A Controlled Language Aproach to Text Optimisation in Technical Documentation

Ecaterina Rascu
Universität des Saarlandes, Saarbrücken, Germany
kati@iai.uni-sb.de

Abstract

In this paper we propose a controlled language approach to text optimisation in the field of technical documentation. Within this approach, we use stylistic paraphrases as instrument to the optimisation process. We present various categories of paraphrasing principles and describe their implementation in the corrector component of a controlled language checker.

1 Introduction

Document production in general, and technical writing in particular are inevitably linked to the concept of text optimisation. From a linguistic point of view, standard language, consistent use of terminology, unambiguous linguistic structures are required to increase the readability, comprehensibility and translatability of technical documentation. In addition, company specific regulations may impose further restrictions on documents in order to promote a consistent corporate image.

Many companies have developed their own style guides or writing rules for technical documentation. These usually include both requirements characteristic of technical writing in general and company specific regulations. Some examples of stylistic requirements are: the avoidance of jargon, the avoidance of complex, conjoined sentences, the use of positive constructions or the use of parallel structures (O'Brien, 2003). Style rules are sometimes formulated in form of so-called controlled languages.

When speaking about style guides, style rules or style checking, the notion of *style* is used to cover all linguistic phenomena which cannot be controlled by means of a prescriptive grammar. Stylistic phenomena are usually treated in terms of appropriate/ inappropriate whereas grammar problems are commonly dealt with in terms of correct/incorrect.

In order to support text optimisation and ensure conformance to standards, various tools such as document management systems, authoring tools, terminological databases or language checkers have been developed. In this paper we propose to extend the style checking functionality of an existing authoring tool, i.e. CLAT, so that it not only detects inappropriate linguistic structures but also provides an adequate reformulation.

In the following sections, we first define controlled languages (CLs) and present the various approaches used for language control in CL checkers. Then, we present the style checking functionality of the CLAT checker. Subsequently, paraphrasing principles are elaborated in order to reformulate structures that are found to be stylistically inappropriate. The implementation of the paraphrasing principles into a corrector is also discussed. Eventually, we evaluate the perormance of the developed paraphrasing strategy.

2 Controlled Language Approach

Controlled languages (CLs) are a subset of the set of natural languages which have been created for specific purposes (Dras, 1999). They have generally been devised for large companies in order to restrict the complexity of the language used in their documentation and thus optimise the document production and the reception processes. Moreover, the use of a CL should facilitate machine processing of firm documentation and ensure that a consistent corporate image is promoted.

Controlled languages restrict vocabulary, syntax, and surface characteristics of texts. Moreover, corporate values like solidity, loy-

alty, quality, fun, creativity, design can be signalled in the written documentation of the company, for instance, by using appropriate linguistic means (Henriksen et al., 2004).

For many companies it is important that the conformance of texts to the specific rule sets defining a CL be verifiable with specific applications, so called conformance checkers or controlled language checkers. CL checkers ideally have two components, a checker and a corrector, which either automatically corrects the text or suggests rewrites to the user (Mitamura and Nyberg, 2001). But not all checkers contain such correction modules. Typical reasons for not integrating a corrector are for example the wish to keep error reports simple and to avoid possible wrong suggestions which would only annoy the technical writer (Wojcik and Holmback, 1996). Moreover, besides reformulating inappropriate structures, checkers might integrate various other strategies meant to support the rewriting process such as detailed problem diagnosis or training through typical reformulation examples.

According to the way they implement the CL constraints, two types of CL checking systems can be distinguished (see also (Mitamura and Nyberg, 2001)):

- *Prescriptive systems* specify all structures that are allowed when using the CL, i.e. the underlying grammar defines *how* texts must be written in the respective CL. Consequently, all sentences which cannot be parsed by the grammar are considered wrong and must be rewritten (Mitamura and Nyberg, 2001). This approach is taken by Caterpillar's Technical English (Kamprath et al., 1998).
- *Proscriptive systems* specify all structures that are not allowed when using the respective CL, i.e. the underlying grammar describes *what* needs to be rewritten. Within these systems, the approach to language checking is more relaxed, since only a partial checking of the documents is carried out. As (Mitamura and Nyberg, 2001) point out, documents still might have other problems which are not covered by the rules of the respective CL. Examples of proscriptive systems are the language checking tools CLAT (Schmidt-Wigger, 1998; Haller, 2000; Reuther and Schmidt-Wigger, 2000) and Diebold's Controlled Language Checker (Moore, 2000).

