TECHNICAL REPORT

HyperVoice
A Phone-Based CSCW Platform

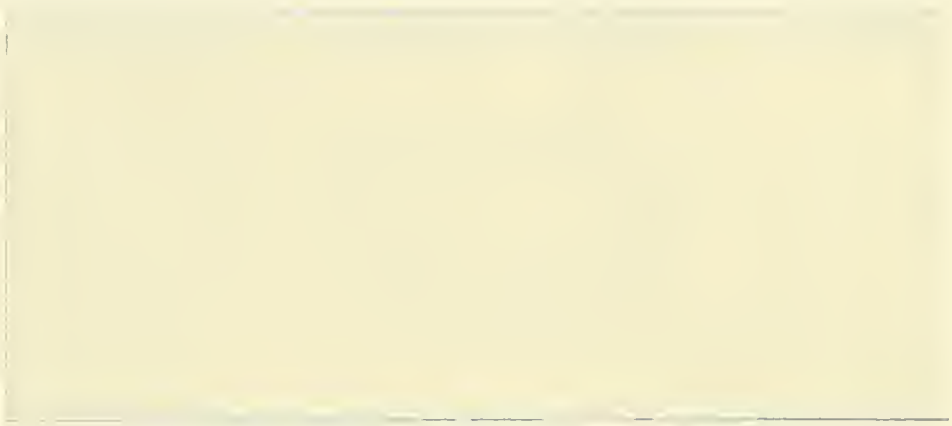
Paul Resnick

CCS TR # 133, Sloan School WP # 3463-92

August, 1992

CENTER FOR
COORDINATION
SCIENCE

Massachusetts Institute of Technology
Sloan School of Management
Cambridge, Massachusetts



HyperVoice
A Phone-Based CSCW Platform

Paul Resnick

CCS TR # 133, Sloan School WP # 3463-92

August, 1992

Acknowledgments

This research was supported by Digital Equipment Corporation, the National Science Foundation (Grant No. IRI-8903034), Matshushita Electric Industrial Co., Boeing, Information Resources, Inc., Electronic Data Systems, Apple Computer Company, and the corporate members of the MIT International Financial Services Research Center.

HyperVoice
A Phone-Based CSCW Platform

Paul Resnick

MIT Center for Coordination Science
1 Amherst Street, Room E40-181
Cambridge, MA 02139
(617) 253-8694 presnick@mit.edu

ABSTRACT

A major shift is underway in how we think about telephones. For decades, they were used solely for one-to-one, synchronous communication. The increasing use of answering machines and voice messaging, however, is shifting the public perception of telephones, thus opening a space for more innovative applications. Five years from now, some of the most interesting and popular cooperative work applications will probably use telephones as the primary means of access. This paper presents evidence that there are practical phone-based cooperative work applications and describes a set of software tools that facilitate the development of such applications.

INTRODUCTION

Telephones are the most ubiquitous, best-networked, and simplest computer terminals available today. This makes telephones an attractive platform for cooperative work applications such as event calendars, issue discussions, task tracking, and question and answer gathering applications, especially those in which local-area network connection of all users cannot be assumed. The limitation of telephones is that they provide only sound for output and only sound and twelve buttons for input. Many skeptics who are familiar with existing telephone interfaces believe that this limitation is so strong as to eliminate the possibility of creating practical phone-based cooperative work applications. A new telephone interface style called Skip and Scan [15], however, opens up the possibility of more complex telephone-based applications, including cooperative work applications.

Field trials of several applications, totaling more than 7000 calls, demonstrate that it is possible to build usable and useful phone-based cooperative work applications. The field trials also highlight some of the factors that will influence the success or failure of applications. These factors include the value of the expressiveness of voice, the need for anonymity, the need to remember large chunks of information, and the distribution of costs and benefits among users.

