

A KNOWLEDGE-BASED APPLICATION DEFINITION SYSTEM

Clifford R. Hollander
Harry C. Reinstein
IBM Scientific Center
1530 Page Mill Road
Palo Alto, California 94304

The design of an experimental consultation system, here called ADS, aimed at providing expert-level assistance to (nonprogrammer) specialists in defining medically-oriented, sensor-based applications is considered. The heart of the system is a knowledge base containing extensive static descriptions of the problem domain and mechanisms for instantiating elements of a subject application.

1. INTRODUCTION

One very promising direction for facilitating more direct involvement by the (non-programmer) problem area specialist in the application definition process is the use of an intelligent computer-based assistant, capable of offering expert-level guidance. This paper considers the design of an experimental knowledge based consultant [1], here denoted as ADS, which incrementally elicits (and presents in flowgraph form) a detailed specification of the subject application. The task domain selected for exercising these concepts involves gathering, analyzing, and displaying data obtained from sensor-based medical instrumentation.

ADS has been heavily influenced in its design by a number of research AI systems [2,3,4,5,6,7,8]. It maintains a substantial knowledge base, containing both static descriptions of the problem domain, obtained from human experts, and dynamic descriptions of the subject application, generated during the consultation. The knowledge base is used to direct a simple, yet powerful, set of procedural mechanisms.

The emphasis here is on achieving superior problem-related expertise, ease of use, and extensibility. A prototype (coded in APL) was only recently begun, therefore most of this discussion pertains to design issues.

2. AN EXAMPLE APPLICATION

Fig. 1 depicts a typical ADS application, namely to display systolic (i.e., maximum arterial) blood pressure. Starting from this partial specification, ADS and the user cooperate in discovering the semantic relationships which complete the diagram:

(1) A 'windowing' criterion is needed to govern the number of values input to MAXIMUM on each iteration. The user decides to process the sample points gathered during a single cardiac cycle. This entails introducing an additional sensor device and an event handler.

(2) For enhanced readability, ADS suggests that several seconds of MAXIMUM-readings be averaged before updating the display screen.

3. EXTERNAL APPEARANCE

The dialog is conducted wholly in terms of primitives familiar within the problem domain, e.g., sensors, displays, sampling rates, averaging windows, events. It is materialized as a succession of interactive panels, each dynamically tailored to reflect the needs of the current dialog state. A panel appears as a collection of screen areas (embodying menus, lists, prompts, messages, graphs, and key-in slots) together with usage restrictions (e.g., value constraint, selection criteria, view-only, scrollability, panel chaining). Utilizing full-screen panels as the units of communication

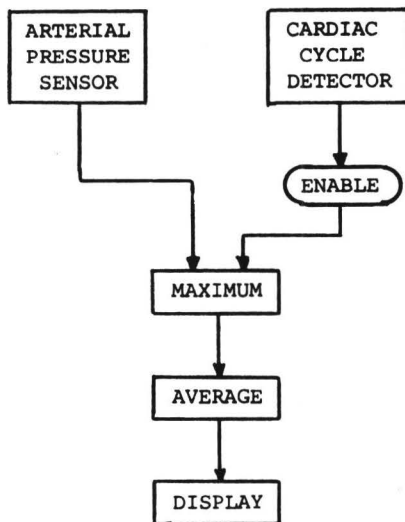


Fig. 1. Systolic Pressure

should make for a more comfortable dialog.

Throughout the definition process, a flowgraph, reflecting the current application description, is developed on a separate graphic screen. Extensive feedback and contextual information is available both from this diagram and from the knowledge structures associated with its components.

Also, an Explanation facility [2] is planned, in order to give the user auxiliary information about the dialog session and the knowledge base.

4. CHOICE OF REPRESENTATION

Knowledge in ADS is divided into two major categories based upon differences in origin, usage, and lifetime. The Resident Knowledge Base (RKB) is built by human experts and contains the static descriptions by which the system models the conceptual primitives which underlie the problem domain. Each such domain entity, has a number of identifiable components, each with certain attributes, and participates in various relationships with other entities. The Application Knowledge Base (AKB) is built during the consultation and contains dynamic descriptions which characterize the subject application in terms of the RKB model. Each application entity is an instantiated AKB version of a relevant domain entity, created as

a result of traversing a portion of the RKB.

The RKB and AKB share a single object-centered representation scheme. Entities are expressed in KRL-like [9] notation and embedded within a semantic network [10], using several explicit edge types for semantic linkage. In the RKB, links are used primarily to express specialization (SPEC edges) and reference (REF edges) relationships among domain entities. Whereas, in the AKB, links tend to denote actual attachments (ATT edges) between the application entities involved. Other kinds of relationships are represented implicitly by introducing relation entities into the network and by partitioning it into subnets [10].