A *hybrid* architecture using both prescriptive and proscriptive specifications becomes necessary when CL checkers are paired with a corrector. In these systems, proscriptive rules detect non-compliant linguistic structures, i.e. *what* needs to be reformulated, whereas the prescriptive ones indicate *how* to reformulate the identified items. Such an architecture has been proposed as an extension to KANT CE Checker (Mitamura and Nyberg, 2001; Mitamura et al., 2003).

In the following section a correction module is proposed for the CLAT language checking tool. As in case of KANT CE Checker, a hybrid approach combining prescriptive and proscriptive CL rules is considered.

In order to develop an appropriate set of rules, we first present similar projects involving German CLs. Then, the style checking component of CLAT is presented in more detail. We show that the proscriptive style checking rules must be paired with appropriate prescriptive rewrite rules in order to be able to produce paraphrases for the stylistically inappropriate sentences. The paraphrasing strategies and the corresponding rewrite rules are presented in Section 3. A small experiment evaluating the performance of the paraphrasing tool is described in Section 4.

2.1 Controlling Style in German Technical Documents

There have been only a few attempts to design and implement a controlled German. *Siemens AG* developed *Siemens-Dokumentationsdeutsch* (SDD), a set of proscriptive rules controlling German grammatical constructions (Schachtl, 1996; Lehrndorfer and Schachtl, 1998) in order to facilitate the production of documentation that was to be subsequently machine processed, i.e. machine translated. The set of rules addressed editing issues such as NPs without determiners in headings, parenthetical constructions or punctuation as well as linguistic properties such as syntactic ambiguity, coordination, word order in conditional clauses, etc. However, SDD exists only as a research prototype and has

not been further developed.

Another approach was taken by Anne Lehrndorfer in her PhD thesis. She elaborated guidelines for a *Controlled German* (Lehrndorfer, 1996). The controlled vocabulary is considered to be a weak-point in the elaboration of a German CL. The syntax rules were elaborated taking into account psycho-linguistic considerations. The rules regulate, among others, word order, the use of the infinitive in instructions, coordination, they forbid relative clauses and indirect interrogatives etc. The suggested syntactic constructs are based on the most typical sentence structures of German and were selected according to content relations rather than considering linguistic criteria. Therefore, as the author mentions, the guidelines are more suitable when writing new documents than for document correction or reformulation (Lehrndorfer, 1996, 174). The thus elaborated *Controlled German* is a theoretical construct and has not yet been extensively tested. It can be considered only as a basis for a practically relevant controlled language.

A proscriptive approach to language control has been adopted for CLAT[1], a CL checker developed at IAI[2] in Saarbrücken. CLAT is a CL checker which was designed to support technical writers in producing high quality technical documentation by checking spelling, grammar, style and terminology in technical documents (Schmidt-Wigger, 1998; Carl et al., 2002; Hernandez and Rascu, 2004). CLAT has been in commercial use since 1998 and has been constantly improved and tuned to the needs of its users. However, the present architecture of the commercial tool does not include a corrector.

In Section 2.2 we focus on the style checking functionality of CLAT whereas in Section 3 we discuss how the tool can be extended to paraphrase inappropriate text fragments.

2.2 Style Checking with CLAT

The main task of the style checking component in CLAT is to identify phrase structures that are ambiguous or difficult to understand. Besides, company specific stylistic requirements are also checked in order to ensure compliance with corporate identity. The style checking functionality uses style constraints which were initially formulated with the help of experts in the field of technical documentation (Schmidt-Wigger, 1998) and were subsequently further tuned to the needs of technical writers. Presently there are 33 core rules checking conformance to general stylistic standards and 40 optional rules which allow the user to activate different sets of style rules in different configurations (Hernandez and Rascu, 2004).

Style constraints in CLAT address the following issues: layout, lexical problems, ambiguity, ellipsis, complexity, order of sentence constituents as well as other stylistic problems. Example (1) is detected by a lexical constraint prohibiting the use of the indefinite pronoun *man* whereas (2) was considered inappropriate on the basis of a syntactic rule checking the use of the verb *sein* (*to be*) followed by a *zu infinitive*. The CL implemented within CLAT states that neither construction is specific enough and may lead to misinterpretation.

(1) Um sich anzumelden, gibt man das Passwort ein.
(2) Die Schrauben sind zu ersetzen.