HyperVoice is an application generator for phone-based cooperative work applications. Several features distinguish HyperVoice from other software tools for building telephone applications:

1) The specification language primitives are at a high level of abstraction. An interpreter automatically determines the details of dialogue sequencing and the text of prompts.

2) Messages have internal structure. Using telephone forms, callers can add new information objects (messages) that consist of several fields. The fields can contain recorded voice, typed-in dates, phone numbers, and quantities, or even links to other information objects. Sorting and filtering operations can act on the non-voice fields.

3) Presentation formats are separate from information objects. Multiple presentation formats can present the same information objects in different ways, depending on the context. In addition, callers can add new information objects without specifying details of how to present the information, since existing presentation formats can be reused.

SAMPLE APPLICATION: EVENT CALENDAR

Consider the following scenario for a phone call to an event calendar application. The computer answers the phone and prompts the caller to select one of six event categories. The caller selects the lectures and seminars category. The system tells the caller that there are twelve announcements in this category. The announcements are arranged in the order of the event

dates and the caller then uses two buttons, 9 and 7, to move forward and back through the announcements. Each announcement begins with a headline, so that the caller can quickly decide whether to listen to the rest of the announcement. If the caller chooses to keep listening beyond the headline, several other "fields" are played back, including the date, the time, the location, a contact phone number, and details. The caller can also press #, the "smart" fast-forward button. Unlike an ordinary fast-forward button, which advances a fixed amount of time, the "smart" fast-forward skips to the next logical segment in the recording, in this case the next field of the announcement.

After listening to some of the announcements, the caller presses 3, which initiates the entry of a new announcement. The caller then fills out a "telephone form." Each field of the event announcement becomes one entry blank in the form. The caller can press 9 and 7 to skip back and forth between the entry blanks. In some entry blanks, the caller speaks information. In other entry blanks, such as the one for the date field, the caller presses buttons to enter data, such as 082192 for August 21, 1992. For typed in data, the system runs validity checks, ensuring, for example, that the caller enters a date in the next 90 days. The caller can review and replace the contents of any of the entry blanks before deciding to save the form. Once saved, the new announcement is added to the lectures and seminars category.

Now consider the scenario of a moderator calling up to make sure that no one has added an inappropriate announcement. Instead of selecting a category, the moderator enters a special code and the system goes to a list of all the event announcements. In this case, however, the announcements are sorted in descending order by the date on which they were entered, so that the newest announcements are at the beginning of the list. When listening to an announcement, the moderator hears all the same fields that regular callers do, plus the 'date added' and 'category' fields. The system automatically had added those fields to each announcement that was posted to the system. If the moderator finds an inappropriate announcement, entry of a special code removes that announcement from the system.

This scenario illustrates a number of important features of applications developed using HyperVoice. 1) The information in the system can come from many sources, since anyone with a touch-tone phone can add a new announcement. 2) The event announcements consist of several pieces. While recording, that allows the system to remind the callers of important information to record, and allows callers to re-record smaller pieces when they make mistakes in recording. During playback, it permits callers to use a smart fast-forward button to skip between fields. 3) Some of the fields of event announcements are symbolic rather than recorded, including the date, date added, and category fields. This allows the system to provide validity checks during data entry and to sort announcements during playback. 4) There are multiple presentation formats for the same information objects. For example, a moderator hears the announcements sorted in a different order, and with additional fields played back.

HYPERVOICE

HyperVoice is an application generator for telephone bulletin-board applications. To create an application, a programmer specifies a login procedure, a collection of linked information objects, how to filter and sort the information, and how callers can add new information objects. A pre-processor automatically generates the text of prompts that need to be recorded. At run-time, an interpreter generates state-machines that determine the details of dialogue sequencing.

Existing telephone application toolkits require programmers to specify state-machine representations directly [5, 13]. These tools provide programming environments analogous to HyperCard. By contrast, HyperVoice provides a specification language that abstracts away from many of the details of dialogue sequencing and navigation prompts. It is analogous to recent research on generating screen-based interfaces from higher-level abstractions [6, 17, 18].

