[*Proceedings of a Workshop on Machine Translation*, July 1990, UMIST]

Running a Robust Dialogue System

William J Black*

October 29, 1991

Abstract

This paper outlines the general approach taken to dialogue management for robustness in the recently started ESPRITII project, PLUS (P5254). The goal of the project is the production of a robust natural language dialogue system integrating linguistic and non-linguistic knowledge in a principled way, based on the pragmatics theories of Grice and Searle, and using results in knowledge-base management systems and logic programming for the maintenance of dynamic contextual knowledge bases.

1 Introduction

The PLUS project aims at producing a natural language understanding component that allows flexible and efficient Human-Computer interaction. The principal characteristic of such a system is robustness in a wide range of situations: the system should be capable of dealing with extragrammatical input (such as 'elliptical' fragments and misspellings), and flexible enough to allow a real dialogue with the user. In order to demonstrate the capabilities of the PLUS system on a realistically sized application, an interactive Yellow Pages Information Service has been chosen as the demonstrator.

PLUS is a project funded by European Commission for four years commencing in November 1990. The participating organizations are: CAP GEMINI INNOVATION, Paris; ITK, Tilburg, the Netherlands; Omega Generation, Bologna, Italy; CAP GEMINI SCS BeCom GmbH, Hamburg; LIMSI, Paris; The University of Bristol; UMIST, Manchester; and the University of Göteborg (Sweden).

In this paper, we begin by describing the theoretical rationale of the project, then consider aspects of the system architecture, before briefly reporting on an empirical study of simulated human-computer dialogues that has already reached the stage of completed data collection, and preliminary analysis. Finally, we report on our evaluation of re-usable components for parts of the software architecture already described.

*Centre for Computational Linguistics, UMIST, Manchester, UK. e-mail bill@ccl.umist.ac.uk

2 Summary

The keynote of the project is to achieve robustness in natural language understanding by treating natural language as a communicative activity whose essential characteristic is to convey a meaning that is both appropriate and relevant contextually. Since the intention of a human user of natural language is to convey an intended message, and since all messages occur in some context, it is crucially important to exploit this context in the derivation of the intended interpretation. This contrasts with other approaches to the problem of robustness, whose approach can be classified as low-level (spelling correctors, on-line lexical acquisition, domain-dependent constraints, semantic grammars, and so on). These low-level techniques miss the heart of the problem, which is to react appropriately in a context created or updated by the fact that a user has typed something at a keyboard with the express purpose of communicating a message.

The key issue will be the exploitation of both pragmatic and linguistic phenomena, such as interpretation with respect to context, and the power of inference tools derived from non-linguistic problems, in order to provide reasoning-based robustness in natural language understanding by integrating these two areas. We believe this to be the single most important area of neglect in current work on natural language understanding.

2.1 Robustness via Pragmatics

A standard natural language interface consists essentially of a lexicon, a grammar and a parser, and usually fails when confronted with input which has not in some way been anticipated by the author of the system. A robust natural language interface must include the standard components, but must also have the capability of reacting appropriately in problematic situations.

The aim of PLUS is to produce a natural language understanding system whose principal characteristic is robustness in a wide range of situations. Such situations involve not only such well-known linguistic problems as ellipsis, reference resolution, unknown words or spelling errors, but also more complex issues such as deriving conversational implicatures, relevance determination, doxastic speaker modelling and other issues relating to the context of the dialogue.

The key issue in this project is the exploitation of both pragmatic linguistic phenomena, such as interpretation with respect to context, and the power of inference tools derived from non-linguistic problems, in order to provide reasoning-based robustness in natural language understanding by integrating these two areas.

2.2 Modularity

One important aspect of a natural language component is the workload needed to integrate natural language interaction in a new application. For some NL systems everything has to be re-built from lexicon to semantic representation and semantic evaluation.

