DOCUMENT RESUME

ED 231 635 SE 042 071

TITLE	Math Network Curriculum Project. Project Summary; Final Report.
INSTITUTION	San Francisco State Univ., Calif.
SPONS AGENCY	National Inst. of Education (ED), Washington, DC.; National Science Foundation, Washington, D.C.
PUB DATE	Jun 83
GRANT	SED-80-12465
NOTE	30p.; For related documents, see SE 042 072-079.
PUB TYPE	Reports - Descriptive (141)
EDRS PRICE	MF01/PC02 Plus Postage.
DESCRIPTORS	*Computer Oriented Programs; Inservice Teacher Education; Mathematical Concepts; *Mathematics Curriculum; *Mathematics Education; *Mathematics Instruction; *Microcomputers; *Middle Schools; Problem Solving; Reports; Teacher Education; Units of Study
IDENTIFIERS	Math Network Curriculum Project

ABSTRACT

This document summarizes the work of the Math Curriculum Project at San Francisco State University. The project developed seven curriculum units for the middle school mathematics program, using microcomputers as a problem solving tool to foster mathematical thinking and develop insights into mathematical concepts. They also created a prototype telephone network that is both a message system and a curricular data base for activities in each unit. Finally, they developed a teacher training model from their experiences in piloting the materials. The report describes the objectives, methods and procedures, outcomes, and dissemination activities of the project. An overview of the units, a network manager manual, a message system user manual, and a list of talks about the project are appended. (MNS)

MATH NETWORK CURRICULUM PROJECT

PROJECT SUMMARY

SED-8012465

This project has developed seven curriculum units for the middle school mathematics program that use microcomputers as a problem-solving tool to foster mathematical thinking and insights into mathematical concepts. We have also created a prototype telephone network that is both a message system and a curricular data base for activities in each unit. Finally, we have developed a teacher training model from our experience with the materials and the teachers who piloted the materials.

Each unit, designed to be a two-week module, is in fact a teacher's guide which includes detailed directions along with the courseware and software needed. Teacher intervention in the non-computer activities that begin each unit is required, and the consistent use of small group instruction makes the units usable in the standard classroom if two microcomputers are present. The network applications require additional equipment but are not essential.

The content includes arithmetic/pre-algebra, geometry, and statistics. Problem solving is integral to all the activities.

A trainers' guide to provide inservice on these materials is provided as an eighth unit and can be effectively used by individuals who are both familiar with microcomputers and have a good mathematics background!

The final report is available through both the Educational Resources Information Center (ERIC) and the National Technical Information Service (NTIS). The project material is also available through the same sources.

June 1983

MATH NETWORK CURRICULUM PROJECT (MNCP)

FINAL REPORT

This project was supported by NSF Award No. SED 8012465 in the Joint National Science Foundation/National Institute of Education program on Mathematics Education Using Information Technology. The work was done at San Francisco State University by three members of the Mathematics Department: Mr. William Finzer, Dr. Jose Gutierrez, and Dr. Diane Resek.

Description of the Objectives

The overall thrust of this project was to encourage the use of microcomputers as problem solving tools in middle school mathematics classrooms. To do this we focused on three specific objectives: Curriculum Development, Teacher Education, and Computer Networking. There was a perceived need for curriculum materials that used the computer within the framework of the regular classroom.

We developed a series of seven teacher's guides (UNITS) to provide support and direction for incorporating the computer in three areas: arithmetic/pre-algebra, statistics, and geometry.. (Appendix A gives an overview of the units.) Each unit is intended to take about two weeks of class time and includes appropriate courseware and software.

A premise for the MNCP curriculum units was that a knowledgeable teacher was essential to direct and support students toward higher levels of abstract thinking and problem solving. As a result of our experience with the teachers in this project we have created an Inservice Guide that structures an inservice program for each of the units. The material in the Inservice Guide is directed at a Master Teacher who, besides having a working knowledge of the mathematics and pedagogy of the units, may be an experienced classroom teacher, a curriculum specialist, a school district or county administrator or a college instructor. The need for some inservice is essential not only because of the computer involvement and the nature of the mathematics, but also because the units incorporate an active approach to the learning that uses manipulative materials and small group instruction.

Computer networking, in this case, is a system which connects computers in the project classrooms with each other via telephone lines. A data base from activities in the UNITS was established by the project classes, and a prototype message system for interaction between and among groups of students, teachers, the project managers, and individuals was developed. The overall networking system is described in Appendix B; the message system is described in Appendix C. It was our intention that the message system serve as a vehicle to motivate student involvement through a shared curriculum, and that it also would provide a communication system to support teachers in their work with the curriculum.

Methods & Procedures

The UNITS

The UNITS were developed around topics appropriate to the middle school mathematics curriculum. The middle school, grades 6 through 8, can provide time within the usual mathematics curriculum that is not normally available in the high school curriculum. Furthermore, the developmental level of middle school students was appropriate for the creation and testing of hypotheses, which are a significant aspect of each of the UNITS.

Imbedded in each unit are elements of active learning, students' use of manipulative materials, and small group activities. Materials and activities were sought out and/or created that provided a concrete representation of the concepts to be presented in the computer applications to follow. Each UNIT was carefully structured around small group instruction to allow the entire class to complete the computer activities within the two week time frame with only two computers in a regular classroom.

Ideas for the UNITS were developed and formalized early in 1981 and work was started on some of them. For the Summer, Joan Akers joined the team as a curriculum developer and the remaining UNITS were designed and written. Concurrently, three programmers were working on the software for specific UNITS. By Fall the UNITS were ready for initial field testing.

The UNITS were tested by eleven classroom teachers who were selected for this project. The co-principal investigators conducted a series of site visits to observe the UNITS in use. As a result of these visits and from very specific suggestions from the teachers, changes and revisions were incorporated. Teachers were asked to test at least two of the UNITS. Nine of the eleven teachers used the materials with at least two of their classes. More than six hundred students were involved in the field testing stage of this project.

Teacher Education

Local school districts were contacted and teachers from interested districts were invited to participate. In the Spring of 1981, eleven teachers were chosen for an inservice program that started with a special section of "Computers in the Elementary Classroom", a course regularly offered at San Francisco State University. Through this course the teachers were introduced to the use of small group instruction and the use of manipulative materials as a part of their own learning. They learned to program in BASIC. During this time the teachers were given two microcomputers to use for their own learning and to familiarize themselves with its use in their own classroom.