This section briefly describes the HyperVoice application generator and presents part of the specification of the event calendar application. For more details, see [14].

The Language

A programmer begins by specifying object types. In the event calendar application, for example, there is an event announcement object type that has eight fields: headline, date, time, location, contact number, details, date added, and category.

Lists are ordered sequences of objects. The same object can appear in more than one list. In the event calendar, for example, there is one list of events for each category, plus a master list that contains all the announcements.

Login

Some applications require login procedures to restrict access and to determine the initial list of information objects to present to different callers. The HyperVoice Login primitive includes two parameters for whether callers need to enter ids and passwords to access the system. If no registration is required, two other parameters specify the initial privileges callers should get and the initial list to present. If registration is required, a parameter specifies a list of User objects. The id and password that a caller enters pick out a User object, which contains fields that specify the initial privileges and list to present.

Presenting a List

A List Format specifies how to present a list of objects. The List Format has a number of parameters, as summarized in Table 1. The filter selects a subset of the list to present. The Sort Formats determine the order in which to present the selected objects. The Item Formats specify which fields of the objects to play back, in what order, and whether to play the field names before the contents of the fields. One Item Format specifier is included for each of the object types that can appear in the list.

A menu is a special case of presenting a list of objects, as determined by the two parameters, 'how to advance' and 'how to select'. Callers can advance to the next item either by waiting until the end of the current item (WAIT) or by pressing a button (SKIP), or both. With numeric selection, callers press **1** to select the first object, **2** to select the second and so on. With positional selection, a single button selects the current object. For a discussion of the relative merits of the menu styles that these two parameters can generate, see [14].

The 'Name for Objects' parameter is used in generating the text of prompts that will tell the caller how to navigate through the list. For example, if the parameter has the word "announcement" as its value, a number of prompts will be generated, including, "For the next announcement, press 9," and, "For the previous announcement, press 7."

The 'Response List?' parameter determines what recordings will be played back to introduce the list. Each list that is not a response list has its own recording to introduce the contents of the list. Response lists, however, are generated automatically when information objects are added, so that the description of those lists needs to be generated automatically as well. HyperVoice plays the phrase "Responses to:" and then the contents of the specified field of the object that the list contains responses to.

Adding New Information by Phone

A List Action specifies how a caller can add new objects. It determines the privileges required to add a new object and the locations in the list from which the addition can be initiated.

The List Action also includes an Extension Format specification (Table 2). The Extension Format determines what kind of object to add, which lists to add it to, and where in those lists to add it. Finally, an Extension Format determines initial values for the fields, which fields will appear in the form, maximum recording lengths, and validity checks on typed in data, such as accepting dates only in the next 30 days.

Parameter Name	Alternative Values	Description
Filter		Selects a subset of the list to present.
Sort Formats		The order in which to present the items.
Item Formats		How to play back each item in the list.
How to Advance	SKIP, WAIT, BOTH	Whether the caller presses a button to advance to the next item, or just waits.
How to Select	NONE, NUMERIC, POSITIONAL	The selection mechanism for menus.
List Action		Action that callers can take to add a new information object.
Name For Objects		Used in generating prompts.
Response List?	NO, Field Name	Is this a list of responses to some other object? If so, the field of the other object to play in the list header.

Table 1: The parameters of a List Format specification.

Parameter Name	Alternative Values	Description
Add to Current List?	YES, NO	Whether the new object will be added to the list currently being presented to the caller.
Other Lists to Add To		
Location to Add	END, BEGINNING	Add the new object at the end or the beginning of the list(s).
Type of Object to Add		
Edit Format		A collection of Field Edit Formats that specify which fields will appear in form, initial values, validity checks, etc.

Table 2: The parameters of an Extension Format specification.

