

# SRI International

## A BRIEF OVERVIEW OF THE CANDIDE PROJECT

Technical Note 450

September 20, 1988

By: Fernando C. N. Pereira and Martha E. Pollack

Artificial Intelligence Center  
Computer and Information Sciences Division

**APPROVED FOR PUBLIC RELEASE:  
DISTRIBUTION UNLIMITED**

This research was supported by the Department of the Navy under Contract N00039-84-C-0524 with the Space and Naval Warfare Systems Command.

The views and conclusions contained in this document are those of the authors and should not be interpreted as representative of the official policies, either expressed or implied, of the Defense Advanced Research Projects Agency or the United States Government.



# A Brief Overview of The Candide Project

Fernando C.N. Pereira  
Martha E. Pollack

## 1 Introduction

Over the past several decades, computer systems have come to embody more and more complex forms of knowledge. For example, a production-line control system must embody knowledge about how the steps in the production line are ordered, what the capacity of each machine in the system is, and what the procedure is for assigning high priority to certain “hot” lots. As another example, consider a system that monitors for malfunctions in a space vehicle and suggests corrective actions. It must embody knowledge about the normal configuration of the vehicle’s components, about the symptoms of component malfunction—for instance, that a sudden pressure rise may indicate that a valve is jammed—and about the steps that should be taken to correct any malfunction. Even systems that are closer to traditional databases may need to embody relatively complex knowledge. A useful personnel database system must not only keep a consistent and complete record of who works where in an organization, but must also recognize that each employee has a single boss who is also an employee, that while by law all employees must earn more than \$3.50 per hour, in fact no one in the company earns less than \$4.15 per hour, and so on.

How is the knowledge that is embodied in a computer system acquired? In most present-day systems, it is painstakingly encoded in the actual algorithms that implement the system. Even in most AI systems, which may have a level of explicit representation of the knowledge, this knowledge must be entered into the system as expressions in some formal computer language. Usually this must be done by a specialized programmer or knowledge engineer. The result is a *knowledge bottleneck*. Significant efficiency could be gained by making it possible for the needed knowledge to be installed by those who are knowledgeable about the domain of some computer system, but who do not necessarily possess expertise about the internal workings or language of the system itself.

The Candide project has been concerned with developing tools that will help to alleviate the knowledge bottleneck problem. Under its auspices, the Candide prototype system was designed and implemented to exemplify the acquisition of procedural knowledge expressed in a combination of flow charts and ordinary English discourse.

The current version of Candide uses the Procedural Reasoning System (PRS) [1] as a testbed. PRS is a reactive, real-time system for reasoning about and performing tasks in dynamic environments. A central component of PRS's knowledge base is its set of procedural networks. Each procedural network encodes information about the steps of some domain procedure; it is used by PRS for performing the procedure. One important application of PRS has been to the problem of equipment malfunction on NASA's space shuttle [2]. Candide exemplifies the kind of interface to PRS that might be used by an engineer who is familiar with the procedures for dealing with space-shuttle equipment malfunction, but who may be unfamiliar with the

formal PRS rule language.

Significantly, Candide is not just a single-utterance interpretation system; it includes capabilities for processing the types of extended natural-language dialogues that are necessary in complex tasks such as knowledge acquisition. Major contributions of the Candide project include the development of a unified framework for semantic and pragmatic processing of natural-language discourse, supported by powerful and general syntactic parsing mechanisms. A number of significant advances in unification-based parsing have been made during the course of this project. In addition, techniques have been developed for reasoning about the domain and the discourse history in order to handle a wide range of important semantic and pragmatic phenomena mentioned below.

## 2 The Acquisition of Procedural Knowledge

As noted above, the current version of Candide provides an interface for building and updating the collection of procedural networks that makes up a crucial part of PRS's knowledge base. Each procedural network encodes information about the steps of some domain procedure which can then be used by PRS for performing that procedure.

Details of the ways in which PRS uses of the networks can be found in [1,2]. Here we simply provide enough information to make the operation of the Candide system clear. Every procedural network comprises two parts: a set of invocation conditions that specify when the encoded procedure is relevant, and a graph that encodes the steps of the procedure itself. Arcs of the graph are labeled in one of three ways: there are achievement arcs, which

are prefixed with the operator “!”, query arcs, prefixed with the operator “?”, and assertional arcs, prefixed with the operator “→”. Each arc type has an associated method of traversal: for example, an query arc can be traversed only if the condition on it is true. What is important to note for our purposes is that the three arc types correspond quite directly to the three major moods of English sentences. Invocation conditions may either be assertions, which denote propositions that, when true, indicate that the procedure is relevant, or they may be imperatives, which denote goals that are likely to be satisfied as a result of performing the procedure.