The Fall of 1981 brought the field testing phase and the second element of the inservice program. The teachers now were to implement the UNITS and to support them in this task, eight three hour inservice sessions were prepared to specifically deal with the UNITS and the teaching strategies underlying their use. The methods used during these sessions were in keeping with the methods of the UNITS themselves. The use of manipulative materials and small group instruction was consistently applied where appropriate as a model for their own teaching.

The resulting product of these inservice sessions with project teachers is an Inservice Guide to be used by a master teacher to provide inservice to other classroom teachers.

The Network

In the Spring of 1981, the development of the NETWORK prototype from scratch was a considerable undertaking. Almost all the software in the package had to be developed locally for the hardware available at the time. A description of the hardware, the aplications, the protocols, and the software involved in the NETWORK is provided in the Network Manager Manual (Appendix B). Examples of the curricular use of the NETWORK are also given.

A separate but significant use of the NETWORK is the message system. The DANDELION MESSAGE SYSTEM USERS Manual. (Appendix C) provides a complete description of its operation.

The NETWORK provided a useful link between participants. The ability to contact a subgroup of teachers with information about a specific UNIT made the implementation and revision of the UNITS proceed smoothly. Student's too were able to interact with each other and in some cases directly with the project managers. At least ten student groups were identified by name.

Outcomes

The project had some necessary limitations that anyone in the microcomputer field will recognize. The material is limited to the machines on which the work was done; in this case we used 16K Gommodore PET computers with 4.0 BASIC. The Network development was done using a 10 megabyte CORVUS hard disk connected to an Apple II Plus. This development did not go beyond the research stage and is not available for dissemination. Nevertheless, significant products were created.

UNITS

Each UNIT is a sequence of classroom and computer activities designed to span a two week period in a regular mathematics classroom. Each UNIT contains the following:

Overview - intent of the UNIT, comments about content, activities and pedagogy.

How to use this guide - description of use, time needed, and special considerations.

Summary of activities and materials - a summary for easy reference.

Appendix - worksheets and other instructional materials.

Each of the activities within a UNIT provides the teacher with the following information:

Format - student groupings for the activity
Time requirements - number of class periods (or minutes) needed
Materials needed - list of materials
Background - discussion of the activity, its intent, and its place in the sequence of activities
Purpose - specific objectives for the activity
Teacher preparation - what to do ahead of time
Activity - step by step directions and suggestions.

The activities are sequenced within each UNIT and student worksheets are provided for each activity. Three of the UNITS require previous experience from another UNIT. With these three exceptions, each UNIT can stand alone as a separate activity. (See Appendix A for these limitations.) The activities are designed to be open ended and can be used by the same students for much longer periods than the intended two weeks. Because of the open ended and exploratory nature of the curriculum, no answers exist. The software, which is not designed to be stand alone software, is self-checking and enables students to set their own objectives and goals with careful guidance from a knowledgable teacher. The required non-computer activities that precede the simulation type activities are necessary for concept development and appropriate computer use.

Each UNIT has its own software. The number of programs varies from one to four per UNIT. There are fifteen programs in all.

Inservice Guide

The Inservice Guide has a similar structure to the UNITS in that it provides a Master Teacher with time, material, and pedagogical requirements to conduct a two hour inservice for each of the seven UNITS.

The Guide presents information on the need for inservice, a description of a Master Teacher, a philosophy which motivated the creation of this type of curriculum and a discussion on the nature and scheduling of the inservice. The one hundred and thirty-five pages also include samples of worksheets from the UNITS to be used by the Master Teacher.

Network

The NETWORK was intended as a communication system and as access to a data base for the curriculum from each of the UNITS. This purpose was served during the course of this project but while it is operational, it is not appropriate for dissemination. Much information is provided in Appendix B on the NETWORK and in Appendix C on the message system, but the nature of the operation of the entire system requires a high degree of training and experience. Should the reader require additional information or would like to consider further development work, he/she should contact the principal investigator directly.

Dissemination Activities

Seventeen talks and papers on the activities of this project were presented to various professional groups and educational institutions. These are listed in Appendix D. Twenty-seven educators and/or educational institutions requested and were sent portions of the curriculum. In addition, a mailing list was created from forty-eight individuals and/or organizations who requested information. These will be sent a summary and information on how to obtain copies of this report.

This report, the entire text of the UNITS and listings of the computer programs that accompany them are being sent to the Educational Resource Information Center (ERIC) and the National Technical Information Service (NTIS) to make the results of the project available to interested individuals and groups.

A further aspect of dissemination more or less attributable to this project is the further activities of the teachers involved. Each has become a resource person within their own school and of the nine, six have conducted inservice classes, three have taught college classes and six have been instrumental in the acquisition of additional computer equipment within their own school districts. This vital resource of a knowledgable teacher will serve the use of information technology in education far beyond the life of all other products.

Math Network Curriculum Project

San Francisco State University

Overview of Units

May 19, 1982

Below is a list of the seven units developed by the Math Network Curriculum Project starting in January, 1981. Following the list is a brief description of individual units. Each unit is designed to take about two weeks of classroom work. Most of the units have been tested in middle school classrooms in the San Francisco Bay Area. The project has also written a 'Guide for Master Teachers' to be used to aid in planning and conducting in-service workshops for teachers in using the curriculum.

1. **Input-Output**
2. **Strategies**
3. **Business**
4. **Sampling**
5. **Data Analysis**
6. **Turtle Geometry**
7. **Turtle Symmetry**

1. **Input-Output**

This unit develops students' abilities to recognize and describe patterns and introduces the notion of expressing mathematical concepts in a symbolic language, **Easy Speak.** The experience of using Easy Speak prepares students to use algebra to express numerical ideas. The motivation for using Easy Speak comes from the power it gives its users to create Input-Output machines, give them to other students to analyse, and to share them with other classrooms of students via the Network.

The unit begins with practice on simple input-output machines many of which grow out of activities with manipulative materials and moves to machines with two-part rules (e.g. 'if the input is an even number, the output is zero; and if the input is odd, the output is one').