Example: The Event List Format

Now consider how to specify the presentation of a category of even announcements, as shown in Table 3. It specifies that the announcements be sorted in ascending order by the date of the announcement. The first six fields of each event announcement will be played back, and all except the headline will have the field name played back before the contents.

There is a List Action (details not shown) that specifies how a caller can add a new announcement. It specifies that the new announcement will be added both to the current list and to the master list that the moderator uses, as described in the scenario above.

A different List Format (not shown) specifies the presentation of the master list of announcements to the moderator. It specifies that the announcements be sorted in descending order by the 'date added' field, so that the moderator can hear the most recent announcements first. The Item Format in the List Format includes two additional fields, 'category' and 'date added'. Finally, the List Format does not include a List Action, since the moderator will not be adding announcements directly to the master list.

Implementation

Programmers specify the primitives described above by filling in screen-based forms in the OVAL system [8], which runs on a Macintosh. These primitives are then written to a database file and transferred to an IBM-compatible PC. A C language program on the PC reads the application specification from the database file and updates the database file whenever callers add new information objects. A commercial add-in card, Watson, handles speech digitization and touch-tone detection.

List Format: Event List Format	
Field Name	Value
Filter	[0 - 90 days]
Sort Order	['Date' ASCENDING]
Item Formats	['Headline' NO 'Date' YES 'Time' YES 'Location' YES 'Contact Number YES 'Details' YES]
How to Advance	SKIP
How to Select	NONE
List Actions	[New Event Announcement]
Name for Objects	Announcement
Response List?	NO

Table 3: The specification of how to present a list of announcements.

SUCCESS FACTORS FOR APPLICATIONS

Previous research on CSCW applications and analysis of the differences between text and voice suggest a number of context variables that will influence the value of HyperVoice applications. Below I discuss both positive and negative factors.

Green Flags

Time-critical information Once entered over the phone, information is immediately available to other callers. For applications with time-critical information, HyperVoice will be more useful than communication systems such as mass mailings that have longer delays to publication.

Need for access from home or while traveling One of the great advantages of the telephone is its widespread accessibility. Applications that benefit from entry or retrieval of information from remote sites will be more likely to succeed, since competing technologies cannot match its accessibility.

Need for expressiveness of voice Voice is more expressive than text; through tone, pitch, and speed, a speaker can convey much information not conveyed by a text transcription. Studies of teleconferencing indicate that voice is the single most important channel to include for collaborative tasks [12] and that voice is a better annotation medium for the more complex, controversial, and social aspects of collaborative tasks [4].

Users have weak composition, keyboarding, or reading skills Unlike text-based systems, phone-based systems do not require these skills.

Opportunity to create a 'honeymoon period' Most communication systems have increasing returns to adoption: the utility of the system to each user increases as the number of other users increases [2, 10]. To achieve critical mass, these systems need an initial honeymoon period in which users adopt based on expectations of future value, when others have adopted, rather than on the current utility of the system. Any opportunity to create a honeymoon period, through sponsorship by a powerful person or through a big splash introduction of the system, will improve its chances of success.

Red Flags

Well-entrenched communication patterns Changes from existing patterns are disruptive and people may not perceive the opportunity for better communication patterns.

Poor distribution of costs and benefits The distribution of incentives is a well-known problem for CSCW systems [7]. While the benefits to the group as a whole may outweigh the costs, the benefits may accrue to

some individuals and the costs to others, who may then refuse to participate.

Need for anonymity A person's voice is more easily identified than a person's writing style. Research indicates that the anonymity of bulletin boards can improve participation from shy people [3] and that anonymous suggestions can enhance brainstorming sessions [11]. Using digital signal processing algorithms, it is possible to disguise voices, but only at the cost of losing the expressiveness and some of the intelligibility.

