Proceedings of the ESPRIT Conference 1991.

A Pragmatics-based Language Understanding System

William J Black[*)], Jens Allwood, Harry C Bunt, Frens J H Dols, Carlo Donzella,
Giacomo Ferrari, John Gallagher, Rainer Haidan, Bill Imlah, Kristiina Jokinell
Jean-Marie Lancel, Joakim Nivre, Gérard Sabah, Tom Wachtel

December 12, 1991

Abstract

This paper presents an introduction for the ESPRIT community of the recently started ESPRIT 11 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. We present the background assumptions, an overview of the system conceptual design, of the empirical research that has already been undertaken on real corpora and of re-usable system components.

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 extra-grammatical 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 organisations are: CAP GEMINI INNOVATION, Paris (the Co-ordinating contractor); 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). The project leader is Jean-Marie Lancel.[1]

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.

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

[*)] Editor, Centre for Computational Linguistics, UMIST, Manchester, UK. e-mail billace@umist.ac.uk 'CAP
[1] GEMINI INNOVATION, Rue de Tocqueville 118, Paris, 75017, France, e-mail lancel@capsogeti.fr

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 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 (hereafter 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.

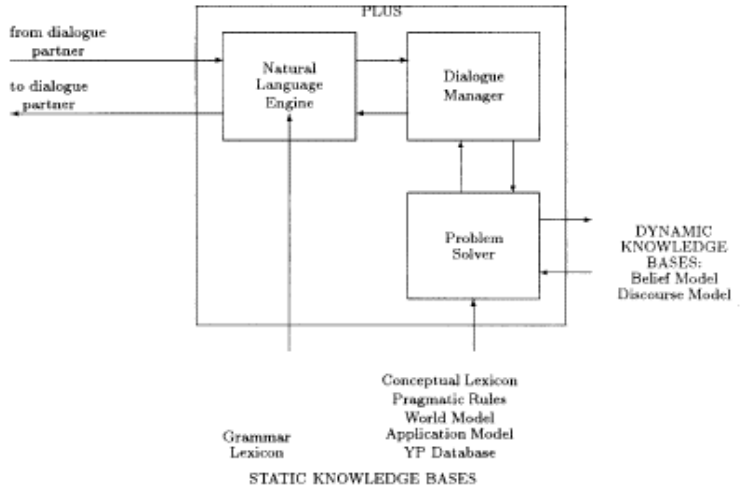

Figure 1: Outline Architecture

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 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 depleted 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[2] grammar and lexicon. The main distinguishing 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. The mirror image of this, in the surface generator, is that the latter 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.

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

Figure 2: Natural Language Engine

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.

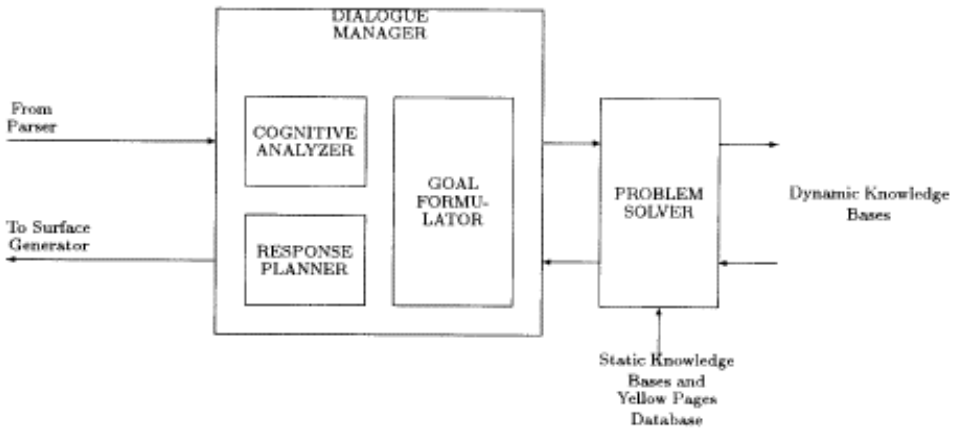

Figure 3: Dialogue Manager.

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.

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.

3.3.1 Plan Recognition

This is a service used by the Cognitive Analyser in attempting to recognise the dialogue partner's intended force. Its operation will be similar to that described in Wilensky (1983). This may or may not be derivative of the planner (next paragraph).