2. **Strategies**

Prerequisite: Input-Output unit

Continuing the exploration of Easy Speak which students begin in the Input-Output Unit, the Strategies Unit emphasizes discovering, inventing, and finally, expressing in the Easy Speak language strategies for playing the game of Nim. In this unit, computer and to store them on the Network. When not making designs on the computer, students use 'Turtle-tractors,' devices similar to protractors, to make drawings on paper.

Besides learning to express geometrical ideas in a computer language, students discover theorems about polygons, learn to command more than one turtle at a time, and become expert at seeing how a complex design is built of smaller parts.

7. Turtle Symmetry
Prerequisite: Turtle Geometry
Because the Turtletalk language makes it easy to command multiple, symmetrically placed turtles, it provides an ideal medium for studying symmetry. In this unit, students learn to identify and draw designs with different kinds of mirror and rotational symmetry. They also study and create 'families' of designs - sets of designs which have common characteristics yet differ in patterned ways.

Guide for Master Teachers[1]
Most teachers will need support in getting started using the units. In addition to the new ideas about using computers and networking, some of the teaching strategies, particularly group work and using concrete materials, are not commonly seen in middle school classrooms. This guide outlines a series of seven workshops which could form, or be integrated into, an in-service program or a pre-service college course.

[1] Turtletalk is a simple graphics language, a subset of the language LOGO developed by the MIT LOGO group. We gratefully acknowledge their work, especially as put forward by Seymour Papert, Hal Abelson, and Andy diSessa.

Network Manager Manual

by
Bill Finzer
Math Network Curriculum Project
San Francisco State University
February 9, 1982

Overview

Computer networking takes advantage of two important capabilities of computers:

- to store and retrieve information
- to communicate with each other over the phone lines

In the Math Network Curriculum Project we are interested in how teachers and students in middle school classrooms might use networking to enhance their learning of mathematics.

Our vision was that teachers and students would gain significantly from being able to communicate with teachers and students in other classrooms. We knew that we wanted some of this communication to be in the form of 'messages' to each other; but that we also wanted them to be able to store the results of their studies in a place that others in the community could access them. To realize these goals, the system described in these pages was developed.

Purpose of this Manual

You will find in this manual:

- a description of the Math Network Curriculum Project (MNCP) Network as we implemented it,
- some considerations for those who may be developing their own network systems,
- maintenance instructions for the MNCP Network, and

It is presumed that anyone reading this manual has had some prior programming experience, and some experience using microcomputers connected to phone lines.

The MNCP Network Implementation

Definitions

We need a common terminology to use to talk about the various aspects of the Network System. Refer to figure 1 to see how these components interconnect.

user: refers to a person (usually a teacher or student in a classroom) with a computer (in our case Commodore Pets with 16K of memory) connected to a phone line through a modem who dials up the **server** (see below) to gain access to some capability of the Network.

user program: is a program loaded in the user's computer which allows that computer to talk properly to the server computer.

server: is a computer which monitors incoming phone calls from users and 'serves' them information which is stored on its disk. The server functions automatically without need for a person to be there except in case of power failure or special needs which involve changing hardware or software. An Apple II micro-computer was the server computer for the MNCP Network.

server program: is a computer program which runs on the server computer. It is this program which deals with the incoming calls, deciding what kind of **transaction** is desired.

transaction: refers to a unit of business between the server program and the user program. An example of a transaction might be the user program asking for one line of a user's message to be retrieved and the subsequent retrieval and sending of that line to the user computer.

manager: refers to a person who is responsible for maintaining the server, the server programs, and the user programs. The manager must understand in detail how everything works so that when things go wrong, s/he can fix them; or, when new capabilities are desired, the manager should be able to determine if they are possible and, if so, provide them.

manager programs: the manager uses various programs run on the server to perform functions like adding a new user mailbox, checking to see if storage on the server's disk is getting low, setting up new data files, etc.

System Capabilities

A teacher or student (i.e. a **user**) in a classroom with a computer connected to a phone line through a modem can use the MNCP Network for two quite different tasks.

