Dynamic Semantics at Work

Rolf Schwitter and Marc Tilbrook

Centre for Language Technology, Macquarie University,
Sydney, NSW 2109, Australia
{schwitt, marct}@ics.mq.edu.au

Abstract. In this case study we show how an unambiguous semantic representation can be constructed dynamically in left-to-right order while a text is written in PENG, a controlled natural language designed for knowledge representation. PENG can be used in contexts where precise texts (e.g. software specifications, axioms for formal ontologies, legal documents) need to be composed. Texts written in PENG look seemingly informal and are easy to write and to read for humans but have first-order equivalent properties that make these texts computer-processable.

1 Introduction

Controlled natural languages are well-defined subsets of natural languages that have been restricted with respect to their grammar and their lexicon. Traditionally, controlled natural languages fall into two major groups: *human-oriented* and *machine-oriented* controlled natural languages.

Human-oriented controlled natural languages have been designed to increase the readability of technical documents for human readers while machine-oriented controlled natural languages have been developed to improve the computational properties of these texts [3]. As a rule of thumb: a controlled natural language that is easy to read for humans is also easier to process for a machine but not necessarily completely processable by a machine.

PENG is a machine-oriented controlled natural language – with a restricted grammar and lexicon – that has been designed to write unambiguous and precise specifications for knowledge representation [10]. The writing process for a PENG text is supported by ECOLE, an intelligent text editor, that guides the user and guarantees compliance to the rules of the controlled language [11].

The language processor of the PENG system translates sentences dynamically into a flattened notational variant of Discourse Representation Theory (DRT) [5] [6] while the user writes the text. That means that after each word form that the user enters, the language processor creates semantic information that is analyzed in the context of a given discourse representation structure (DRS) and then either updates the current information state or delays this process if necessary. Additionally, the language processor generates look-ahead categories and sends them to the editor. These look-ahead categories provide predictive syntactic hints that inform the user about choices on how to continue the current input string. The language processor is connected via a server with a

theorem prover (OTTER; [8]) and a model builder (MACE; [7]) that run in parallel and allow for question answering and acceptability checking (i.e. consistency and informativeness checking) of a specified piece of knowledge.

In this paper, we will focus on how the discourse representation is dynamically constructed using a flattened notation for DRT conditions. Because PENG is a controlled natural language with well-defined syntactic properties, it allows for threading of information states in left-to-right order.

The remainder of this paper is structured as follows: In Section 2, we will present the controlled natural language PENG and show how the user works with this language. In Section 3, we will introduce a flattened notation for DRT conditions and discuss the advantages of this notation for a practical application. In Section 4, we will introduce a dynamic approach to constructing DRSs that supports the incremental writing process. In section 5, we will extend the introduced approach substantially and discuss implementation issues for various linguistic phenomena. In Section 6, we will show how Lewis Carroll's *Grocer Puzzle* can be solved in PENG. Finally, in Section 7, we will conclude and emphasize the advantages of the presented approach.

2 The Controlled Language PENG

PENG uses an unification-based phrase structure grammar as syntactic scaffolding for the DRS construction and a lexicon with (user-defined) content words and predefined function words.

2.1 The Grammar of PENG

The grammar of PENG currently consists of over 100 phrase structure rules that are processed by a chart parser which performs top-down parsing. The same rule formalism is used for building syntactic, semantic and pragmatic structures. This rule formalism uses an attribute-value notation to represent feature structures associated with the constituents. Below is a phrase structure rule that shows that a simple PENG sentence *s0* is composed of a noun phrase *n3* and a verb phrase *v3* followed by a full stop *fs*:

```
s0([crd:no,drs:D,para:P1-P4,tree:[s0,T1,T2,T3]]) -->
  n3([crd:_,arg:I,spec:Q,ana:A,drs:D,sco:S,para:P1-P2,tree:T1]),
  v3([crd:_,arg:I,drs:S,para:P2-P3,tree:T2]),
  fs([cat:fs,para:P3-P4,tree:T3]).
```

This rule encodes – among other things - the flow of semantic information and subject-verb agreement of the constituents. The attribute *arg* with the value *I* (i.e. a variable that stands for an entire feature bundle) constrains the argument structure of a sentence. The attribute *drs* guarantees that the semantic information *D* of the sentence is shared with the noun phrase (and eventually with its determiner). The attribute *sco* assures that the scope *S* of the noun phrase

is passed to the verb phrase. The task of the attribute *para* is to construct a paraphrase for the input string and finally the attribute *tree* is used to build up a syntactic tree during processing time.