Need to scan large information chunks If there is no way to break information chunks into small pieces, then it will take longer to listen to a message than for a good reader to read or scan a written version of it. If, on the other hand, the information divides naturally into small, meaningful segments, then a consistent set of telephone buttons may allow callers to accomplish the fast changes of attention that eye gaze shifts accomplish in visual scanning. This is the idea behind the Skip and Scan interface style [15].

Need to remember large information chunks The presentation of information by phone is ephemeral. If callers need to take information with them, they will have to engage in the tedious process of transcription.

FIELD TRIALS

HyperVoice has generated working prototypes of many different applications, including a joke collector, several versions of an issue discussion application, event calendars, a question and answer line for teachers, and a task tracking application. These prototype applications demonstrate the utility of the HyperVoice primitives as a specification language for a diverse set of applications. The design process for several of the prototypes also demonstrated the utility of HyperVoice's high-level primitives for discussion of alternative designs with non-programmers.

Table 4 summarizes field trials of several of the prototypes. The field trials demonstrate the possibility of usable and useful telephone-based cooperative work applications. Observations from both the successful and unsuccessful field trials are consistent with the success factors described above.

Issue Discussions

In an issue discussion application, callers can not only listen to what others have recorded, but also record responses. Similar to selecting a category of events, a caller to an opinion forum navigates through one or more menus to select a topic. A topic is a list of comments, each having headline and contents fields. Each time a caller adds a comment, an empty list of responses is automatically created. Then, a future caller who listens to a recorded comment presses a button to go to the responses to that comment.

The first issue discussion was used as an adjunct to a class discussion. Despite some technical and user interface difficulties, a majority of the class called and a number of them recorded comments, including responses to others' comments. The second issue discussion, on the topic of intellectual property rights on software user interfaces, was demonstrated at the Interactive Experience at CHI '90. Experts on the topic recorded comments in advance. Most conference-goers, however, were unwilling to publicly record an opinion on such a controversial topic since their voices might be recognized.

Application Name	Message Types	Participatory Design?	# of calls	# of messages added	duration of trial
Issue Discussion 1: class discussion	Comment	No	~20	~10	4 days
Issue Discussion 2: Intellectual Property	Comment	No	~150	3	3 days
Issue Discussion 3: U-TALK	Comment	No	1030+	152+	2 months +
Mandela Task Tracking	Status Report	Yes	1	0	7 days
Mandela Event calendar and Volunteer signup	Message (unstructured)	Yes	1378	more than 200	10 days
Peace and Justice Events Hotline	Event Announcement	No	4578 +	more than 300	1 year +
Curriculum Questions and Answers	Lesson Plan, Question, Response, Success Story, Meeting Announcement, Comment	Yes	72 by head teacher; 66 by others	57 by head teacher; 10 by others	6 weeks

Table 4: Summary of field trials. Those marked + are ongoing.

The final issue discussion, U-TALK, (pronounced "You talk") was open to the entire MIT community. The letters spell out the internal MIT phone number for the system. I initially hoped for serious discussion of issues such as academic honesty. While some people recorded serious comments, and even one poem, others took advantage of the expressiveness of voice to record music and other entertaining sounds. For example, in response to a question about their worst experience at MIT, two people shouted in unison, "Everything!" To preserve anonymity, some people tried to disguise their voices.

Task Tracking

The people coordinating Nelson Mandela's visit to Boston in June 1990 had the opportunity to use a HyperVoice application to help track the status of plans for events. One or two people recruited from local corporations and non-profit organizations were responsible for each of the approximately ten events. In addition, there were several overall coordinators, including a publicity person and an overall operations coordinator. Because they were an ad hoc group, the only technologies that they all had in common were telephones and fax machines.

Interestingly, the HyperVoice specification language provided a convenient language for discussing alternative designs with the head of the operations committee. We considered several designs, varying the fields of a status report, who would be allowed to add new status reports, and who would be allowed to listen to them. We decided on one list of status reports for each event, with open access, both reading and writing, to all members of the operations committee.