During style checking, stylistically inappropriate phrases are highlighted and the technical writer is prompted to reformulate the sentence. An error message as well as a typical example illustrating the problem and its stylistically adequate reformulation are also provided. A typical reformulation example for *sein + zu infinitive* is given below[3]:

\# An dieser Stelle sind die Einstellungen nochmals zu überprüfen.
⇒Überprüfen Sie an dieser Stelle nochmals die Einstellungen.

From the point of view of linguistic processing, style checking in CLAT involves three stages:

1. morphological analysis of the input text,

[1] All references in the article will be made to the German implementation of CLAT. The tool is also available for English, and in less developed form, for French, Italian, Spanish, and Swedish.

[2] Institut für Angewandte Informationsforschung, Saarbrücken

[3] Stylistically inappropriate sentences are marked with "#". Their reformulation is introduced by "⇒".

2. morpho-syntactic disambiguation of the analysed input using KURD[4], a shallow-parsing formalism (Carl and Schmidt-Wigger, 1998; Carl, 2001; Carl, 2005),

3. style checking, i.e. marking improper structures with an error code using KURD. On the basis of this information, CLAT offers at a later point a reformulation example and/or a detailed error message.

The linguistic structures marked with an error code are also the starting point for the paraphrasing process discussed in Section 3.

3 Paraphrasing

The main task of a corrector is to generate correct paraphrases for the stylistically inappropriate sentences. Mitamura and Nyberg suggest that in a proscriptive system each constraint must be paired with a prescriptive rewrite rule (Mitamura and Nyberg, 2001). However, often different reformulations are needed depending on the context.

Prescriptive optimisation rules, so-called paraphrasing principles, were formulated taking into account relevant research in various fields such as controlled languages (Lehrndorfer and Schachtl, 1998), text typology (Göpferich, 2002), speech act theory (Searle, 1970) as well as evaluating the TETRIS corpus available at IAI containing reformulation examples proposed by experts in the field of technical documentation. Subsequently, a set of stylistic paraphrases was implemented starting from these more general paraphrasing principles. The implemented stylistic paraphrases deal with the following phenomena.

3.1 Specification of Lexical Meaning

The stylistic constraints introduced to deal with specification of lexical meaning address two phenomena: lexical ambiguity and lexical vagueness. The distinction between lexical ambiguity and lexical vagueness is defined as a matter of whether two or more meanings associated with a phonological form are distinct, i.e. ambiguous, or united as non-distinguished sub-cases of a single more general meaning (Tuggy, 1993, 273). We illustrate the specification of meaning by providing reformulations to problems detected by two CLAT style rules.

For example, when using *betätigen* (*operate*) in a technical document, the reader is deprived of more specific information concerning the intended mode of operation. In case a button or a key is involved the paraphrasing principle in 3 is proposed.

(3) *Button/Taste betätigen*
⇒ *Button/Taste drücken*

(3a) #Zum Start eines Moduls betätigen Sie die Taste →.
⇒Zum Start eines Moduls drücken Sie die Taste →.

Three paraphrasing principles were formulated in order to specify the meaning of the indefinite pronoun *man*. The default principle (4) replaces the pronoun *man* with the polite form of the personal pronoun as shown in (4a). In (5) *man* is part of generalising phrases such as *man kann sagen* or *man merkt bis heute*. Following the principle of economy of expression, we propose to delete these phrases from the input structures (see example (5a)). The paraphrasing principle (6) offers a reformulation for *man* occurring in definitions. As illustrated in example (6a) the accusative or dative object of the input sentence becomes the subject of the reformulation whereas the finite verb is replaced by the corresponding form of the verb *sein*[5].

(4) *man* $V_{fiv,sg}$ ⇒ *Sie* $V_{fiv,pl}$
(5) *man* in generalizing phrase ⇒ NIL
(6) *man* + V_{fiv} + $NP_{obj,acc;dat}$ in definitions
⇒ $NP_{subj,nom}$ + $sein_{fiv}$

[4]KURD is an acronym representing the basic actions of the formalism: **K**ill–**U**nify–**R**eplace–**D**elete.

[5]The abbreviations used in the examples are as follows: V–verb, NP–noun phrase, fiv–finite verb, sg–singular, pl–plural, subj–subject, obj–object, acc–accusative, dat–dative. "," is used to indicate conjunction of values whereas ";" shows disjunction

(4a) #Um sich anzumelden, gibt man das Passwort ein.
⇒Um sich anzumelden, geben Sie das Passwort ein.