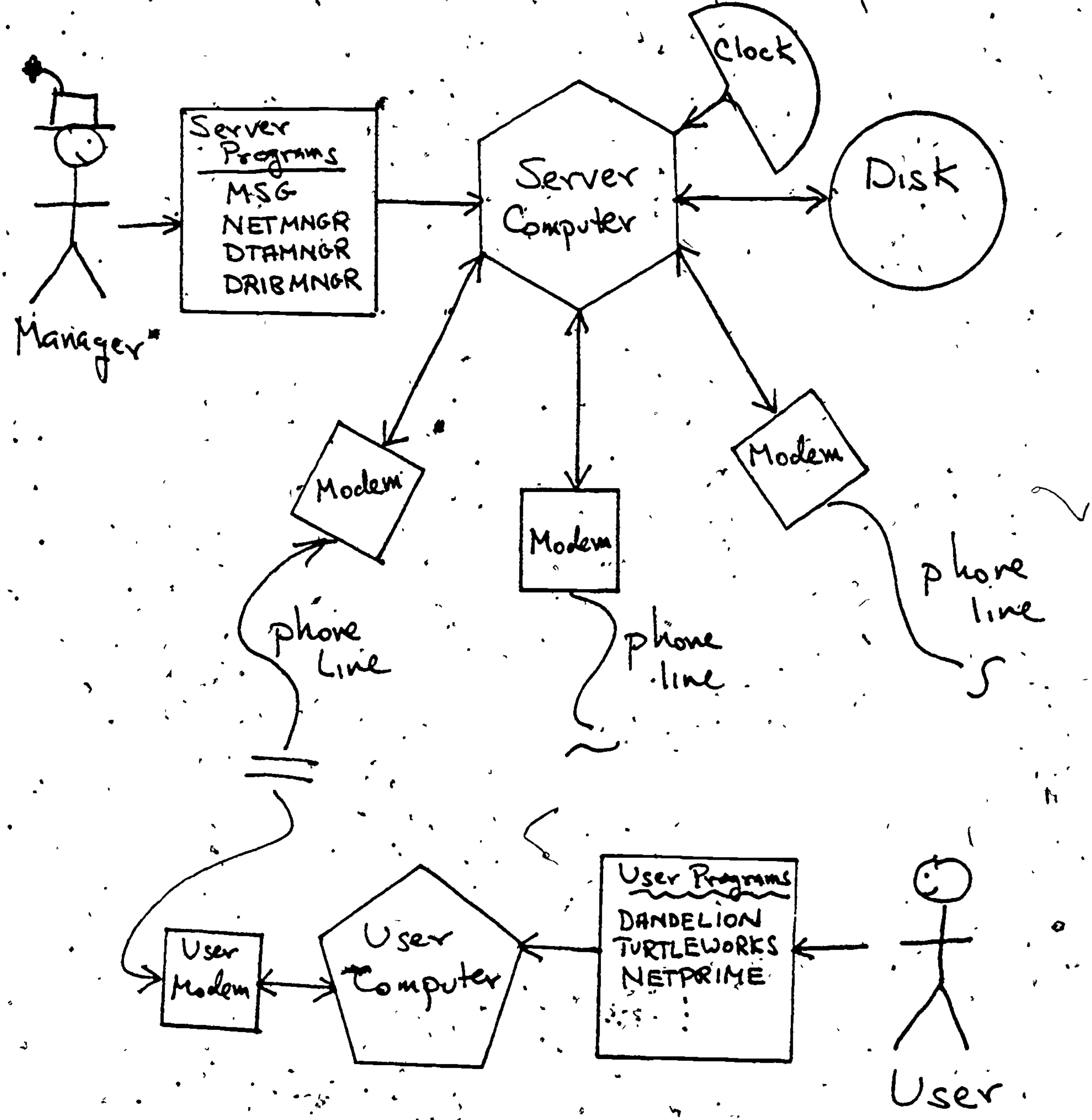


figure 1
MNCP Network Configuration

Dandelion - an Electronic Message System

With his/her computer running a program called Dandelion, the user gains access to an electronic message system. Each teacher and each classroom of students has a 'mailbox' on the server Apple. The user send messages to any mailbox or any group of mailboxes (called 'distribution lists'). A user with a mailbox can, of course, look at the messages that may be waiting in his/her mailbox. In addition, there is a special mailbox called 'bulletins' in which messages of general interest may be posted. Dandelion provides the basic communication link between people in the MNCP community. Through it, people arrange meetings, share ideas about curriculum, ask questions about programs or teaching materials, report successes or records reached in working with classroom computer programs, conduct penpal-like correspondence, job hunt, and sell cookies and computer equipment. Refer to the Dandelion User Manual for more information about this system.

Applications Which Store and Access Data

There are various other programs which users can run on their computers which store and retrieve information to and from the server's disk. In the current implementation of the MNCP Network there are seven of these user programs. To take three examples:

Turtleworks is a program which allows the user to program in a graphics oriented language (a subset of LOGO) to explore problems in geometry. The Network can be used to store sets of 'TurtleTalk' commands (which can be considered a program) on the server's disk. This means that the user can later retrieve these commands to continue working on them. It also means that other users can look at the work of any individual user to get ideas from it or simply to appreciate it. Users may look at any stored programs but may only delete programs that they themselves stored.

NetPrime allows storage and retrieval of information used in a simulation of Network TV ratings. Individual users store their entries in the simulation which consist of a schedule of TV shows for Tuesday night's prime time. Any user may then retrieve up to five sets of schedules onto his/her computer and have them compete against each other. Typically, a classroom of students decides as a group what their schedule should be before trying it out on the Network.

DataMaker allows users to create data files and store them on the Network. A typical data file might consist of the information gathered on a questionnaire given to all the students in a class. Because the information is stored on the Network, other classrooms may load it into their computers and see, for example, how this other classroom data is similar to or differs from theirs.

Hardware

Figure 1 shows the basic layout of the MNCP Network. A single Apple II with a Corvus hard disk and three MicroModems serves incoming calls from 9 classrooms in San Francisco and South San Francisco. This Apple II uses a Mountain Hardware clock to generate interrupts for polling the three incoming lines and for time and date 'stamping' transactions. The classroom computers are Commodore Pets equipped with Novation D-Cat (direct couple) modems.

Clearly, a variety of different hardware configurations would have served equally well. There are, however, important constraints on each component.

Constraints on the Server

Twenty-four hours a day the server monitors possible incoming calls on each of its three modems. Each time it gets a call it handles 'transactions' between it and the calling computer. It 'serves' the calling computer. The following list details the capabilities that the server must have.

1. **Storage Capacity** - Having a lot of mass storage available is essential. The MNCP server currently uses about one megabyte of storage on its Corvus disk (which has 9.6 megabytes total capacity). With more users, of course, more storage space will be required.

2. **Channels** - Currently, up to three users may call in simultaneously to the server. About 10% of the time that the server is busy, there are two callers on line. Less than 1% of the time, there are three callers on line. With more users, again, there would be more demand on the multi-channel capacity of the network. It is **very important**, especially in serving relatively naive users, that the incoming lines be seldom, if ever, busy. We elected to make this possible by having a single Apple II have three incoming lines. This had significant implications on the design of the system. In particular, it led to complexity in the server program and to relatively slow response of the server in dealing with transactions. Another solution would have been to have one Apple II for each incoming line. Since the Corvus disk system supports sharing (through its 'Constellation'), each Apple II could have accessed the same information on the disk. Complexity would have been introduced in making sure that one server did not destroy information needed by another.

3. **Speed** - The server must be able to respond quickly enough to user needs that the user does not feel frustrated. The MNCP Network barely meets this requirement, mostly because of the complexity of dealing with several channels in one server. The server program is written in UCSD PASCAL, a language which runs significantly faster on the Apple than does BASIC. BASIC, then, is ruled out as an implementation language because of speed.

4. **Complexity** - The server program is necessarily somewhat complex. It has a multitude of tasks to perform (even if multi-channel capacity is not included). This means that the server program must be particularly carefully written and well-documented, so that it can be maintained easily and new capabilities can be added with relative ease. It was for these reasons that we chose to write this program in UCSD PASCAL rather than assembly language or BASIC. Except for the complications introduced by the multi-channel requirement, this proved to be a good choice. On larger systems, there are languages which support 'asynchronous processes.' These languages would have made much easier the task of supporting multiple channels. At present, however, there are no implementations of systems which have asynchronous processes on micro-computers, and our budget dictated against buying a larger computer to be used as the server.

