Integration of Correction Modules in a Controlled Language Application

Svetlana SHEREMETYEVA

LanA Consulting ApS
Copenhagen, Denmark
lanaconsult@mail.dk
+45 33 25 04 41

Abstract. The paper describes correction modules for AutoPat, - an authoring system for patent claims. The specificity of AutoPat is that it relies on human-computer content specification in controlled language. The quality of the textual input is a crucial point in getting a high quality AutoPat output. Our correction modules handle both the quality of human input and the final system output. The human input is passed through a spell-checker, a grammar checker and a content checker. An application-tuned grammar checker is run over the system final output of the generation module.

1 Introduction

The quality of input is a crucial point for an NLP application, e.g., machine translation and authoring, whose primary design focus is getting a high quality output. A wide range of activities can be found in the area of developing correction (proofing) tools for textual input. Such tools fall in two major categories, - a) spell- and grammar checkers to deal with spelling and grammar mistakes and b) controlled language checkers for rewriting linguistically correct text into a controlled language input to ensure that texts conform to the desired vocabulary and grammar constraints. This, in turn, improves the chance of achieving high quality output (Bernth, 1998; Mitamura & Nyberg, 1995).

Correction tools are applied both for pre-editing and postediting (Mitamura, 1999; Bernth, cf.) and tend to be integrated into an NLP application as its essential components. For example, in the KANT machine translation system the controlled grammar is built into a grammar-checking component, which uses the same parsing engine as the source text analyzer (Mitamura, cf.).

Correction tools are often created for interactive checking of texts, and attempt to offer alternatives whenever possible (Nyberg et al., 2003).

Controlled language in machine translation and document authoring can be used to both improve the coverage of the system and to develop better proofing tools. In other words, in checking rules responsible for linguistically correct output one can exploit restrictions of the application controlled language thus saving on knowledge acquisition and improving the quality of the corrections. We attempt just that.

The paper describes correction modules for AutoPat, - an authoring system for patent claims. The specificity of AutoPat is that it is equipped with a knowledge elicitation scenario and relies on joint human-computer content specification in controlled

language. The quality of the textual input is a crucial point in getting a high quality output. Our correction modules are application-tuned; exploit restrictions of the AutoPat controlled language and designed to handle both the quality of human input and the final system output. They share the system knowledge base over a rich space of semantic, syntactic and morphological features and are integrated into the system at different stages of the authoring process. The human input is passed through a spell-checker, a grammar checker and a content checker. The grammar checker is run over the system final output to correct possible mistakes of the generation module. All correction has been implemented for the English language but can be extended to other languages.

In what follows we fist give an overview of the application and the restrictions of the AutoPat controlled language. We then describe the lexicon feature space, and show how the controlled language environment is used in AutoPat correction tools.

2 AutoPat overview

AutoPat is an authoring system for composing patent claim designed to produce a high quality output, a claim text that is linguistically correct and meets strict requirements to its structure imposed by Patent Law. It generates a single, albeit possibly very complex, sentence with a well specified conceptual, syntactic and stylistic/rhetorical structure. Figure 1 illustrates a fragment (claims can be over a page long) of an AutoPat output.

A cassette for holding excess lengths of light waveguides in a splice area ***comprising*** *a cover part and a* ***pot-shaped*** *bottom part* ***having*** *a bottom disk and a rim* ***extendin****g perpendicular to said bottom disk, said cover and bottom parts* ***are superimposed*** *to enclose jointly an area* ***forming*** *a magazine****….***

Figure 1. A fragment of a patent claim text generated by AutoPat.

In the process of a computer interview the system elicits knowledge about the invention via an interface, to which the user supplies words or phrases filling the slots of predicate templates. These templates are user interface images of predicate/argument structures stored in the lexicon; every slot shown in the interface corresponds to a certain case-role in the lexicon. For example, the slot « where » corresponds to the case-role « place », the slot « what for » is in fact the case-role « purpose », etc., see Figures 2 and 3.

The predicate templates are displayed in the interface following the user selection in a predicate menu. As immediate feedback, the engine generates and displays one short sentence in English for each claim feature that the user has described in the interface. Simultaneously AutoPat builds a deep content representation and then transforms it into a final surface claim text as shown in Figure 1 (Sheremetyeva and Nirenburg, 1996; Sheremetyeva, 2003). The system thus lets its users compose a patent claim ready for submission to a patent office by limiting user input to the lexical and phrasal level.

3 AutoPat controlled language

AutoPat controlled language draws heavily on the patent claim sublanguage. It is designed on a 9- mio word corpus statistics, which, on the one hand, makes it extremely user-friendly and, on the other hand, allows for a great amount of effort saving in knowledge acquisition and better system performance.

