


FIGURE 4-18. Multi-day appointment with alarm and other details window open

Let's assume that your dad lets you know that he'll definitely be able to make the visit. Now that you're sure he'll be here, you can open the appointment, tap *alarm+* again to see the appointment's details, and change the status to *confirmed*. When you tap *save* to close the appointment again, the mysterious question mark is gone, replaced by the more satisfying multi-day image that you've seen before.

 **Jump in Anywhere.** A multi-day appointment appears on the day view of every day that it includes. If you want to change anything about the appointment, including its name, duration, or anything else, you can get to the appointment from any of its days.

Now that you're sure your father's coming, you can really look forward to his visit. In fact, you can use another datebook feature to reflect the excitement. If you open the appointment again and then tap *alarm+* one more time, you can use the appointment priority feature to tell Magic Cap that this appointment is very important by giving it high priority. When you save the appointment, the week and month views still show it listed as any other multi-day item, drawing a line from the start date through the end date, but in the day view the datebook reminds you that his visit has high priority by stamping the appointment with an exclamation point. He'd be really pleased with that.

Customizing with Rules

As mentioned earlier, the standard signals for appointments are the alarm sound and an announcement that stays open until you close it, but you can customize this behavior by using the datebook rules. You can choose to have both the sound and the announcement, or just one or the other. If you don't like the alarm sound, you can replace it with another sound that you're more fond of.

As you might recall, there are different rules for each scene, so make sure the datebook is open when you want to change its rules. After you open the datebook, tap the lamp, then *rules*. When you tap there, you'll see the first three rules that the datebook lets you set, as pictured in Figure 4-19. By tapping in the box or circle at the left, you can turn each rule on or off. The first two rules determine what will happen when an appointment alarm goes off.

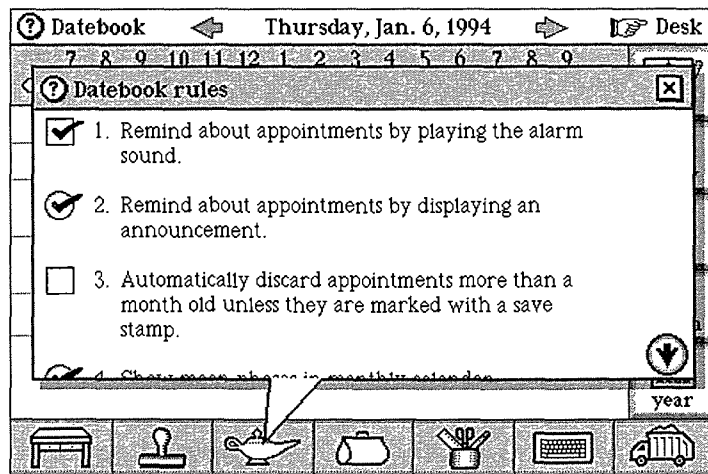


FIGURE 4-19. Datebook rules

The on-screen announcement is a great reminder for people who are easily startled, or for anyone who spends a lot of time in meetings or libraries. If you're more the aural type, you can use a sound to remind you instead of (or in addition to) the announcement. To turn the sound alarm off, just tap the check box at the start of *rule 1, Remind about appointments by playing the alarm sound*. When the check mark goes away, the rule is turned off, and you won't hear a sound to remind you about appointments.

Now, when an appointment alarm goes off, you'll get an announcement in a box on the screen. Even if the communicator is turned off, it'll come back on to display the reminder, and the announcement will stay there until you close it, which helps to make sure you don't miss it.

There are two kinds of rules in the datebook, those with circle check boxes and those with squares. The rules with circle check boxes are simply turned on or off; you can't customize them any further. The rules with square

check boxes are flexible—you can play with them. The rule about playing a sound for an alarm can be customized by tapping the rule's text. Once you do that, you can pick any of the built-in sounds for the alarm, as you can see in Figure 4-20.

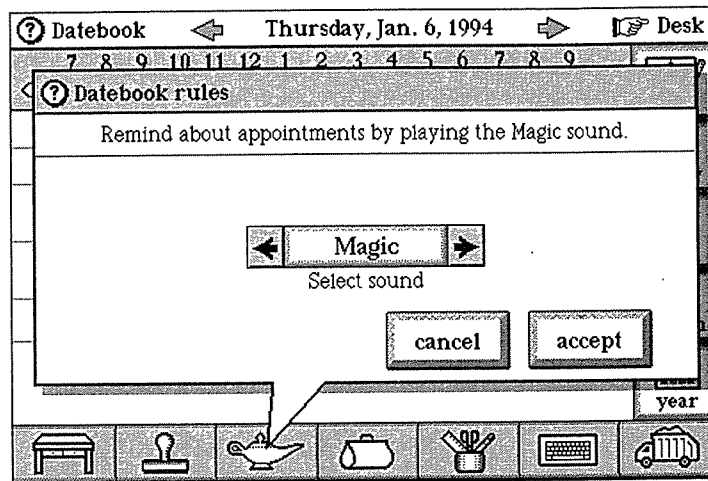


FIGURE 4-20. Customizing the sound played as an appointment reminder

When you've got the rule the way you want it, you can *save it*, making it the ruling sound, or you can *save a copy* of it and have a brand-new rule that uses this sound as the reminder, leaving the rule about the alarm sound just as it was. You could be really obnoxious and have both sounds (or even more), but that isn't a good idea for anyone who might be in a staff meeting when the alarm goes off.

