless make it explicit as follows:

```
:SPEECHACT ASSERTION
```

This is a shorthand enabled by the following SPL-macro code:

```
:speechact (sa / ASSERTION :polarity Positive)
```

We can see that a negative sentence would require a different SPL declaration:

```
:SPEECHACT DENIAL
```

which again uses an SPL-macro code:

```
:speechact (sa / ASSERTION :polarity Negative)
```

Imperative mood

```
:SPEECHACT IMPERATIVE
```

This is enabled by the following macro:

```
(imperative:speech-act-id (?sa / Command :polarity
Positive))
```

If we want to do it without the macro, the SPL should contain the following:

```
:speechact (sa / COMMAND :polarity Positive)
```

A negative imperative we can get by the following:

```
:speech-act-id COMMAND :polarity Negative
```

Another way to specify imperative in the SPL is the following, and we can add the positive or negative declarations as well:

```
:COMMAND-Q COMMAND
```

Infinitive mood

The original grammar does not support infinitive as a possibility for an independent clause simplex. A possible solution is to include Infinitive as another realisation of the speech act Command. In Czech, we can issue a command in the infinitive. e.g. *Stát!* (*Stop*). So we can expect an SPL bundle of the following form:

```
:speechact INFINITIVE
```

In the infinitive mood in Russian, a requirement speech act can be used, expressed with *следует* (*one should*):

```
:speechact REQUIREMENT
```

3.3.2.2 Voice

In as not possible to determine diathesis, in particular voice, using the original grammar and SPL implementations. We will probably use an SPL macro which will take active, passive and medio-passive (reflexive) as values and set the appropriate grammatical features.

Active voice

Since active voice is the default, we do not have to specify it in the SPL.

Medio-passive (reflexive passive) voice

The SPL line below could be used to cause the choice under the AGENCY system to be made in the right way, namely, to choose MIDDLE instead of EFFECTIVE verb:

```
:CAUSED-PROCESS-Q INDEPENDENT
```

This statement leads to conflation of MEDIUM and SUBJECT, which is our aim. In addition, we have to ensure that the reflexive verb form (Russian) or particle (Czech, Bulgarian) are generated. As all processes in our intermediate prototype texts are “not-verbal”, only the constraint that the process should be realised by “not-verbal” verb holds.

3.3.3 Form of Address

Each of the options for personal style can be varied for person and number. Another dimension of classification is into polite and familiar form of address in second person. As stated earlier, the polite form implies plural number.

Politeness

Polite vs. familiar form of address is a choice relevant for the second person, and it is a decision of polite vs. familiar style of addressing the readership. It determines the grammatical number used, which then does not really depend on whether there is actually one addressee or more.

Once the speech act is fixed and the default IMPERATIVE-INTERACTANT gives the verb "second person", under the system IMPERATIVE-MOOD-TYPE a choice should be made between polite and familiar style, which is the choice between plural and singular number of the verb form.

However, the politeness feature has not been included in the original grammars, so including it as such requires the implementation of a new system, which would link the polite with plural number, and familiar with singular number (all for second person).[17] Once this is done, we can specify the politeness feature in the SPL by something like the following:

```
:Politeness-Q Polite or :Politeness-Q Familiar
```

As stated earlier, the polite form is the default in written instructions for general adult public in our languages. Therefore, it possible to specify POLITE as default. In this case the choice would not have to be included in every SPL.

Number

The plural number results from the following specification:

```
:MULTIPLICITY-Q MULTIPLE
```

Person

In our texts, when choosing the person in personal style, we have to determine whether first or second person should be used. This can be achieved by the following SPL declarations:

```
:MEMBER-SET-Q INCLUDED

:MEMBER-SET-Q NOTINCLUDED
:MEMBER-SET-QQ INCLUDED
```

The first declaration means that the speaker is included, and therefore the realisation will be in first person. The second couple of declarations means that speaker is excluded and hearer is included, which together leads to realisation in second person.

We have to modify these systems in the grammar so that the person and number are treated in full detail, because there aree differences in the AGILE languages which are not present in English.

4. Text Structuring for the Intermediate Prototype

In this section we discuss how we intend to specify the structure of a text to be generated by the intermediate prototype. We begin by making a distinction between *text structure elements'* being the elements from which a (task-oriented) text is built up, and *text templates*, which condition the way text structure elements are to be linguistically realised (Section 4.1). Subsequently, we focus on the relation between T-box concepts on the one hand, and text structure elements

[17] Implementing such a system is part of Task 7.2 concerned with the implementation of tactical generators for the intermediate prototype.

on the other hand. We are specifically interested in those T-box concepts that are used to configure the content specified in an A-box, like the PROCEDURE and METHOD* concepts. This is because we want to have a close connection between how content can be defined in an A-box and how that content is to be spelled out in a text using a particular organisation, so as to ensure that the intended content is indeed reflected by the text (Section 4.2).

4.1 Text Templates and Text Structure Elements

In the introduction to the present report we already alluded to the idea that texts may vary in their distribution and lexico-grammatical realisation of content. The corpus investigations we carried out for the languages under study in AGILE provided empirical support for the view that the observable alternations can be classified into two main groups, as follows:

- *Choice of grammatical means*: whether lexico-grammatical choices are predetermined in certain ways by the style of text.
- *Distribution of content and lexical realisation*: whether information is realised explicitly in the most straightforward way with respect to the A-box, so that there is only a minimal inference load required to interpret a piece of text, or whether information is left more or less implicit. And, what the realisation is of the explicitly conveyed information.

Here we propose two notions that should enable us to capture these differences in a principled fashion. These two notions are the notion of **text structure element** and the notion of **text template**.

A text structure element is a predefined component that is to be filled by one or more particular parts of the user's definition of the content to be spelled out by the text. Using the reader-oriented terminology common in technical authoring guides, we will thus distinguish text structure elements like TASK-INSTRUCTIONS, SUBTASKS, etc. (for a full listing of the text structure elements we use for the intermediate prototype see Section 4.1.1).

Orthogonal to the notion of text structure element is the notion of text template. Whereas text structure elements capture *what* needs to be realised, the idea of a text template captures *how* that content needs to be realised. Thus, a template defines a *style* in which content is to be realised.

There are choices, though, that do concern realisation but which are relatively independent of the style in which a text is to be generated. We will call these *general realisation constraints.* A good example of such a choice is whether side effects, like the appearance of a menu, should be explicitly mentioned or whether they can be left more or less implicit (cf. Section 3.1). Thus:

(36) ***Explicit side effects:***

(a) *The* File *menu appears.*

(37) ***Implicit side effects:***

(a) *Select* Save *in the* File *menu.*

(b) *Subsequently we select* Save *in the* File *menu.*

In the next section we provide a formal definition of text structure elements distinguished in AGILE.