PENG sentences can be simple or complex and can be interrelated anaphorically with definite expressions. The main restrictions of the language are the use of present tense verbs and the control of plural constructions by explicit disambiguation markers [10]. Other restrictions that are important are the use of variables instead of personal pronouns, and the scope of quantifiers and negation. In PENG, a quantifier has always scope over all subsequent quantifiers in a sentence and a verb phrase negation has scope over the entire verb phrase. The order of quantifiers can be manipulated by constructors such as *there is a* and *for every*, for example *For every cyclist there is a race that is challenging*. These restrictions enable the scope of all quantifiers to be determined from the surface order.

2.2 The Lexicon of PENG

The lexicon of PENG is made up of content words and function words. Content words are defined by the user and can be added or modified during the writing process with the help of a lexical editor. The structure of content words can be simple (e.g. *cyclist*) or compound in various forms (e.g *motorcyclist, tri-cyclist, champion cyclist*) like in full English. Semantically, compound words are treated as single units in the lexicon and their internal structure is not further analysed. In PENG, each content word can be associated with a set of (strict) synonyms and abbreviations. During processing time, synonyms and abbreviations are automatically replaced by the primary content word that has been specified in the lexicon and its corresponding logical representation. This replacement is reflected in a paraphrase that the PENG system generates for each sentence.

2.3 Writing in PENG

The writing process for a PENG text is supported by ECOLE, a look-ahead text editor [11]. After each word form that the user enters, the editor indicates what kind of syntactic structures can follow the current input string. Let us assume that the user is in the process of writing the noun phrase:

The American champion cyclist Lance Armstrong ...

After entering the word *champion*, the text editor displays the following kinds of look-ahead categories:

```
proper_noun
verb
auxiliary:[does]
noun:[cyclist]
preposition:[of]
relative_pronoun:[who]
variable:[e.g. X1]
```

At this point, the user can employ all lexicalized content words and function words that belong to these categories to carry forward the input string, for example:

The American champion Lance ...
The American champion works ...
The American champion does ...
The American champion cyclist ...
The American champion of ...
The American champion who ...
The American champion Y23 ...

This way the user is guided and does not need to learn and remember the restrictions of the controlled language. If the user enters a content word that is not in the lexicon, then the spelling checker of the PENG system fires up and checks whether that word is misspelled or not. The user either corrects the spelling of the word or can add the word (in case of a content word) to the lexicon with the help of a lexical editor.

2.4 Anaphora resolution in PENG

In PENG only definite noun phrases, proper nouns, and variables can be used as anaphoric expressions. Personal pronouns are not allowed in PENG but non-ambiguous variables can be used instead of pronouns, for example:

The cyclist A1 beats the cyclist A2. A1 is faster than A2.

Noun phrases must be accessible to be referred to by anaphoric expressions: indefinite noun phrases that are in the scope of a negation or an universal quantifier are not accessible for anaphoric reference from subsequent sentences in PENG. If the text contains – for example – the accessible noun phrase:

The American champion cyclist Lance Armstrong ...

then the following anaphoric expressions can be used to corefer with the closest antecedent:

The American champion cyclist Lance Armstrong ...
The American champion cyclist Lance ...
The American champion cyclist Armstrong ...
The American champion cyclist ...
The American champion ...
Lance Armstrong ...
Armstrong ...
Lance ...

In PENG anaphora resolution is done dynamically while the user writes the text. That means whenever a complete noun phrase has been processed, the anaphora resolution algorithm checks if there exists a subsumption relation between the noun phrase and a (closest) accessible antecedent. If no antecedent can be found, then the noun phrase is teated as an indefinite noun phrase introducing a new discourse referent. To support the user, all accessible noun phrases could be displayed in a similar way as this is done for look-ahead categories so that the user could simply make a selection from a list of accessible noun phrases. However, this functionality has not yet been implemented.

3 A Flattened Notation for DRT conditions

Formally, a DRS is an ordered pair `<U,Con>` where `U` is a set of discourse referents and `Con` is a set of conditions [5]. For processing reasons, we represent a DRS as a term of the form `drs(U,Con)` consisting of a list `U` of discourse referents $[I_1, I_2, \ldots I_n]$ denoting entities and a list `Con` of conditions $[C_1, C_2, \ldots C_n]$ that describe properties or relations that these discourse referents must satisfy. DRSs can occur as constituents of larger (complex) DRSs. Complex DRS conditions are those involving implication, disjunction, and negation [6].