Summary

The datebook is the place to tell Magic Cap about your appointments, and if necessary, have Magic Cap help you tell everyone else who needs to know. There are several built-in kinds of appointments to choose from. You can note birthdays that will automatically repeat every year, even though they're only entered once. Business trips can be scheduled across several days with a single entry, and personal notes are easily attached that stay with the appointment throughout its duration. Magic Cap can help schedule a meeting, and while you're entering that appointment, the touch of a button automatically creates and sends an invitation to the meeting's participants.

Your datebook becomes a gentle reminder of things you need to do, moving tasks from day to day until you complete them. You can even design your own type of appointment, reducing the amount of time spent entering the same event, like ball games or concerts.

There are various lists included in the datebook that cut down on typing because you can choose anything from the lists with just a touch. There are descriptions of the kind of appointments you can make, from serious (*staff meeting*) to light-hearted (*hot date*). There is a list of locations to choose from, ranging from *school* to *secret hiding place*. The datebook remembers everything you add, so the next time you want to schedule your acupuncture appointment, or enter the name of your favorite Chinese restaurant, it will be added to the lists of descriptions for you to choose from.

When choosing the participants of an event, the datebook displays the names in your name file, and even gives you the opportunity to add a new card for a new contact while entering that appointment. An appointment can tag along on an invitation to a meeting, and if

the invitee responds positively, it will deposit itself in the invitee's Magic Cap datebook.

You can customize each event with notes. You can give an appointment instructions to repeat itself at certain intervals, and you can ask your datebook to remind you about the appointment with a sound or announcement.

Like the rest of Magic Cap, the datebook is designed to reflect the way you live and work. While you're at work, you might need to remind yourself to stop at the video store on the way home to pick up one of the movies little Zuzu wants to see. When you're at home, you'll need a place to write down the date of the big communication meeting at work, as well as some ideas about what to include in your presentation. The datebook works with the rest of Magic Cap to help you run your life.

Chapter 5



Name Cards

Keeping In Touch with Everyone

I'm usually pretty good at remembering the birthdays of family members and friends, especially when they happen to be on otherwise memorable days, like December 7 or April 1. I'm also good at remembering to buy birthday cards for friends early enough to mail them to arrive on time. If I were really together, I'd also remember to jot down my friends' addresses before I went to the card store so that I could buy them and mail them all in one trip. But by the time I get home, write out the card, get the address, and mail it, I usually need a belated birthday card. I've never been able to get in the habit of carrying an address book around with me all the time. Magic Cap's name file lets you have all the addresses you need (even if it's only once a year) right there with you all the time, neatly tucked inside a communicator that you'll be carrying around with you anyway.

The name file gives you a place to list lots of names, addresses, and telephone numbers that you can look up whenever you need to use them. Magic Cap itself also uses the name file to hold the details of how to make electronic contact with people, companies, and services. When you address an electronic mail message or a fax to someone with your communicator, you choose the recipient from the name file.

Even without considering electronic mail, the name file is a versatile address book. It has room for the names and addresses of all the important people in your life, and it is powerful enough to replace your paper address books (it seems that many people have more than one). The name file doesn't have cute pictures of cats on it like a paper address book might, but Magic Cap's personalizing features probably make up for that.

The name file can have cards for people you work with, people your spouse works with who you might need to be in touch with, companies that you do business with, or information service providers that send you stuff via electronic mail. The name file is flexible enough to hold many addresses, telephone numbers, and electronic mail addresses on every card.

Communicating

Remember that the information in the name file isn't just for you; your communicator uses the name cards to know how to reach your contacts electronically. You can use your communicator to get in touch with just about anyone in the name file, whether that person has a Magic Cap communicator, a traditional electronic mail address, a fax machine, or just a telephone.

The name file can automatically dial telephone numbers for you. All you have to do is find the card for an associate, hold the communicator next to the telephone's mouthpiece, and then tap a telephone number on the name card. Magic Cap will dial the number for you (this trick doesn't work with all Magic Cap communicators and all telephones). Your child can borrow your communicator to write a thank-you note to his grandmother for a birthday present; because you convinced her to buy a communicator, the note gets delivered quickly with electronic mail.

Entering Name Cards

One of the first things you'll want to do with a new communicator is fill up the name file with the names of your co-workers, associates, and friends. After all, it's not a very personal communicator without them. Let's start by adding a name card for our co-worker Susan Anthony. To start entering her new card, tap the name file on the desk. The name file zooms open and shows you the card for the last contact you made. When you tap *new*, you'll get to choose whether to make a new card for a person, company, group, or information service (Figure 5-1).

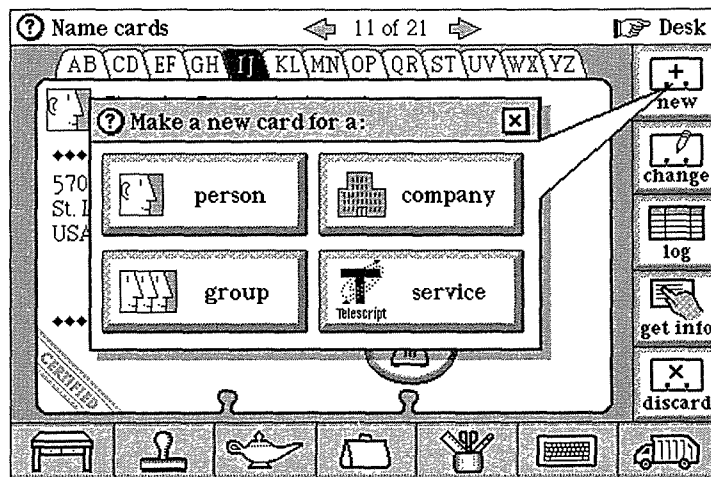


FIGURE 5-1. Window offers choices to make a new card

When you choose *person*, a blank card for the new person is made; a window to enter the first and last names appears; and the on-screen keyboard automatically opens, ready for you to start typing a first name. As you type the first two letters, *S u*, Magic Cap guesses that you're

typing *Susan* and automatically completes the word for you. Tap *last name*, then type *Anthony*, then *done* to finish entering the name.



