Artificial Intelligence Review **14:** 277–281, 2000.
Stephen J. Hegner et al. (eds.), Intelligent help systems for UNIX.
© 2000 *Kluwer Academic Publishers. Printed in the Netherlands.*

Editorial
Intelligent Help Systems for UNIX: Natural Language Dialogue

This is the last of a series of three special issues on *intelligent help systems for UNIX*.[1] This issue addresses *natural language dialogue* whereas the previous issues focussed on *computational models and systems* and *planning and knowledge representation*, respectively. In this collection, the focus is not upon issues of parsing and production per se, even though several of the systems described herein have significant capacities in this dimension. Instead, work here has evolved more intimately within the context of consultation systems, a topic seldom dealt with by other natural language systems. Nevertheless, the issues discussed are of general concern within natural language processing.

In the first of two papers by **Chin**, the UCEgo component of UC[2] is described. It implements an “active” intelligent agent with its own goals and plans to act on behalf of UC. UCEgo volunteers information, corrects user misconceptions and rejects unethical requests when appropriate. It adopts goals when it notices that the user lacks necessary knowledge, or has incorrect beliefs and then plans to volunteer information or correct the user’s misconception as appropriate. Plans are pre-stored skeletons that are indexed under the types of situations in which they are typically useful. Plan suggestion situations include the goal for the plan, plan preconditions, and appropriateness conditions.

In his second paper, **Chin** presents the UCExpress answer-expression component of UC. Strategic issues in generation are the primary topic; specifically, the problem of expressing answers once they have been computed is addressed in detail. Following the philosophy of natural language systems that use an internal representation language, the machinations or thought processes of such systems are performed in terms of this language. Thus, in response to a user’s request, such a system formulates a set of propositions or a piece of network denoting the “conceptual response” to the user’s query. This conceptual response must then be conveyed to the user in natural language through a process of generation. However, direct translation of formulas in an internal representation language to natural language is generally unwise. The propositions comprising a conceptual response are likely to contain much information that is already known to the user. In addition, there are usually several styles or formats in which a conceptual answer may be conveyed. Hence the natural language system must decide not only which conceptual response to convey, but also which format to convey it in. Chin calls this process of pruning and formatting “answer expression.” The result of answer expression is a new set of propositions to be conveyed to the user in natural language, a process of tactical generation which is not described herein. Actually, the term “answer expression” is unnecessarily restrictive. Given that an agent of any kind has something to express, the problem always arises of selecting both the appropriate portion to convey and a format in which to convey it. Thus, answer expression is not so much a process specific to answering questions in the act of consulting as it is part of a general process of deciding how to express oneself in context. Much of the analysis of answer expression in the paper appears to be formulated in a way that makes it applicable to the general case of expression, and not just the specific cases to which it is applied.

Mayfield focusses on plan recognition in dialogue systems – the process of explaining why an utterance was made in terms of the plans and goals that its speaker was pursuing in making the utterance. He proposes three criteria for making judgements about the merits of explanations: applicability, grounding and completeness. *Applicability* refers to the needs of the system that will use the explanation, *grounding* refers to situating the explanation in what is known of the speaker and the dialogue, and *completeness* refers to how comprehensively the explanation covers the goals that motivated the production of the utterance. These criteria are applied in designing and evaluating a plan recognition algorithm and its associated knowledge base.

The work of **Martin** is concerned with a very different natural language topic, namely, handling metaphorical language. The focus is the manner in which metaphors in the UNIX domain are represented in MIDAS (Metaphor Interpretation, Denotation, and Acquisition System), a computer program developed and based on the explicit representation of knowledge about metaphors. It may be surprising to find metaphoric language as a prominent concern of building systems that converse about operating systems. However, a compelling case is made that much of ordinary language, even that used in quasi-technical domains, is intrinsically metaphoric. In doing so, the traditional dichotomy between metaphoric and conventional language is challenged, and the notion of metaphoric word senses is introduced. Much of Martin’s work is concerned with representing the “conceptual metaphors” that underlie these word senses and which relate senses of various lexical items to one another. The most significant application of these metaphoric representations is knowledge acquisition. In particular, it is shown how a system with such representations can hypothesize a new word sense of a word with which it is already familiar. Such a capability turns out to be especially useful in learning the language with which people talk about computational matters, since, in this domain, the use of word senses that are extended in precisely this manner seem to proliferate. This work is an important step in the direction of designing systems which can extend their own vocabulary, a problem that will become increasingly important as natural language systems become deployed in the real world.