The committee members, however, did not use the phone application. Follow-up telephone calls to them indicate several reasons. First, the system was ready too late in the process. Communication and coordination structures, however inadequate, had already been established. Second, the operations coordinator introduced the system briefly in the middle of a lengthy meeting. Several people could not remember the system being introduced at all. Thus, the introduction process did not create enough excitement to spawn a honeymoon period. Third, there was an incentive distribution problem. The work required to post status reports would have fallen on the event coordinators, while most of them perceived that the benefits would all accrue to the overall operations coordinator. Finally, some of the event coordinators wanted to retain as much control as possible over their events. As a result, they did not want to open their plans to others' scrutiny.

Event Calendars

A more successful HyperVoice application publicized the schedule of events for Nelson Mandela's visit to Boston. In this case, the general public was not permitted to add new event announcements. The event organizers, however, added and removed announcements by phone, and this proved to be quite useful for handling last minute changes to the schedule. Printed flyers and newspaper and radio spots advertised the phone number. Besides listening to the schedule of events, callers could also listen to recorded requests for volunteers and leave (unstructured) messages to sign up.

A second event calendar, called the peace and justice event hotline, allowed the general public to post announcements using telephone forms. While a few people had trouble with the structured input (they recorded their entire announcements in each entry blank of the telephone form), most had no trouble.

Leaflets and announcements at political events around Boston publicized the phone number. During the Gulf War, the emcee at a city-wide rally described it as the best source of up-to-date information about events. While transmission of information about event announcements does not require the expressiveness of voice, a number of people remarked that they liked hearing many different voices on the system. More than a year later, the system is still in active use.

Curriculum Questions and Answers

A group of 38 elementary school math teachers used a HyperVoice application to communicate about a new math curriculum that they were using. The HyperVoice specification language again turned out to be useful in discussing alternative designs with a non-programmer, the head teacher who developed the curriculum. The key decisions again were the structure of information objects and who would be able to listen to and add them. We decided to have publicly accessible lists of success stories, meeting announcements, and general comments, and a list of lesson plans that only the head teacher could add to.

Each lesson plan had an attached list of questions that any teacher could add to, and each question had an attached list of responses that they could add to. The motivation for this was that the head teacher routinely handled the same question from more than one teacher, causing her to be on the phone for several hours each night. By making the questions and answers public, she hoped not to handle the same question repeatedly. Thus, this application embodies some of the ideas in the Answer Garden system [1].

The question and answer system made use of the ability to provide multiple presentation formats for information objects. The head teacher promised to call in each night and respond to any questions that other teachers had recorded. There were as many as forty different lesson plans on the system at any one time. Without a special method of access, she would have had to check all forty question lists for new questions.

Instead, each new question was also added to a master list that she checked. In addition, each new question had a field that contained a pointer to its associated lesson plan. That way, the head teacher could listen to the new question and hear what the associated lesson plan was.

The actual usage differed quite a bit from the planned usage. Almost half the teachers never called. Those who did call listened to the lesson plans that the head teacher posted and listened to the meeting announcements. A few of them recorded success stories or congratulations to other teachers on wedding plans, but only two recorded questions. This application, then, gave the head teacher the additional task of recording lesson plans, without reducing the number of repeat questions that she handled. After six weeks, she stopped recording new lesson plans.

A number of plausible explanations can be made of the usage patterns. First, the system was introduced six weeks into the school year, after communication patterns were already set. Second, while the 38 teachers were distributed among many schools, there were usually two or three in each school, and the whole group met for one day a month, so that many teachers may not have felt the need for greater communication. Third, teachers generally do not get much feedback from peers, so that asking publicly about teaching puts them in an unfamiliar and vulnerable position. Thus, the lack of anonymity of recorded voice appeared to be an important factor. Fourth, at approximately ten minutes each, the recorded lesson plans were too long. To absorb all the detail in a recorded lesson plan, teachers would have had to take extensive written notes. Despite that, some teachers did use lesson plans from the phone system in their classrooms.