Sweating the Details. Because entering names is one of the most common and tedious tasks you'll have to do with your communicator, Magic Cap's designers spent a lot of time making it as fast and convenient as possible. When you're entering a name card, lots of things happen to make your life easier. Some of them are so subtle that you may not even notice.

First, when you begin to enter the name, the box for *First name* already has a typing point in it, and the on-screen keyboard is already open, sparing you from having to worry about those two details. Next, you might notice that the keyboard is already shifted to uppercase for starting the name, the way most people like it. It'll switch back to lower case after typing the first letter.

Typing with the on-screen keyboard has been designed to feel comfortable and fast. The keys feel very responsive, working right away when you touch them and sounding a satisfying snap that suggests an electric typewriter.

Another touch for making typing convenient is automatic completion of words. As you might remember from Chapter 3, Magic Cap keeps lists of common words in various categories, such as first names, last names, cities, ZIP codes, and more. As you enter information, Magic Cap checks to see if your entry looks like one it already knows about. For example, as you enter the first two letters of *Susan*, Magic Cap sees that its list of first names contains only one entry that starts with *Su*, so it guesses *Susan* for the whole name but keeps the rest of the letters selected. If you're really typing *Sunshine* or

you don't even notice that it's guessed and you keep typing the rest of *Susan*, Magic Cap's guessing won't get in your way as you continue typing. The letters you type will simply replace the letters it guessed.

As you enter new words, they're automatically added to the list of words that Magic Cap will guess when you're typing. You can get finer control over the words by looking at the Word Lists book in the library. See Chapter 9 for more information on using the Word Lists book to customize your words.

In addition to automatically completing words, Magic Cap knows about a couple of other things that can speed up your entries. If you're entering someone's work address and Magic Cap already knows the address for the person's company, it will guess that address so you don't have to type it. If you enter a city that it's already seen, it will try to guess the ZIP code. If you enter a two-letter state abbreviation but forget to capitalize the second letter, it will remember for you. It tries very, very hard.

After you tap *done*, the new card appears with places to add work and home addresses, work phone, work fax, and home phone (see Figure 5-2). You can add even more stuff by opening the stamper and getting stamps for other addresses and phone numbers—the name file will even accommodate your gadget-happy friend who has ten different phones. We'll look into that a little later.

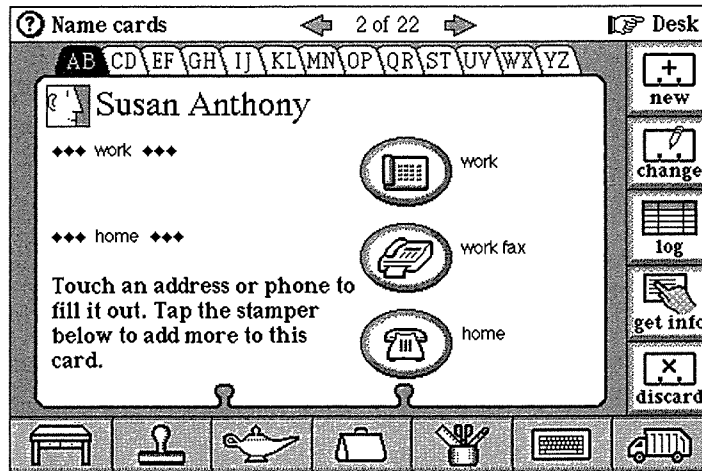


FIGURE 5-2. New name card awaiting more information

It Can Be Taught

In its efforts to help you enter information as quickly and easily as possible, Magic Cap picks up a few tricks as it goes along. Let's say we're now ready to enter Susan Anthony's husband, Mark. Tap *new*, then *person*, then start entering his name: first *M a r k*, then the last name. *A n* is as far as you get when Magic Cap guesses that the last name is going to be Anthony and fills in the rest for you. Of course, it learned the name Anthony from your previous entry and suggested it here to try to save typing. In this case, it guessed right, so you can just tap the *done* button and Mark Anthony is entered.

What if Magic Cap guesses wrong and suggests a word that's not what you want? You can find out by entering another person, Hans Anderson, to the name file. Tap *new*, and then *person* to start entering the new name

card. Typing *H a n s* is uneventful; this seems to be the first time Magic Cap has heard of that uncommon name. But when you type *A n* to start the last name, Magic Cap guesses Anthony, as usual (see Figure 5-3).

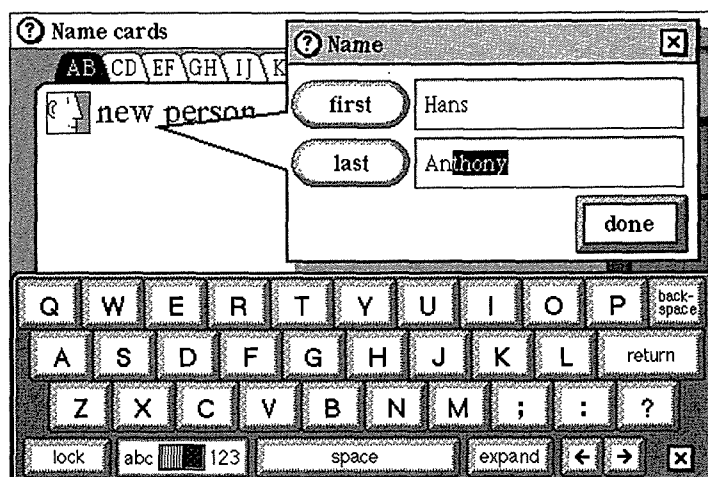


FIGURE 5-3. Magic Cap guesses for automatic completion

What do you have to do to correct this wrong guess? Nothing at all. As you type the next letter, the *d* appears and the rest of the wrong guess vanishes. You continue typing the rest of the name: *e r s o n*, then tap *done*. Magic Cap makes the new name card and learns a new last name, Anderson.