Quilici investigates how users' mistaken beliefs about UNIX commands can be corrected. He shows how knowledge about specific UNIX-related plans and abstract knowledge about how beliefs can be justified can be represented and how it can be used to form justifications for advisor beliefs and to understand justifications given for user beliefs. Knowledge is captured in *justification patterns*, which are domain-independent knowledge structures that are similar to the abstract knowledge structures used to understand the point behind a story. Justification patterns enable the advisor to understand and formulate novel belief justifications, giving the advisor the ability to recognise and respond to novel misconceptions.

To sum up, the problem of how to enable a consultant to become "active" and intervening when it believes the user to have misconceptions is one which must be addressed. It is illuminating to compare this requirement of an "active" consultant to that embodied in *active* (as opposed to *passive*) help systems, as described in papers in the first special issue in this series. The papers in the current issue also bring to the forefront the importance of planning in terms of both recognition and production for understanding and responding to the user's utterances. Finally, we see how phenomena such as metaphor in language can even seep into specific domains like UNIX, and that for full understanding it needs to be processed.

The articles in this issue are derived from papers originally presented at a workshop entitled *Knowledge representation in the UNIX help domain*, organised by Peter Norvig, Wolfgang Wahlster and Robert Wilensky at the University of California, Berkeley, USA, in December, 1987. The workshop was funded, in part, by the International Computer Science Institute and all participants were invited. The area of intelligent help systems for UNIX provides such a fruitful example domain of application for many AI techniques that we have decided to publish this work, which is still timely, more widely and particularly now where we have many spoken dialogue systems applied to such fixed domains (see Bernsen et al. 1998, BusinessWeek 1998).

Each article here has been reviewed by the editors and has been subsequently revised; furthermore, all authors have been asked to include a section on *recent developments* on their work. Related work which may be of interest to the reader can be found in Kobsa and Wahlster (1988) which focusses on user modelling and in Maybury and Wahlster (1998) which is a comprehensive publication on intelligent user interfaces. Members of the Kluwer editorial and publishing staff are to be thanked for their help in producing this issue. It would not have been possible without Melanie Willow (Journals Editorial Office), André Diepenbroek (Desk Editor, Editorial Department), and Vanessa Nijweide (Kluwer Publishing Editor, Humanities and Social Sciences). Melanie and André have worked very hard with authors and reviewers to enable this to happen.

The Editors, October, 2000.

Stephen J. Hegner
Umeå University
Sweden

Paul Mc Kevitt
University of Ulster (Magee), Derry
Northern Ireland

Peter Norvig
NASA Ames Research Center
California, USA

and

Robert Wilensky
University of California at Berkeley
USA

Notes

1. UNIX is a trademark of X/Open, Inc.
2. For an overview of UC (UNIX Consultant), see the paper of Wilensky et al. in the first special issue of this three-part series.

References

Bernsen, N-O., Dybkjær, H. & Dybkjær, L. (1998). *Designing Interactive Speech Systems: From First Ideas to User Testing* Springer-Verlag: New York, NY.

BusinessWeek (1998). *Speech Technology is the Next Big Thing in Computing, Special Report*. BusinessWeek, Feb. 23. New York: McGraw-Hill.

Kobsa, A. & Wahlster, W. (1988). *User Models in Dialog Systems*. Springer-Verlag: Berlin.

Maybury, M. & Wahlster, W. (eds.) (1998). *Readings in Intelligent User Interfaces*. Morgan Kaufmann Publishers: Los Altos, CA.