4.1.1 Text Structure Elements

Our definition of text structure elements differs slightly from the definitions used in the previous deliverables of Work Package 5 (TEXS1/TEXM1), as reported in [AGILE 5.1]. The reason for this difference is that the underlying domain model has changed. We first present the text structure elements, then the T-box concepts.

TASK-DOCUMENT

A task-document has two slots:

- TASK-TITLE (obligatory)
- TASK-INSTRUCTIONS (obligatory)

where "TASK-INSTRUCTIONS" means at least one INSTRUCTION.

INSTRUCTION

An instruction has three slots:

- TASKS (obligatory)
- CONSTRAINT (optional)
- PRECONDITION (optional)

where "TASKS" means at least one TASK.

TASK

A task has two slots:

- INSTRUCTIONS (obligatory)
- SIDE-EFFECT (optional)

In comparison to the definitions in TEXS1 the above proves to be simpler, which is due to the recursive nature of the T-Box concepts defined in the (revised) domain model for the intermediate prototype. The AGILE domain

model, as previously described in [AGILE 2.1], has undergone various changes in order to

1. provide a more uniform, recursive scheme for modelling text structure (in comparison to the previous proposal), and to
2. cover the text structures as found in the corpora studied in the context of this deliverable (see the earlier section).

We now employ just four concepts to define the admissible A-box configurations, instead of the seven concepts used earlier. These configurations provide the structured contexts in which actions and events take place. These concepts are PROCEDURE, METHOD*, PROCEDURE-LIST and METHOD-LIST. We define them as follows[18]:

PROCEDURE

A procedure has three slots:

- GOAL (obligatory, to be filled by a USER-ACTION)
- METHODS (optional, to be filled by a METHOD-LIST)
- SIDE-EFFECT (optional, to be filled by a USER-EVENT)

METHOD

A method has three slots:

- CONSTRAINT (optional, to be filled by an OPERATING-SYSTEM)
- PRECONDITION (optional, to be filled by a PROCEDURE)
- SUBSTEPS (obligatory, to be filled by a PROCEDURE-LIST)

METHOD-LIST

A Method-List is essentially a list of METHOD*s:

- FIRST (obligatory, of type METHOD*)
- REST (optional, of type METHOD-LIST)

PROCEDURE-LIST

A Procedure-List is essentially a list of PROCEDUREs:

- FIRST (obligatory, of type PROCEDURE)
- REST (optional, of type PROCEDURE-LIST)

---

[18] The asterisk on METHOD* prevents a clash with the LISP symbol Method.

In comparison to the previous set of concepts, which were modelled on the basis of the DRAFTER-2 project, we no longer have a distinction between PROCEDURE and COMPLEX-PROCEDURE, or METHOD and COMPLEX-METHOD (see also [AGILE 5.1]). This simplifies the T-box at the cost of complicating the A-box, in the sense that more entities are sometimes needed in order to model an instruction.

For instance, a simple goal-subgoal relationship like `Choose Save to save the drawing' requires the following A-box structure:

```
procedure
GOAL `save the drawing'
METHODS method-list
        FIRST method
              SUBSTEPS procedure-list
                       FIRST procedure
                             GOAL `choose Save'
```

Furthermore, the new attribute CONSTRAINT on a method enables us to make a distinction between methods applicable under Windows and methods applicable under DOS or UNIX, as in the following example from IMD- Text 1:

(38) 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.

4.1.2 Text Templates and General Conditions on Realisation

At this point, let us recapitulate our discussion so far. Above we saw that text structure elements provide a more fine-grained organisation of what content needs to be realised. More precisely, they determine distribution of content in an abstract sense. Furthermore, in Section 4.1 and in Section 1.1 we already alluded to the viewpoint that we can use text templates to prescribe the choice of grammatical means in the realisation of content, thus determining the style of the text. In order to do so templates may require the introduction of additional discourse markers to make text structure explicit (for example, as in **informative** parts of text). Finally, independent of a particular text template there may be choices influencing the level of explicitness in informing the reader –for example, on observable side effects, as we saw earlier.

In Section 2 on corpus investigation we indicated that among the features attended to in the investigation were the following:

- Expressing or hiding the intentional agent of an action (agency, voice)
- Ways of expressing the readership (person, number, voice)
- Mode of realising instructions (mood, voice)

- The complexity of linguistic expressions (group and clause complexity, rank shifting)

These aspects were dealt with in a more principled way to specify the variation of text style for the intermediate prototype in Section 3. We discussed there two different styles in which instructions can usually be realised, being the *personal* or the *non-personal* style. More specifically, we distinguished an instruction's heading and its style of realisation from the instruction's steps and their realisation in a particular style. We also explained how there seems to be an interdependency between the style in which the instruction steps are generated and the style that seems appropriate for the heading.

The purpose of a text template is to relate these choices of style to the actual elements from which our text structures are composed – the text structure elements. In other words, a text template should specify the following aspects:

- **Realisation of instruction heading** – constraints that are to be imposed on the TASK-TITLE of a TASK-DOCUMENT. As we already pointed out in Section 3.2, for the purpose of the intermediate prototype we will restrict ourselves to realising headings as nominal groups.
- **Realisation of instruction steps** – constraints that are to be imposed on the TASK-INSTRUCTIONS of a TASK-DOCUMENT. These constraints regard
    - *Style*:
        - *Personal*: Realise TASK-INSTRUCTIONS, meaning TASKS and INSTRUCTIONS,
            - Using indicative mood (possibly including dropped subjects), or imperative mood.
            - Using a polite form of address (second person plural) or a familiar way of addressing the reader.
        - *Non-Personal*: Realise TASK-INSTRUCTIONS, thus TASKS and INSTRUCTIONS, in indicative mood in passive voice or using an infinitive.

Recall that in Section 3.3 we already specified the statements to be put into an SPL in order to reflect the various aspects of a particular style. Using these statements, a text template can impose constraints on the realisation of a text structure element by means of KPML's REALIZE-WITH statement. For example, if we want to see the TASK-TITLE realised as a nominal group, then we can use

```
(REALIZE-WITH TASK-TITLE
(:EXIST-SPEECH-ACT-Q NOSPEECHACT))
```

with ":EXIST-SPEECH-ACT-Q NOSPEECHACT" being the SPL component responsible for the realisation of the TASK-TITLE as a nominal group (cf. 3.3.1).

In addition to constraints on lexico-grammatical realisation, we can also specify in a template the *layout* of individual text structure elements. KPML provides us with the ANNOTATE function, which can be used to specify in an SPL that the constituent(s) realising a particular piece of the SPL's content

should be annotated with a particular set of tags. Following the practice of the previous Work Package 5 deliverables (TEXS1-Cz, TEXS1-Ru, TEXS1-Bu, cf. [AGILE 5.1]) we can use HTML tags to specify a particular layout, which can be viewed when an annotated set of sentences is loaded into, for example, a web browser.

For example, the following mapping between text structure elements and HTML tags could be used in a text template delineating a procedural style of instructional text. We give the layout as HTML tags with unspecified (i.e. default) values.

