Behavior Research Methods, Instruments, & Computers
2003, 35 (2), 263-268

A computer-aided digital audio recording and encoding system for improving the encoding of verbal reports

ROBERT J. CRUTCHER
University of Dayton, Dayton, Ohio

The increasing use of verbal reports in psychological research requires tools for improving the ease and reliability of collecting and coding verbal report data. An approach is described that maintains the verbal report data in digitally recorded audio form throughout the collecting and encoding processes. A new computer-aided encoding tool, CAPAS, is described, which randomly selects and plays individual protocol segments and stores computer keyboard-entered codes in an SPSS-formatted data file.

Verbal reports lend themselves to use in diverse task domains and provide important behavioral data (see, e.g., Austin, 1999; Chi, 1997; Crutcher, 1998; Crutcher & Ericsson, 2000; Ericsson & Simon, 1993). For example, they provide a high temporal density of behavioral data critical to the evaluation of psychological theories (Anderson, 1987; Crutcher, 1994, 1998; Ericsson & Simon, 1993; Simon & Kaplan, 1989). The increasing use of verbal report methodologies requires that methods for collecting, coding, and analyzing these reports must improve accordingly. A number of theoretical issues have been raised by previous researchers, such as the need for rigorous task analysis before reports are collected and the importance of localized encoding schemes, which require that a given protocol be coded with minimum reliance on other protocol segments or surrounding context (Crutcher, 1994; Crutcher, Ericsson, & Wichura, 1994; Ericsson & Simon, 1993). My central concern here is the use of computer hardware and software tools to improve the efficiency and ease of collecting and coding verbal reports while adhering to these important theoretical constraints. Many aspects of the collecting and encoding of verbal report data are very time-consuming (Bainbridge & Sanderson, 1995; Chi, 1997; Crutcher et al., 1994). Here I describe a new approach to collecting and coding verbal protocols that maintains the verbal reports in a digitally recorded format throughout the process of collecting, segmenting, and coding the reports, while enforcing the localized encoding principle (Crutcher et al., 1994; Ericsson & Simon, 1993). A new computer software tool named CAPAS (Computer-Aided Protocol Analysis System), along with standard computer software tools, eliminates the need for conventional cassette tape, tape players, and transcription machines and increases the speed and efficiency of collecting and coding verbal reports.

Computers have been used in two types of systems for encoding verbal report protocols: automatic computerized encoding systems, in which encoding is accomplished entirely by a computer program (Waterman & Newell, 1971, 1973) and semiautomatic encoding systems (Bainbridge & Sanderson, 1995; Bhaskar & Simon, 1977; Crutcher & Ericsson, 2000; Crutcher et al., 1994; Ericsson & Simon, 1993; Fisher, 1988; James & Sanderson, 1991; Sanderson, James, & Seidler, 1989). Waterman and Newell's (1971) PAS-I is one of the most ambitious systems of the first type. In this program, a problem space is first defined, in order to limit the number of possible coding categories onto which the protocol statements are mapped. The protocol statements are segmented manually, processed by a linguistic parser, mapped onto a set of propositional relations, and reduced to a smaller set of semantic primitives. These primitives are eventually translated into a problem behavior graph that captures a person's successive knowledge states and the productions required to move from one state to another. The strength of a completely automated approach is that all protocol segments are encoded similarly, with consistent application of the coding rules. In addition, because the rules must be incorporated into the program itself for it to work, evaluation of the rules is straightforward. Unfortunately, such programs have been implemented only in well-defined task domains for which linguistic parsers can be developed.

Semiautomatic encoding systems, in which human coders classify the reports but are aided by computer tools that increase the objectivity, reliability, or efficiency of human coders, are much more common. In a semiautomated encoding system, the coding decision responsibility is split between a human coder and the computer program, with relative responsibilities of coder and program varying from one system to another. At one extreme, the program may store considerable domain

Inquiries regarding availability of the CAPAS program as well as other correspondence should be addressed to Robert Crutcher, either via e-mail (crutcher@udayton.edu) or via U.S. mail (Department of Psychology, University of Dayton, 300 College Park, Dayton, OH 45469-1430).

knowledge and assume major responsibility for coding decisions, with the human coder simply confirming or denying and perhaps revising coding decisions made by the program. At the opposite extreme, the program may have little or no domain knowledge but merely presents individual protocol segments and perhaps possible coding categories, leaving major responsibility for encoding decisions exclusively to the human coder.