Figure 1 depicts a portion of a highly simplified procedural net. It has been adapted from one application of PRS: monitoring for and responding to equipment malfunctions on NASA’s space shuttle. The figure shows a small portion of a network that encodes the procedure for dealing with the failure of one of the jets in the shuttle’s reaction control system (RCS). The invocation conditions in the figure correspond to a set of three assertions: “The RCS jet warning light is on.”, “The CW alarm is on.”, and “A jet is faulty.” Then, the topmost arc in the net, the one emanating from the start-node, corresponds to an imperative, “Close the manifold.” The two arcs emanating from node S1 correspond to an interrogative, “Is there high usage in the RCS?”. PRS should traverse the lefthand arc if the answer to this query is no, and traverse the righthand arc if the answer is yes. Finally, the arc emanating from node S3 corresponds to an assertion “The jet driver has an electrical failure in the on position.” What is important to note is that the Candide system can translate these ordinary English expressions, used in context, into PRS’s internal language. The use of context is crucial: Candide can determine, for instance, that the manifold to be closed, in

the arc emanating from the start node, is the one that is connected to the faulty jet mentioned in the invocation conditions. The user does not need to specify this fact explicitly. Not only does this save time for user, by allowing him to make use of ordinary, colloquial English (so that he does not need to say, for instance “Close the manifold that is connected to the jet that is faulty”)—it also frees him from needing to know how this information is stored internally: he does not need to know that the relation between jets and manifolds is the “attached” relation.

To build a procedural network through Candide, someone familiar with the procedures relevant to some domain first describes the invocation conditions for the procedure, and then specifies each of the arcs of the network by giving its start and end nodes and stating the conditions on that arc. Complex objects, conditions, goals, actions, and propositions are thus specified in English, while their temporal and causal relations to one another are specified graphically. Candide automatically translates each English condition into a statement in the language used by PRS. The translated condition is displayed on the network; the English statement is also stored for the convenience of the user. The construction of arcs and nodes is achieved by the graphical interface tool Grasper [3].

### 3 The Architecture of Candide

The Candide system is implemented in Prolog and Zeta-Lisp on Symbolics 3600-series computers. Figure 2 provides a schematic diagram of the system. Processes are shown in rectangles, and data stores in curved-corner rectangles. Candide has been embedded in Grasper, which itself can be called

INVOCATION-PART:  
 (AND (\*\*FACT (& (LIGHT \$E5)  
 (WARNING \$N102.)  
 (JET \$N101.)  
 (RCS \$P86.)  
 (THS \$Z46.)  
 (THS\_OF \$P86. \$Z46.)  
 (JET\_OF \$Z46. \$N101.)  
 (CULPRIT \$N102. \$N101.)  
 (WARNER \$N102. \$E5))))  
 (\*\*FACT (ON \$E5))  
 (\*\*FACT (& (ALARM \$E8)  
 (BACKUP \$N103.)  
 (BACKER\_UP \$N103. \$E8)  
 (CW \$P87.)  
 (PURPOSE\_OF \$E8 \$P87.)))  
 (\*\*FACT (ON \$E8))  
 (\*\*FACT (JET \$R222.))  
 (\*\*FACT (FAULTY \$R222.))