In our flattened notation for DRS conditions we treat concepts such as `cyclist(I)` as typed individuals `obj([cyclist],I)`. Concepts do not introduce predicate symbols anymore and can therefore be referred to by simple terms (see also [2]). The domain of discourse is divided into the domain of objects and the domain of eventualities (= events and states). The domain of objects has a lattice-theoretic structure and is subdivided into groups, individuals and mass objects [9]. Here are a few lexicon entries for DRS conditions in PENG that reflect this structure:

```
[obj([cyclist],I),struc(I,atomic)]
[obj([champion,cyclist],I),struc(I,atomic)]
[obj([grocers],I),struc(I,group)]
[pred(E,[wins],I1,I2),evtl(E,event)]
[prop(T,[in],I1,I2),role(T,time)]
[prop([industrious],I)]
```

This flattened notation has three main advantages: First, quantification over complex terms that would require higher-order quantification can be conducted via first-order quantification. Second, the flattened notation simplifies the formalization of logical axioms to express various forms of linguistic and non-linguistic knowledge. Third, this notation increases – as a neat side-effect – the efficiency of the inference processes.

4 Dynamic Construction of DRSs

In Kamp's [5] *original* DRT the processing of a sequence of sentences S_1, S_2, ... S_n is carried out through a DRS construction algorithm that starts with the

syntactic analysis of the first sentence S_1 and then transforms it with the help of DRS construction rules into a DRS K_1 which serves as the context for processing the second sentence S_2. This approach is sequential and does not emphasize the dynamic aspect of transforming information states while syntactic constituents are parsed.

Johnson and Klein's [4] reformulation of the DRS construction algorithm offers a solution here (see also [1]). In their approach each syntactic constituent of a sentence is related to an incoming and outgoing DRS. This relation is modeled by a difference list of the form *DrsIn-DrsOut*. This data structure threads the semantic information through the phrase structure rules of a (definite clause) grammar. The outgoing DRS is constructed from the incoming DRS (which contains information about available antecedents) plus conventional semantic information derived from the actual constituent. The meaning of a constituent can then be defined as the change in the DRS, after the constituent has been processed.

Threading of semantic information through a grammar can get complex, especially if a sentence consists of a number of quantifiers and constructors, since different scope-bearing elements of a sentence result in nested DRSs. As we will see in the next section, sometimes updating of a DRS needs to be delayed, particularly if we want to generalize over a set of phrase structure rules and avoid multiplying out these rules.

5 Implementation Issues

In this section, we will discuss how a DRS is dynamically constructed in PENG. To make the following discussion self-contained, we will first present a pseudo-grammar that will later be used for threading difference lists through its structure. We will first focus on the crucial role that determiners play in this process and then describe in a stepwise manner how the sentence

The American champion cyclist Lance Armstrong wins no race in April.

is translated into a DRS. This sentence consists of a rather complex definite noun phrase *The American champion cyclist Lance Armstrong* in subject position and a verb phrase *wins no race* with a temporal modifier *in April*. The definite noun phrase is composed of an adjective (*American*), a compound common noun (*champion cyclist*), and a proper noun (*Lance Armstrong*) in appositive position. This noun phrase can be described by the following (simplified) phrase structure rules:

```
n3  --> d0, n2, { anaphora_resolution }.
n2  --> a2, n1.
n1  --> n0, ap.
ap  --> pn.
n0  --> xcn, ccn.
pn  --> xpn, cpn.
```

The first phrase structure rule indicates that anaphora resolution is done after the entire noun phrase has been processed. If no antecedent can be found, then a definite noun phrase is treated in PENG as if it were an indefinite noun phrase. As the phrase structure rules foreshadow, compound words such as *champion cyclist* and *Lance Armstrong* are processed token by token as they are entered by the user (see Section 5.2 for details).

The verb phrase of the sentence is composed of a transitive verb *wins* that subcategorizes for a (negative) noun phrase *no race* whereas the prepositional phrase *in April* is syntactically attached to the verb. The (simplified) phrase structure rules have the following form in PENG:

```
v3  --> v2.
v2  --> v1, p2.
v1  --> v0, n3.
p2  --> p0, n3.
```

As we will see, these phrase structure rules will be expanded in a way that allows us to deal – among other things – with optional constituents such as prepositional and adverbial modifiers in an uniform way. This is important, since these optional constituents can open a new complex DRS space that embeds DRS conditions that have been derived from preceding constituents. As we will see in the following sections, the key question is to decide when to open a new DRS space and when to close it.