(5a) #Generell kann man sagen, dass die meisten CD-ROM-Laufwerke unterstützt werden.
⇒Generell werden die meisten CD-ROM-Laufwerke unterstützt.

(6a) #Unter Netzwerkdrucker versteht man meist Drucker, die ein Printserver-Netzwerkinterface eingebaut haben...
⇒Netzwerkdrucker sind meist Drucker, die ein Printserver-Netzwerkinterface eingebaut haben...

The paraphrasing rules, i.e. the actual implementations of the paraphrasing principles will be explained in Section 3.5.

3.2 Specification of Illocutionary Force

Vagueness in illocutionary force occurs when illocutionary indicators allow for a clear specification of the speech act class but not of the illocutionary force of an utterance. For example *sein + zu infinitive* can in many contexts be interpreted either as obligation or possibility. Experimental results show that *sein + zu infinitive* is commonly used with the sense of obligation in text genres pertaining to technical documentation (Kuntz, 1979, 217). Therefore, we suggest as default reformulation the imperative (7.1) or the infinitive used as imperative (7.2). However, it is not possible to reformulate *sein + zu infinitive* in subordinate clauses by using the imperative or the infinitive used as imperative. Therefore, distinct paraphrasing principles are formulated for main (MC) and respectively subordinate clauses (SC). Reformulated examples are given in (7.1a), (7.2a), and (8a).

(7 .1) *Sein + zu infinitive* in MC
⇒ imperative

(7.2) *Sein + zu infinitive* in MC
⇒ infinitive used as imperative

(8) *Sein + zu infinitive* in SC
⇒ *müssen* + infinitive

(7) #Dabei ist *Dateiname* durch den Namen der zu druckenden Datei zu ersetzen.

(7.1a) ⇒Dabei ersetzen Sie *Dateiname* durch den Namen der zu druckenden Datei.

(7.2a) ⇒Dabei *Dateiname* durch den Namen der zu druckenden Datei ersetzen.

(8a) #Welcher Druckbefehl zu nehmen ist, hängt vom Anwendungsprogramm ab.
⇒Welchen Druckbefehl Sie nehmen müssen, hängt vom Anwendungsprogramm an.

3.3 Reduction of Structural Complexity

We illustrate this process by means of a simple paraphrasing principle which is responsible for splitting up coordinated main clauses into separate sentences.

(9) coordinated MCs ⇒ separate sentences

(9a) #Der Kernel wird geladen und Sie werden aufgefordert, die erste Moduldiskette einzulegen.
⇒Der Kernel wird geladen. Sie werden aufgefordert, die erste Moduldiskette einzulegen.

3.4 Layout

Optimising layout is not necessarily a linguistic task. Certain linguistic structures, however, can offer clues for a better organisation of technical documents. For instance paraphrasing principle (10) specifies that coordinated subordinate clauses are to be paraphrased as lists as shown in (10a).

(10) coordinated SCs ⇒ list

(10a) #In der Liste erscheint beispielsweise, ob grundlegende Einstellungen neu vorgenommen oder ob Konfigurationsdateien verschoben wurden oder ob bekannte Programme modifiziert wurden.

⇒In der Liste erscheint beispielsweise:
- ob grundlegende Einstellungen neu vorgenommen wurden oder
- ob Konfigurationsdateien verschoben wurden oder
- ob bekannte Programme modifiziert wurden.

3.5 Linguistic Processing

In the preceding sections we presented some paraphrasing principles elaborated in order to optimise technical texts. The paraphrasing principles are actually implemented into a corrector, i.e. a paraphrasing component the task of which is to generate reformulations for stylistically inappropriate sentences. In case the input structure is marked with an error code indicating that a stylistic constraint is not respected, it is passed to the paraphrasing component. Paraphrasing in CLAT operates on input structures that have passed several linguistic processing steps: morphological analysis, morpho-syntactic disambiguation, and style checking (cf. section 2.2).

The paraphrasing component makes use of an ordered set of paraphrase rules implemented in KURD, a shallow parsing formalism. These rules describe the necessary rearrangement of an input structure so that an appropriate sentence can be produced. The paraphrase rules are applied successively and if a rule applies, the input structure is modified. The subsequent rules are thus applied to the modified input structure. Only one paraphrase can be produced for each input sentence since no backtracking is performed.