Text Structure Element	**Layout**	**HTML Tags**
TASK-TITLE	Heading level *N*	<H*N*> … </H*N*>
(TASK-) INSTRUCTIONS	Ordered list	<OL> … </OL>
INSTRUCTION	(Ordered) list item	<LI>
CONSTRAINT	Description list item	<DL> <DD>… </DL>
PRECONDITION	(Ordered) list item	<LI>
SIDE-EFFECT	Description list item	< DL> <DD>… </DL>
TASK	Text	

Figure 4 - Example of HTML-style specification of TSE layout

Finally, besides the constraints on realisation imposed by a text template we may have general constraints on realisation – as we often noted already. In the case of the intermediate prototype, an example of such a general constraint is the explicit or implicit realisation of a side effect.

To round off our discussion on text templates, let us make some remarks in regard to the conclusions provided at the end of [AGILE 5.1]. There, on page 25, we mentioned that we would concentrate (among other things) on a model enabling a more flexible generation of alternative forms and on those aspects of text planning necessary to decide whether a given content should be expressed by a sequence of simplex clauses or by a complex clause. It should be clear that the first point is conveniently captured by the idea of text templates as we defined it in this section. The second point is covered by the (template-independent) general constraints on realisation.

4.2 Mapping Between Text Structure Elements and T-Box Concepts

This mapping will enable us to provide a flexible definition of rules for text structuring of a content defined by an A-box.

For the intermediate prototype we employ the following mapping between T-Box concepts and Text Structure Elements.

- TASK-TITLE ↔ GOAL of topmost PROCEDURE
- TASK-INSTRUCTIONS ↔ METHODS of topmost PROCEDURE
- SIDE-EFFECT ↔ SIDE-EFFECT of a PROCEDURE
- TASK ↔ a PROCEDURE's GOAL
- CONSTRAINT ↔ CONSTRAINT of a METHOD
- PRECONDITION ↔ PRECONDITION of a METHOD
- INSTRUCTIONS ↔ METHODS
- INSTRUCTION ↔ METHOD

By way of an example, let us consider again the A-Box for *Choose Save to save the drawing*:

```
procedure
  GOAL `save the drawing'
  METHODS method-list
          FIRST method
                  SUBSTEPS procedure-list
                              FIRST procedure
                                  GOAL `choose Save'
```

This A-Box would translate into the following arrangement of text structure elements, given the mapping we provided above:

TASK-TITLE "Save the drawing"
TASK-INSTRUCTIONS
INSTRUCTION
TASK "Choose Save"

This could, using the layout given in Figure 4 above for example, result in the text shown in Figure 5:

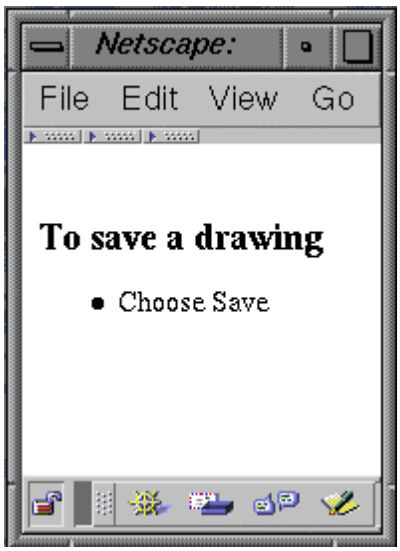


Figure 5 Example of generated text

4.3 Elaborate Example of Generating a Text Structure

We close this section with a more elaborate example of how we generate a text structure, given a content specified by an A-box. For this purpose, we use a shortened version of the A-box of the first text for the intermediate prototype. This A-box actually corresponds to the texts we used in Section 3 to illustrate possible style alternations in the context of the intermediate prototype.

Below we commence by providing the promised A-box. Our primary aim with the A-box is to provide a picture of a more complex configuration of content, using the concepts we discussed earlier in Section 4.1.1. To that end, in the A-box we give below we do omit some of the concepts that are irrelevant to our discussion (notably, some of the concepts that are used to set up list-like structures). Subsequently, given this complex configuration, we proceed by providing the corresponding text structure in terms of text structure elements, following the mapping of Section 4.2. Once we have this text structure, we provide the realisations corresponding to the structure and following a particular style. We end the example by showing the text in a possible layout, using the mapping to HTML tags as in Figure 4.

4.3.1 The A-Box

The A-box we given below in Figure 6 is a shortened version of the A-box for Text 1 for the intermediate prototype, in two respects. Firstly, its content corresponds to the text used in Section 4.1.1. Secondly, not all the concepts that can be found in the original A-box are represented here. As for the latter, whenever we omit some concepts we indicate their omission by means of "…". With regard to typesetting, we display configurational concepts using *italics*, and content concepts using **boldface**.

```
PROCEDURE
  GOAL:
    CREATE
    ACTEE:
    … STYLE
      OWNER:
        MULTILINE
    ACTOR:
    … USER
  METHODS:                  %% This is the way to specify lists of methods,
  METHOD-LIST               %% as we already defined earlier. In the A-box
  FIRST:                    %% hereafter we will omit the extra concepts
    METHOD*                 %% needed to define lists (METHOD-LIST, FIRST,
      PRECONDITION:         %% REST) in order to shorten the A-box.
      PROCEDURE
        GOAL:
          OPEN-SCREEN-OBJECT
          ACTEE:
            GUI-ACTEE
            RANGE:
              DIALOGUE-BOX
            LABEL:
              Multiline Styles
          ACTOR:
          … USER
        METHODS:            %% Eventually, the following part of the A-box
        … METHOD*           %% will be realised as
            CONSTRAINT:  %% Windows: From the Object Properties toolbar
              OPERATING-SYSTEM   %%or Data menu, choose Multl. Style"
                LABEL:       %%Observe the coordination involving
                  Windows    %%a disjunction- content wise, there are 2
            SUBSTEPS:        %% methods given for the Windows case. The
            … PROCEDURE      %% A-box part defines two METHODs, both of
              GOAL:             %% which have as CONSTRAINT the
                                %% OPERATING-SYSTEM
                CHOOSE      %% Windows. To the left of this comment is
                  ACTEE:        %% the first method.
                    DISPLAYED-ACTEE
                    RANGE:
                      OPTION
                    LABEL:
                      Multiline Style
                  OPTIONS:
                    GUI-ACTEE
                  RANGE:
                    TOOLBAR
                  LABEL:
                    Object Properties
                  ACTOR:
                  … USER
        … METHOD*           %% And here is the second part for Windows.
            CONSTRAINT:
              OPERATING-SYSTEM
                LABEL:
                  Windows
              SUBSTEPS:
              … PROCEDURE
                  GOAL:
                    CHOOSE
```

```
                              ACTEE:
                                 DISPLAYED-ACTEE
                                 RANGE:
                                    OPTION
                                 LABEL:
                                    Multiline Style
                              OPTIONS:
                                 GUI-ACTEE
                                 RANGE:
                                    MENU
                                 LABEL:
                                    Data
                              ACTOR:
                           … USER

                  METHOD*
                     CONSTRAINT:
                        OPERATING-SYSTEM
                        LABEL:
                           DOS and UNIX
                     SUBSTEPS:
                     … PROCEDURE
                        GOAL:
                           CHOOSE
                           ACTEE:
                              DISPLAYED-ACTEE
                              RANGE:
                                 OPTION
                              LABEL:
                                 Multiline Style
                           OPTIONS:
                              GUI-ACTEE
                           RANGE:
                              MENU
                              LABEL:
                                 Data
                           ACTOR:
                              … USER
SUBSTEPS:
… PROCEDURE
     GOAL:
        ADD-TO
        ACTEE:
        … ELEMENTS
        RECIPIENT:
        … STYLE
        ACTOR:
        … USER
     METHODS:
     METHOD*
       SUBSTEPS:
       …  PROCEDURE
          GOAL:
            CHOOSE
            ACTEE:
               DISPLAYED-ACTEE
               RANGE:
                  OPTION
```