The PLUS architecture clearly differentiates what is specific to the language (lexicon, grammar), what is specific to the application (application model) and

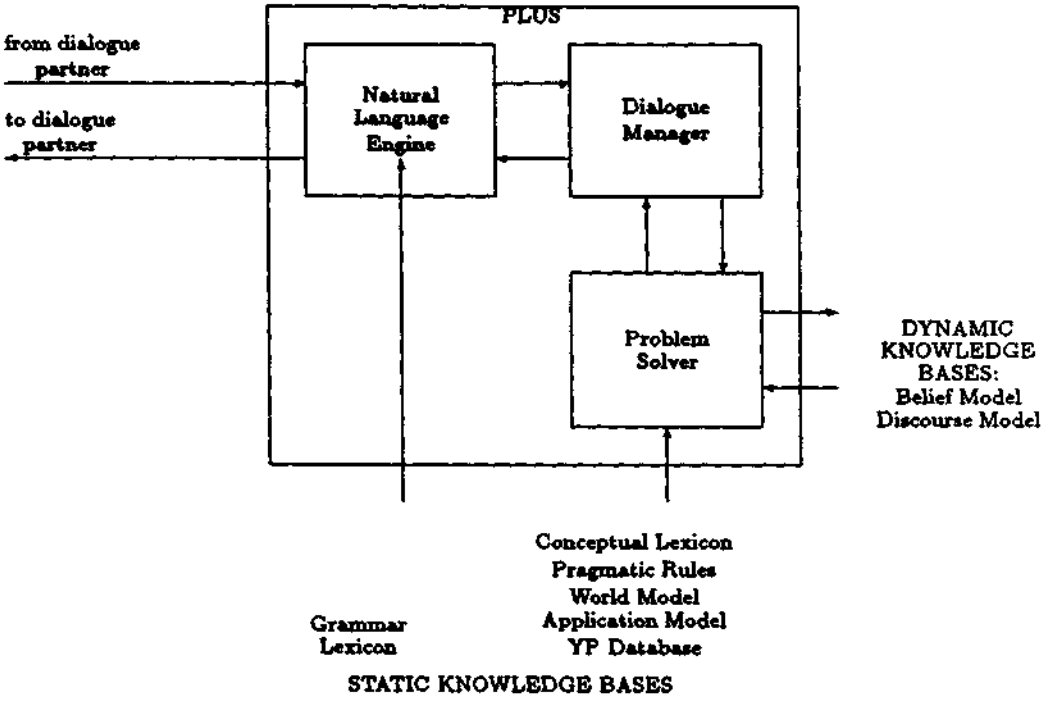

Figure 1: Outline Architecture

what is independent of both language and application (basic knowledge), such as conversational rules and deductive mechanisms.

This separation minimises the cost of building a new application. Only the part specific of the application needs to be developed, language specific parts and basic knowledge need only extensions or additions. This modularity reduces also the cost of adding a new language to the capabilities of the system. This makes the evolution towards a multilingual tool feasible.

3 Architecture

Figure 1 shows the conceptual relationships between the major declarative and functional components. It also shows where the main internal interfaces in the architecture are: *viz.* between the natural language engine and the dialogue manager, and between the dialogue manager and the problem solver. It is important to define the relation between the Natural Language Engine (here-after NLE) and the Dialogue Manager (DM) clearly, as this marks the crucial distinction between PLUS and other similar projects: the radical shift from a syntax-semantics-based language understanding to a pragmatics-based one.

It is proposed that ideally the NLE would operate symmetrically with respect to analysis and generation. This means that the static data stored in the knowledge bases could be the same for both tasks: the NLE would use the same grammar