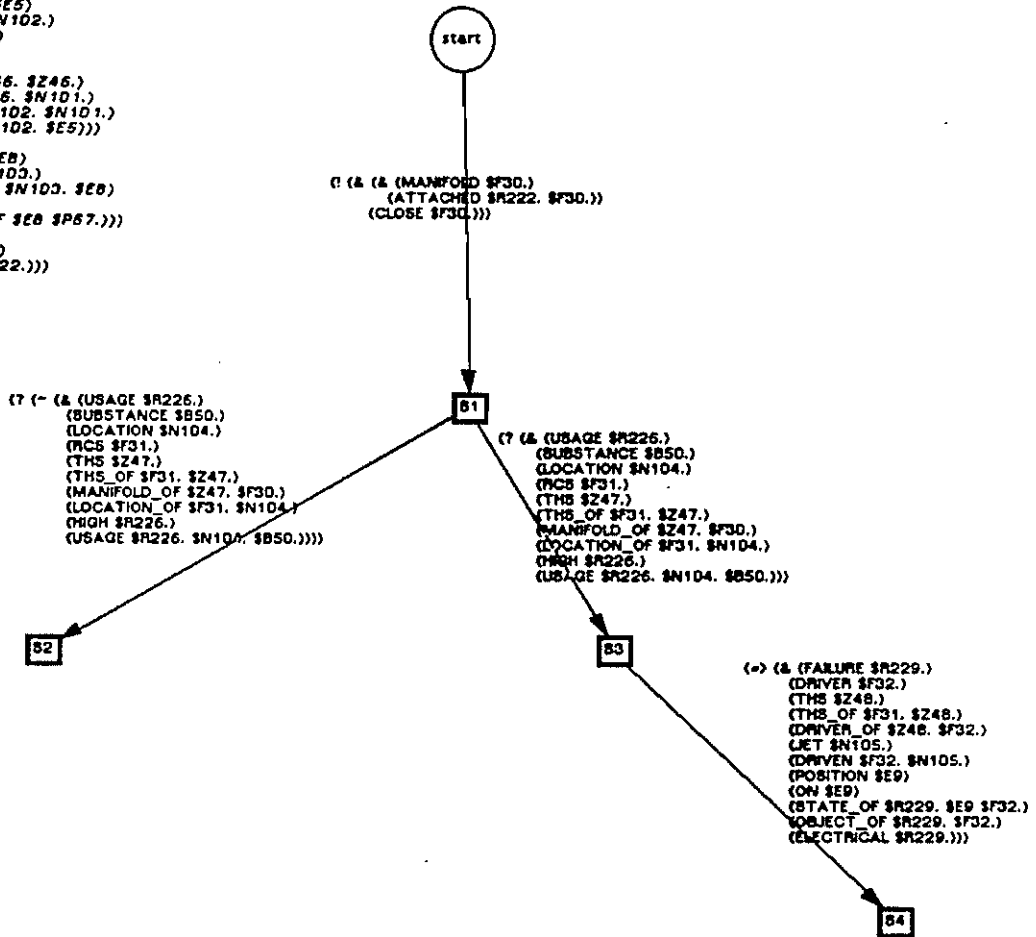


Figure 1

directly from PRS.

The Candide system can be decomposed into three major components: a syntactic parser (PATR-II), a system for semantic and pragmatic interpretation (CANDIDE-SPI), and a routine that transforms the logical forms produced by CANDIDE-SPI into expressions in PRS's internal language (TRANSFORM). The latter two components have been kept distinct from one another in an effort to maintain Candide's generality and portability. The logical forms produced by CANDIDE-SPI are general enough that they can subsequently be translated into a variety of formal languages that might be used by different end systems. Because TRANSFORM is a small, highly domain-dependent module, we do not discuss it further below.

## 4 Syntactic Analysis

Syntactic analysis in Candide is performed by PATR-II, a state-of-the-art syntactic parsing system that implements unification-based parsing. The development of PATR-II was begun prior to the start of the Candide project; a general discussion of the system, and of unification-based parsing in general, can be found in [4]. Here we will just mention some of the improvements made to PATR-II under the auspices of the Candide project.

Unification-based parsing involves the construction and multiple use of structures representing complex syntactic categories. The same complex category may have to be incorporated into many alternative analyses of the input sentence, imposing on the category different constraints that must be segregated. The most straightforward means of enforcing this segregation is to create a distinct complete copy of the category before trying to incorpo-