SAPA (Bhaskar & Simon, 1977) is an example of a semiautomatic system in which much of the domain knowledge is contained in the program itself. Possessing a model of the subject's problem solving strategy as it processes a protocol segment, SAPA predicts the next operation that the subject will apply but offers the coder the option of correcting the program's decision and returning the program to the correct path. Examples of semiautomatic systems in which no specific domain knowledge is contained in the program are SHAPA (James & Sanderson, 1991; Sanderson et al., 1989) and MPAS (Crutcher et al., 1994). These systems aid in the encoding process by handling large numbers of protocols and focusing the coder's attention on individual segments one at a time, but leave coding decisions exclusively to the human coder. The advantage of systems such as SHAPA or MPAS over systems such as SAPA consists in their wider applicability, because domain specific knowledge is not built into the system. A potential disadvantage of such systems is that often they cannot eliminate all of the subjective aspects of coding entirely. Whereas a machine program can be designed to encode a specific segment in the same way regardless of the surrounding context, a human coder's encoding can be influenced by context. For example, information from previously coded segments from the same participant or experimental treatment condition may influence the coding of a current segment. In addition, whereas automated programs can easily code hundreds or even thousands of protocols rapidly without tiring, human coders code much more slowly and tire after even relatively small numbers of protocols, particularly if the coding process is done by hand. Finally, human coders may fail to apply coding rules consistently unless the rules are explicit and can be applied to the protocols straightforwardly. Nevertheless, if semiautomatic systems are properly implemented and if the principle of localized encoding is implemented, these differences can be minimized.

Implementing the Localized Encoding Principle

The MPAS (Multiple Protocol Analysis System) program (Crutcher et al., 1994) is an example of a system that rigorously implements the localized encoding principle. Whenever possible, it is preferable to code verbal reports without reference to the surrounding context. This is referred to as the *localized encoding principle* and is a key feature of MPAS. MPAS forces the coder to code each protocol segment independently and without reliance on contextual information. This is accomplished by storing all of the protocol segments for a group of subjects in an experiment in a single file before coding begins. During coding, MPAS randomly selects and presents individual protocol segments to the coder. Information concerning the identity of each protocol segment (e.g., subject number, condition, etc.) is kept hidden during the encoding process.

The MPAS program also implements the localized encoding principle in a second way: By improving the ease and efficiency of coding verbal protocols, MPAS makes it possible to actually code many more protocols from a larger pool of subjects than would be practically possible without the program. As the number of subjects and protocol segments in the stored pool of protocols increases, the likelihood of a coder remembering and using information from previous segments to code a current segment decreases.

A General Description of CAPAS

Despite its efficiency in coding large numbers of individual protocol segments, MPAS, like most computer-aided coding tools, requires that one first transcribe all of the verbal reports before coding begins. Not only is this transcription process extremely time consuming, it can introduce errors into the data. CAPAS (Computer-Aided Protocol Encoding System) is a new computer tool inspired by the former MPAS program that employs the same localized encoding principle in its approach.[1] CAPAS dramatically improves on MPAS by eliminating the need for transcribing the verbal reports. CAPAS accomplishes this by maintaining the recorded verbal reports in digital audio format and presenting (or playing) the individual segments in randomized order for encoding. Unlike MPAS, though, CAPAS is designed to code relatively brief verbal protocols rather than longer verbal protocols such as one might collect in problem-solving tasks. CAPAS is appropriate for coding protocols collected in memory or learning strategy experiments in which brief think-aloud or retrospective verbal reports for many hundreds or thousands of trials are collected. CAPAS would also be appropriate for coding verbal interactions of relatively brief duration. In principle, CAPAS can play back audio segments or protocols of any length; however, longer protocols, if they cannot be broken down into smaller segments for coding, are probably more easily coded from a written transcription. It is also important to note that CAPAS does not automatically encode the recorded protocols. The researcher must previously define a coding scheme before using CAPAS to code a set of protocols, and these codes must consist of five character strings of letters or numbers.

The use of CAPAS to code protocols is straightforward and will be described in a moment. However, before coding can begin, all of the individual protocols or protocol segments for an experiment must be stored in individual sound files (AIFF or Sound Designer II format) which must be named so that each file name identifies a subject number and a protocol or protocol segment number (e.g., 1-1, 1-2, 1-3, 1-4, . . . 2-1, 2-2, 2-3). This is so that CAPAS can associate each entered code with the appropriate protocol.