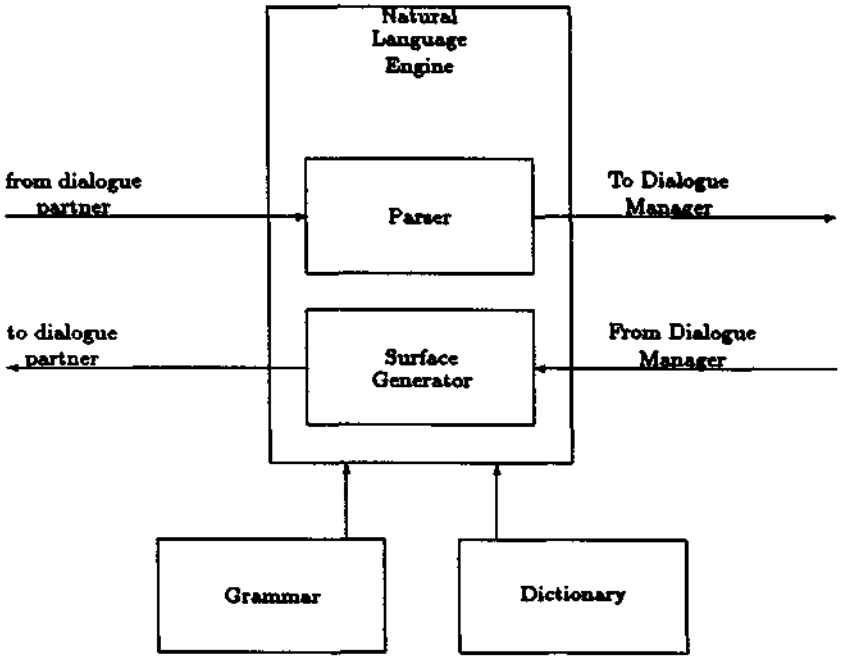

Figure 2: Natural Language Engine

and dictionary for parsing and generation. The DM would also ideally use the same pragmatic knowledge (and the help of the Problem Solver: PS) to build an utterance meaning from the NLE output in analysis and to formulate a literal meaning for the NLE input in generation. In neither case are we committed to the use of the same inference mechanisms, but we also acknowledge that there may be some differences in the grammar as used for generation in comparison with that used for parsing. Every effort will be made to use the same grammar, but if it proves infeasible, even with automatic translation between the two, we could maintain them separately. On the other hand, the NLE and DM can be differentiated on the basis of the knowledge bases they have access to. The grammar and the dictionary are 'private' to the NLE, and the others (including the remaining static knowledge bases and the dynamic knowledge bases) are private to the DM, via its Problem Solver interface.

The Dialogue Manager is further decomposed into three principal subcomponents, and the Natural Language Engine into two. These are: in the Dialogue Manager, the *Cognitive Analyser (CA),* the *Goal Formulator (GF)* and the *Response Planner (RP);* and in the Natural Language Engine, the *Parser* and *Surface Generator.*

3.1 The Natural Language Engine

The internal structure of the NLE can be depicted as in Figure 2, which slightly simplifies the picture in omitting to show the information flow between the grammar and dictionary and the separate components of the NLE. However, it does emphasise that it is the same[1] grammar and lexicon. The main distinguishing

[1] at least in conceptual terms: However, we acknowledge and are addressing the practical difficulties involved.

feature of the parser from the type widely described in the literature of natural language interfaces is that we require it to produce a rather underspecified yet still formal literal semantic and pragmatic meaning representation, in particular abstaining from such operations as spelling correction, sense disambiguation, pronoun resolution, modifier and quantifier scope disambiguation, because these all belong to the domain of contextual, pragmatic, analysis. To facilitate this, the natural language lexicon is built on the basis that entries correspond to lexemes, not word senses, and the pragmatics components have access to a separate 'conceptual lexicon' which maps lexemes to world and application concepts.

The mirror image of the parser is the surface generator, which accepts a relatively complete specification of the form and content of the generated utterance, exercising no real choices, which are all made in the pragmatic component: the Response Planner (RP),which deals with the issues that belong to both strategic and tactical generation as usually described. The surface generator is therefore able to use much of the recently reported work on generation from logical form with unification grammars, and not require the more complex procedural notions of the more traditional systemic generators.

3.2 The Dialogue Manager

Figure 3 illustrates the internal composition of the Dialogue Manager at a Conceptual Level. The Cognitive Analyser is responsible for all interpretation of the utterance of the dialogue partner, including inferring the user's goals as well as beliefs. The Goal Formulator is responsible for deciding on a strategy for responding, including making enquiries from the Yellow Pages database in order to have the information to respond, and determining what kind of dialogue act is an appropriate way to respond. The Response Planner is responsible for what is described as both Strategic and Tactical Generation in the literature, including rhetorical planning, anticipating implicature and presupposition derivation, sequencing, partitioning into linguistic units, focusing, sentence type choice and planning of anaphoric referring expressions (not necessarily in that order).

3.2.1 Cognitive Analyser

The cognitive analyser can be conceptualised as a function from 'literal meanings', background knowledge states and contexts to contexts. Thus its inputs include all the knowledge bases in the architecture with the exception of the natural language engine's grammar and dictionary. The dynamic KBs that comprise the context, being updated during the dialogue, are the dialogue history and the belief model.

The task of the Cognitive Analyzer is to find the most relevant utterance meaning to the context, using its world knowledge (which for the present discussion we will regard as subsuming the application-specific knowledge). The CA analyses the dialogue partner's turn in accordance with the dialogue grammar and pragmatic rules, and tries to find out the communicative intention of the dialogue partner who used the particular expression.

The output of the CA is a set of new system beliefs that are to be incorporated into the Belief Model. The output of the CA thus goes to the contextual knowledge bases and triggers the knowledge base updating procedures.