5.1 The role of determiners

The bulk of work that the DRS construction algorithm conducts is done in the rules for the determiners. Despite their minor syntactic role, determiners are the most important constituents for establishing the DRS of a sentence. Semantically, a determiner has two arguments: a restrictor and a scope. The restrictor consists of the conditions derived from the remaining noun phrase (= *n3 - d0*). The scope – itself probably consisting of a complex DRS – can be composed of the conditions outside the noun phrase (in our case semantic information derived from the verb phrase). Below is a (simplified) grammar rule for the definite determiner *the*:

```
d0([...,
    drs:D1-D3,
    res:[drs([],[])|D1]-D2,
    sco:D2-D3,
    ...]) -->
    { lexicon([lex:Determiner,...],[...]) },
    Determiner.
```

A definite determiner does not introduce a new DRS. However, since we do not known in advance whether the entire definite noun phrase is used anaphorically or not, we add the semantic information derived from the remaining noun

phrase into an empty DRS *drs([],[])* that we place in front of the restrictor's incoming DRS *D1*. This DRS serves as a store that can be accessed by the anaphora resolution algorithm once the entire definite noun phrase has been processed. After anaphora resolution, the restrictor's outgoing DRS *D2* will contain the resolved DRS conditions that are then passed on to the scope's incoming DRS. After processing the verb phrase, the scope's outgoing DRS *D3* will contain the semantic information for the sentence as a whole.

Other determiners such as the negative determiner *no* manipulate the DRS in more complex ways:

```
d0([...,
    drs:D1-[drs(U1,[drs(U2,C2) -> drs([],[~drs(U3,C3)])|C1])|D3],
    res:[drs([],[])|D1]-D2,
    sco:[drs([],[])|D2]-[drs(U3,C3),drs(U2,C2),drs(U1,C1)|D3],
    ...]) -->
   { lexicon([lex:Determiner,...],[...]) },
   Determiner.
```

Here the restrictor pushes an empty DRS *drs([],[])* in front of the incoming DRS *D1* and makes this the active information space where all discourse referents and conditions for the remaining noun phrase are collected. The scope takes the restrictor's outgoing DRS *D2* and pushes a new empty DRS in front of it and makes this again a new active information space where all discourse referents and conditions outside the noun phrase are collected. The DRS for the restrictor *drs(U2,C2)* and the DRS for the scope *drs(U3,C3)* are then embedded into a complex outgoing DRS condition:

```
[drs(U1,[drs(U2,C2) -> drs([],[~drs(U3,C3)])|C1])|D3]
```

This DRS consists of an implication (`->`) and a negation (`~`), and will finally represent the meaning of the negative determiner.

5.2 Processing the restrictor

After processing the definite determiner, the restrictor *R1* is passed to the constituent *n2* where the remaining semantic information for the noun phrase will be acquired:

```
n3([...,arg:[I|A],drs:D,sco:S,...]) -->
  d0([...,ana:yes,drs:D,res:R1-R3,sco:S,...]),
  n2([...,arg:[I|A],drs:R1-R2,...]),
  { anaphora_resolution(...,I,R2,R3,...) }.
```

The remaining noun phrase consists of the nominal constituent *American champion cyclist Lance Armstrong* that is composed of an adjective and two compound words. The structure of the compound words will not be further analysed semantically but the semantics is represented as a list of terms instead

of a predicate (see Section 3). This representation makes it easy to add additional axioms later that specify the relation between these terms – if necessary. The following (simplified) phrase structure rules are responsible for processing a compound noun such as *champion cyclist* dynamically:

```
n0([...,drs:D1-D2,...]) -->
  xcn([...,len:[]-Tokens,drs:D1-D2,...]),
  ccn([...,len:Tokens-[],...]).

xcn([...,arg:[ind:I|R],len:[]-Tokens,
     drs:[drs(U1,C1)|D1]-[drs([I|U1],[C3,C2|C1])|D1],...]) -->
  { lexicon([lex:[Token|Tokens],...],
            [...,arg:[ind:I|R],...,con:[C3,C2]]) },
  [Token].
```

Compound nouns are processed token by token. After the user enters *champion*, the first grammar rule *n0* triggers a lexicon lookup via *xcn* and retrieves the semantic information for the entire compound noun. That information – *I* for the discourse referent and *C3,C2* for the conditions – is then pushed on the outgoing DRS:

```
[drs([I|U1],[C3,C2|C1])|D1]
```

The remaining tokens *Tokens* of the compound noun are written on a list that is processed recursively as soon as these tokens are entered by the user. This way only one lexicon lookup is necessary for processing a compound noun.