This preprocessing of the recorded verbal protocols can be accomplished with digital recording and editing software (e.g., Bias' Peak[2]) or digital recording hardware and audio editing software together. A variety of approaches are possible, and each has merits. One approach is to use the digital recording and editing program to record the verbal protocols while running a participant and, while recording, to mark the individual protocol boundaries. The Bias' Peak application, for example, allows one to mark audio regions with a simple keystroke during recording and then later export the marked regions to individual sound files. The criteria for segmenting the recording are determined by each researcher's needs: For example, in my current experiments, recordings are segmented on the basis of the individual memory retrieval trials. However, recordings can also be marked and segmented on the basis of pauses or sentence units, depending on the specific goals of the researcher. Peak's export function can append prefixes to file names and produce a set of individual sound files labeled with the appropriate naming scheme required by CAPAS, where the first number is the subject number and the second is the protocol or protocol segment number.

Alternatively, recordings can be made with an external digital recording device (such as a CD/RW recorder) to make recordings that are later transferred to the desktop computer via a high-speed data connection (e.g., USB or Firewire). Afterward, a digital audio editing tool is used to mark and segment the audio file into discrete audio regions and then export them to individual sound files as described previously. A potential advantage of this approach is that a hardware recording solution is more reliable than a software recoding solution; there are no computer crashes. However, the downside of first recording on digital hardware and then transferring to the desktop machine is that the recorded audio must still be segmented and exported, which requires additional time. An ideal solution to the latter problem is a CD/RW recorder that allows marking of track segments on the recorded CD/R disk media during the recording process. Most CD/R recording machines have a remote controller with a "create new track" button. During the recording process, the experimenter simply presses this button each time a new protocol begins. The limitation of no more than 99 tracks to a disk simply requires that a new CD-R must be used after the limit is reached. An additional advantage of this approach is that the recorded protocols are automatically archived. Transfer of the individual protocol tracks to the desktop is accomplished with a simple select and copy operation that produces a separate audio file for each recorded track. All that remains is to rename the files.

Once the individual audio files for all participants in an experiment are created, they are placed together in a "files" folder along with the CAPAS application. As CAPAS starts up, a new coding session can be initiated (see Figure 1), and all of the names of the audio files from the "files" folder are automatically read into a list (see Figure 2) and stored in an external backup file.

This list is randomized and used as an index to select and present the individual audio segments to the coder as well as to keep track of how many protocols have been coded and how many remain to be coded. Each protocol is presented by *playing* the corresponding audio file for the coder. After listening to a protocol, the coder enters a code (see Figure 2), which CAPAS stores and associates with that specific protocol, writing everything to an external tab-delimited file. Only a single code may be entered for each presented protocol or protocol segment, and this code must consist of a string of 5 alphanumeric characters (e.g., 11101).

Coders sometimes need additional information while coding a set of protocols. For example, in one of our

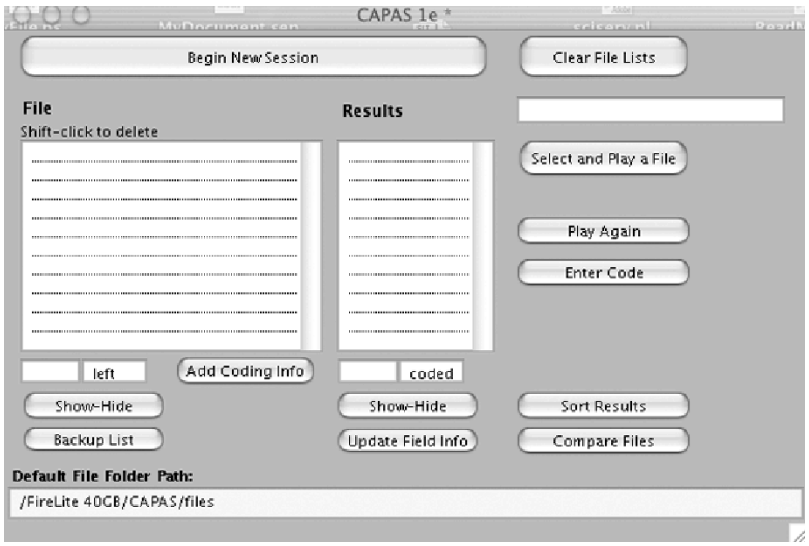

Figure 1. CAPAS interface upon starting up the program.