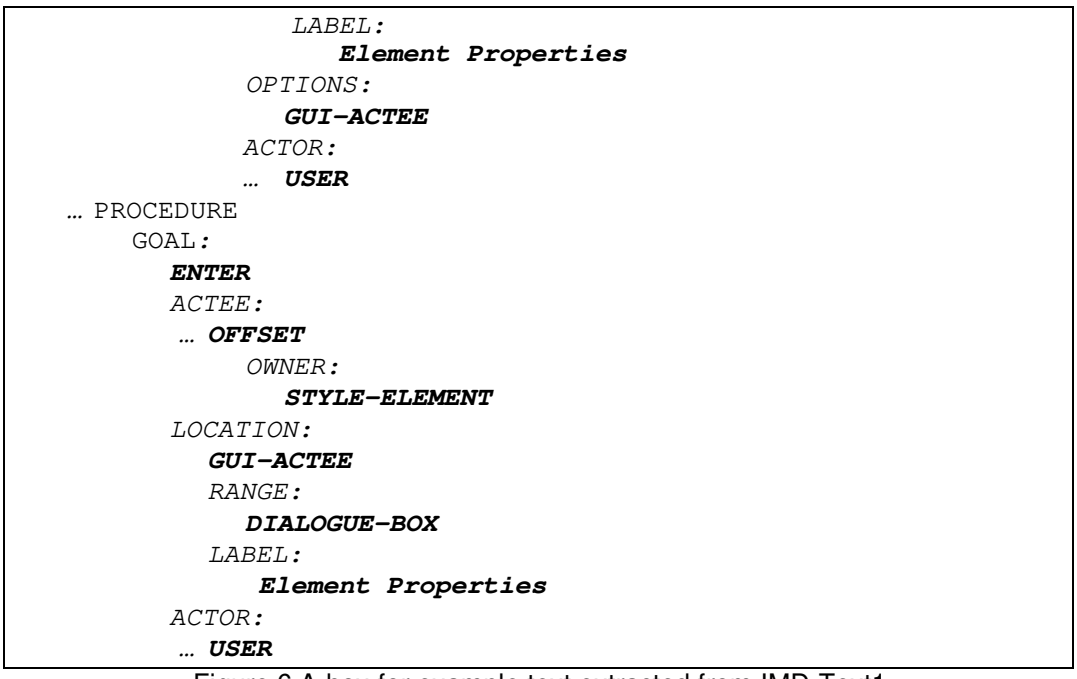
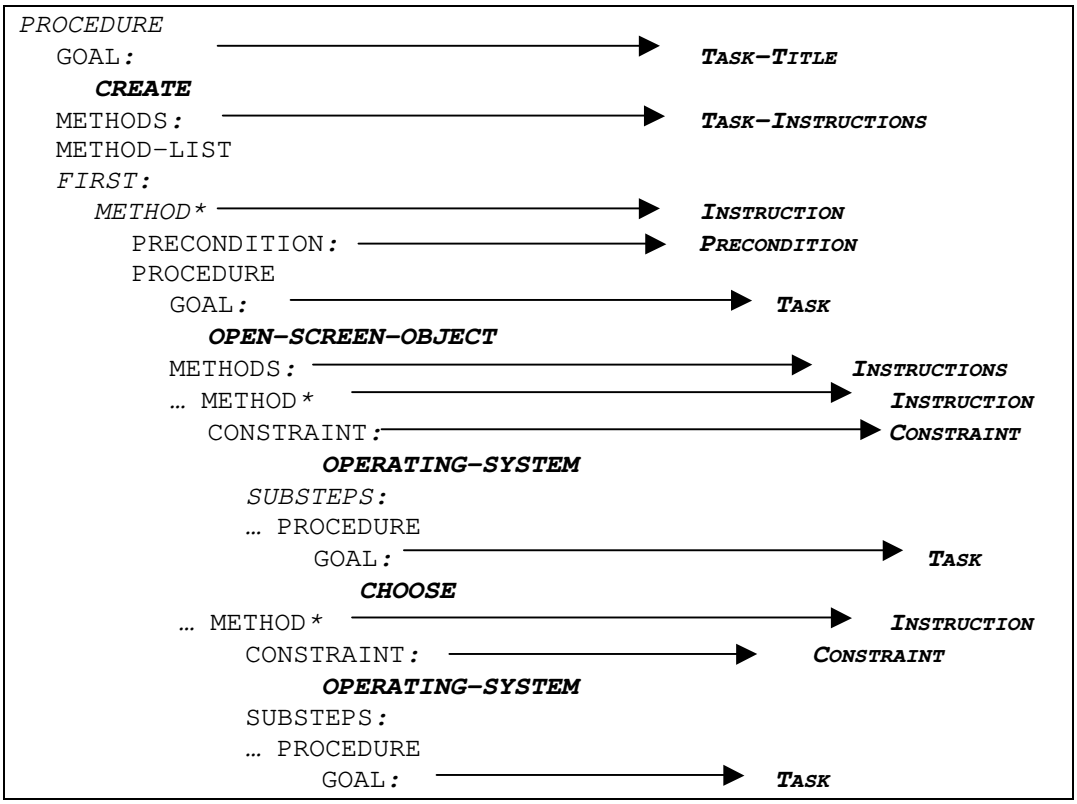


Figure 6 A-box for example text extracted from IMD-Text1

4.3.2 The Text Structure

Consider again the A-box above. Using the mapping we provided in Section 4.1.1 we can create a text structure, composed of text structure elements, as follows, see Figure 7. (Since the mapping only concerns the concepts used for configuring content in the A-box, we omit from the A-box below all concepts that are irrelevant to configuration.)