The next time you enter a last name that starts with *An*, Magic Cap won't make a guess at all. Instead, it will wait for the third letter of the last name. If you type a *t*, it will guess Anthony, and if you enter a *d*, it will guess Anderson. If you enter another letter entirely, it won't guess at all, but it will learn a new name for future guesses.

To make better guesses, Magic Cap keeps track of words in several different categories and only guesses

them where appropriate. For example, it will never guess a last name when a job title is requested. Magic Cap comes from the factory with lists of common words in such categories as cities, job titles, and names (first and last). As you enter more name cards, the lists build. You can see and change the lists by looking at the Word Lists book in the library; there's more about that in Chapter 9.

Adding Addresses and Telephone Numbers

When you enter a new person, Magic Cap provides spaces for home and work addresses and home, work, and work fax telephones. Now you can fill in some of those addresses and phone numbers on Hans Anderson's name card.

Tap *work address*; the keyboard appears automatically, as usual, and you can type in a job title, company name, address, city, state, and ZIP. As you type each item, Magic Cap is working to make it easier for you by shifting the keyboard to uppercase when it thinks that's the right thing to do, guessing the rest of words you're typing, and doing other tricks to try to reduce the amount of typing you have to do.

To enter a phone number, tap *work phone*. As the keyboard and the entry window appear, Magic Cap has already suggested an area code. If it's the wrong one, you can just replace it. When you enter phone numbers, you don't have to bother with niceties such as parentheses around the area code or a hyphen after the first three digits of the number; Magic Cap will format the number for you. You can even use the mnemonic letters that appear on a phone dial next to the numbers, so when you enter 555-RIBS, the phone number of your favorite source of charred mammal flesh, Magic Cap will understand that you mean 555-7427. Figure 5-4 shows Hans Anderson's card with his address and phone number entered.

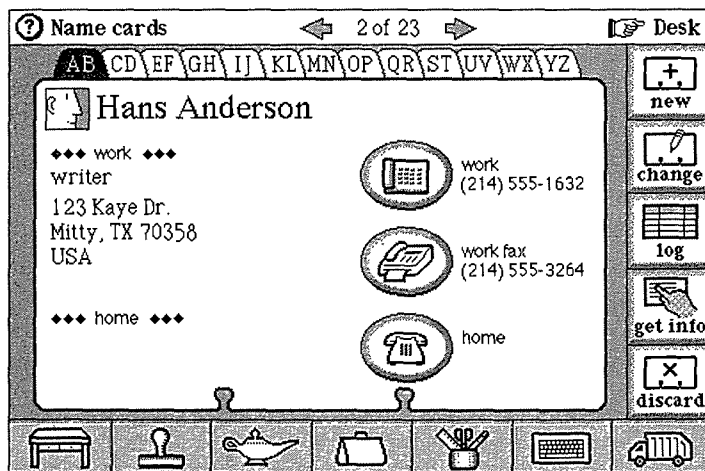



FIGURE 5-4. New name card with phone and address information

 **I Got You, Babe.** In working to make Magic Cap as useful and magical as possible, Magic Cap's designers tried to think of everything that people would do with their communicators. One interesting problem arose when entering name cards for people with only one name, like Madonna (and certainly she would be in lots of name files). Because these folks had no last name, they were filed incorrectly. The Magic Cap team noticed the problem and fixed it, and now single-name people are filed correctly—Madonna shows up after Chris MacAskill and before Julie Madsen, right where she belongs. The Magic Cap team referred to this problem as the Cher bug.

If you fill in all the addresses and phone numbers provided and you need some more, you can add them easily at any time. To add them, tap the stamper that's always on the bottom of the screen and you'll see lots of different kinds of phones and addresses to choose from, including fax number, pager, cellular phone, and work address (see Figure 5-5). If you want to add an address or phone that you don't see, you can use the *other* items and type in your own description.

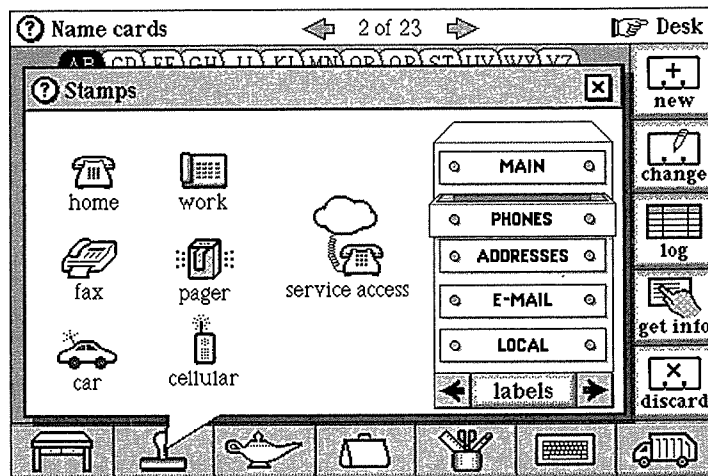


FIGURE 5-5. Phones drawer in the stamper shows additional choices

Each time you add an item from the stamp drawer, the image hops onto the card and zooms open for your typing. Let's say you want to add a cellular phone number for Hans Anderson. Tap the stamper, the *phones* drawer, and then the *cellular* stamp. The little phone leaps onto the name card, the keyboard appears, and a window opens

with a place for you to type the phone number. When you tap *done*, the window closes and the phone number has been added to Hans's card.