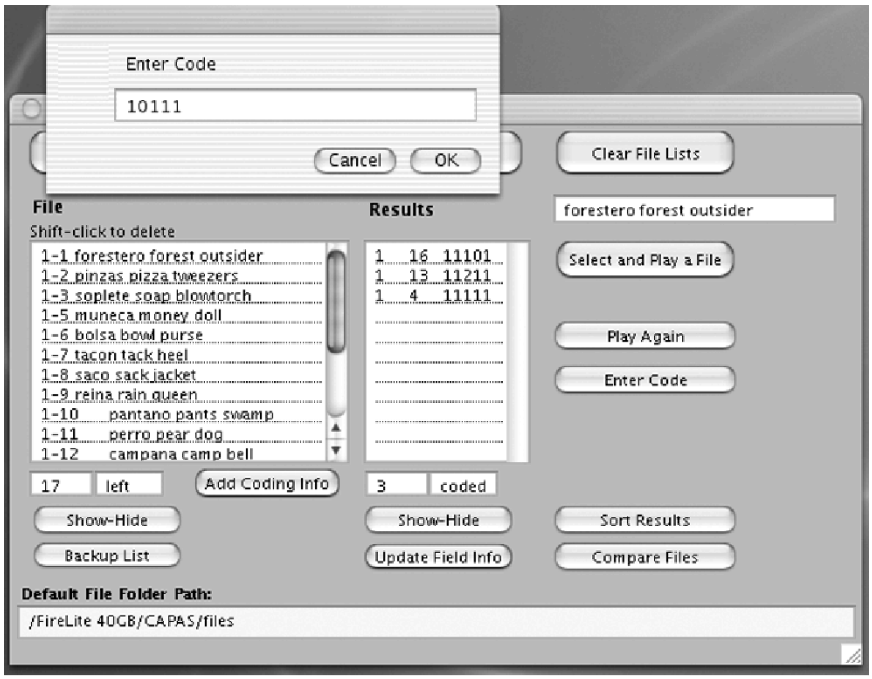

Figure 2. CAPAS interface after reading in audio file names and coding a verbal protocol.

memory retrieval experiments in which word pairs were learned using specific mediators or images, the coder needed to know the word pair for a trial as well as the original mediator used to learn the pair in order to code the correctness of the response and whether or not the original mediator was mentioned. Any additional information that the researcher would like a coder to see during the coding process can be placed in an external file so that each line of the file corresponds to a protocol for an individual subject on a specific trial (i.e., the first line of the file will align with audio file 1-1, the second line with audio file 1-2, etc.). Before beginning coding, the coder clicks on an Add Coding Info button (see Figures 1 and 2) to ensure that this information is displayed for each protocol during the coding process.

At any time during coding, the coder may quit CAPAS and resume coding at a later time. When coding resumes, the coder clicks on an Update Field Info button to resume coding exactly where coding ended. After coding all protocol segments, CAPAS alerts the user that all protocols have been coded, at which point the user clicks on a button to save all subject and trial information, the protocol codes, and the additional coding information to a tab-delimited ASCII text file for analysis by a statistical package.

A Comparison of CAPAS to MPAS in Analyzing Verbal Report Data From a Laboratory Study

I now briefly describe the use of CAPAS in coding some protocol data from a previously conducted laboratory study (Crutcher & Ericsson, 2000) in which MPAS was used to code the verbal protocol data. The task of interest here is memory retrieval of vocabulary pairs learned using keyword mediators (e.g., the Spanish–English pair *perro–dog* can be learned using the keyword mediator *pear* by first noticing the similarity in sound of *perro* and the keyword *pear* and then relating *pear* and *dog* by generating an interactive visual image of a "dog eating a pear," for example). Verbal reports were collected during the learning of these pairs as well as later, when participants were tested on the pairs with a cued recall procedure that presented the Spanish word as a cue and asked the participant to recall the corresponding English translation. Participants sometimes gave a retrospective verbal report of whatever they remembered thinking from the time when the cue appeared on the computer screen (e.g., *perro*) until their response (e.g., *dog*). These verbal reports were brief, on the order of a few seconds, and from as short as just a few words to as long as four or five lines or phrases. An example protocol is displayed in Table 1.

To assess the efficiency of CAPAS, I conducted an informal comparison of CAPAS and MPAS in coding a small sample of these verbal reports. From the set of 512 previously transcribed protocols, 20 protocols were selected at random and re-recorded. This recording was then used to test the two programs and measure the total time necessary to transcribe and code (in the case of MPAS) or segment and code (in the case of CAPAS) the recorded protocols. The length of the recording was 4 min 9 sec.