3.3.2 Planning

This is a general purpose planning component that is used in both the Goal Formulator and the Response Planner. The planner will operate in a propose-criticise cycle, and can simplify and merge subplan.

3.3.3 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[3] 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.

4 Corpus collection and analysis

The goal of the corpus collection task within the PLUS Project is to Provide the project with relevant information for the design of system components. In addition, it will provide useful information for test and evaluation of the resulting system.

4.1 Corpus Collection

In designing the corpus collection task, the following parameters were considered:

- **Languages.** Since portability to new languages is a major concern of the PLUS project, It is important to try to separate, as far as possible, language-dependent and language-independent aspects of dialogues. Hence, dialogues are collected in three different languages: *English*, *French,* and *Swedish,*

- **Dialogue types:** By *dialogue types* we mean different configurations of participants and communication channels. First, we have so-called *Wizard-of-0z terminal dialogues,* where the subject interacts through a terminal with what he believes to be a computer system. This is the most important type for the PLUS project, since it is the one which most closely resembles the dialogues that will be possible with the final system. However, for comparison, we also collect *human-human terminal dialogues* and *human-human telephone dialogues.* The comparisons are important *inter alia* because many of the theoretical approaches (e.g. in pragmatics) that will be exploited in PLUS have been developed mainly with human-human (especially spoken) communication in view.

- **Scenarios:** The term *scenario* refers here to the specific dialogue task that the subject is confronted with. The scenarios have been chosen from the same subdomains of the Yellow Pages that have been selected for the demonstrator, namely: *car hire, restaurants,* and *personal insurance.*

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

4.2 Analysis

The dialogue corpora collected in PLUS are first subjected to a preliminary analysis and subsequently to a more detailed analysis within the work packages devoted to pragmatics, natural language engine and knowledge base management.

Analysis of the dialogues with regard to pragmatic phenomena gives useful information for designing such components as *dialogue grammars* and *rules of conversation*. Analysis of syntactic and semantic aspects of the corpora will be used in designing the parts of the system analyzing and generating language,
Specifically in producing grammars and lexica. Information from the. corpus analysis can also contribute to the Modelling of the application.

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 possible and appropriate, PLUS will reuse software, data and algorithms as well as formalism, techniques and ideas resulting from previous research in the following areas: pragmatics, formalisms for expressing linguistic knowledge, linguistic repositories (grammars, lexicons), natural language parsers, interpreters and generators, knowledge representation and management, automated reasoning.

5.1 Pragmatics

The evaluation of the pragmatics components of LOQUI, SUNDIAL, CFID, CARAMEL, ESTEAM and TENDUM allowed a collective study of the various approaches to pragmatics inside the PLUS project.

The systems described exhibit different architectures. However some common components can be identified:

- a dialogue grammar,
- a mechanism to map single utterances onto a complex logical form,
- a dialogue memory,
- a mechanism for reference resolution,
- a planner or a dialogue manager,
- a task model.

Each evaluated project was designed for specific aims, and has clear boundaries. Nevertheless it has been shown in the evaluation that there is a set of commonalities, namely dialogue grammar, dialogue memory and dialogue management. Noticeable differences exist in belief modelling (granularity of user modelisation). Differences can also be seen for the representation of dialogue acts. The conducted study indicates that significant theoretical results are available which the PLUS project can build on.

5.2 Linguistic formalisms

The question of which linguistic formalism to choose cannot be answered on the basis of a purely technical evaluation, since most formalisms are based on theoretical ideas which cannot be judged right or wrong per se; discussion of their reusability in PLUS must therefore focus on their adequacy given the general PLUS approach to

language understanding and on practical considerations, such as the availability of software tools to support working with the formalism in question.

Of the formalisms in use at the sites of the PLUS consortium, GPSG, HPSG, DPSG and UCG all meet the general criteria of being well-defined, general, state of the art, and well-documented. Each of these formalisms also satisfies the requirement of being purely declarative, thus allowing the use of the same grammar and lexicon both for parsing and generation.

Compared to some other formalisms, HPSG and UCG have the advantage of adopting the nowadays predominant *lexicalist* view on grammar theory, which leads to grammars with small sets of rules that are better manageable than classical rule-oriented grammars.