RELATED WORK

HyperVoice builds on previous research in two areas, telephone-based interfaces and screen-based information-sharing tools. Overall, this project generalizes the notion of semi-structured information objects and presentation formats from text-based systems to a new medium, and develops a telephone interface style that is suited to the entry and presentation of semi-structured objects.

One previous project investigated the collection of semi-structured information objects over the telephone [16]. The PhoneSlave conducted conversations with callers to elicit structured answering machine messages. The system asked each caller a series of questions ("Who's calling please?"; "What is this in reference to?"; "At what number can he reach you?"; etc.) After playing a question, it recorded whatever the caller said, until it detected a long pause, then went on to the next question. The system automatically filled in the date and time of the call. The structured messages were retrieved using graphical tools on a computer screen. HyperVoice generalizes this dialogue to include entry of non-voice data such as dates and even links to other objects. HyperVoice also generates interfaces that take advantage of message structure in presenting information objects for playback by telephone.

A number of asynchronous text-based systems help coordinate the activities of a set of people. Applications have included conversation management [19], task tracking and scheduling.

Malone et al [9] showed that most such applications can be constructed from the following features: 1) sharing of semi-structured information objects; 2) filters and, more generally, agents, that automatically select some objects for presentation to particular users at particular times; 3) views that specify how to present visually single objects or collections of objects in ways that are helpful to particular users at particular times.

HyperVoice generalizes the notion of semi-structured objects in these systems to include voice as a data type. It also generalizes the concept of views, a visual notion, to that of presentation formats, and provides presentation formats that are particularly suited to telephones.

FUTURE RESEARCH

Future plans call for the development and evaluation of additional applications and the integration of two other widely accessible technologies, fax and email. The planned applications include an aid to scheduling meetings and task allocation applications (e.g., a distributed sign-up sheet). Some of these will require extensions to the HyperVoice specification language.

To integrate fax and email with the telephone, I am developing an architecture for sharing semi-structured information objects that will allow users to participate with any combination of these technologies, without requiring any individual user to have all three. For example, a user could enter a semi-structured object using a telephone form, but could send in a fax to be the contents of one of the fields, and an email message to be the contents of another. An extension of this idea would be to make the telephone an alternative mode of access to Lotus Notes or Oval [8] databases, for both entry and retrieval.

CONCLUSION

The main message of this paper is that it is possible and worthwhile to use the telephone as a platform for input and retrieval of semi-structured information objects. HyperVoice provides a language that is useful both in considering alternative designs and in specifying applications. The field trials of applications demonstrate that even some prototype-quality systems were usable and useful. The telephone is a worthy platform for further research on CSCW applications and for the development of commercial products.

ACKNOWLEDGMENTS

Tom Malone provided invaluable guidance and assistance throughout this project. Charlie Welch maintains the Peace and Justice event hotline. Wendy Lee did the fundraising and publicity for U-TALK. Victoria Bill, Leslie Clark, Mary Leer, and Lauren Resnick made the teachers' question and answer application possible and Jolene Galegher and Mark Ackerman helped analyze it. Kevin Crowston, Rob Fichman, Chris Kemerer, Jintae Lee, Wanda Orlikowski, Mike Plusch, Chris Schmandt, Bob Virzi, and Joanne Yates also contributed the content of this paper.

REFERENCES

1. Ackerman, M. S. and Malone, T. W. Answer Garden: A Tool for Growing Organizational Memory. In *Conference on Office Information Systems.* (Cambridge, MA, 1990). ACM, pp. 31-39.

2. Arthur, W. B. Competing Technologies: An Overview. In *Technical Change and Economic Theory,* Volume . G. Dosi, Ed. Columbia University Press, New York, NY, 1987.

3. Chaiklin, S. and Schrum, L. Community-Based Telecommunications. In *Third Guelph Symposium on Computer Mediated Communication.* (Guelph, Ontario, Candada, 1990).

