In the first version, the student begins by choosing an *agent*, either Edith Piaf or Charles Degaulle. Then, for each of a set of 'critical situations' in the story the agent offers to use one of a number of arguments consonant with the agent's character. The student selects the one argument she thinks will be most effective in the given situation and then types, in French, an explanation of her choice. The student then receives two kinds of feedback: After explaining her choice, the story continues and so she gets immediate feedback as she sees the effect of her argument on the situation; Additionally, her explanation is sent to an instructor for grading, providing her with delayed but detailed feedback on her analysis and writing skills.

In the second version the student argues *directly* on behalf of Philippe. Here the student generates her own argument; for each of the situations in the story a list of possible topics, around which the student's argument could center, is presented. The student selects the topic she feels will be most effective and expands it into a full argument. Again the student immediately sees the effect of her argument and receives later comments from her instructor.

COMPARISON OF THE 'DIRECT' AND 'AGENT' VERSIONS

In both implementations students need sufficient comprehension skills to understand the situations depicted in the story. Additionally, they must be able to analyze the meaning of these situations within French culture. They are also required in both cases to construct coherent, extended arguments in French.

However, the implementations differ in the use the students are required to make of their analyses. The Agent version relieves the students of the pressure to formulate arguments on their own and so they are in a more passive role. In the Direct version students are allowed a more active role in constructing what they feel is the best version of the argument. The Direct version is not however, unequivocally preferable since the story can reach a successful outcome even in cases where students were not able to construct a good version of their chosen argument. We felt this was pedagogically problematic and so wanted to construct the Agent version to avoid this problem. It does so in that the agent always offers a well formed argument, leaving the students to construct text which does not effect the outcome, but which instead explains the reason for chosing a particular argument; and this text is graded by a human instructor.

OUTCOME & SUMMARY

In tests of the software we have found that students enjoyed using both versions, felt that they would like to see the software incorporated into their curricula and felt encouraged to continue with their studies after having used the software. We feel that these results suggest that the software does therefore provide the kind of positive educational experience which we had hoped to create.

CONTACT INFORMATION

Beth Adelson
Department of Psychology
Rutgers University
Armitage Hall
Camden, NJ 08102 U.S.A.
609-757-6080 (work)
adelson.chi@xerox.com

[1] Thanks to Gilberte Furstenberg of the Athena Language Learning Project for her thoughtful and creative contributions. The interface described here is not related to the interface originally created by the Athena Language Learning Project for the disc "A la rencontre de Philippe". For information on that interface contact Dr. Janet Murray, MIT, Cambridge, MA.

SOUND

AURALIZATION OF PARALLEL PROGRAMS

Larry Albright, Jay Alan Jackson, & Joan Francioni
University of Southwestern Louisiana

INTRODUCTION

Auralization is the sonic equivalent of visualization. It is the attempt to convey the essence of massive amounts of scientific data in a manner that can be easily interpreted by humans. Visualization appeals to human sight, auralization appeals to human hearing.

Parallel program trace data is the domain chosen for this first attempt at auralization. Much visualization work has been done in the area of debugging parallel programs (Heath, 90). This is mostly in the form of graphs and animations which depict such program behavior as interprocessor communication and processor utilization.

Some of the reasons for considering auralization as an alternative to visualization are the following. 1) Audio information does not need to be directly attended. It fills the space which the human occupies, freeing human motion and sight for other tasks. 2) Audio information is inherently sequential, thus making it ideally suited to the portrayal of data which represent functions of time. 3) Multiple strands of information may be presented at once in audio. The history of the music of western civilization provides evidence that humans are capable of processing a vast array of audio information at once if it is properly presented. 4) With the advent of MIDI, the Musical Instrument Digital

Interface, and inexpensive hardware synthesizers, a wide range of audio information may be quickly and efficiently generated, stored, and reproduced. This is in stark contrast to many of the expensive, time-consuming graphical rendering processes.