Specific consequences from the general PLUS approach for the choice of linguistic formalism concern the requirement that the NLE parser should produce meaning representation based solely on the information that is found in the input and the linguistic repositories; in particular, these representations should be *formal yet underspecified* for those semantic aspects for which the input is unspecified. Such semantic representations have been shown to be feasible in the systems LOQUI, TENDUM, and CLE, for parsers using phrase-structure type grammar formalisms.

With respect to computational adequacy, all formalisms mentioned here have been implemented and shown to be computationally feasible. However, for a sizeable flexible categorial grammar it is hard to avoid local explosions of spurious ambiguities.

HPSG (Pollard & Sag, 1987) is an eclectic syntactic theory that incorporates ideas both from GPSG, Categorial Grammar, and Government-Binding Theory, and employs a powerful, flexible formalism that allows expression of insights from other approaches. HPSG thus seems to offer the most *extensible* approach to grammar and lexicon development, and has therefore been chosen as the linguistic formalism of the PLUS Natural Language Engine.

5.3 Linguistic repositories

For PLUS, the available linguistic repositories concern the GPSG, HPSG, DPSG and UCG grammars, lexicons and morphological tools developed for the systems LOQUI, TENDUM, ACORD and SUNDIAL. Other lexicons and morphological tools may be relevant if their theoretical assumptions do not conflict too much with those of the chosen linguistic formalism.

Existing environments for grammar development and testing for HPSG-style grammars, such as ELU (Estival, 1991) should facilitate the further development of these repositories.

5.4 Natural language analyzers and generators

Several parsers and generators from LOQUI, TENDUM, SUNDIAL, CARAMEL or ACORD are available for reuse in PLUS, either as software modules or as algorithms to be reimplemented. Concerning the type of parser and generator to be developed, the chart parser technique and the bottom-up head-driven generator strategy are promising reusables that can be employed in combination with an HPSG-based granimar-lexicon formalism.

5.5 Knowledge representation and management

For knowledge representation and management (including the representation and application of various kinds of 'context,' information), PLUS will rely on the use of the Conceptual Modelling Language (CML).

CML, as formally defined in Stanley (1986), is a general approach to conceptual modelling,, integrating concepts from Artificial Intelligence, logic programming, and deductive data bases. CML was developed in the Esprit projects LOKI and DAIDA; in the latter project it was used in a software engineering context for system modelling and called SMI, 'System Modelling Support Tool'). The design and implementation of CML, was mainly influenced by research in the field of deductive data bases (Gallaire, 1984) and logic programming (Kowalski, 1979).

CML meets the PLUS requirements of being well-defined, general, state of the art, extensible, computationally feasible, and well-documented (Haidan, 1990). Extensions to be made to the currently existing CML system concern the use of multiple knowledge bases; certain aspects of negation, inconsistency checking and metaprogramming, and the implementation of the envisaged abductive and hypothetico-deductive query procedures.

5.6 Automated reasoning

CML, is based on first-order logic and the idea of expressing higher-order concepts into this logic by means of predefined 'metapredicates'. This allows the definition of defaults, exceptions and selective closed world assumptions.

The reasoning procedures developed with CML, comprise implentatations of algorithms for deduction (Poole, 1986) and event reasoning (Sergot, 1986). Extensions of the CML, reasoning techniques with hypotheticodeductive reasoning are planned, and should result in a system which has more powerful reasoning capabilities then those of purely deductive or abductive reasoning in isolation.

6 Conclusions

The first six-months phase of the PLUS project was devoted to training, evaluation of reusable components, corpus collection, external requirements and system design. In the second six-month phase, work is now focusing on the detailed design of the system components, the prototyping of some of the key elements in the architecture and the conceptual modelling of selected Yellow Pages sub-domains.

The aimed output of the project in 1994 is a software environment for developing natural language dialogue, applications. One requirement is that this environment be extensible to new languages.

With present natural language interfaces, users are not able to express freely their needs, and get sensible answers: failure in the dialogue or unadequate answers are frequent. Present, experimentations show that robustness of the dialogue is a key concept for wide acceptance of this new mode of interaction. The PLUS project stresses the use of natural language dialogue as one of the best ways to provide efficient and friendly human-computer interaction.

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. 1

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

Lancel, J. M., Allwood, J., Bilange, E., Black, W. J., Bunt, H. C., Dols, F.J.H., Donzella, C., Gallagher, J., Haidan, R., Sabah, G., Wachtel, T. (October 1990). *PLUS Technical Annex, Part II.*

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.