Constraints on the Modems

There are two brands of modems in the MNCP network. The server uses MicroModem II, manufactured by D.C. Hayes. This modem allows the server to **answer** the phone when it rings. Since it is a 'direct-couple' modem, it has very low noise characteristics compared to acoustically coupled modems. It proved completely satisfactory for our uses.

Each user computer was equipped with a Novation D-Cat modem. Since the user is present, auto-answer capabilities were not necessary for the user modem. Again direct-coupling meant that we had no noise problems. The difficulties that we encounted with this modem were that we had to write our own 'UART' software which then took away memory space from application programs, and that the physical construction of the modem was quite flimsy, thus some of them broke in use by teachers and students. In retrospect, we wish we had used an IEEE modem such as the one provided by Commodore or manufactured by Livermore Star.

Constraints on the User Computer

Our computers in the classroom were Commodore Pets with 16K of RAM. Virtually any of the available micros with this amount of memory or more would have been adequate for the networking aspects of our project. Note, however, that the user computer is **not acting as an ordinary computer terminal** when it is using the Network. It is actually running a program, and it is the program that conducts transactions with the server computer. This means that as we expand to include other computers in the network, applications software which currently runs on the PETs must be rewritten to work on other computers. MNCP software for user computers is written in MicroSoft BASIC, a language which proved quite adequate for the curriculum needs of the project. We were, however, constantly running into the 16K memory limitations of the user machines. We did not choose to use a model with more memory because, given that our users were loading programs with tape cassette recorders, we thought that the time of loading a 16K program would be all that could be tolerated.

Software

It is important to contrast the way interactions between the server and user computers take place with typical interactions between a 'host time-sharing computer' and a user terminal. In the latter case, every keystroke that the user makes on his/her terminal is sent to the host computer. The user is interacting **directly** with the host computer. In the case of a server-user interaction, however, the user interacts directly with his/her **own computer**. The program in that computer decides when it is appropriate to interact with the server computer. To take an example, when a user wants to send a message to someone, none of the typing in and editing of the message involves the server computer until the user decides that the message is actually ready to be sent; then the user computer sends the lines of the message to the server.

Protocol

To manage this interaction, there must be an agreed upon 'protocol' between the user program and the server program. Once there is a protocol, the particular kind of computer the user has, or the language that runs on it does not matter. The only thing that matters is that the correspondence between server and user follow the protocol. In the MNCP Network, the first character transmitted always decided whether the transaction had to do with the message system or with the data files. The next byte indicated the particular kind of transaction that was to be carried out. As of this writing, detailed documentation of the MNCP protocol is to be found in the comments of the PASCAL program which runs on the server.

Implementing on New Machines

While at present we have implemented the MNCP programs only for PET computers, implementing them for another computer in another language only involves making sure that the network protocol is satisfied during transactions between user and server.

Writing New Applications

17

Suppose that someone invents a new use for the data file capability of the server. Implementing that use is relatively simple. The data file is created and initialized on the server disk; the server program is augmented so that it can know about the new data file; and the user program is written so that it will use the already agreed upon protocol for accessing data files.

Developing Your Own Network

At this time, the MNCP Network is an experimental system which may or may not become available through commercial sources. Suppose that you are interested in setting up a networking facility in a school, district, county, or region. Let us consider some of the possibilities that exist.

Becoming an MNCP Participant

If your users have Commodore 16K Pets and you are located near San Francisco (so that phone costs do not become prohibitive) then you could become a participant in the MNCP Network. Each user would need a modem, a phone line, and software available from us. A disk drive is not necessary, though it is, of course, convenient. You would then gain access to a supported system including the Dandelion message system and curriculum applications in middle school mathematics. New applications developed by you could also be supported at the MNCP end.

Single Campus Networking

If all your users are in one school, you can purchase a commercial network system (one is available from Corvus as of this writing) which will allow multiple servers and for servers and users to share a high-speed network cable. One of the servers on the network could be a message and data server. Another can be a printer server. Considerable software support comes with such systems, but you would need to develop your own particular applications packages for the curriculum uses you have in mind.

Bulletin Boards

There are available quite a number of bulletin board programs for micro-computers. A bulletin board allows posting of messages for other users to browse and read. Some will support sharing of BASIC programs. Some support individual mailboxes. There are several disadvantages to currently available bulletin board systems. You cannot modify them to support other applications; e.g: sharing of Turtletalk programs. In addition, only one user can be calling in at a time.

Linked Nets

There exists a model for what one would really like in the way of a network for schools. That is the Defense Department's ArpaNet. The ArpaNet is a world-wide network which links together smaller networks. Each link in the ArpaNet is a 'gateway' which is a special server computer whose job it is to route messages from the local network to the appropriate gateway in another part of the global network. The goal would be that in each school building there is a coaxial cable with several 'outlets' in each room. Any computer could be plugged into any outlet. With your computer plugged in, you could reach any server on the local net. So you could send material to be printed to a printer server, material to be stored to a disk server, etcetera. You could send a message to a student or teacher in your school by sending it to a message server. Now, if you want to send a message to a person in another school entirely, your local network's gateway server reads the address of your message and finds a phone path to the gateway where that person has his/her mailbox and sends it there. So you could send and receive information to and from a very large community of educators in your region, or across the country. Given that such a system exists now (the ArpaNet) it seems only a matter of time before such facilities are available more generally, and especially to schools.

MNCP Network Maintenance

The purpose of this section is to detail for you how to start and maintain the MNCP Network. Included are explanations of the various files and programs that make up the network. It is assumed here that you are somewhat familiar with the UCSD PASCAL operating system on the Apple II and that you have already connected your CORVUS disk and got PASCAL installed on it. You should also know about creating new 'volumes' on the CORVUS.

What you Need

You need the following hardware:

- an Apple II computer,
- a Corvus hard disk properly connected to your Apple II,
- from one to three Micromodem II's put in slots 2, 5, and 4 (so if you had two modems, they would be put in slots 2 and 5), and
- a Mountain Hardware clock in slot six set to interrupt every 1 millisecond. (See their documentation for how to do this.)