MAPPING TRACE DATA TO AUDIO

In our implementation, we mapped various types of parallel program trace data which is generated by PICL (Geist, et al, 1990) into musical parameters. If we consider the PICL data as the domain of the mapping, then some of the elements of the domain include the number of processors, the identity of each processor, the type of action (e.g. send message, receive message, processor idle). Similarly, with music as the range of the mapping, some of the elements of the range can be identified by common music composition parameters: melody, harmony, rhythm, timbre, texture, dynamics, spatial separation. Following are some of the mappings we employed.

PROCESSOR IDLE TIME

For several types of parallel programs running on four, eight, and sixteen processor hypercubes, we auralized processor idle time trace data. Each processor was mapped to a unique pitch. One synthetic "instrument" was used for all processors. This "instrument" consisted of a bell-like attack followed by a string (violin) section sustain. The longer the processor was idle, the louder its sustained note became. The marked difference between attack and sustain provided the needed aural cues to easily hear the onset of a processor's idle time. The increasing loudness for processors which were idle an excessive amount of time served to draw the listener's attention. The wide pitch range of the orchestral string section made it appropriate to map as many as sixteen processors to the same timbre. A jazz type chord consisting of stacked thirds was used to assign pitches to the processors.

INTERPROCESSOR COMMUNICATION

Interprocessor communication is recorded in each send and receive that is done by each processor. Some of the information which we would like to portray is 1) the sender's identity and the receiver's identity at the time the message is sent; 2) the time the message is received; 3) a sense of how many messages are pending at any time in the program. One mapping assigned a unique melody to the "send" operation, a unique melody to the "receive" operation, a unique instrument and a unique pitch to each processor. When a processor sent a message, the send melody was transposed to the sender's unique pitch and played on the sender's instrument, with the final pitch of the send melody sustained until the message was received. When the message was received, the receiving processor played the receive melody on its instrument, but transposed to the sender's pitch. The transposition to the sender's pitch helped make the aural connection between sender and receiver. At the same time, the sender played an "end of transmission" melody, and, if no further messages were pending (to the same or other processors) terminated its sustain note.

CONCLUSION

The results of this initial study confirm that auralization of parallel program behavior can be a useful debugging and performance analysis aid. Future research includes the evaluation of the effectiveness of these tools for larger numbers of parallel programming experts, solving the problems of mappings from much larger numbers of processors (greater than 64), combining auralization with visualization for a synchronized, integrated presentation, and the application of auralization to other fields of science.

REFERENCES

Francioni, J.F., L. Albright, and J.A. Jackson, "Debugging parallel programs using sound," in Proceedings of the ACM/ONR Workshop on Parallel and Distributed Debugging," December, 1991.

Francioni, J.F., L. Albright, and J.A. Jackson, "The sounds of parallel programs," in Proceedings of the Sixth Distributed Memory Computing Conference, Portland, OR, April, 1991.

Geist, G.A., M.T. Heath, B.W. Peyton, P.H. Worley, "A user's guide to PICL: A Portable Instrumented Communication Library," Technical Report, Oak Ridge National Laboratory, October, 1990, ORNL/TM-11616.

Heath, Michael T., "Visual animation of parallel algorithms for matrix computations," in Proceedings of the Fifth Distributed Memory Computing Conference, Charleston, SC, April, 1990.

CONTACT INFORMATION

Larry Albright
Computer Science Department
University of Southwestern Louisiana
Lafayette, Louisiana 70504
(318) 231-6638
albright@gator.cacs.usl.edu

SCREEN, PHONE, AND VOICE USER INTERFACES

Richard Halstead-Nussloch - IBM
Mary Jacobson - Georgia Tech
Chan Chuongvan - IBM

ABSTRACT

Office principles often have multiple ways to complete typical tasks. For example, scheduling can be done on a paper calendar, electronically, by phone store-and-forward, etc. Each of the ways has unique user-interface characteristics and limitations, as well as a set of economic costs and benefits. We recently completed a study where 18 users compared traditional screen-based, phone-based, and voice-activated interfaces to calendar facilities. The voice-activated and phone-based user interfaces showed the