NL Assistant: A Toolkit for Developing Natural Language Applications

Deborah A. Dahl, Lewis M. Norton, Ahmed Bouzid, and Li Li

Unisys Corporation

Introduction

We will be demonstrating a toolkit for developing natural language-based applications and two applications. The goals of this toolkit are to reduce development time and cost for natural language based applications by reducing the amount of linguistic and programming work needed. Linguistic work has been reduced by integrating large-scale linguistics resources—Comlex (Grishman, et. al., 1993) and WordNet (Miller, 1990). Programming work is reduced by automating some of the programming tasks. The toolkit is designed for both speech- and text-based interface applications. It runs in a Windows NT environment. Applications can run in either Windows NT or Unix.

System Components

The NL Assistant toolkit consists of

1. a natural language processing engine (Dahl, 1992)
2. lexical and semantic servers based on Comlex and WordNet (Grishman et.al, 1993, Miller, 1990)
3. template files which serve as the basis for new applications
4. a graphical toolkit for entering linguistic and application-related information
5. algorithms for automatic rule generation based on developer input.

Natural Language Development

Two strategies address the goal of minimizing the amount of linguistics expertise required to develop applications with the NL Assistant toolkit. To reduce the amount of lexical information that the developer must add, large-scale lexical resources have been integrated with the toolkit. These include Comlex and WordNet as well as additional, internally developed resources. The second strategy is to provide easy to use editors for entering linguistic information.

Servers

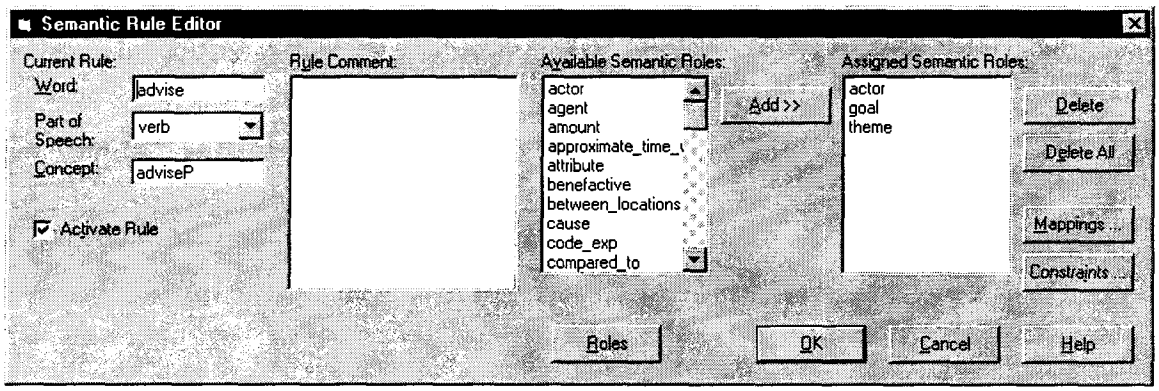
Lexical information is supplied by four external servers which are accessed by the natural language engine during processing. Syntactic information is supplied by a lexical server based on the 50K word Comlex dictionary available from the Linguistic Data Consortium. Semantic information comes from two servers, a KB server based on the noun portion of WordNet (70K concepts), and a semantics server containing case frame information for 2500 English verbs. A denotations server using unique concept names generated for each WordNet synset at ISI links the words in the lexicon to the concepts in the KB. When the engine is connected to the servers, whenever lexical information for specific words cannot be found in the local engine, it is requested from the servers. When the servers cannot supply lexical information for a particular word, various heuristics are used to hypothesize the missing information. When hypothesized information is wrong, it can be corrected by using the editors.

Editors

Although the servers minimize the amount of linguistic work that needs to be done to develop an application, they do not eliminate it. The main reason for this is that a particular application will make use of words that do not exist in the dictionary. For example, in our initial applications these have included words such as the verbs 'OEM', 'interface', 'download', and 'customize'. To improve the

ease of use of the linguistic development environment, several special-purpose editors have also been implemented. These include editors for lexical, semantic, and knowledge-base work.

The figure below illustrates an editing session in the semantic rule editor for the verb *advise*.



Application Development

Linking the application-independent semantic representation to the back-end application software is the task of the application module. To reduce the amount of time required to develop an application and the amount of expertise required, we have structured the application module into a set of several different types of rules. (Norton, et. al., 1996) The tasks these rules have to perform are: (1) map the user's utterance into an utterance class which consists of pragmatically equivalent utterances. (2) determine, based on the user's utterance, the state of the dialog, and any other information relevant to the application (such as the state of a database) what to do next (3) perform the next action or set of actions. Actions include saying something to the user, retrieving information from a database, and resetting the dialog to a new state. These rules are written in Prolog. The toolkit provides an editor for editing and managing these rules. In addition, the toolkit provides tools for automatically generating rules in the special case of applications which do not control a dialog.

Applications

We will demonstrate a web-based text application on the topic of the NL Assistant product. We will also demonstrate a speech recognition application for mortgage quotations, our Mortgage Assistant product.

Metrics

The largest application we have developed to date has 37 answer classes on the topic of the NL Assistant product. It used 683 training sentences and achieved a score of 83% first answer correct and 88% first or second answer correct on a live test with 144 queries. It required approximately two person-months to develop.

References

Dahl, Deborah A. ``Pundit—Natural language interfaces''. In G. Comyn, N.E. Fuchs, and M. J. Ratcliffe, eds. Logic programming in action. Springer-Verlag, 1992.

Grishman, Ralph, Catherine Macleod and Susanne Wolf. ``The Comlex syntax project''. ARPA Human Language Technology Workshop. Morgan Kaufmann, 1993.

Linebarger, Marcia C., Lewis M. Norton, and Deborah A. Dahl. ``A Portable approach to last resort parsing and interpretation''. ARPA Human Language Technology Workshop, 1993.

Miller, George A. ``Five papers on WordNet''. International Journal of Lexicography. 1990.

Norton, Lewis M., Carl E. Weir, Ahmed Bouzid, Deborah A. Dahl, and K.W. Scholz. ``A methodology or application development for spoken language systems''. Proceedings of ICSLP96. Philadelphia, PA, 1996.