You can continue to add new people to your name file. The more names you enter, the smarter Magic Cap gets about guessing words. This works especially well when you're adding members of your family (you'll really love it if you have a long last name and lots of brothers and sisters).

Name Card Commands

Like many Magic Cap scenes, the right side of the screen is filled with buttons that activate various commands. We've already worked through the first (and most important) command, *new*. The rest are *change*, *log*, *get info*, and *discard*.

After you enter a name card's initial information, you know that it's only a matter of time before something changes, whether it's a new phone number or address, or a whole new last name for a friend who just got married. To avoid changing something inadvertently, you have to tap *change* before typing in a change to any information on a card that's already been completed. While you're changing information, the items on the name card are drawn with boxes around them (see Figure 5-6). When you're done changing a card, tap *change* to lock it up again. If you're just adding information, such as an additional phone number, you don't have to tap *change* first. While you're changing, you can also throw away any information on the name card.

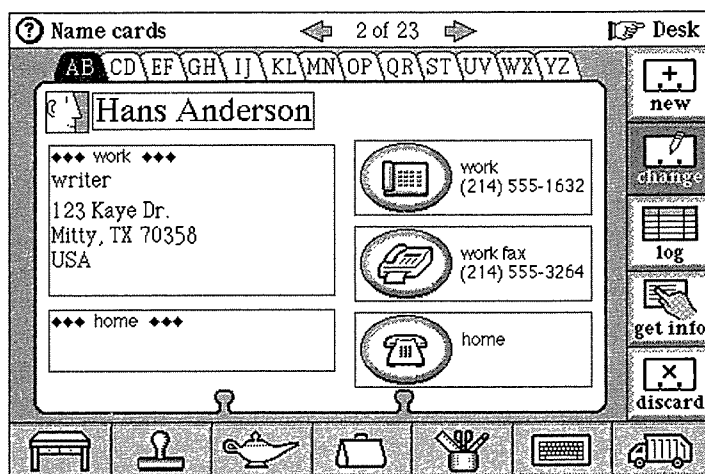



FIGURE 5-6. Making changes to a name card

 **Change for the Better.** You can also use *change* as a shortcut to save yourself some typing. You might want to have separate name cards for members of a couple (we'll call them Louis and Jennifer). Even though they have the same home phone number and address, they have separate birthdays. Start by making a name card for Louis, including home address and phone number. Tap *change*, slide the address to the tote bag, and then hold down the option key while sliding a copy back out of the tote bag and dropping it on Louis's card. The option key makes sure that there's still a copy of the address in the bag.

Next, move to Jennifer's card and slide the address out of the tote bag and onto her card. You can repeat this process for their phone number. This trick lets you type their home and work information only once, and then copy it to use on separate name cards.

The *log* button lets you see a record of communication with someone. When you tap *log*, you'll see a list of the times you contacted the person or company on that name card. The *get info* button asks PersonaLink to try to find a name in its directory. If PersonaLink succeeds in finding the person, you'll get a certified name card for your name file.

The last button on the right is *discard*, which lets you get rid of the name card you're looking at. After you confirm that that's what you really want to do, Magic Cap folds up the card and tosses it into the trash.

Looking at Name Cards

While browsing through the name file, you can see several different views of name cards, all designed to look like they came from printed address books. The name file highlights the alphabetical tab with the appropriate letter pair at the top of the screen (for Hans Anderson, that's *A B*). You can also see how many cards are in the file and which card is showing, as well as left and right arrows to move forward and backward through the file.

If you want to see the names you have for any two-letter tab, you can tap right on the tab to go to that index card. When you do, you'll see a lined index card with a list of names beginning with *A* and *B*, as in Figure 5-7. There's also a power user shortcut: If you hold down the option key while tapping on a tab, you skip past the index card for that tab and go straight to the first card of the second letter listed. If there are no entries for the second letter of the pair, you'll just see the index card. So, if you option-tap on the *A B* tab, you'll see the first name card that falls under *B*, if there is one.

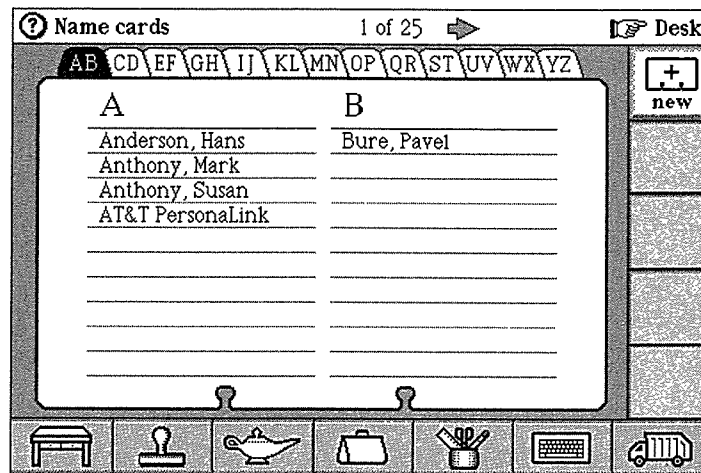


FIGURE 5-7. Index view of name cards

When you're looking at an index card, you can open any individual card by tapping it. When you do that, the card you chose will appear. When you're looking at a name card, you'll also notice that the appropriate tab is highlighted at the top of the screen. In fact, when you tap *new* to make a new card, the name file highlights the *OP* tab (for *new person*, of course) as the card appears. This works when entering a card for a company, too, putting the company card under *MN* (*new company*) until you type in the company's name.