For processing and coding of the protocols with MPAS, it was necessary to first transcribe the 20 protocols with a software transcription program. This tran-

Table 1
Example of Individual Protocol File Encoded With MPAS and CAPAS

—perro pear dog—
dog
thought about a pear
and then a dog with it
eating a pear\

scription process required 17 min 3 sec. Subsequent coding of the transcribed protocols and generation of the final tab-delimited data file took an additional 3 min 53 sec, for a total processing time of 20 min 56 sec, or roughly 21 min total.

For the processing and coding of the verbal reports with CAPAS, Bias' program Peak was used to mark the sound regions corresponding to individual protocols while recording the verbal reports. Afterward, Peak was used to export the marked sound regions as 20 individual audio files in approximately 2 min 20 sec. Placing the files in a folder and then using CAPAS to code the reports and generate a data output file required an additional 5 min 51 sec, for a total processing time of 8 min 11 sec, or roughly 8 min. The total time for processing and coding 20 brief protocols was thus reduced from approximately 21 to 8 min, an absolute reduction of 13 min and a relative savings of approximately 62%.

Summary and Conclusions

As a tool to improve the efficiency of coding verbal protocol data, CAPAS appears to have significant potential. CAPAS showed a 62% savings in coding a set of protocols, in comparison with using the text-based MPAS program.

Although CAPAS has clear advantages over programs such as MPAS, which require that one first transcribe verbal protocols, some limitations do need to be acknowledged. First, coding protocols directly from the raw audio recordings is probably practical only when the protocols are of brief duration, as in retrospective reports gathered in memory or strategy experiments (e.g., Crutcher & Ericsson, 2000; Siegler, 1987, 1988). Longer protocols may exceed the capacity of a coder to process and code accurately. Empirical studies could test this. Second, transcribing protocols has the advantage that later, after coding, one can easily scan through the transcribed protocols to look for additional information or patterns in the protocols. Third, a possible limitation of CAPAS concerns the localized encoding principle: Some contextual information may be available that is absent when one is using MPAS. During the coding process, MPAS conceals all information concerning the identity of a participant as well as the treatment condition. CAPAS, though, because it plays the actual recorded protocols, provides voice identification information that might sometimes remind the coder of a subject's previous report. Anecdotally, I can report that in coding many participants' protocols at the same time, it is rather difficult to recall details of any given subject's previous protocols. A potential solution to this problem may be available, though. Currently, I am exploring several possible ways to apply frequency transforms to the recorded audio so that information identifying the speaker of a given protocol is disguised or masked during playback. One final limitation of CAPAS deserves comment. Though CAPAS produces a considerable time savings in coding verbal protocols in comparison with the traditional approach of first transcribing and then coding reports, recently some researchers have described coding systems that achieve even greater efficiency by relying on voice recognition technology to directly code audio data at the time of collection (e.g., White, King, & Duncan, 2002). Such systems are of course limited by current voice recognition software capabilities and, for the time being, are limited to experimental domains in which the range of verbal responses comprises single word vocabularies (e.g., color responses in a Stroop paradigm) or to verbal data from speakers who have been explicitly trained on the program for many hours.

As for the future, some specific changes and improvements in CAPAS are planned. First, whereas MPAS can be used to code verbal protocols of any length, for now CAPAS seems practical only in domains in which brief protocols are employed. I plan to merge the two programs into a single program that can be used to code recorded protocols directly (as CAPAS does now) as well as to code transcribed protocols (as MPAS does). Second, I plan eventually to rewrite CAPAS so that it can present the coding categories to the coder during the coding process to minimize miscoding errors.

In conclusion, CAPAS is a useful new tool for coding and analyzing verbal protocol data in domains where the protocols are of relatively brief duration. It improves dramatically on previous programs such as MPAS by reducing the time necessary for processing and coding protocol data. In addition, CAPAS enforces the localized encoding principle, ensuring that the verbal report data are coded and analyzed as rigorously as possible.

REFERENCES

Anderson, J. R. (1987). Methodologies for studying human knowledge. *Behavioral & Brain Sciences*, **10**, 467-505.

Austin, J. A. (1999). Exploratory protocol analysis of verbal behavior of expert and novice data analysts: Using concurrent verbal reports to examine data analyst verbal behavior. *Journal of Organizational Behavior Management*, **18**, 61-81.

Bainbridge, L., & Sanderson, P. (1995). Verbal protocol analysis. In J. R. Wilson & E. N. Corlett (Eds.), *Evaluation of human work: A practical ergonomics methodology* (pp. 169-201). Philadelphia: Taylor & Francis.