After processing the entire noun phrase, the outgoing DRS *R2* of *n2* has the following form:

```
[drs([A],
     [prop([american],A)
      struc(A,atomic)
      obj([champion,cyclist],A)
      named([lance,armstrong],A)])|D1]
```

The anaphora resolution algorithm of *n3* checks now whether the information in front of *D1* is accessible in *D1* or not.

5.3 Processing the scope

The scope of the noun phrase will be realized by the semantic information derived from the verb phrase. The attribute *drs* is used to pass the scope *S* from the noun phrase into the verb phrase (see the grammar rule *s0* in Section 2.1).

The semantic information for the verb phrase is processed by the following phrase structure rules:

```
v3([...,drs:D,...]) -->
  v2([...,drs:D,...]).

v2([...,drs:D,...]) -->
  v1([...,drs:D,sco:S,sco:hold:S1,...]),
  p2([...,drs:S,sco:S1,...]).

v1([...,drs:D,sco:S,sco:hold:S1,...]) -->
  v0([...,drs:S1,...]),
  n3([...,drs:D,sco:S,...]).
```

The important thing here is that in *v0* a new difference list *S1* with an incoming and outgoing DRS is constructed that collects the semantic information for the verbal event:

```
drs:[drs(U1,C1)|D]-[drs([E|U1],[C3,C2|C1])|D]
```

However, this information is not immediately used as scope for the subcategorized noun phrase *n3*, since it is not yet known at this stage of processing whether the verbal event will later be modified (as it is the case in our example) or not. Instead, the difference list *S1* is stored using the attribute *sco:hold* and passed on to the prepositional phrase *p2* via *v1*. In summary, the prepositional phrase *p2* receives the scope *S* of the noun phrase *n3* as DRS and the DRS of the verb *v0* as scope *S1*.

If the sentence would not contain a prepositional modifier, then the following additional rule would fire and guarantee that the scope of the noun phrase unifies with the semantic information for the verb stored on the holding list:

```
v2([...,drs:D,...]) -->
  v1([...,drs:D,sco:S,sco:hold:S,...]).
```

After processing the scope for the verb phrase, the outgoing DRS has the following final form:

```
[A,B]
prop([american],A),struc(A,atomic),obj([champion,cyclist],A),
named([lance,armstrong],A),struc(B,atomic),named([april],B)
  [C]
  obj([race],C),struc(C,atomic) ->
  ~ [D,E]
    prop(D,in,E,B),role(D,time),pred(E,[wins],A,C),evtl(E,event)
```

As the resulting DRS shows, the DRS conditions derived from the verb occur on the right hand side of the implication and fall under negation. This is a consequence of the quantificational potential of the negative determiner.

6 Check it or Proof it!

Texts written in PENG can be checked for consistency and informativeness after each new sentence that the user enters. Apart from that, the user can also query a specification. For example, the text in Fig. 1 is a reformulation of Lewis Carroll's *Grocer Puzzle* in PENG that can be solved by formulating a question in controlled natural language.