AutoPat controlled language is what can be called a relaxed controlled language in that it relies on

- a partially controlled lexicon, including a closed lexicon of predicates (boldfaced in Figure 1), and a "free" lexicon to form predicate arguments
- an implicitly controlled grammar which is associated with a controlled set of predicate/argument templates rather than with syntactic sentence-level constraints
- authoring memory (Allen, 1999), which AutoPat creates every time the user fills a predicate slot.

4 Feature space

The AutoPat correction modules work on the output of the system analyzer over a rich space of features. Based on the user input the analyzer produces an internal representation of the claim content in the form:

text::={ template){template}*
template::={predicate-class predicate ((case-role)(case-role}*)[1]
case-role::= (status value)
value::= {word tag}*

where

predicate-class is a semantic class of a predicate (such as, meronymy, location, connection, etc.) as specified in the predicate lexicon,

predicate is a string corresponding to one of predicates from the predicate lexicon,

case-role is a predicate argument of a certain *status*.

status identifies a semantic status of every case-role as "agent", "place", "mode", etc. thus specifying a case frame for every predicate

value is a string, which fills a case-role.

tag is a label from the lexicon that codes a typed feature structure. For example, nouns tags have the structure:

[POS
[Noun
[object [plural, singular]
process-ing [plural, singular]
process-other [plural, singular]
substance [plural, singular]
measuring unit [plural, singular]
parameter [plural, singular]
other [plural, singular]]]]]

Correction tools are run over a feature space, which include the semantic, syntactic and morphological features coded in tags. When run on the template slot fillers phrase borders and the status of the case-role the phrase fills augment the feature space..

[1] template in the content representation is retrieved from a predicate lexicon following the user's predicate selection from a system menu. The interface presentation of this template is shown in Figure 3.

5 Correction modules

As mentioned earlier our correction modules are application-tuned, exploit restrictions of the AutoPat controlled language and designed to handle both the quality of human input and the final system output. Corrections are thus performed twice, - at the pre-generation stage, and at the postgeneration stage. Checking procedure is interactive to notify the author when the author's input may not be appropriate, and attempts to offer alternatives whenever possible, which the user can approve by a single mouse click.

We cannot but stress again that user input quality is of special importance in AutoPat. The generator can only be expected to produce high quality claim texts provided the user correctly fills the system predicate templates. The generator treats the case-role fillers as blocks and determines the order of these case-role fillers and predicates in the output claim text following a case-role linear pattern stored in the predicate lexicon (Sheremetyeva cf.).

For example, if the ill formed filler "*steel rim at least one" in the template shown in Figure 2 had not been corrected the grammar mistake would have been transferred to the final claim text. This applies to misspelled words as well. Else, if the phrase "*basically perpendicular*" in the template shown in Figure 3 had been typed in the slot "What" the invention feature-sentence would have been generated with the wrong word order: "**basically perpendicular to said bottom disk extends at least one steel rim for rotation"*. Wrong or omitted prepositions also cause problems. For example, if in the template shown in Figure 3 the phrase filling the slot "What for" (case-role "purpose") had been left uncorrected, it would have distorted the meaning of the invention feature (sentence) as shown in Figure 3: *" at least one steel rim extends basically perpendicular to said bottom disk rotation"*.

During our testing period we discovered that in spite of all the user-friendliness of the system we cannot always expect an absolutely correct input of a regular user who is not a linguist, might not be a technical writer (but, e.g., an inventor), and/or not necessarily a native English speaker[2]. After all, the user can simply mistype or omit words. We tuned our correction modules to the following types of the erroneous user input:

a) spelling mistakes (corrected by the AutoPat spell-checker module)

b) slot fillers with omitted or wrong prepositions

c) ill formed slot fillers most frequently input by non-native English speakers, who are influenced by their native language structures. Here the situation is reverse to that of pre-editing in MT, when well-formed English phrases/sentences are sometimes intentionally distorted to get closer to the target language. We have to rewrite such distorted phrases into well-formed controlled English.

d) well-formed slot fillers erroneously put in wrong template slots.

To deal with mistakes of types b), c) and d) we developed what we call a content checker. The content checker differs from a regular grammar checker in that it works over a more powerful feature space than a free text grammar checker. It "knows" phrase borders and case-role semantic status.

---

[2]AutoPat was tested by both native English speakers and non-native English (Russian, German and Japanese) speakers. Rules a) and b) are written to cover actual mistakes done by a non-native English speaker.