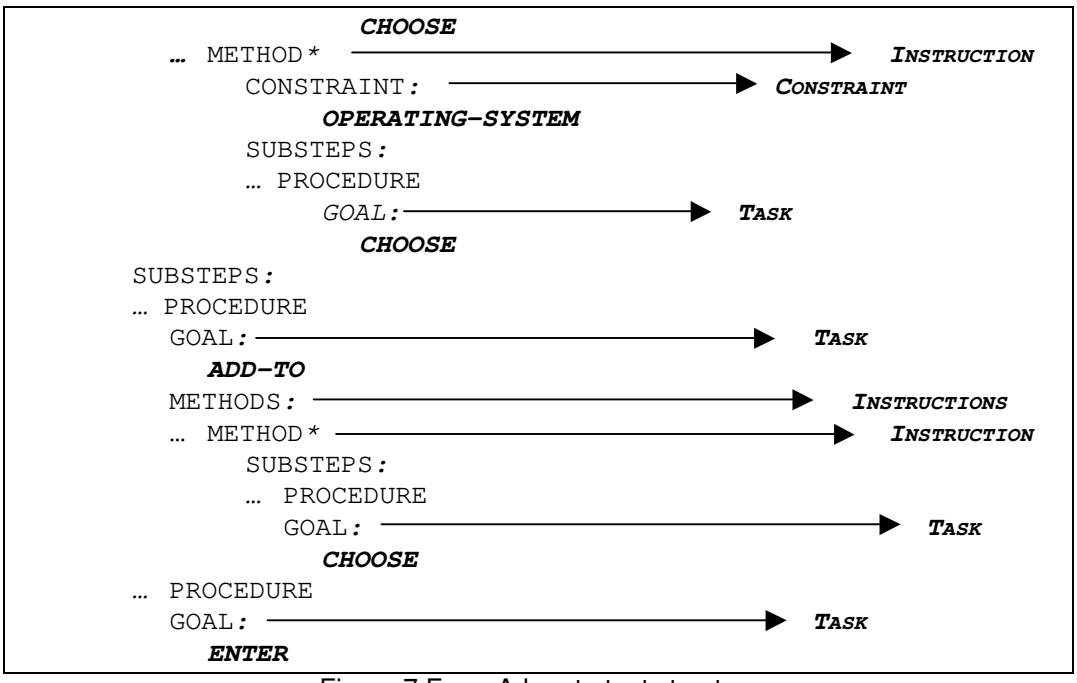


Figure 7 From A-box to text structure

4.3.3 Realisation and Layout

An example of the realisation that results from the text structuring process discussed above is shown below for English and Czech.

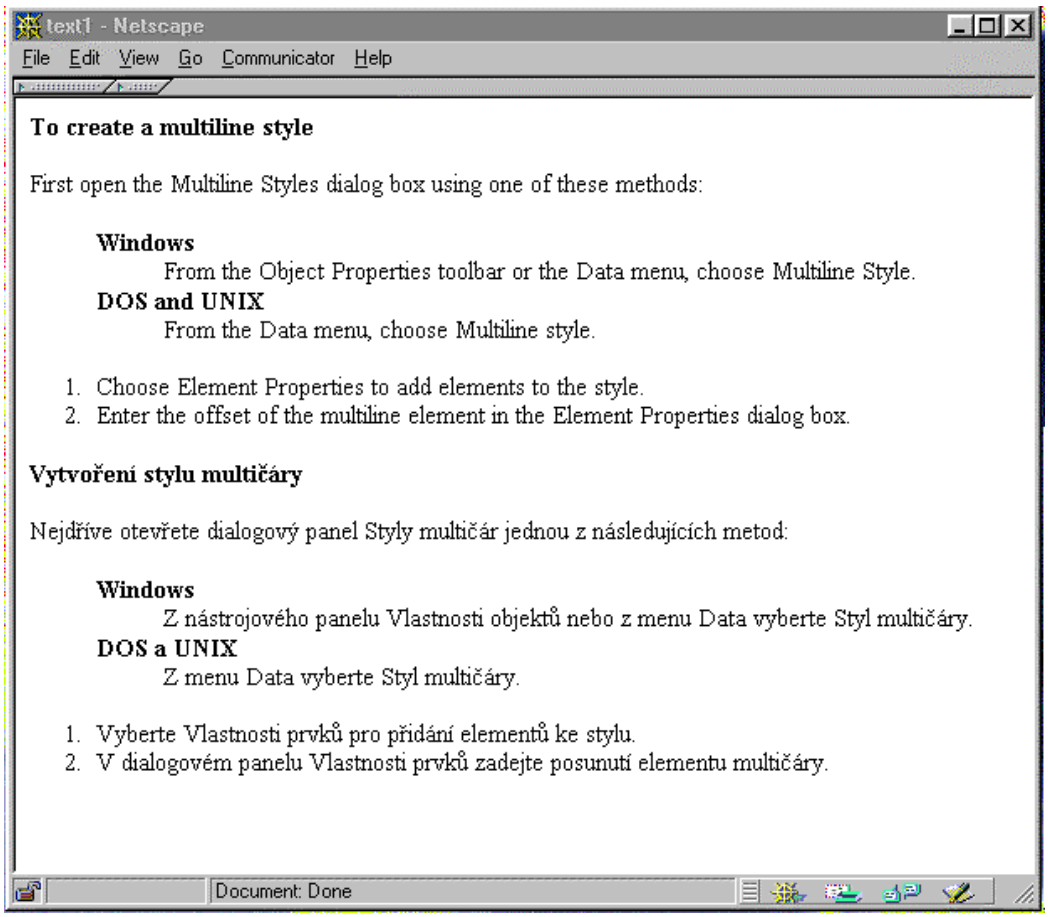


Figure 8 Example of text realisation

5. A Prelude to the Text Structuring Module (TSM)

In this section we outline the strategy we want to employ in the intermediate demonstrator in order to achieve "end-to-end" generation, following the approach to text specification we spelled out in Section 4. As alluded to in Section 1.2, the general thought behind "end-to-end" generation in AGILE is that the user specifies an A-box via the authoring interface, and obtains as output a text in a particular style. Crucial to this enterprise is the Text Structuring Module. This module takes as input the user-defined A-box, and yields a set of SPLs that serve as input to a language-specific tactical generator.

We leave the exact details of the implementation of the Text Structuring Module (TSM) for a more extended discussion that we will present in the forthcoming deliverable TEXM2. Below we first provide an overview of the general architecture of the TSM. Subsequently, we briefly discuss the design of systemic networks for text structuring. We end the section with a consideration on how to get from text structure to a set of SPLs.