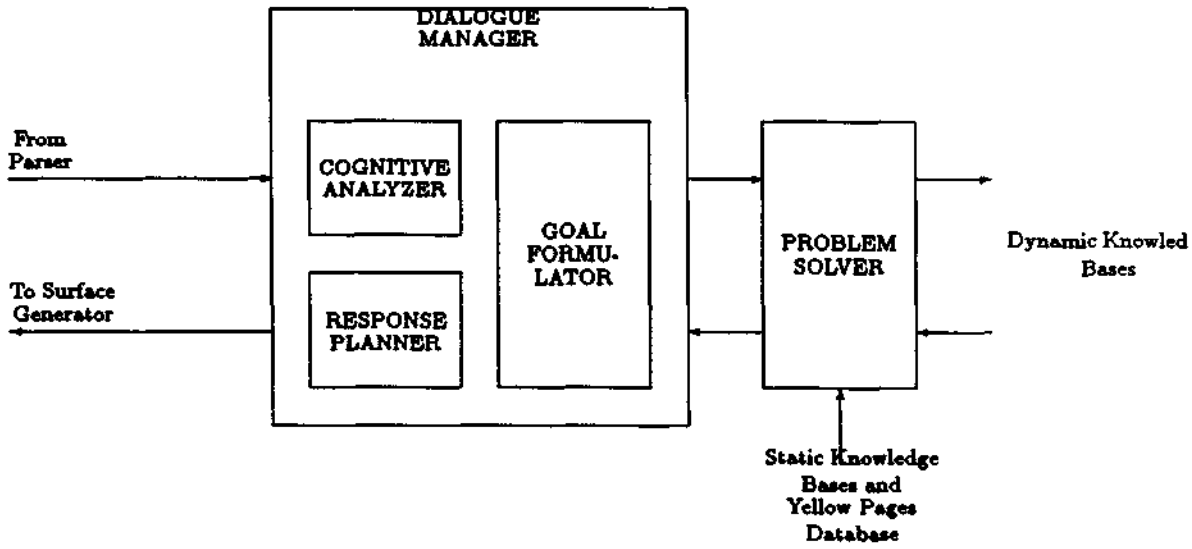

Figure 3: Dialogue Manager.

The internal structure of the Cognitive analyser is based on the elaboration of the meaning from literal to contextual in what can be conceived as three stages. Firstly, the content part of the meaning has to be mapped from the literal predicates of the literal meaning logical form language to the conceptual level language. Secondly, the intended pragmatic force must be determined, and finally additional implicatures/inferences are derived as necessary for the coherent incorporation of the new beliefs into the belief model.

3.2.2 Goal Formulator

The goal formulator decides what to do next at a given moment of the dialogue. It plans the strategy of responding to the communicative act of the dialogue partner utterance. It has an ultimate goal that is determined by the application model (in the YP application: 'give a name/address/ telephone number to the user'), and it solves the problem of how to reach the ultimate goal within the limits given by the world model and the current, updated dialogue history and belief model, assuming the Gricean maxims of quality and relevance. The goal formulator plans the steps or sub-goals to attain the main goal. It 'knows' about the preferences among the sub-goals, and relying on general problem solving techniques, it passes the selected goals with an indication of priority to the Response Planner

The Goal Formulator maps from a context to a specification of system goals. The input is a knowledge base representing the context derived from the application of the output of the Cognitive Analyser to the previous context (the "initial context"). This represents the context as created by the dialogue partner input (the "amended context").

The output is a specification of a new goal context representing a state of affairs the system will strive to reach. The difference between the amended context and the goal context represents the system's goals as determined by the effect

of the dialogue partner input on the initial context. These goals are output to the Response Planner.

3.2.3 Response Planner

The RP is responsible for planning the literal meaning of the next system utterance in reaction to the current dialogue situation. It also has access to the results of the Goal Formulator's enquiries to the Yellow Pages Database. Therefore, the identified inputs and outputs are: Input - system goals:informative or sub-dialogue; Output - Literal meaning + additional information.

This computation incorporates the following functions: Planning, critiquing of plans, evaluation of effects on dialogue partner, deep generation and dialogue history update. The effects and needs of the Response Planner implies that it can access the all knowledge sources excluding the application database.

3.3 Problem Solver

General Purpose reasoning tools are used for three purposes in the PLUS conceptual architecture: Plan recognition, in the course of 'cognitive analysis'; Planning of its own actions and responses, at various levels; Knowledge-Base access and update. Of these, the last is a major re-usable component of the system, based on CML, which was developed in the Esprit projects LOKI and DAIDA.