Because you need to alphabetize both people and companies in the same file, the name file's alphabetizing rules work equally well for companies and people (even people who only have one name, like Cher and Dad).



Forgiving Users. As Magic Cap was being developed, potential users were brought in and asked to try out features while video cameras rolled. The development team then watched the tapes to see how well the user

interface worked. This experience proved amazingly revealing and often painful, as the developers' clever designs sometimes delighted, but sometimes missed the mark completely, causing the designers to start over and try again.

One interesting problem appeared when users were typing in the first and last name for a new person's name card. Often, users would type the first and last name on the same line in the entry window, separated by a space, even though that line clearly says *first name*. This created a horrible situation. Instead of a person whose first name is Sergei and last name is Makarov, you'd get someone with no last name at all, which would cause the card to be misfiled and mess up any addressing to that person. How could users be taught not to do this?

Somebody had a better idea. Instead of requiring users to be taught something, they taught Magic Cap to see if two names were entered in the *first name* box and none in the *last name* box. If so, Magic Cap assumes that the user just typed the whole name at once and takes what the user typed and divides it into first and last names. This feature is so subtle that most users who run into it will probably not even notice that it's happened, but they'd surely notice if it didn't work right.

Current Contact

Magic Cap remembers the name from the last name card you see and keeps track of it as the *current contact*. Right now, the current contact is our old friend Hans Anderson, because we're still looking at his card. When you close the name file to go to another scene, Magic Cap remembers the card you were looking at. The next time you want to send a message or a fax, or make a

telephone call, Magic Cap guesses that you may well want the current contact to get the message, as Figure 5-8 shows when a new message is created.

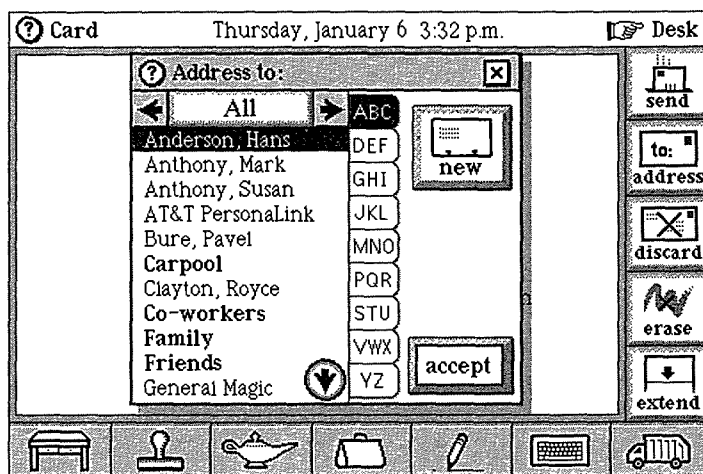


FIGURE 5-8. Current contact shown in new message name chooser

If you don't want to use the current contact, you can pick any name from the name chooser, of course, but Magic Cap guesses that since you just added the name to your file, or just got done looking at it, you might want to get in touch with that person. If you choose another name, the current contact changes to show your new preference. If you go to the datebook to schedule an appointment, the current contact is highlighted in your choices of *who*. The next time you open the name file, you'll see the last card you were looking at when you had the name file open, and that card then becomes the current contact.

Here's an example of how this current contact stuff can work even if you're not sending any messages. Let's say you get a phone call from Megan Marlowe, inviting

you to a meeting tomorrow to discuss a possible work project. You tap the datebook to schedule the meeting. Because you haven't entered Megan in your name file yet, you can do that without having to close the datebook, as shown in Figure 5-9. While looking at the name chooser in the datebook, you can tap *new* (the button even looks like a name card) to add her name to your file. Later, after you're off the phone and you look in the name file, it opens to her card so you can add her address and telephone information.

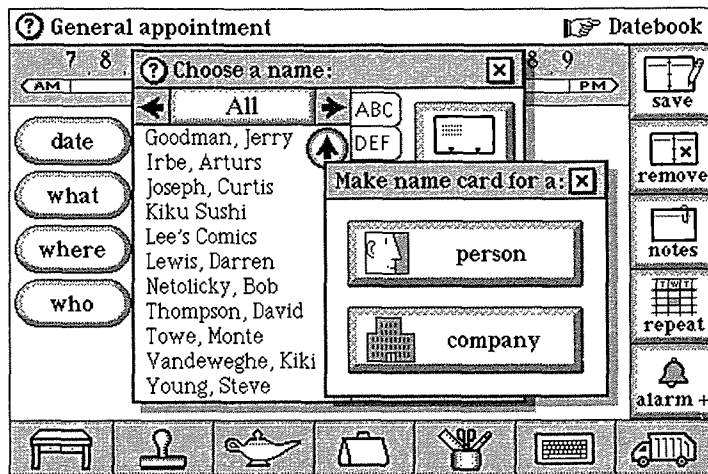


FIGURE 5-9. Adding new name cards from within the datebook

How You Get Name Cards

One way to add cards to your name file is to enter them directly, but that isn't the only way. As you've seen, you can enter new names while you're doing other things without having to stop and open the name file. For example, when you're in the datebook, you can schedule an appointment and create the name card while making

the appointment. The next time you open the name file, you'll find a name card for the person or company you just added.

Whenever you get a message from PersonaLink, you'll get a name card for the message's sender. If you already have a card for the sender, Magic Cap makes sure that you've got the latest information about the sender. This helps keep your name file up to date as you keep in touch with people who send messages to you. After all, this is a communicator, not a notepad.

When you start using Magic Cap, one of the first things you do is enter your own name card to personalize your communicator. After you type your name, Magic Cap adds a *certified* sash across the bottom-left corner (see Figure 5-10), which means the card has "official" information, that is, it's either about the person using the communicator (that's you) or it was obtained from some official source (usually PersonaLink).

