TELOS, A LANGUAGE FOR BUILDING WELL-STRUCTURED AI MODELS

L. Travis, M. Honda, R. LeBlanc and S. Zeigler
Computer Sciences Department
University of Wisconsin - Madison
Madison, Wisconsin 53706

TELOS is an attempt to provide powerful abstraction mechanisms and other structuring facilities within a language that provides the special capabilities needed for AI research. TELOS includes PASCAL as a subset and also is implemented in PASCAL. A full description is available in [1] and [2],

Like most other AI languages, TELOS includes facilities needed for experimentation with large stores of general knowledge, tentatively modifiable and associatively referenceable, and with various planning and reasoning strategies. However, in contrast to other AI languages whose design has focused on building in certain powerful high-level constructs, the design of TELOS has focused on building in powerful abstraction mechanisms with which these particular high-level constructs, as well as numerous others, can be defined and implemented with reasonable ease.

The usefulness of abstraction in all kinds of human intellectual activity has long been observed, and programming is certainly no exception. Only recently, however, have programming languages begun to appear with features specifically designed to facilitate abstraction of the several different kinds needed in the programming process, in particular, data abstraction and control abstraction as well as procedural abstraction. TELOS implements a set of data, control, and procedural abstraction mechanisms specifically tailored to AI requirements.

The data abstraction capabilities provided in TELOS add to the already powerful data type extension capabilities of PASCAL. The user defines a problem-specific data type by including details of representation and implementation within a definitional scoping called a "capsule". The procedures and functions which realize possible primitive operations on objects of the type being defined are an integral part of the definition, and the objects are characterized and used in terms of these defining operations. Features are provided for specifying and controlling how objects of a capsule-defined type are to be fitted into the TELOS associative data base.

An AI programmer might use TELOS capsules to define data types ranging from low-level lists (of the kind built into most other AI languages) to theoretically significant data structures. Such high-level abstract type definitions might implement, for example, case-slotted prototypes (frames), concept-dependency graphs, or production "conditionals". By emphasizing abstraction mechanisms rather than high-level constructs (e.g., frames as a basic data type, as in KRL), it has been possible to minimize theoretical bias in the language. Consequently TELOS should be usable for investigation of numerous competing theories, for example, in the area of knowledge representation.

The control abstraction capabilities provided in TELOS enable convenient user definition of the novel kinds of control regimes which are investigated in AI research, that is, those which realize alternative problem-solving strategies. Just as TELOS capsules localize data representation details, TELOS "overseers" localize interprocess control-transfer and communication details needed to realize a desired control regime.

An AT programmer might use TELOS overseers to define control regimes ranging from blind, depth-first backtracking (of the kind built into many recent AI languages) to quasi-parallel processing to sophisticated planning strategies. Features are provided which enable the procedures and processes controlled by an overseer to be identified and invoked descriptively as well as by name.

Besides its abstraction mechanisms, TELOS contains other facilities which can contribute to the building of well-structured programs, starting with the rich set of program structuring facilities already available in PASCAL. If used correctly, these facilities can result in modularized, hierarchical programs with reasonably comprehensible control and data flows.

Many language capabilities very useful in AI programming directly conflict with principles of good program structuring. The TELOS design attempts to mediate the conflicts with compromises which give AI programmers some of the advantages of both worlds. Examples: (1) TELOS allows the construction of a heterarchical control regime but centralizes into one location its communication and control-transfer activities. (2) TELOS allows specification of conditional interrupts (i.e., demons) but subsumes them under a general, unified event handling scheme. The scheme enables signal, escape (error), and suspend events to be explicitly declared and activated, and (whether explicit or not) to be operative only within certain clearly circumscribed and easily determined scopes. (3) TELOS allows creation of alternative data base versions (i.e., contexts) but requires that switching of the current execution context be accomplished with the structured form:

InContext <context> Do <statement>

Several benefits should accrue for AI from such a language design,

(1) AI programs are highly complex. A language like TELOS provides many aids to managing and containing program complexity.

(2) Improved AI programmer productivity, as is possible with the features TELOS provides, will mean improved AT research productivity.

(3) Much of AI programming involves putting preliminary ideas and hypotheses into programs and then changing the programs as suggested by experience with them. The improved program comprehensibility possible with TELOS can make this kind of evolutionary programming much easier and more efficient.

(4) One of the motivations of work on AI is development of a general theory of intelligence. Such theory development will be expedited if the program abstractions representing high-level theoretical constructs are as distinct and as well-defined as they can be in TELOS programs.

References

[1] Travis, L., M. Honda, R. LeBlanc and S. Zeigler. 1977. Design rationale for TELOS, a PASCAL-based AI language. Proceedings of SIGPLAN-SIGART symposium on AI and programming languages.

[2] ———. 1977. TELOS design specifications. UW-Madison, Academic Computing Center Technical Report.