3.3.1 KBMS

The operational KBMS is built using a CML (Haidan and Maier, 1990) subsystem which permits access to a set of loaded knowledge bases (generated by the CML Support System) and perform the operations needed by the PLUS system.

A given data base can be interfaced to a KB such that part of the factual knowledge in this KB physically resides in the data base but can be accessed by the same[2] query mechanism used to access the knowledge in the KB. Thus the factual data in the data base appear to be in the CML KB in a way completely transparent to the PLUS system. This mechanism is used to interface the Yellow Pages Data Base to the Application KB for retrieval of its factual contents.

On top of the basic functions of query and update of knowledge bases (with consistency checking), a set of higher level meta-reasoning mechanisms are built, including abduction, temporal reasoning, planning, reasoning about beliefs, tracing and explanation. Meta reasoning is used as well to access and reason about different KBs, their informational contents and their interrelations e.g. compatibility of knowledge contained in them or capability to help solving a given problem. It is based on concepts of provability and mutual knowledge.

As well as the internal reasoning mechanisms, the KBMS service has well-defined interfaces both to the Dialogue Manager and to the application database.

[2] although for reasons of granularity of representation, it may be necessary to have automatic translation between external and CML query languages

4 Corpus collection and analysis

As input to the design and evaluation of the system, the consortium has carried out a Wizard-of-Oz simulation of the system, as a basis for collecting samples of realistic dialogues.

5 Reuse of existing components

PLUS aims to further the state of the art in machine understanding of natural language by (a) implementing the above mentioned strategy of relying heavily on context information and pragmatic knowledge; (b) building on previous research in natural language processing, knowledge representation and automated reasoning.

Where appropriate, PLUS is reusing software, data and algorithms as well as formalism, techniques and ideas resulting from previous research in the following areas:

- formalisms for expressing linguistic knowledge;
- linguistic repositories (grammars, lexicons);
- natural language parsers, interpreters and generators;
- knowledge representation and management;
- automated reasoning.

Particular choices that have been made in these areas include HPSG (Pollard & Sag, 1987) for the linguistic formalism, and an adaptation of the semantics described there to deal with quantification in a more straightforward way, and to introduce discourse indices. A more formal semantics applies to the representation of beliefs for the reasoning components of the system. This is a synthesis of the EL- formalism for semantic representation used in the TENDUM system (Bunt, 1985), an 'attitudes model' and the Davidsonian approach used in LOQUI, CLE and ACORD. The parser and generator are not specialised to the PLUS environment, except in so far as the former is able to deal with unrecognised words via defaults in the lexicon.

6 Current Status

The project is now in its second six-month phase, and work is well under way on the detailed design and prototyping of many of the key elements in the architecture. The adaptation of re-usable natural language engine and basic inferencing tools is scheduled for completion eighteen months from the start of the project, with detailed analysis of the key pragmatic functional components and system knowledge bases starting in the same period, based on a solid foundation of empirical data.

This paper is an abridged and slightly updated version of one edited by the same author for the Proceedings of the 1991 Esprit Conference.

References:

Bunt, H.C. (1985) *Mass terms and model-theoretic semantics.* Cambridge University Press, Cambridge, UK.

Estival, D. (1991) Declarativeness and Linguistic Theory. In: *Proc. of the First In. Conf. on Knowledge Modelling and Transfer,* Sophia-Antipolis, April 1991. IOS Press, Amsterdam.

Haidan, R. and R. Meyer (1990). Requirements Modelling and System Specification in a Logic-based Knowledge Representation Framework. DAIDA (P892) Project Report.

Gallaire, H., Minker, J. and Nicolas, J-M. (1984). Logic and Databases: A deductive Approach. *Computing Surveys,* No.2, Vol.16.

Kowalski, R.A. 1979. *Logic for Problem Solving.* North-Holland

Pollard, C. & Sag, I. (1987) *Information-Based Syntax and Semantics.* CSLI, Stanford.

Poole D.L. and Goebel A.R. (1986) Gracefully Adding Negation and Disjunction to Prolog. Proc. 3rd Int. Conf. on Logic Programming, Springer, 1986

M. Sergot and R. Kowalski (1986) A Logic-Based Calculus of Events. *New Generation Computing,* No.4, pp.57-95.

Stanley M. T. (1986) *CML: A Knowledge Representation Language with Application to Requirements Modelling.* PhD Thesis, Dept. of Computer Science, University of Toronto.

Wilensky, R. (1983) *Planning and Understanding.* Addison-Wesley, 1983.