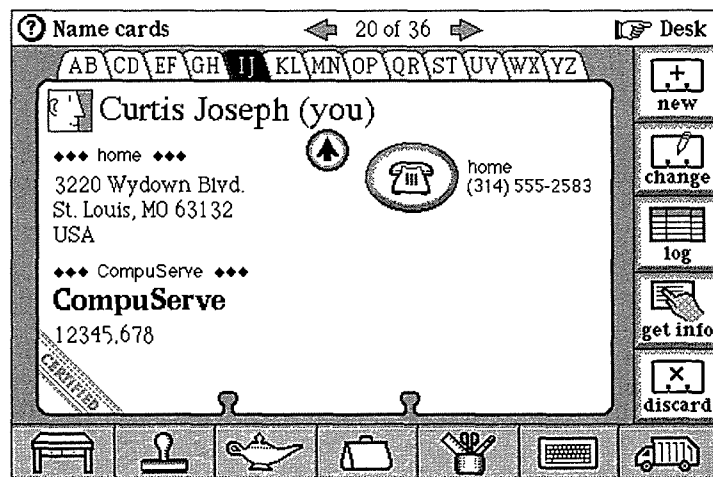


FIGURE 5-10. Certified name card for the owner of the communicator (you)

After you've typed in your name, you can go back any time and add other information, like addresses for home and work, all your phone numbers, and any electronic mail addresses you might have acquired. Then, whenever you send a message, your name card tags along and updates the recipient's name file with your latest information. Magic Cap knows that you've got the latest information about yourself, so it will replace outdated stuff in someone else's file with anything new and improved from you.

You can also add name cards for companies. In fact, you'll find a few company name cards already in your brand-new name file, giving you a way to get in touch with some of the companies that worked on making your communicator. Go ahead—give them a piece of your mind.

You can also add cards that represent Telescript-based services. These services will be few at first, but if General Magic succeeds in making its software platforms popular, many more services will spring up. You'll be able to join these services to get new kinds of information from distant places. Usually, you'll get a name card for every service that you join, but you may be able to get information from some services by manually entering a name card and then sending a message. Watch your in box (and the skies over downtown Magic Cap) for information on future services.

Electronic Mail Addresses

Earlier you saw that the stamper is filled with lots of choices for addresses and phone numbers, but you should look in the *E-mail* drawer to find the real magic of a personal communicator. In that drawer, pictured in Figure 5-11, you'll find logos for lots of traditional electronic mail

services, such as America Online, Prodigy, and MCI Mail. As you enter new name cards, you can add addresses for these traditional services, because not all your contacts will be available on Telescript-based services.

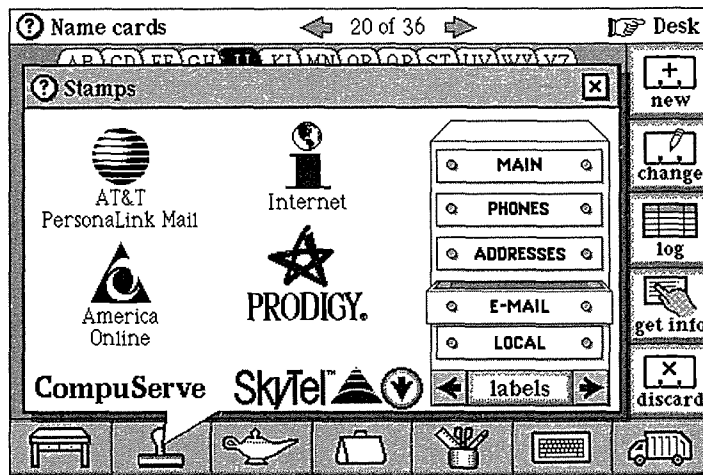


FIGURE 5-11. E-mail drawer in the stamper shows some information services

You can use these stamps to send messages to your contacts who have addresses on many different kinds of electronic mail. There are two ways to reach people using different services. First, your communicator has software packages that let you talk directly to some electronic mail services, including PersonaLink. Second, you can use these services to forward mail to other services by using *gateways*, special connections from one mail service to another.

Using gateways, you can get from PersonaLink to almost any other electronic mail service. For example, if an associate of yours has a CompuServe address, you could enter that address on your associate's name card. Then,

you could create a new message addressed to your associate. One of your delivery choices for the message would be *CompuServe via PersonaLink*, which means that your message would go to PersonaLink, then through a gateway to CompuServe. This is similar to the process that takes place when you get money from an automated teller machine that doesn't belong to your bank.

Of course, anyone who's not a member of a Telescript-based service doesn't get the full experience that a Magic Cap user gets. In other words, there's no room for animated kitty cats on traditional electronic mail. Messages usually get stripped down to just bare text when passing through gateways, but at least the essence gets transmitted. If you tell your friends and associates what they're missing, they'll have more incentive for getting a Magic Cap communicator.



Missing Link. There's a magical connection that lets most electronic mail services talk to each other: Internet. Almost every service, including PersonaLink, has a gateway to Internet, which is the sprawling worldwide web of computers that connects millions of users. Because most services can exchange mail through Internet, you can usually get mail from one place to another even if you and your recipient are on different services.

Groups

Magic Cap lets you associate groups of names together for easy communication. There are three built-in groups: Family, Friends, and Co-Workers, and you can add your own groups. After you've made a group, you can send a message to all the group's members at once,

as we did in Chapter 2. This message can use different ways to get to each member: Some may be available on PersonaLink, others might have traditional electronic mail, and some slackers will only have fax machines.