Bhaskar, R., & Simon, H. A. (1977). Problem solving in semantically rich domains: An example from engineering thermodynamics. *Cognitive Science*, **1**, 193-215.

Chi, M. T. (1997). Quantifying qualitative analyses of verbal data: A practical guide. *Journal of the Learning Sciences*, **6**, 271-315.

Crutcher, R. J. (1994). Telling what we know: The use of verbal report methodologies in psychological research. *Psychological Science*, **5**, 241-244.

CRUTCHER, R. J. (1998). The role of prior knowledge in mediating foreign vocabulary acquisition and retention: A process-analytic approach. In A. F. Healy & L. E. Bourne, Jr. (Eds.), *Foreign language learning: Psycholinguistic studies on training and retention* (pp. 91-111). Mahwah, NJ: Erlbaum.

CRUTCHER, R. J., & ERICSSON, K. A. (2000). The role of mediators in memory retrieval as a function of practice: Controlled mediation to direct access. *Journal of Experimental Psychology: Learning, Memory, & Cognition*, **26**, 1297-1317.

CRUTCHER, R. J., ERICSSON, K. A., & WICHURA, C. A. (1994). Improving the encoding of verbal reports by using MPAS: A computer-aided encoding system. *Behavior Research Methods, Instruments, & Computers*, **26**, 167-171.

ERICSSON, K. A., & SIMON, H. A. (1993). *Protocol analysis: Verbal reports as data* (rev. ed.). Cambridge, MA: MIT Press.

FISHER, C. (1988). Advancing the study of programming with computer-aided protocol analysis. In G. Olson, E. Soloway, & S. Sheppard (Eds.), *Empirical studies of programmers: 1987 workshop* (pp. 198-216). Norwood, NJ: Ablex.

JAMES, J. M., & SANDERSON, P. M. (1991). Heuristic and statistical support for protocol analysis with SHAPA Version 2.01. *Behavior Research Methods, Instruments, & Computers*. **23**, 449-460.

SANDERSON, P. M., JAMES, J. M., & SEIDLER, K. S. (1989). SHAPA: An interactive software environment for protocol analysis. *Ergonomics*. **32**, 1271-1302.

SIEGLER, R. S. (1987). The perils of averaging data over strategies: An example from children's addition. *Journal of Experimental Psychology: General*, **116**, 250-264.

SIEGLER, R. S. (1988). Strategy choice procedures and the development of multiplication skill. *Journal of Experimental Psychology: General*, **117**, 258-275.

SIMON, H. A., & KAPLAN, C. A. (1989). Foundations of cognitive science. In M. I. Posner (Ed.), *Foundations of cognitive science* (pp. 1-47). Cambridge, MA: MIT Press.

WATERMAN, D. A., & NEWELL, A. (1971). Protocol analysis as a task for artificial intelligence. *Artificial Intelligence*, **2**, 285-318.

WATERMAN, D. A., & NEWELL, A. (1973). PAS-II: An interactive task-free version of an automated protocol analysis system. In *Proceedings of the Third IJCA*, (pp. 431-445). Menlo Park, CA: Stanford Research Institute.

WHITE, D. J., KING, A. P., & DUNCAN, S. D. (2002). Voice recognition technology as a tool for behavioral research. *Behavior Research Methods, Instruments, & Computers*, **34**, 1-5.

NOTES

1. CAPAS was created using Revolution 1.1.1, and it runs within that environment. Revolution is a cross-platform HyperCard-like programming environment designed for authoring and running multimedia programs on multiple operating systems, including all versions of Windows and Mac OS as well as many versions of UNIX. Although written to run under Mac OS X and OS 9, with minor modifications CAPAS should run under any version of Windows or UNIX. Revolution can also be used to easily modify CAPAS to suit a specific user's needs. Further information about CAPAS as well as copies of the current CAPAS program, which requires Revolution 1.1.1 to run, are available free upon request from the author (crutcher@udayton.edu). In addition, a stand-alone program version of CAPAS 1.0 for Mac OS X and OS 9 that does not require Revolution 1.1.1 is also freely available. Stand-alone versions of the program for Windows will be available in the future.

2. On the Macintosh platform, Bias's Peak program is one of many possible digital audio recording and editing programs that may be used; other possibilities include SoundEdit and Sound Designer. I prefer Peak because of its advanced features for marking sound regions and automating the exporting of these regions to individual sound files. On the Windows side, there are many options as well (e.g., Sound Forge and Cubase VST), but I am not familiar enough with them to recommend one over another.

(Manuscript received November 14, 2002;
revision accepted for publication February 8, 2003.)