4. Chalfonte, B. L., Fish, R. S. and Kraut, R. E. Expressive Richness: A Comparison of Speech and Text as Media for Revision. In *CHI '91: Conference on Human Factors in Computing Systems.* (Seattle, WA, 1991). ACM Press, pp. 21-26.

5. Magnum Software Corporation. TFLX Quick Manual: A Simple Overview of Magnum TFLX and its Picture Programming System. 21115 Devonshire Street, Suite 337, Chatsworth, CA 91311, 1990.

6. de Baar, D. J. M. J., Foley, J. D. and Mullet, K. E. Coupling Application Design and User Interface Design. In *CHI '92 Conference on Human Factors in Computing Systems.* (Monterey, CA, 1992). ACM, pp. 259-266.

7. Grudin, J. Why Groupware Applications Fail: Problems in Design and Evaluation. *Office: Technology and People.* 4, 3 (1989), pp. 245-264.

8. Lai, K.-Y., Malone, T. and Yu, K.-C. Object Lens: A "Spreadsheet" for Cooperative Work. *ACM Transactions on Office Information Systems.* 6, 4 (1988), pp. 332-353.

9. Malone, T. W., Grant, K. R., Lai, K.-Y., Rao, R. and Rosenblitt, D. Semi-structured Messages are Surprisingly Useful for Computer-Supported Coordination. In *Computer-Supported Cooperative Work: A Book of Readings.* I. Greif, Ed. Morgan Kaufmann, San Mateo, CA, 1988, pp. 311-331.

10. Markus, M. L. Towards a 'Critical Mass' Theory of Interactive Media: Universal Access, Interdependence and Diffusion. In *Perspectives on Organizations and New Information Technology,.* J. Fulk and C. W. Steinfeld, Ed. Sage Publications, Newbury Park, CA, 1990, pp. 194-218.

11. Nunamaker, J. F., Dennis, A. R., Valacich, J. S., Vogel, D. R. and George, J. F. Electronic Meeting Systems to Support Group Work. *Communications of the ACM.* 34, 7 (1991), pp. 40-61.

12. Ochsman, R. B. and Chapanis, A. The Effects of 10 Communication Modes on the Behavior of Teams During Co-operative Problem-solving. *International Journal of Man-Machine Studies.* 6, (1974), pp. 579-619.

13. Repenning, A. and Sumner, T. Using Agentsheets to Create a Voice Dialog Design Environment. In *Symposium on Applied Computing.* (Kansas City, MO, 1992). ACM Press, pp. 1199-1207.

14. Resnick, P. HyperVoice: Groupware by Telephone. Ph.D. thesis, Department of Electrical Engineering and Computer Science MIT, 1992.

15. Resnick, P. and Virzi, R. A. Skip and Scan: Cleaning Up Telephone Interfaces. In *CHI '92: Conference on Human Factors in Computing Systems.* (Monterey, CA, 1992). ACM, pp. 419-426.

16. Schmandt, C. and Arons, B. A Conversational Telephone Messaging System. *IEEE Transactions on Consumer Electronics.* CE-30, (1984).

17. Szekely, P. Template-based Mapping of Application Data to Interactive Displays. In *ACM Symposium on User Interface Software and Technology.* (Snowbird, Utah, 1990). ACM, pp. 1-9.

18. Wiecha, C., Bennett, W., Boies, S. and Gould, J. Tools for Generating Consistent User Interfaces. In *Coordinating User Interfaces for Consistency.* J. Nielsen, Ed. Academic Press, San Diego, CA, 1989, pp. 107-130.

19. Winograd, T. A Language/Action Perspective on the Design of Cooperative Work. In *Computer Supported Cooperative Work: A Book of Readings.* I. Greif, Ed. Morgan Kaufmann, San Mateo, CA, 1988, pp. 623-653.