You may use the Corvus Constellation to allow other computers to share the Corvus disk, but you must set the polling parameter of the Constellation to the maximum allowable time (255 cycles). Corvus' documentation explains how to do this. The purpose is to prevent the Constellation from going on to the next computer while your server Apple is interrupted by the clock to look at a modem line.

Let us suppose that you are running from a volume on your Corvus called 'NETWORK:.' When you start, this root volume should have the following files:

SYSTEM.PASCAL (supplied by APPLE)
SYSTEM.MISCINFO (supplied by APPLE)

```
SYSTEM.LIBRARY (supplied by MNCP)

NETMNGR.CODE (supplied by MNCP)
DTAMNGR.CODE (supplied by MNCP)
DRIBMNGR.CODE (supplied by MNCP)
MSG1.V9.CODE (supplied by MNCP)
MSG2.V9.CODE (supplied by MNCP)
MSG3.V9.CODE (supplied by MNCP)
```

The Network programs do not care what you call your volumes, but they will expect all the relevant files which you are about to create to be on the 'prefix' volume. Typically the prefix volume will be the same as your root volume. The size of the prefix volume should be quite large. We use a prefix volume 2048 blocks long. Since each block is 512 bytes, that means we have about 1 megabyte of storage available. With more users and/or more applications, you will need more storage.

Getting Started

Your first task is to create mailboxes for your users and to initialize the files which will hold their mail. Then you will initialize data files and dribble files (used to record the activity of the system which is useful when you want to know how it is being used). Finally you can start the networking program itself.

1. Execute the program NETMNGR. Among the options at the top of the screen is one for U(sers. Choose that (by the single-keystroke U). You are now about to make N(ew users. By choosing that option, you will be asked for the name, password, and status of the new user. Names and passwords should be in all UPPERCASE and can be at most 16 characters long, and status may be one of M(anager, T(eacher, S(tudent, G(roup, or O(ther. After having entered this data, you are asked for confirmation that it is all OK. Included among your users should be one called BULLETINS which has no password and status O(ther. When you are done, you can L(ist all or some of the users you have created. If you have made any mistakes, you can R(emove those users and make them again as N(ew users. Finally, when they are all set up properly, you should Q(uit. You will be asked if you want to save the changes you have made, to which you should reply Yes, at which time the information will be written out on the disk to a file named USER.HDRS.

2. You are now back to the main part of NETMNGR. You next need to place your users on appropriate 'distribution lists.' To do that, choose the D(ist option. Now you want to create a N(ew list. An example of such a list might be TEACHERS. Then you want to E(dit this list and A(dd the appropriate users, one at a time, to it. Follow this procedure for each list you desire for your system. Finally Q(uit.

3. Back again at the top level of NETMNGR, you want to I(nitialize the remaining files. You will be asked how long to make each file. Typical responses might be:

   400 headers
   2000 bdys
   400 directory entries

   Normally, you will respond Y(es to the final question as to whether to initialize the User files as well. You will return automatically to the top level of NETMNGR.

4. You are now finished with NETMNGR so Q(uit from it. You should find yourself in the top level of PASCAL.

5. Your next task is to initialize the data files. The program DTAMNGR is what allows you to do this. X(ecute DTAMNGR. Choose the I(nitialize option. The various data files that your server programs knows about will be listed by name. As of this writing, they are

   1) Multiturtles
   2) Prime Time
   3) Nim Speak
   4) Easy Speak
   5) At Your Service
   6) Business Speak
   7) Data Analysis

   Choose each in succession by number. You will be asked how many records to allocate for the directory of each file (100 is a typical response) and how many data entries per file to allocate (1200 is typical). When finished with all of them, Q(uit to return to toplevel of PASCAL.

6. Finally you need to initialize the dribble files which will record system activity for you. X(ecute DRIBMNGR and choose the I(nitialize option. Allocating 1000 entries for the dribble file is typical and will allow the system to run for several months without overflowing with dribble entries. Q(uit and to return to PASCAL.

7. Now you are ready to see if your system works. There are three versions of the program MSG on the root volume (MSG1.V9.CODE, MSG2.V9.CODE, and MSG3.V9.CODE), one for each possible number of modems you might have connected to your system. Choose the appropriate one and X(ecute it. For example, if you have two modems, then X(ecute MSG2.V9. Eventually, the screen should show

various system information on it, including the date and time, number of channels available, etc. The cursor will be winking along updating this information. (Note, you will need to have properly set the current time on your clock using the utility program provided with the clock.)

8. Load up Dandelion on one of your user computers and try calling in. While you are connected, the Apple screen should show the data received and transmitted, and the number of cycles since the last activity.

9. When you want (for some reason) to stop the MSG, you should make sure that no one is on-line at the moment and then RESET your Apple. (Later versions of the server program will allow a more graceful exit, but, for now, you must reset in order to stop the interrupts from the clock from messing up normal, non-networking operation.)

Maintenance

Basically, the MNCP Network will run with very little intervention from its manager. There are, however, a few things to do from time to time.

1. Ocassionally, one of your users will stop using the system and thus messages will accumulate to them without ever getting read. You can detect this condition using NETMNGR, the U(ser option and L(isting all users. The last column is the number of messages that they have. A user with a lot of messages should be contacted and asked if they wish to be kept on the system. If not, you should dial up as them and delete all their messages. Then go to the NETMNGR program and D(elete them from the list of users. Also D(elete them from any distribution lists they might be on.

2. Use the NETMNGR program to E(xamine the three files MSG.HDRS, MSG.BDYS, MSG.DIR to make sure that they continue to have plenty of space on them for incoming messages. If not, you will need to reinitialize the system with more space on it. (Currently, this is a deficiency in the system. It should be made possible to increase the size of a file without having to lose current messages.)

3. Every 20,000 cycles during the day and every 100,000 cycles at night, an entry in the dribble file is made to show how much activity there has been since the last dribble. Eventually this file will fill up. The program DRIBMNGR allows you to make a printout of the dribble file and to reinitialize it.

Dandelion Message System User Manual

Bill Finzer

Math Network Curriculum Project

San Francisco State University
August 15, 1981

Introduction

The dandelion owes its success in nature to its ability to spread itself quickly far and wide, its seeds riding lightly on currents of air. We of the Math Network Curriculum Project hope that through this Dandelion message system for the PET computer, your ideas, questions, and suggestions may also propagate quickly through the electronic currents of phone wires to take root in others' minds, producing timely and constructive responses.