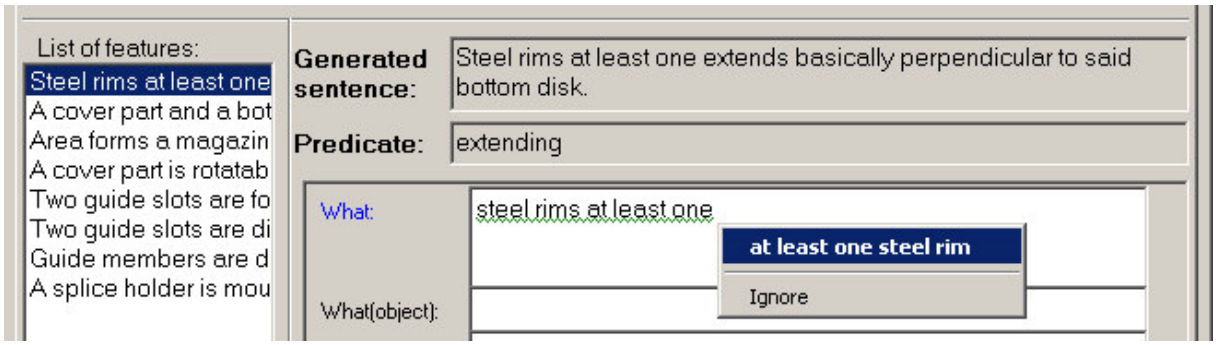

Figure 2. A diagnostic message in the user interface called by the rule a) above. The grammar checker identified an ill formed phrase and produced interactive messages suggesting correction actions. Correction is done automatically by mouse-clicking

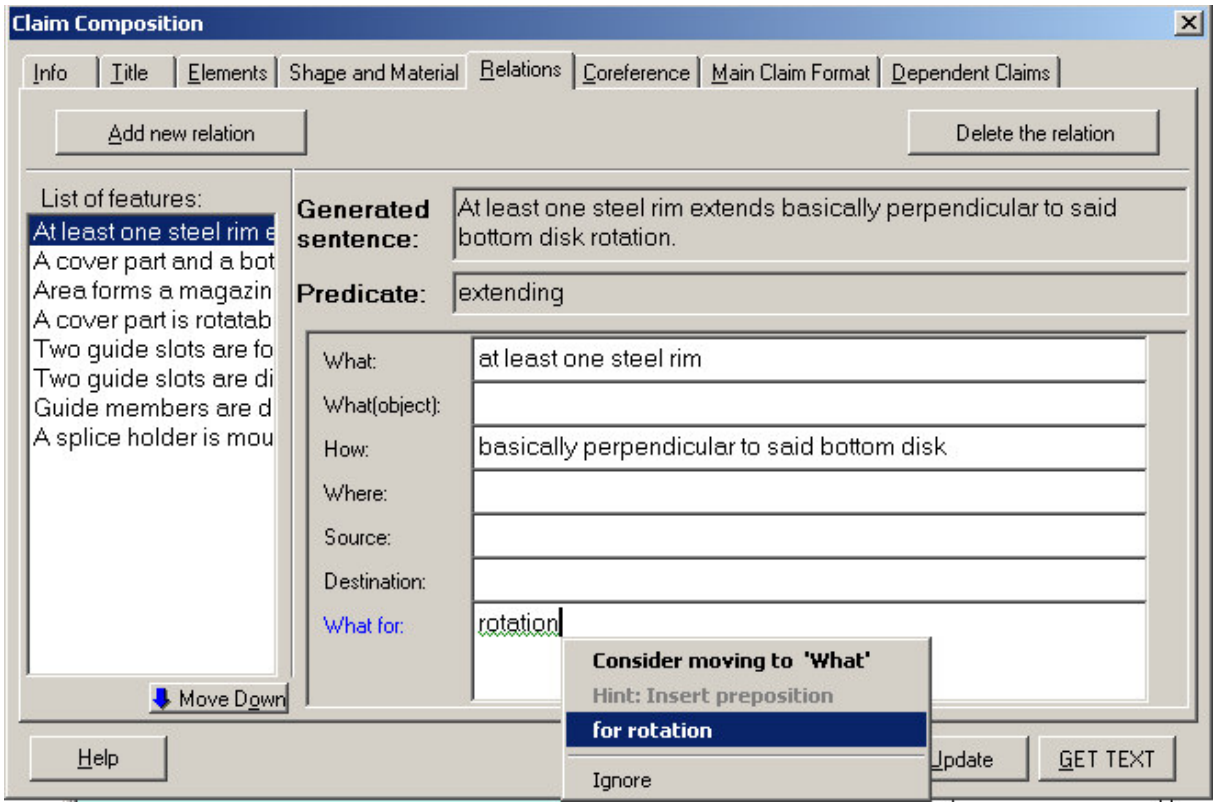

Figure 3. A screenshot of the user interface with a predicate template. The content checker identified a wrong filler in a template slot "purpose" and produced interactive messages suggesting correction actions (see rule b) above).