Figure 1 presents a simple lexical paraphrase rule, one of the possible implementations for paraphrasing principle (4). The linguistic information is encoded by means of feature bundles consisting of attribute-value pairs[6]. The rule in the figure replaces the value of the *lu* feature (i.e. *man*) by the polite form of the personal pronoun and the number information in the description of the verb is adjusted accordingly, i.e. it receives the value *plu*.

The rule consists of three parts, a rule identifier (*r330_1*), a description part and an action part. The description part consists of conditions that must match successive nodes in the input and marks with capital letters the nodes that need to be modified (here A and B) in the action part. Conditions are formulated in KURD with the help of two quantifiers: the existence quantifier *e* and the all quantifier *a*. The action part specifies the operations to be performed on the marked nodes in case the rules apply. In rule *r330_1* the description part specifies that the pattern needs to begin with the pronoun *man* and end with a finite verb form in the present tense. Zero or more elements other than a finite verb may intervene between these two (∗*e*{*vtyp*∼=*fiv*}). In the action part, feature bundles in the nodes marked with A and B are modified by means of the replace operator *r*.

```
r330_1 =
Aa{c=w,sc=pron,lu=man},
*e{vtyp~=fiv},
Be{c=verb,vtyp=fiv,tns=pres}
:  Ar{lu=Sie},
   Br{nb=plu}.
```

Figure 1: Paraphrase Rule

In a last step, a token generator produces the surface forms of text fragments from the morphological representation that has been modified during the paraphrasing process.

4 Evaluation

The paraphrasing strategy outlined in the previous sections was tested on an excerpt from a computer manual[7]. The test text *T* consists of 43.381 words. Within the test we evaluated the paraphrases produced for inappropriate structures detected by ten style rules of CLAT's style checking component. Thus, a total of 321 non-compliant sentences were identified, 237 of

[6] the features used in the figure are: c–category, sc–subcategory, lu–lexical unit, vtyp–verb type, tns–tense, nb–number.

[7] Behlert & al.: SUSE LINUX 9.2 – Administrationshandbuch (Copyright SUSE LINUX AG)

which were correctly reformulated. 44 structures were wrongly paraphrased and 40 more were not reformulated at all.

We computed precision as the ratio of the correct paraphrases over all paraphrases produced by the system and recall as the ratio of the correct paraphrases over all annotated paraphrases. The f-score is (*2 * precision * recall*) / (*precision + recall*). Figure 2 summarises the results of the experiment.

	T
f-score	0.79
precision	0.84
recall	0.74

Figure 2: Evaluation

The inappropriate paraphrases were mainly due to an insufficient context disambiguation. For instance, structures identified by the stylistic rule *Avoid the use of "sollen"* may, among other, express warnings or suggestions/orders and must be paraphrased differently depending on the context. Insufficient identification of the illocutionary indicators in the input sentence lead to incorrect paraphrases as illustrated in example (11).

(11) #Auf keinen Fall sollten Sie überhaupt keinen Swap-Speicher anlegen.
#Auf keinen Fall legen Sie überhaupt keinen Swap-Speicher an.

The system failed to produce paraphrases for a range of stylistically inappropriate structures due to lack of appropriate paraphrase rules. For a small percentage of these "misses", paraphrase rules can and will be implemented in the future. But for the was majority, no reliable paraphrase rule can be formulated, because the linguistic information available in the context is insufficient. For example, no paraphrase was produced for the sentence (12) which is non-compliant with the style rule *Avoid using the verb "durchführen"*, since no object of the verbal derivate *Überprüfungen* is present in the sentence. A successful reformulation of the same stylistic problem is illustrated in example (13).

(12) #Anhand der installierten Datenbank lassen sich auch Überprüfungen durchführen.

(13) #Das Sichern der Daten ist als Systemadministrator root durchzuführen; nur root hat die Rechte, alle lokalen Dateien zu lesen.
⇒Die Daten sichern Sie als Systemadministrator root. Nur Root hat die Rechte, alle lokalen Dateien zu lesen.

5 Conclusions

In this paper we presented a controlled language approach to text optimisation in technical documentation. A hybrid approach is proposed in order to complement the style checking functionality of the CLAT CL checker with a corrector able to reformulate the inappropriate text fragments detected by the system. Paraphrases are used as instrument to the optimisation process.

An evaluation of the optimisation process reveals that precision needs to be further increased by paraphrase rules more efficient in detecting the illocutionary indicators contained in the input. For a corrector, high precision is essential, since wrong rewrite suggestions are extremely annoying for technical authors and decrease the acceptability of CL checkers (see also (Wojcik and Holmback, 1996)).