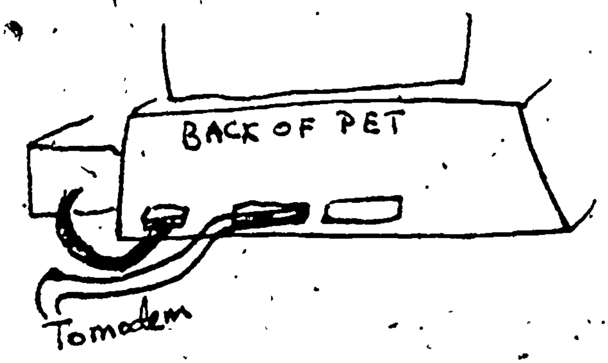
The Dandelion Message System works, in many ways, just like the postal system. You send messages which get put in other people's mailboxes. When other people send you a message, it appears in your mailbox. You can browse the messages in your mailbox by looking at their 'headers' much as you might look at the outsides of envelopes. Usually you will examine the contents of your messages. Some messages you will keep and others you will throw away.

There are significant differences between Dandelion and the postal system, however. First of all, the messages you send are deposited **immediately** in the receiver's mailbox. Secondly, you can send a message to a large number of people simultaneously, if you wish. Lastly, it is possible to post messages in a 'public' mailbox (called **bulletins**) where everyone can browse just as they might browse a school bulletin board.

Getting Started

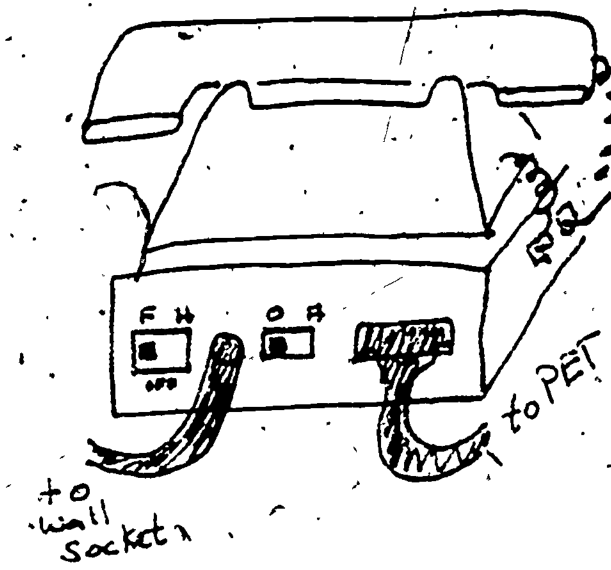
To use Dandelion, you will need:

- A Commodore PET computer with at least 16K of memory running BASIC 4.0
- A Novation D-Cat modem with connector to the PET User Port
- A phone which has the clip-on connections to the handset
- A tape of the Dandelion program



The modem is the thin black rectangular box. It must be connected to three things: the PET, the phone, and an electrical power outlet.

1) Use the gray cable to connect the modem to the PET. The funny-shaped end with the pins goes to the back of the modem and the green connector with the goop on it goes to the 'user port' of the PET. See picture to the left to see where this is. Pay attention to the direction the cable comes from the PET.



2) Put your phone on top of the modem. You need to connect the phone to the modem using the two plugs on the left side of the modem. One curly cord must go from the handset to the modem and the other must go from the phone body to the modem.

3) Finally, plug the modem in using its power cord with the transformer on the plug end. The other end of the power cord connects into the small round hole in the back of the modem.

Now that the connections are made, you need to set the switches on the modem properly.

1) **Set both of the small black switches on the back of the modem to the right.** The right hand switch is the power switch while the left switch determines whether the modem is going to 'originate' the call or 'answer' one.

2) **Both of the large black buttons on the front of the modem should be pushed in.** When you want to use the phone for normal voice communication, you can let both these buttons out. You don't have to disconnect it from the modem.

Finally, load your Dandelion tape into the PET as you would any other program. This is a long program and so takes about 5 minutes to load.

To SAVE the Dandelion program on tape so that you will have a backup copy, simply use the normal PET SAVE procedure, but do it **before** you **RUN** the program. (You can also SAVE after RUNning if you are careful to stop the program using the QUIT command.

Using Dandelion

When you first RUN Dandelion, you will see a Welcome message and a reminder about how to set up your modem. The MNCP telephone number is also displayed. You are instructed to dial this number. There is also a blinking 'W' which is Dandelion's way of telling you that it is Waiting for something to come to it over the phone line.

After you have dialled, wait 10 to 20 seconds while the MNCP computer answers and sends its first message to your PET. You will see the message displayed on your screen. It will tell you how many people are using the message system at that moment.

> If you find you are waiting longer than 20 seconds, it may be because the line is busy. Push the right black button on the front of the modem so that it is out. Listen to your phone. If you hear a busy signal, then the MNCP computer already has as many users as it can handle. If the phone is still ringing, then the MNCP computer is 'down' or being used for something else at that time. Call back later in either case.

Underneath the greeting from MNCP, is the instruction to

Press RETURN to go on

After you do so, you will see a whole set of options. These are explained below. You choose an option by typing the first letter of its name and then [**RETURN**].

LOGIN

Usually the first thing you do will be to **LOGIN**. This just means that you are letting your PET know who you are so that it can properly look things up in your mailbox on the MNCP computer. Note that you should **not** capitalize your name.

After typing L and [**RETURN**] you will be asked for your mailbox name. You must be a recognized user of the MNCP system to have a mailbox name. Usually these names are last names of people. However, there are some names that stand for groups of people like

designgroup
or
chessinterest

There is also a mailbox whose name is

bulletins

which is a place that anyone can browse.

After you type in your mailbox name, Dandelion will see if that is a name recognized by the MNCP mail system. If so, it will tell you how many messages you have. If you have a password, it will also ask you for that password. (You will notice that nothing appears on the screen as you type in your password. Passwords are supposed to be secret, to make it difficult for other people to look at your mail.)

BROWSE

If you have some messages in your mailbox, you can **BROWSE** them. You will be asked which message number to start at and which one to end at. This is useful because sometimes you will accumulate a lot of messages and you may only want to look at the most recent of them. If you want to browse backwards, simply make the second number smaller than the first.