5.1 General Architecture of the TSM

Following the earlier discussion of text structuring for AGILE in [AGILE 5.1], we consider the TSM to consist basically of the following three parts:

1. **Systemic networks for Text Structuring**. By means of the text structuring networks, an appropriate text structure is generated for a given A-box.
2. **A program that divides the content of a text's A-box over a number of smaller A-boxes**. The way the division is made is determined by the text structure generated by the systemic networks, as under (1).
3. **A program that generates SPL-code from a given A-box**. The set of A-boxes constructed under (2) functions as input to the program mentioned under (3). Thus, the output that the program under (3) generates is a *set* of SPLs. In fact, these SPLs altogether specify the sentences out of which the text (spelling out the A-box) is made up, due to (sequential) interdependence clarified in the description of the TSM's parts.

The idea behind this general architecture is to make as much use as possible of the KPML environment, rather than opting for employing external modules. The reason for doing so is that, in this way, we do not need an interface between the text structurer and KPML. Consequently, it is much easier to share information throughout the process of generation.

5.2 Systemic Networks for Text Structuring

The major component of the Text Structuring Module is formed by a set of systemic networks. Following the approach that was taken in deliverable TEXM1 reported in [AGILE 5.1], we are constructing a region that defines an additional level of linguistic resources for the level of 'genre', thus dealing with text structuring. The region enables the building up of text structure in a way that is similar to the way the grammar builds up grammatical structure. It was already

noted in [AGILE 5.1] that such a region can be easily "placed on top of" a tactical generator. This makes it easy to integrate it not only with the lexico-grammars that are being developed for AGILE but - conceivably- also with other grammars developed using the KPML environment.

The region, called CADCAM-INSTRUCTIONS, builds on the resources developed for the initial prototype. Following from the viewpoint that text templates and text structure elements are essentially orthogonal ideas, the region consists of two interacting parts.

One part deals with interpreting the A-box in terms of text structure elements. The systems of this part are placed under the TASK UNIT TYPE system. By traversing the network that these systems make up, we obtain a text structure that conforms to the definitions we provided in section 4.1.1 above, and which provides a textual structure for the content defined in the A-box. The network making up this part of the CADCAM-INSTRUCTIONS region is displayed in Figure 9.

The other part of the region deals with the text templates and general conditions on realisation that we discussed earlier in section 4.1.2. The aim of this part is to impose constraints on the realisation of the text structure elements that are being introduced by traversing the other part of the region. Naturally it will depend on our choice of a particular text template and on our setting of the general conditions which constraints will be imposed. These choices will be made through interaction between the user and the system.

What should be clear from setting up the Text Structuring Module this way is that it will indeed be very easy to share information between the high-level task of text structuring (or, as it is sometimes called, "text planning"), and the actual tactical generation of individual sentences. The reason is that both tasks are performed within the KPML environment. The so-called generation gap can thus be minimised since the systems in the CADCAM-INSTRUCTIONS region are in this case able to exert control over the way the grammar will realise a sentence.

5.3 From Text Structure to SPL

After we have obtained a text structure including constraints on how its parts are to be realised, we need to take some further steps before we can invoke the tactical generator.

The first step is to take the text structure and the A-box, and to divide the A-box into parts that are associated with the text structure elements from which the text structure is composed. We already pointed out in [AGILE 5.1] that this can be done by means of ID-inquiries ("identity").

The aim of the second step is to create SPLs, using the parts of the A-box individuated in the previous step and the constraints on realisation already imposed during the generation of the text structure. To that end, for each of the given parts of the A-box we commence by setting up an ideational SPL that realises the content as expressed by that part of the A-box. The SPL is ideational in the sense that it expresses only the content and nothing more. Therefore, once we have an ideational SPL, we need to render the constraints imposed by the text

structure (in particular, by the text template) into terms of SPL code, and include the outcome into the SPL. The resulting SPL then in principle realises the content in a way as prescribed by the text template (and the general realisation conditions).

The final outcome of these two steps is a set of SPLs that specify the realisation of the A-box in terms of the generated text structure. In order to obtain the text, then, we run this set of SPLs through a tactical generator.