An entity contains a set of named slots, which elaborate its attributes and components via sets of (nested) descriptive clauses. A clause consists of a (1) value or value constraint; (2) datatype, e.g., heartbeats/minute, time series; (3) reference to another slot, entity, or subnet; (4) source designation, e.g., user-supplied, filled from another slot; (5) specialization constraint. The order of clauses within a slot is irrelevant. Some examples of clauses:

356, in SYMPTOM-LIST	(values)
NUMERIC, WAVEFORM	(datatypes)
<u>ref-to</u> PRESSURE-DATA	(REF)
<u>fill-from</u> MINVALUE of RANGE	(source)
<u>spec-of</u> SENSOR with MODE=ANALOG	(SPEC)

Considering only SPEC edges, the RKB is a set of hierarchies, one for each major class of primitives, e.g., sensors, displays, functions, connections, data. With respect to these hierarchies: (1) An entity inherits all slots known to its parent. (2) A leaf is a terminal node; it corresponds to an actual domain object. (3) A prototype is a nonterminal node; it stands for an entire (sub)class of objects.

5. BEHIND THE DIALOG

The dialog is an elaboration process in which the elements and flow of the application are recursively analyzed, using the RKB as a model, extracting the details necessary to synthesize the AKB. In each segment of the analysis, a SPEC path is traced from a prototype down to one of its leaves. Along the way, the slots of each entity encountered are used to propagate the information needed to instantiate a new application entity. The entity paths for the

systolic pressure example would be:

DEVICE	FUNCTION
SENSOR	ARITHMETIC
PRESSURE	*MAXIMUM
*ARTERIAL	*AVERAGE
...	*EVENT-HANDLER
DETECTOR	...
*CARDIAC-CYCLE	DISPLAY
	FIELD
	*NUMERIC

where indentation indicates specialization and an asterisk precedes each leaf.

The clauses comprising a slot are processed according to their type. Value, datatype, and source related clauses are used to fill AKB slots (including user interactions). Specialization clauses control selection among alternative SPECS. Reference clauses (to other domain entities) are logged on a work list and serve as the origins of new traversal segments.

6. KNOWLEDGE ACQUISITION

The knowledge acquisition facility is organized around an Editor and a Checker. The Editor offers the problem domain expert full-screen interactive capabilities for examining, building, and modifying the RKB and (during a definition session) the AKB. Attention may be focussed upon knowledge structures at the subnet, entity, or slot/clause levels. The Editor is the sole overseer for activities performed at the subnet and entity levels.

The Checker is called to syntactically validate the contents of any new or altered slot. It makes use of a formal grammar for clauses together with a simple parser. A modest amount of semantic validation is also performed at this time, e.g., checking conformance between a slot-name and an entity-name in a reference clause.

7. AN ASIDE

Application definition may be viewed as a first step in a larger automatic programming process. In this light it is reasonable to consider extending ADS so that it generates executable programs corresponding to an elicited application description.

8. CONCLUSIONS

We have tried to design a consultant which is capable of offering expert-level assistance to nonprogrammers in specifying applications. The heart of the system is a knowledge base with both static and dynamic components, which drives a high-level problem definition dialog.

REFERENCES

- [1] Feigenbaum, E., The art of AI: themes and case studies in knowledge engineering, Proc. 5th IJCAI, M.I.T., 1977, 1014-1029.
- [2] Shortliffe, E., Computer-based Medical Consultations: MYCIN, Elsevier, New York, 1976.
- [3] Buchanan, B.G. and Feigenbaum, E., DENDRAL and meta-DENDRAL: their applications dimension, Stanford Heuristic Programming Project HPP-78-1, Feb. 1978.
- [4] Martin, N. et al, Knowledge management for experiment planning in molecular genetics, Proc. 5th IJCAI, M.I.T., 1977, 882-887.
- [5] Lenat, D., AM: an artificial intelligence approach to discovery in mathematics as heuristic search, Stanford Heuristic Programming Project HPP-76-8, 1976.
- [6] Duda, R. O. et al, Development of a computer-based consultant for mineral exploration, SRI Intl., Ann. Report, Oct. 1977.
- [7] Bennett, J. et al, SACON: A knowledge-based consultant for structural analysis, Stanford Computer Science Dept. CS-78-699, Sept. 1978.
- [8] Green, C. and Barstow, D., On program synthesis knowledge, Artificial Intelligence 10, 3 (Nov. 1978), 241-280.
- [9] Bobrow, D. and Winograd T., An overview of KRL, a knowledge representation language, Cognitive Science 1, 1 (Jan. 1977), 3-46.
- [10] Fikes, R. and Hendricks, G., A network based representation and its natural deduction system, Proc. 5th IJCAI, M.I.T., 1977, 235-246.