References

Michael Carl and Antje Schmidt-Wigger. 1998. Shallow Postmorphological Processing with KURD. In *Proceedings of NeMLaP3/CoNLL98*, Sydney.

Michael Carl, Johann Haller, Christoph Horschmann, Dieter Maas, and Jörg Schütz. 2002. The TETRIS Terminology Tool. *TAL, Structuration de terminologie*, 43(1):31–35.

Michael Carl. 2001. *Example-based Decomposition, Generalization and Refinement for Machine Translation*. Ph.D. thesis, Universität des Saarlandes.

Michael Carl. 2005. *KURD*. IAI, electronic working paper 38. forthcoming.

M. Dras. 1999. *Tree Adjoining Grammar and the Reluctant Paraphrasing of Text*. PhD Thesis, Macquarie University.

Susanne Göpferich. 2002. *Textproduktion im Zeitalter der Globalisierung. Entwicklung*

einer Didaktik des Wissenstransfers. Stauffenburg Verlag, Tuebingen.

Johann Haller. 2000. Sprachtechnologie für die Automobilindustrie. In W. Wills, editor, *Weltgesellschaft – Weltverkehrssprache – Weltkultur*, pages 250–263. Stauffenburg, Tübingen.

Lina Henriksen, Bart Jongejan, and Bente Maegaard. 2004. Corporate Voice, Tone-of-Voice and Controlled Language Techniques. In *Proceedings of LREC–2004*, volume 5, pages 1621–1624, Lisbon, Portugal.

Maryline Hernandez and Ecaterina Rascu. 2004. Checking and Correcting Technical Documents. In Séverine Vienney and Mounira Bioud, editors, *Correction automatique: bilan et perspectives*, number 29 in BULAG, pages 69–84. Presses universitaires de Franche-Comté.

Christine Kamprath, Eric Adolphson, Teruko Mitamura, and Eric Nyberg. 1998. Controlled Language for Multilingual Document Production: Experience with Caterpillar Technical English. In *Proceedings of CLAW–1998*, pages 51–61, Pittsburg, PA.

Helmut Kuntz. 1979. *Zur textsortenmäßigen Binnendifferenzierung des Fachs Kraftfahrzeugtechnik*. Kümmerle Verlag, Göppingen.

Anne Lehrndorfer and Stefanie Schachtl. 1998. Controlled Siemens Documentary German and TopTrans. In *TC Forum*, volume 3.

Anne Lehrndorfer. 1996. *Kontrolliertes Deutsch. Linguistische und sprachpsychologische Leitlinien für eine (maschinell) kontrollierte Sprache in der Technischen Dokumentation*. Gunter Narr Verlag, Tübingen.

Teruko Mitamura and Eric Nyberg. 2001. Automatic Rewriting for Controlled Language Translation. In *Proceedings of NLPRS–2001. Workshop on Automatic Paraphrasing: Theory and Application*.

Teruko Mitamura, Kathryn Baker, Eric Nyberg, and David Svoboda. 2003. Diagnostics for Interactive Controlled Language Checking. In *Proceedings of EAMT/CLAW–2003*, pages 87–94.

Corinne Moore. 2000. Controlled Language at Diebold, Incorporated. In *Proceedings of CLAW–2000*, pages 51–61, Seattle, WA.

Sharon O'Brien. 2003. Controlling Controlled English. An Analysis of Several Controlled Language Rule Sets. In *Proceedings of EAMT/CLAW–2003*, Dublin, Ireland.

Ursula Reuther and Antje Schmidt-Wigger. 2000. Designing a Multi-Purpose CL Application. In *Proceedings of CLAW–2000*, Seattle, WA.

Stefanie Schachtl. 1996. Requirements for Conrolled German in Industrial Applications. In *Proceedings of CLAW–1996*, pages 143–149, Leuven, Belgium.

Antje Schmidt-Wigger. 1998. Grammar and Style Checking for German. In *Proceedings of CLAW–1998*, pages 76–86, Pittsburg, PA.

John R. Searle. 1970. *Speech Acts. An Essay in the Philosophy of Language*. Cambridge University Press.

David Tuggy. 1993. Ambiguity, polysemy, and vagueness. *Cognitive Linguistics*, 4(3):273–290.

Richard H. Wojcik and Heather Holmback. 1996. Getting a Controlled Language Off the Ground at Boeing. In *Proceedings of CLAW–1996*, pages 22–31, Leuven, Belgium.