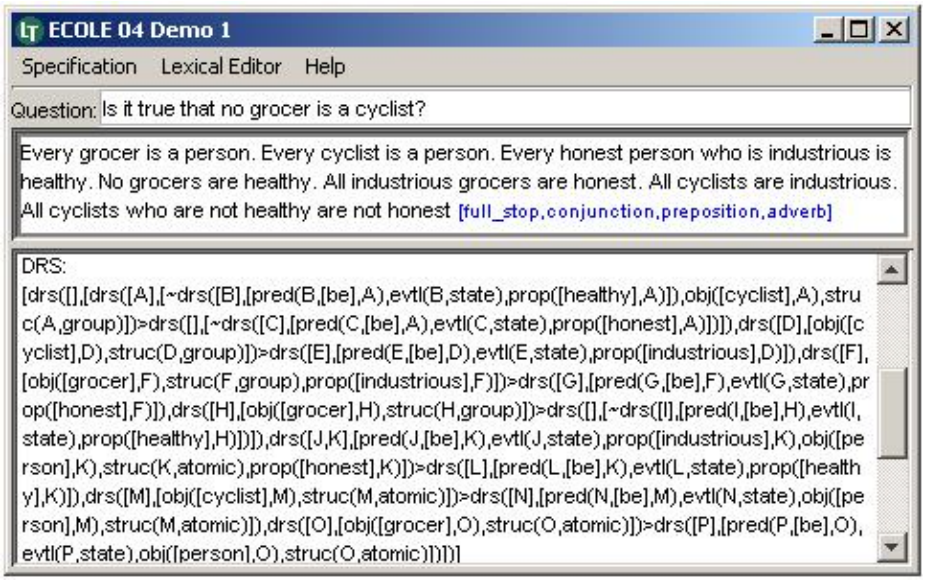

Fig. 1. The Text Editor ECOLE with Look-ahead Categories and DRS

The DRS in Fig. 1 is first translated (in linear time) into a set of first-order formulas. Apart from the DRS, the reasoning services of PENG use additional lattice-theoretic (linguistic) axioms for the inference tasks. For instance, the following axiom

```
(all X Y ((struc(X,atomic) & part_of(X,Y)) -> struc(Y,group))).
```

is used to relate a noun phrase (e.g. *every X*) that introduces an atomic object into the domain to a noun phrase (e.g. *all Xs*) that introduces a group. In our case, the answer to the puzzle is deduced with OTTER, a resolution-style theorem prover for first-order logic with equality [8].

OTTER automatically generates clauses for these formulas, selects inference rules and strategies and solve the puzzle (i.e. derives the empty clause).

7 Conclusions

In this paper we introduced a flattened notation for DRT and showed how discourse representation structures can dynamically be constructed for a controlled natural language in left-to-right order while the text is written. The result is an unambiguous text in controlled natural language that has the same formal properties as the underlying formal representation language. Such a controlled language can serve as an interface language to any kind of knowledge systems.

Acknowledgments

The research reported here was supported by the Australian Research Council, Discovery Project DP0449928. The authors would also like to thank to all the folks at the Centre for Language Technology at Macquarie University for many discussions and valuable comments on the PENG project.

References

1. Black, A.W.: Some Different Approaches to DRT. Draft. Centre for Cognitive Science, University of Edinburgh, June 27, (1997)
2. Hobbs, J.R.: Discourse and Inference. Draft. USC Information Science Institute, Marina del Rey, California, November 3 (2003)
3. Huijsen, W.O.: Controlled Language - An Introduction. In: Proceedings of CLAW 1998, Pittsburgh (1998) 1-15
4. Johnson, M., Klein, E.: Discourse, anaphora and parsing. In: Proceedings of the 11th International Conference on Computational Linguistics. Coling 86. Bonn (1986) 669–675
5. Kamp, H.: A theory of truth and semantic representation. In: J. Groenendijk et al. (eds.) Formal methods in the study of language. University of Amsterdam (1981) 277-322
6. Kamp, H., Reyle, U.: From Discourse to Logic, Dordrecht: Kluwer (1993)
7. McCune, W.: MACE 2.0 Reference Manual and Guide. ANL/MCS-TM-249. Mathematics and Computer Science Division, Technical Memorandum No. 249, Argonne National Laboratory, Argonne (2001)
8. McCune. W.: OTTER 3.3 Reference Manual. ANL/MCS-TM-263, Mathematics and Computer Science Division, Technical Memorandum No. 263, Argonne National Laboratory, Argonne (2003)
9. Schwertel, U.: Plural Semantics for Natural Language Understanding. A Computational Proof-Theoretic Approach. PhD Thesis, University of Zurich (2003)
10. Schwitter, R.: English as a Formal Specification Language. In: Proceedings of the Thirdteenth International Workshop on Database and Expert Systems Applications (DEXA 2002). Aix-en-Provence (2002) 228–232
11. Schwitter, R., Ljungberg, A., Hood, D.: ECOLE: A Look-ahead Editor for a Controlled Language, Proceedings of EAMT-CLAW03, Controlled Language Translation, May 15-17, Dublin City University (2003) 141–150

The submitted date: February 26, 2004
The accepted date: March 12, 2004
The revised date: April 24, 2004