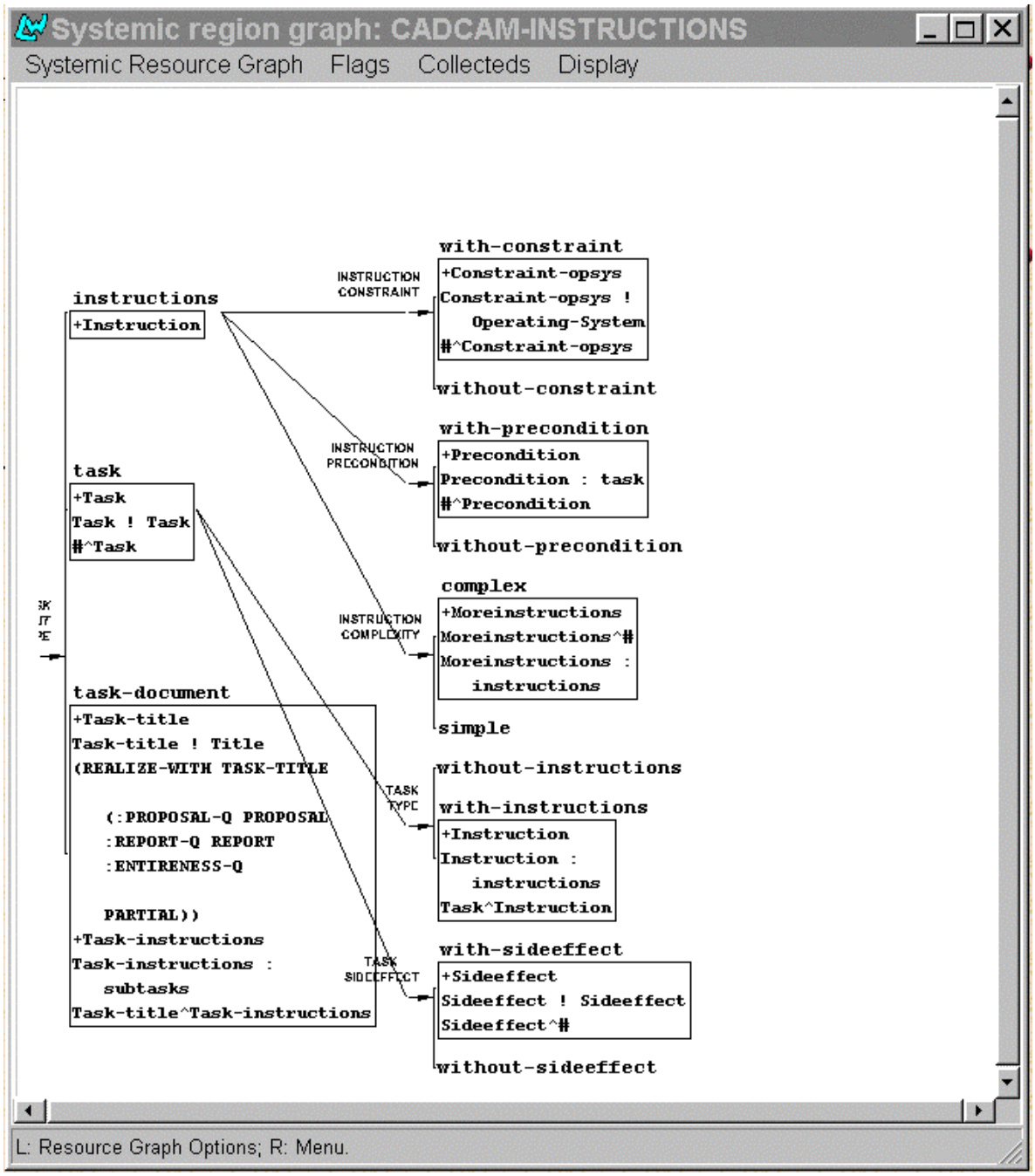


Figure 9- CADCAM-INSTRUCTIONS / Text Structure Elements network

6. Conclusions and Further Work

We have presented the results of Task 5.2 within the Agile project work package WP5 concerned with text structuring specifications for the intermediate prototype, that is, the TEXS2-Bu, TEXS2-Cz and TEXS2-Ru deliverables. In addition, we have presented our initial ideas concerning the implementation of the text structuring module, which is to be described in detail in the next Task 5.2 deliverable, namely TEXM2.

In order to provide orientation in the report, and in order to place the contents of the report into a wider context, we provided an elaborate introduction, in which we explained the problem of text structuring from the viewpoint of an SFG framework in general, and we sketched the particular approach pursued in the AGILE project.

In Section 2 we presented the results of the corpus investigation carried out in order to determine the text structure variation to be covered by the intermediate prototype. First of all, we discussed the characteristics of texts that we consider to belong into the class of instructional texts. This discussion was triggered by the observation that not all parts of texts which are intuitively conceived of as instructional in fact constitute instructions for performing some task in strict sense. Therefore, we adopted an earlier classification of instructional texts into procedure and elaboration sub-genres, and we further classified the former into procedural texts proper and informative texts.

Our corpus analysis concentrated on the variation of lexico-grammatical features in procedural and informative instructional texts other than those included in the corpus extracted from the AutoCAD manual and studied in the work package WP3 in order to inform the development of the initial demonstrator [AGILE 3.1]. We studied both original and translated texts, and found alternative ways of realising instructions which appear to be determined by text style. These include the choice of mood, agency and voice.

On the basis of the corpus study, we determined the range of text style variation to be covered by the intermediate prototype. Besides the lexico-grammatical choices, we also included alternations concerning the explicit realisation of side-effects. Explicitly expressed side-effects constitute the elaboration sub-genre, which occurs interleaved with the procedure sub-genre.

In Section 3, we described the text structure alternations for realising instructions chosen for the intermediate prototype in detail for each language and we presented the components of SPL formulas corresponding to the linguistic realisations. The latter constitute an interface between the text planning process and the tactical generation in the course of text generation in AGILE.

In Section 4, we discussed the organisation of texts in terms of text structure elements, text templates, and general realisation conditions. Based on a mapping between text structure elements and T-box concepts that are used to configure the content defined in an A-box, we presented a way to derive a text structure from the way an A-box is organised. This text structure, which is composed from text structure elements, elucidates how content is to be distributed over the text.

Next, we introduced text templates as means to specify how the text structure elements are to be realised. Making use of the corpus study of Section 2 and the detailed expose on variation in Section 3 we elaborated on what exactly a text template needs to specify in order for the text to be realised in a particular style.

Finally, in Section 5 we presented a prelude to the implementation of the Text Structure Module. The implementation of the TSM is to be discussed in detail in the TEXM2 deliverable.

We can conceive of various issues to be addressed in future work concerning the realisation of content in the context of a (larger) text structure. In Section 4 we already indicated that the approach taken here enables more flexibility in generating various styles of text, and that it allows for different levels of informativity. Thus, the approach covers a large part of the phenomena to be dealt with in the context of the intermediate prototype (again, refer to Section 2), and addresses two of the issues raised in the conclusions of [AGILE 5.1, p.25].

The issues still left open regard the generation of anaphoric references instead of complete realisations, and a full incorporation of the information structure model presented in [AGILE 6.2, Section 2.7]. We intend to deal with these issues (already raised in TEXS1/TEXM1) to some extent in the TEXM2 deliverable.

However, certain interesting and important features of the realisations of instructional texts to be controlled in the process of text structuring remain outside the scope of the intermediate prototype. We will address them in the development of the final prototype.

References

[Bateman 1993] Bateman John. The Dandelion project -an interim report. Technical report, GMD/IPSI.

[DiMarco and Hirst 1993] DiMacrco, C. and Hirst, G. A computational theory of goal-directed style in syntax. In: *Computational Linguistics*, Vol. 19, Number 3, 1993. Pp. 451-499.

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

[Halliday 1985] Halliday Michael A.K. *An Introduction to Functional Grammar*. Arnold.

[Halliday and Hasan 1985] Halliday Michael A.K. and Hasan Ruqaiya. *Language, Context and Text: a social semiotic perspective*. Language and Learning Series. Deakin University Press, Geelong, Victoria, 1985. Also: Oxford University Press, 1989.

[Hartley and Paris 1996] Hartley, Anthony and Paris, Cécile. Two Sources of Control over the Generation of Software Instructions. In: *Proceeding of the ACL Annual Meeting*, Santa Cruz, June 1996. pp. 192-199.

[Martin 1992] Martin, James R. English *Text: Systems and Structure*. Benjamins, Amsterdam.

[Matthiessen and Bateman 1991] Matthiessen, Christian M.I.M. and Bateman, John. *Text Generation and Systemic Functional Linguistics: experiences from English and Japanese*. 1991. Frances Pinter Publishers and St. Martin's Press, London and New York.

[Pemberton et al. 1996] Lyn Pemberton, Louise Gorman, Anthony Hartley and Richard Power (1996). Computer Support for Producing Software Documentation: Some Possible Futures. In: T. van der Geest and M. Sharples eds.), *The New Writing Environment: Writers at Work in a World of Technology*. Springer Verlag.

[Popov 1962] Попов К. - Съвременен български език. Синтаксис. "Наука и изкуство", София, 1962. In Bulgarian.

[Power and Scott 1997] Richard Power and Donia Scott, 1997. *NLG tools to support technical authors and translators*. A manuscript submitted to the ANLP'97 conference.

[Power et al. 1994] Richard Power, Lyn Pemberton, Anthony Hartley and Louise Gorman (1994). *User Requirements Analysis. DRAFTER Project* deliverable WP2 IED4/1/5827, IITRI, University of Brighton, UK.

[Ramm and Villiger 1995] Ramm Wiebke and Villiger Claudia. *Global Text Organisation and Sentence-Grammatical Realisation.* CLAUS report No. 61, University of Saarland. ISSN 0941-083X.