# THE CANDIDE ARCHITECTURE

PRS

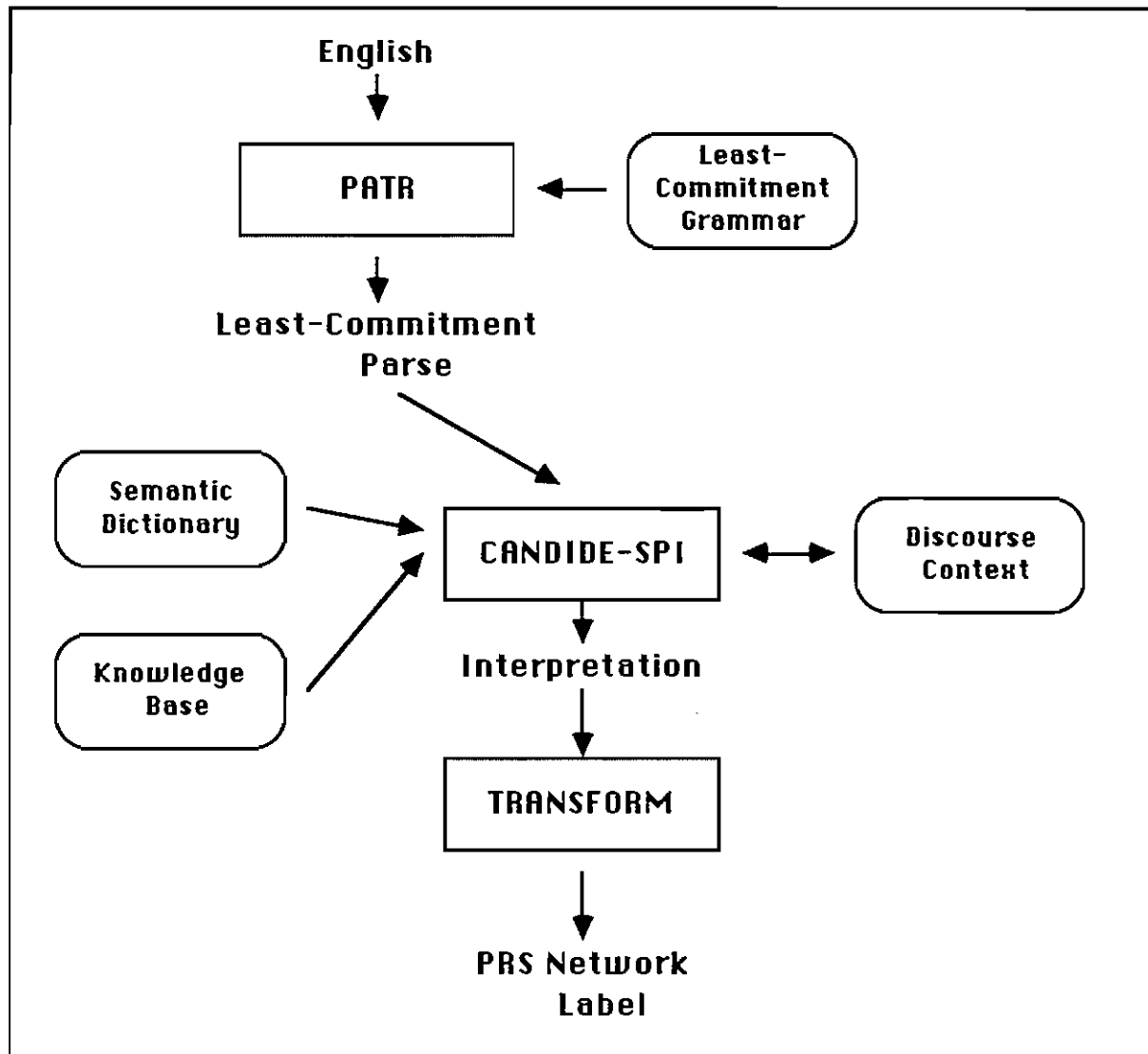


Figure 2

rate it in an analysis. This is very costly in space and time for copying, so we were led to consider alternative approaches, including structure sharing and lazy copying. In structure sharing, each alternative use of a category is represented by a pair of the original category and the set of additional constraints imposed by that particular use of the category. In lazy copying, a new copy of a category is created only after the unification that determines whether the category can be incorporated into an analysis has succeeded. We developed a promising first implementation of the structure-sharing method, but in the end we opted for lazy copying in PATR-II, because it could be incorporated much more easily in the existing system. With this and other data structure modifications, the speed of PATR-II was improved by a factor varying between 5 and 20, leading to parsing times in the 1-10 second interval for typical sentences.

Other advances made to PATR-II during the Candide project include the introduction of restriction mechanisms for parsing and the enhancement of the morphological analyzer. The definition of the restriction method for uniformly controlling the amount of top-down information used in parsing led to dramatic improvements in the efficiency of the PATR-II implementation. A complete overhaul of the morphological analyzer allowed it to make direct use of morphological rules rather than precompiled automata.

## 5 Semantic and Pragmatic Analysis

The second major component in the Candide system, CANDIDE-SPI, performs semantic and pragmatic interpretation. The primary input to this system consists of feature structures that encode the syntactic analysis of

an utterance. In addition, CANDIDE-SPI produces representations of the discourse context during the processing of each utterance; and the discourse context for each utterance is always available during processing of the subsequent one. Intermediate representations are conditional interpretations, which consist of a partial interpretation (called the *sense*) plus a set of assumptions about subsequent pragmatic processing. Semantic interpretation rules operate in a top-down recursive fashion on the input feature structure, converting portions of it to partial interpretations under assumptions of subsequent pragmatic processing. Separate pragmatic rules then discharge the assumptions, in the process further altering the interpretations. The pragmatic rules can also read from and write to the discourse context.

