PORTABLE SOFTWARE MODULES FOR SPEECH RECOGNITION

(DARPA PHASE II SBIR Contract DAAH01-91-C-R297)

John Shore, Principal Investigator

Entropic Research Laboratory
600 Pennsylvania Ave. S.E., Suite 202
Washington, D.C. 20003

PROJECT GOALS

The main goal of this work is to create programming libraries for speech recognition R&D that not only facilitate rapid prototyping, but also raise the probability that the prototypes will be robust and portable enough to serve beyond the prototype stage. Specific technical objectives for the first year include:

(1) Refine the software engineering method that was developed during Phase I. (2) Choose an existing speech recognition system as a "reference" target. (3) Produce a working subset of the utility modules. (4) Produce a working subset of the numerical algorithms modules. (5) Write an open specification for the file formats and I/O functions. (6) Produce example user-level programs (including at least one auditory model).

RECENT RESULTS

Since contract award (1 Sept. 1991), we have concentrated on refining the software engineering method and building a significant set of low-level utiility modules. The method yields reusable, modular C libraries with these properties:

- Modules are simple enough to be designed, written, tested, understood, and maintained by a single person (at a time).
- Testing adding, removing, replacing, or modifying modules are simple operations.
- The result of combining several modules into a library is itself useable as a module in a "higher level" library.
- Modules are organized into a "uses hierarchy" that makes the module dependencies explicit.

Our mehod does not require an elaborate CASE environment. Rather, we stick to ANSI C, a full-featured make program (at present, *gnumake*), and a collection of relatively simple shell scripts. The development was carried out in the context of writing (and re-writing) 10 utility modules, complete with test programs and documentation. The more important modules are these:

msgio: primitive functions for exchanging ASCII messages with the user

usrmsg: functions for verbosity-constrained user messages

estop: functions for halting an application

events: exception handling

signals: turns UNIX signals into events

emalloc: dynamic memory allocation

estrings: strings and string arrays

types: numeric data types

arrays: multidimensional arrays

PLANS FOR THE COMING YEAR

We are now engaged in the development of modules for signal representation and signal processing algorithms. Design goals include providing for automatic type conversion if a type-specific signal processing function is not available, having uniform treatment of signals regardless of their storage location (memory, disk, or mixed), and providing run-time parameterization of the number of records (e.g., frames) dealt with as a single unit.

The next tasks will be to specify and implement the file system, to implement the command-line user-interface modules, and then to complete several user-level programs that demonstrate and illustrate various aspects of the approach. We will also be selecting an existing speech recognition system to serve as a target for the optional second year of the project.

A detailed progress report (and plan for the optional second year) will be available in May, 1992.