Transportable Natural Language Interfaces for
Economic Knowledge Representation System

Lutz Prechelt, Finn Dag Buo, Rolf Adams
(prechelt|finndag|adams@ira.uka.de)
Institut für Programmstrukturen und Datenorganisation
Universität Karlsruhe, W-7500 Karlsruhe, Germany

Appeared in the proceedings of the
Ninth IEEE Conference on
Artificial Intelligence for Applications
Orlando, Florida, March 1-5, 1993

1 Introduction

There have been several attempts to create natural language interfaces for databases [15, 10, 5].
Of these natural language interfaces to databases, none is transportable, that is, they are
dependent knowledge about the database. The algorithms to implement a transportable
natural language interface to databases require a complete model of the domain. We developed
a system that requires a complete model of the domain for the natural language interface knowledge representation.
A new approach uses the idea of a 'generative model' to generate a natural language interface for the application.

A general approach is presented for building transportable natural language interfaces for question answering systems based on a KL-ONE-like knowledge representation. An example system YAPR, is described. The YAPR knowledge representation of concepts and relations is annotated with minimal syntactic information to generate a semantic case frame grammar with inheritance of cases. The generated grammar directs a case frame parser, which processes written input into instantiated case frames. These instantiations are easily translated into knowledge base queries. The same method is applicable to other object-oriented knowledge bases and other parsing techniques. The original contribution of this work is to show an approach with which natural language interfaces can with low effort be adapted to work with any new knowledge base. While most other system require a complete model of the domain for the natural language interface knowledge representation, we derive most of this information from the application's knowledge base. This technique reduces the amount of work needed to create the interface to about additional 15 percent after building the knowledge base for the application kernel.

AI topic: knowledge acquisition, knowledge representation, natural language interface
Language/Tool: Unix, C++
Status: implementation complete, evaluation in progress
Effort: about 4 person years
Impact: quick development of restricted natural language interface for certain classes of knowledge-based applications (15% additional knowledge acquisition work).
express facts) and assertional knowledge (facts about
individuals (instances) in the application domain).

The terminological knowledge consists of concept defi-
nitions and role definitions. A concept can be thought
of as an abstract set of individuals. The concrete in-
dividuals that belong to a concept are called the in-
stances of that concept. A role is a binary relation
from a concept \( A \) to a concept \( B \), i.e., a set of
of instances. \( A \) is called the domain of the role,
\( B \) is called the range of the role. \( \psi \)
defined with constructor
of the set of all
res
and small, but suffice to construct all information the
natural language interface needs. In this section we
describe the type of information the annotations con-
tain and how it is used in YAK to generate the case
frames. See [6, 14] for a detailed description.

There are two main types of information present in the
annotations: (1) information about individual words
and (2) information about grammatical construc-
tions. The word information associates each
word with its natural language syntax
simple phrases that rep-
resent a variety of the
concepts...
Only the words Zugriff and Lesen need to be in the dictionary, the compound is algorithmically broken into these components.

4.2 Grammatical Construction Annotations

All the above annotations merely describe phrases that represent individual concepts; no case frames are built from them. The source of cases for the case frames are the roles. This is where the information about grammatical constructions is used, which tells what cases to insert where. Similar Annotations could be used to generate the information that is otherwise needed by other natural language processors.

As an example, the *Age* of an action, e.g., for an animal.
that has to be used in a query if the case containing it
has been filled. Most of the instantiations can simply
be processed in a top-down manner, only the instan-
tiations for certain grammatical constructions require
some more complicated processing. In any case, this
processing is purely mechanical. No further semantic
processing is necessary. Most ambiguities that remain
after parsing need not be explicitly resolved
since the wrong interpretations return no answer at all.

When a parsing technique
using shall be used,
proach, the lexicon can be reduced to a dictionary (with- 
without semantic information); the semantic information 
can be derived via the annotations. The conceptual 
schema consists of sort information and constraints on 
the arguments of nonsort predicates. With our ap- 
proach, no such schema is necessary at all; the infor-
mation can completely be deduced from the knowledge 
base itself. The database schema consists of infor-
tion that enables the mapping of the inter-
presentation to a query expression 
language. With our appr 
mapping can be di-
derived f
on.
Flexicore. *Data Abstraction and Object-Oriented 

TEAM An experiment in the design of transpor- 
table natural language interfaces. *Artificial Intel-

on Artificial Intelligence Applications, Miami Be-
ach, Florida, February 1991.* IEEE Computer So-
ciety Press.

Kommandos mit YAKR Studienarbeit, Uni-
sität Karlsruhe, Institut für Progr-
und Datenorganisation, Da-

[13] Intz Prechelt. *Ei-
Deutsch.