CANDIDE-SPI contains routines for handling a range of pragmatic phenomena, including pronoun resolution, definite reference, quantifier scoping, resolution of compound nominals, and resolution of syntactic ambiguity such as prepositional-phrase attachment. The system is also capable of handling interactions amongst these phenomena—a problem that has proved challenging for many existing interpretation systems. To support the various pragmatic processes, a set of routines for handling the discourse context and the system's knowledge base were developed. The discourse context comprises three components: an immediate context, representing both type and syntactic information about the entities referred to in the utterance currently undergoing processing, a local context, representing similar information about the entities referred to in the immediately preceding context segment, and a global context, representing just type information about entities referred to throughout the discourse. The latter is configured as a stack; in the testbed system for acquiring procedural nets, this configura-

tion can be implicit since it is effectively defined by the net. The discourse context allows us to make use of current theories of reference, such as the centering theory of pronoun resolution [5] and the focus-based theory of definite reference resolution [6].

To solve pragmatic problems, CANDIDE-SPI makes use of two knowledge stores: a knowledge base and a semantic dictionary. The knowledge base comprises both a type hierarchy and a specification of the roles associated with any type. The encoding of role information includes a specification of the algebraic relations between any entity and the entities that may fill some role of it. To resolve definite reference properly, it is important to know whether a role is a function or a relation of an entity of some type, and if the former, whether or not it is invertible—if the latter, whether or not it is many-to-one. This information is encoded in the knowledge base. Relations amongst the role-fillers are also encoded, and used in the analysis of linguistic modification and ellipsis resolution.

The semantic dictionary contains information about selectional restrictions on verbs, relational nouns, deverbal nouns, and prepositions, as well as prepositional coercion rules. As in the knowledge base, relations amongst the arguments of any vocabulary item can be encoded.

## 6 Further Information

Further information about the Candide project is contained in a number of papers, listed as references [7] - [14] below.

## References

- [1] Georgeff, M.P. and A.L. Lansky. Procedural reasoning. *Proceedings of the IEEE, Special Issue on Knowledge Representation*, pp. 1383-1398, 1986.
- [2] Georgeff, M.P. and A.L. Lansky. A system for reasoning in dynamic domains: Fault diagnosis on the space shuttle. Technical Note 375, Artificial Intelligence Center, SRI International, Menlo Park, CA, 1985.
- [3] Lowrance, J.D. GRASPER II Reference Manual. Internal Document, Artificial Intelligence Center, SRI International, Menlo Park, CA, 1987.
- [4] Shieber, S.M. *An introduction to unification-based approaches to grammar*. Center for the Study of Language and Information (CSLI) Lecture Notes No. 4, Stanford, CA, 1986.
- [5] Grosz, B.J., A.K. Joshi and S. Weinstein. Providing a unified account of definite noun phrases in discourse. *Proceedings of the 21st Annual Meeting of the Association for Computational Linguistics*, Cambridge, MA, pp. 44-50, 1983.
- [6] Grosz, B.J. and C.L. Sidner. Attention, intentions, and the structure of discourse. *Computational Linguistics* Vol. 12, No. 3, pp. 175-204, 1986.
- [7] Pereira, F.C.N. A structure-sharing representation for unification-based grammar formalisms. *Proceedings of the 23rd Annual Meeting of the Association for Computational Linguistics*, Chicago, IL, pp. 137-144, 1985.

- [8] Karttunen, L. and M. Kay. Structure sharing with binary trees. *Proceedings of the 23rd Annual Meeting of the Association for Computational Linguistics* Chicago, IL, pp. 133-136, 1985.
- [9] Shieber, S.M. Using restriction to extend parsing algorithms for complex-feature-based formalisms. *Proceedings of the 23rd Annual Meeting of the Association for Computational Linguistics* Chicago, IL, pp. 145-152, 1985.
- [10] Bear, J. Morphological recognizer with syntactic and phonological rules. Technical Note 396, Artificial Intelligence Center, SRI International, Menlo Park, CA, 1986.
- [11] Pereira, F.C.N. Toward a deductive theory of sentence interpretation. Unpublished manuscript, 1988.
- [12] Pollack, M.E. and F.C.N. Pereira An integrated framework for semantic and pragmatic analysis. *Proceedings of the 26th Annual Meeting of the Association for Computational Linguistics* Buffalo, NY, pp. 75-86, 1988.
- [13] Pell, B. Candide User's Manual. Internal Document, Artificial Intelligence Center, SRI International, Menlo Park, CA, 1988.
- [14] Pereira, F.C.N. and M.E. Pollack A compositional, declarative system for semantic and pragmatic analysis. In preparation.