[Ramm et al. 1995] Ramm Wiebke, Rothkegel Annely, Steiner Erich and Villiger Claudia. *Discourse Grammar for German*. Deliverable R2.3.2. of WP 2 'Grammar Integration', ESPRIT Basic Research Project 6665 DANDELION, University of Saarland.

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

[Sharoff and Sokolova 1995] Sharoff S. and Sokolova L., Analysis of rhetorical structures in technical manuals and their multilingual generation. In: *Proc. of the workshop on multilingual generation at IJCAI'95*, R. Kittredge (ed.) , Montreal,1995. pp. 119-128.

[Sokolova, to appear] Соколова, Е.Г. *О дискурсной структуре и стиле текстов инструкций на русском языке* , to appear in DIALOG'99.

[Vanderlinden and Scott 1995] Vanderlinden, K. and Scott, D. *Raising the Interlingual Ceiling with Multilingual text generation* In: *Proc. Maltilingual text generation,, workshop at IJCAI'95,* Montreal, August, 1995. pp. 95–101.

Agile documentation:

[Agile project 1997] *Agile project. Automatic generation of instructions in languages of Eastern Europe*. Technical report, Commission of the European Community, Brussels, DG XIII. Technical Annex.

[AGILE 2.1] Power, Richard. *Preliminary model of the CAD/CAM domain*. June 1998.

[AGILE 3.1] Hartley, Anthony et al. *Tagging and analysis of instructional texts in the software domain*. June 1998.

[AGILE 4.1] Skoumalová, Hana et al. *Lexical-morphological specifications and resources*. June 1998.

[AGILE 4.2] Kruijff, Geert-Jan et al. *Modelling Lexical Resources in KPML for Generating Instructions in Slavic Languages*. October 1998.

[AGILE 5.1] Bateman, John et al. *Generation of simple text structures in Bulgarian, Czech and Russian*. June 1998

[AGILE 6.1] Bateman John et al. *Specification of grammatical resources for the Initial Demonstrator.* June 1998.

[AGILE 6.2] Adonova, Elena et al. *Formal specification of extended grammar models*. March 1999.

[AGILE 7.1] Bateman, John et al. *Grammatical Resource Implementation for Bulgarian, Czech and Russian*. June 1998.

Corpora

Bulgarian corpus:

[AutoCAD 1996] Zirbel J.H and S. B. Combs. AutoCAD Release 13 for Windows (in Bulgarian). SoftPress 1996.

[JavaScript 1996] Гудман Д. - Програмиране в Интернет чрез JavaScript, ч. I, Computer Times Ltd, 1996.

[Windows95 1998] Ливингстън Б., Д. Строб. Тайните на Windows 95. Техника, София, 1998.

[ПЛОТ3 1989] ПЛОТ 3 - Система за тримерно моделиране, проектиране и чертане. Ръководство за потребителя. София, 1989.

Czech corpus:

[AutoCAD] Uživatelská příručka AutoCAD, verze 13. AutoDesk, Prague Czech Republic. In Czech.

[Baltazar manual] SGP Systems, 1995. SGP Baltazar 3.15 – příručka. SGP Systems, Uherské Hradiště, Czech Republic. In Czech.

[conservation] Půhoný, Karel. 1982. Konzervace a ukládání potravin v domácnosti. Státní Zemědělské Nakladelství, Prague Czech Republic. In Czech.

[drink recipes] Burian, Vlado. 1974. Malý receptář kávy a čaje. Merkur Nakladatelství, Prague Czech Republic. In Czech.

[Ether. manual] 3Com and Autocont, 1995. Návod k instalaci: síťové adaptéry řady Etherlink III. Autocont, Ostrava Czech Republic. In Czech.

[general recipes] Janků, Marie and Janků, František. 1990. Sandtnerová: Kniha rozpočtů a kuchařských předpisů. Art-Servis, Prague Czech Republic. In Czech.

[cheese recipes] Kněz, Václav and Sedláčková, Hana. 1991. Sýry a příprava sýrových pokrmů. SNTL, Prague Czech Republic. In Czech.

[Italy guide] Bartoněk, Antonín et al., 1990. Itálie – průvodce do zahraničí. Globus, Prague Czech Republic. In Czech.

[mountain guide] Birner, Zdeněk and Páv, Antonín. 1981. Krušné hory a západočeská lázeňská oblast. Olympia, Prague Czech Republic. in Czech.

[PC manual] Autocont, 1995. Příručka uživatele – osobní počítač Autocont. Autocont, Ostrava Czech Republic. In Czech.

[T602 manual] Software602, year of publication unknown. Uživatelská příručka Text602. Software602, Prague Czech Republic. In Czech.

[Tesla manual] Příručka k satelitnímu přijímači Tesla. Tesla. 1997. In Czech.

[Trabant guide] Šlehofer, Vlatislav. 1971. Údržba a opravy vozů Trabant 600 a 601. SNTL, Prague Czech Republic. In Czech.

[Win guide] Holčík, Tomáš. year of publication unknown. Microsoft Windows for Workgroups 3.11 – podrobná uživatelská příručka. Computer Press, Prague Czech Republic. in Czech.

[Word guide] Novák, Petr. 1994. MS Word 6 pro Windows. Grada Publishing, Prague Czech Republic. in Czech.

[Word manual] Microsoft Corporation, 1994. Uživatelská příručka Microsoft Word verze 6.0. Microsoft, Ireland. In Czech.

[Works guide] Čáp, Jan. 19??. Microsoft Works 4.0 – základní průvodce uživatele. Computer Press, Prague Czech Republic. In Czech.

Russian corpus

[aircraft manual] Эксплуатация гидравлического оборудования самолета ТУ 204 (фрагмент документации) 1994. In Russian.

[auto manual] Васильев А.В., Гофман Е.Ю., 1993. Эксплуатация и ремонт современных автомобилей. Москва. In Russian.

[LoCoDoNS] Specification for a Doppler effect navigation equipment software. Dassault Aviation. Parшs, 1993 (an in-house Russian translation is provided in the course of a joint project).

[MS Word manual, English] Microsoft Word 6.0 for Windows; User manual. Microsoft Press, 1993.

[MS Word manual, Russian] Руководство пользователя по Microsoft Word 6.0 для Windows. Microsoft Press, 1994. In Russian.

[SPRUT user guide] Иоселиани Г.Р., Прокопенко Ф.В., Система автоматизированного проектирования СПРУТ. Руководство пользователя. (Москва, МГТУ им. Баумана), 1996. In Russian.

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 realised 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.

A. 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.

B. 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 OK за връщане в диалоговия прозорец Multiline Styles. При това диалоговият прозорец Multiline Properties се затваря.
   (b) Избира се OK 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 OK 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.

C. 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.[19]

[19] 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 realisation. Moreover, "šrafuj" is a 2nd person singular form, i.e., a realisation 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.

D. Russian

IMD Tex 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 Tex 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 Tex 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 Tex 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 Tex 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.