The content checker thus takes care of correct filling of predicate templates. It checks whether the user put a filler phrase in the right template slot and whether this filler meets predicate/argument selectional restrictions. The content checker restores or corrects filler's prepositions, and, where necessary, semantics. For example, the phrase "*for the*

table", though syntactically correct would have been ruled out as a filler of the slot "What for" (case-role "purpose"), as the "purpose" case-role requires an abstract head noun.

Our correction technique is a combination of analysis and pattern matching. Correction modules take unambiguously tagged case-role fillers output by the analyzer and match them against domain specific set of rules reflecting controlled grammar in the system knowledge base. In case a pattern is found a correction module returns a diagnostic message or warning and the corresponding segment is marked. The AutoPat correction modules incorporate different diagnostic messages. Most of the messages suggest rewrites or moves to a different slot, which the user can approve by a single mouse click. Currently the user is supposed resolve any problems which are found by interacting with the system, but we have started experimenting with completely automatic correction modules.

Below we give examples of grammar (a), and content (b) checking rules (see also user interface screenshots in Figures 2 and 3):

a)
```
IF T=~Nsg{1}~Npl{1}~Qu{1}~Num{1;"one"}
THEN
    HINT ~Qu~Num~Nsg*Npl
```

b)
```
IF  (T=~NF{1}) AND (PURPOSE)
THEN
   HINTMOVE "Consider moving to  'What' " ,SUBJ
   HINTTEXT "Insert preposition"
   HINT +Prep(for) ~NF
```

where

the rule a) reads " If a string of tags starts with one singular noun followed by one plural noun followed by one quantifier and ends in one numeral with the gloss "one", then reorder the string so as it starts with the quantifier followed by the numeral, followed by the former first noun in singular and ends in the second noun whose form is changed into noun, plural"; and

the rule b) reads "If the current string of tags stats with one noun of the semantic class "functional" and if this string is the filler of the case-role "purpose" then either move this filler to the case-role "what"- subject or insert the preposition "for".

As mentioned earlier corrections are performed twice, - at the pre-generation stage when spelling, grammar and content checkers are run in turn on the analyzed human input (tagged strings in predicate templates), and at the postgeneration stage on the tagged claim text generated by the AutoPat engine.

Only one of the correction modules, - the grammar checker, is used to correct the generator output for obvious reasons. Correction rules of the grammar checker include rules tuned to predictable mistakes of the user and AutoPat engine. At the final stage of AutoPat processing the grammar checker in fact performs as an interactive/automatic posteditor.

.

6 Conclusion

We presented correction modules integrated in AutoPat, - a generator of patent claims, whose efficiency is conditioned by the controlled language framework. Correction modules include an application tuned spellchecker, grammar checker and content checker. The modules are designed to handle the quality of human input and the final system output. Corrections are performed twice, - at the pre-generation stage, and at the postgeneration stage. Correction rules are crafted based on the typology of human mistakes and predictable mistakes of the generation engine.

In general, both the grammar- and content checkers rewrite erroneous human input and generator output into a controlled grammar text. The predicate lexicon is strictly controlled by the system, while the argument lexicon is to some extent controlled by the AutoPat authoring memory (Sheremetyeva, cf.). Pre-generation corrections make the human input more "comfortable" for the generation engine; postgeneration correction produces a controlled language claim text meeting all legal requirements which is easier for further processing, e.g. machine translation or information retrieval.

Preliminary evaluation results show a reasonably small number of proofing failures, mainly due to the incompleteness of analysis and correction rules. We are currently enhancing our knowledge base with new linguistic features and analysis/correction rules to make our correction modules fully automatic.

References

Allen, J. (1999). Adapting the concept of "Translation memory" to "Authoring memory" for a Controlled Language writing environment. Proceedings of the 21st Conference of Translating and Computer" 10-11 November . London.

Bernth, A. (1998). "Easy English: Preprocessing for MT". In proceedings of the Second International Workshop on Controlled Language Applications (CLAW-98).

Mitamura, T. (1999). Controlled Language for Multilingual Machine Translation. Proceedings of Machine Translation Summit VII, Singapore, September 13-17.

Mitamura, T. and Nyberg, E. (1995). "Controlled English for Knowledge-Based MT: Experience with the KANT System". In proceedings of the 6th International Conference on Theoretical and Methodological Issues in Machine Translation (TMI-95).

Nyberg E., T Mitamura, D. Svoboda, J. Ko, K. Baker, J. Micher (2003). An Integrated system for Source language Checking, Analysis and Terminology management. *Proceedings of Machine Translation Summit IX, September. New-Orleans.USA*

Sheremetyeva S.and S.Nirenburg (1996) Interactive Knowledge Elicitation in a Patent Expert's Workstation. IEEE Computer. Vol.7.

Sheremetyeva S. Towards Designing Natural Language Interfaces (2003). *Proceedings of the 4th International Conference "Computational Linguistics and Intelligent Text Processing"* Mexico City, Mexico, February 16-22.