You will see the 'headers' to your messages displayed on the screen. These will tell you some useful information about the message itself, such as **who** sent it, **when** it was sent, what the **number** of the message is, and what the **subject** is. It is **not**, however, the message itself. The header is just like the **outside** of the envelope.

Sometimes you will seem to get stuck in a long sequence of browsing which you would like to terminate because you just saw the message you want to examine. You can **'escape'** from browsing by holding down the **'shift'** key and pressing the **'@'** key. When Dandelion has finished displaying the next message header, it will stop browsing and let you go on. The **'shift @'** key is called the **'escape'** key and will let your **escape** from many places in Dandelion where you don't want to be. If you don't like where you are, try it!

EXAMINE

This is the command that lets you look at the actual message itself. You will be asked for the **number** of the message. This is the number that got displayed when you did the browsing. If you have 5 messages, then the most recent message is numbered 5. If you forgot the message number, then you can **escape** from here and go back and browse again.

You will see the header displayed again and then the lines of the message itself will be displayed one at a time as they are received by Dandelion.

You have probably noticed that while Dandelion is sending and receiving information over the phone line the 'cursor' changes to S for 'sending' and R for receiving, in addition to the W for waiting. If you listen to the phone line (with the right modem button out) you can hear this process going on.

After the last line is displayed you will be asked if you want to delete this message. **Be careful here!** If you answer y, this message will be lost from your mailbox and there will be no way of getting it back!

SEND

When you send a message, the program asks you for the names of the receivers of the message. You can type these in one at a time, hitting **[RETURN]** after each one, or you can type several in on one line with commas in between. In any event, Dandelion will check each potential receiver of your message to see if it is recognized by the MNCP mail system.

You may send a message to the receiver **bulletins.** This places your message in a public place where anyone can look at it. You could use this to announce a brand-new program you have created, place a challenge for high-score on Lemonade, or advertise a free puppy.

There are also **'distribution lists'.** These are whole long lists of mailboxes maintained by the MNCP mail system. Distribution lists end with an 'up-arrow' character. For example, **teachers^** is the name of the distribution lists with all the teachers names on it. (You will notice that it takes a bit longer to send a message to a long list of people than to just one or two, but not too much longer.)

If you have not logged in, you will now be asked who you are. This is the name that will appear next to 'From:' in the message header. If you **have** logged in, then your log-in name will appear here.

Next you must supply the subject of your message. This should be a short phrase, indicating to the receiver what you are about to say. You are limited to one line

Finally, you can type the message itself. You can format the way the message looks on the screen any way you like using spaces, returns, etc. The DEL key will backspace as far back as the beginning of the message. If you get totally disgusted, you can start from the beginning of the sending process by using the **escape** key (shift @). When you are finally ready to actually send the message, **hold down the shift key and press the**

'&' key. This tells Dandelion to 'accept' the message as shown on the screen and send it. You will see the sending happening as the cursor changes from S to W to R over and over again for each line of your message.

There is a limit on how many lines long your message can be. This limit is 25 lines, or about a PET screenful.

DELETE

You will want to get rid of most of the messages you receive. Of course you can do this at the time you examine them. But you can also do it later using this command. You will be asked, as usual, for the number of the message to delete. Then the header will be displayed and you will be asked if this is the message you wish to delete. Typing 'y' here will cause the message to be lost from your mailbox.

LOGOUT

Sometimes there will be another person waiting to use Dandelion when you are done. You should use **LOGOUT** before letting them use the computer. This will ensure that they can't sit down and start browsing your messages.

GOODBYE

This is the graceful way to leave Dandelion. It gives Dandelion a chance to set things up properly for the next program loaded into the computer. If you instead use the RUN/STOP key to end Dandelion, the next program you load in may not work correctly. You are also instructed at this point to please **hang up the phone**. This is very important, since, if you don't, the MNCP computer will stay listening to you. You phone bill will get larger and others will not be able to use your line in. Because of the fact that some users will, inevitably, forget to hang up on occasion, the MNCP computer will hang up on you if it doesn't receive anything from you for fifteen minutes. So, be warned! Don't take more than 15 minutes to type in that message!

Appendix D

Talks and Papers Presented

April 1981	Resek. "Microcomputers in the Mathematics Classroom" University of Arizona, Mathematics Department Tempe, Arizona
June 1981	Resek and Finzer. "Math Network Curriculum Project" Joint NSF-NIE Conference on Technology in Math Education Berkeley, California
October 1981	Finzer. "Some Statistical Simulations" Computer Using Educators (CUE) Fall Conference San Jose, California
November 1981	Resek. "Computers in Math Education" EQUALS Workshop, Lawrence Hall of Science Berkeley, California
November 1981	Gutierrez. "Training Teachers in the Use of Microcomputers" Panel, California Educational Computer Consortium (CECC) San Francisco, California
December 1981	Resek. "Math Network Curriculum Project" Northern California Mathematics Council Asilomar, California
December 1981	Finzer. "Turtleworks - a geometry unit" University of California at Santa Barbara Goleta, California
January 1982	Finzer. "Turtleworks - a geometry unit" Commodore Conference for Educators San Francisco, California
February 1982	Resek. "Using Computers for Pre-Algebra" NSF Inservice Teacher Project University of California at Santa Barbara Goleta, California
March 1982	Gutierrez. "The Math Network Curriculum Project" California Educational Computer Consortium California State Polytechnic San Luis Obispo, California
April 1982	Finzer. "Graphics Languages" Computer Using Educators Spring Conference San Jose, California
Spring 1982	Resek. A workshop series for teacher inservice, Berkeley Unified School District. Berkeley, California

October 1982	Finzer. NSF Inservice Teacher Project University of California at Santa Barbara Goleta, California
October 1982	Gutierrez. "A Statistics Unit for Middle School" Computer Using Educators Fall Conference San Jose, California
November 1982	Gutierrez. Inservice workshop for teachers Kentfield School District Kentfield, California
February 1983	Resek. "Introducing Algebra through Microcomputers" SESAME Colloquium University of California, Berkeley
April 1983	Resek. "Introducing Algebra through Microcomputers" National Council of Teachers of Mathematics Detroit, Michigan