Let's say you want to create a group for your department, and you've already entered name cards for the people you want to have in the group. Start by tapping *new* and then *group*, and then type *The Department* as the group's name. When you tap *done*, you'll see Figure 5-12, an empty group name card.

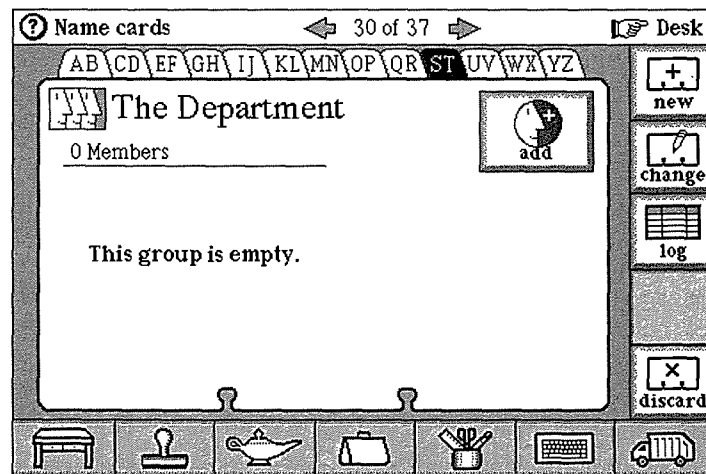


FIGURE 5-12. Name card for newly added group

You can add members to the group by tapping *add*. The familiar name chooser appears, and you can pick a new group member and tap *accept*. When you do, the name chooser closes and the name you picked is added to the group. To add another, just tap *add* again and choose your next name.

If you're adding a bunch of group members at once, you might find it tedious to keep picking them out one at a time. A shortcut can help you here. As you tap *accept* to add the new group member, hold down the option key. This trick adds the member to the group, but leaves the name chooser open, ready for you to add someone else.

You can also use the group card to remove unwanted members from your group. To do this, tap the member you want to remove, then tap *remove*, and that member is outta there. If you're looking at the group and you want to see a member's name card, you can tap the member's name to select it and then tap *look up* to go directly to the member's name card.

Stamped Groups

There's another way to manage groups. To demonstrate it, we'll create a group that's a mailing list for Halloween cards (once you have electronic mail, you find yourself sending messages for all sorts of occasions). First, you make the new group by tapping *new*, then *group*, then typing *My Pumpkin Friends*, then *done*. Now, let's do something a little different: Tap the stamper, open the *occasions* drawer, and then slide the pumpkin stamp out and onto the image in the upper-left corner of the group card. The image slurps into place and becomes attached to the group.

Now you can add your friends Susan Anthony and Hans Anderson to the group. Tap *add*, choose Susan, tap *accept*, and then repeat the process for Hans (Figure 5-13). Go take a look at Susan's name card. When you do, you'll find that the pumpkin image has automatically been added to her card as a reminder that she's in the Halloween group.

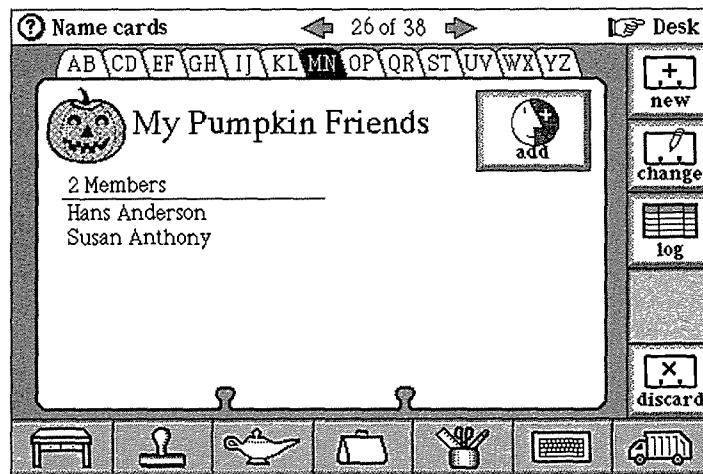



FIGURE 5-13. Name card for stamped group

This brings up an interesting question: What if you go to another name card and simply add a pumpkin stamp to the card? If you do just that, say on Mark Anthony's name card, and then go back to the *My Pumpkin Friends* group card, you'll see that you've added Mark Anthony to the group, just by dropping the appropriate stamp on his card. You can also remove members from the group the same way—just slide the group's stamp from the member's name card to the trash. So, when you associate a group with a stamp, you can add and remove group members simply by adding and removing the stamp.

 **Stamp of Approval.** Here's a quick trick to add a bunch of members to a stamped group. Slide a copy of the group's stamp to the tote bag. Then, as you visit the card of each prospective member of the group, hold down the option key while you slide out of the tote bag. This will pull out a copy of the stamp, which you can then drop on the name card to add that name to the group.

The built-in groups, Family, Friends, and Co-Workers, work just like any other stamped groups. The Co-Workers group has one magic trick that the others don't have, though. When you make a new name card for a person whose company name is the same as yours, Magic Cap will automatically add the person to the Co-Workers group. If you don't really want the person in your Co-Workers group, you can slide the Co-Workers stamp off the name card and into the trash.

Customizing and Other Tricks

You can use any of the goodies from the stamp drawer or magic hat to decorate name cards, including sounds, songs, and animations, which is much tougher to do in a printed address book. If you see an image that reminds you of someone in your name card file, you can drop it in the upper-left corner of that person's name card and your custom image will replace the generic one.

There are a few other shortcuts in the name file to make it easier to move around. In addition to using the left and right arrows at the top of the screen to flip between cards, you can use option-right arrow to move to the last card or option-left arrow to move to the first .

If your communicator is plugged into a phone line, you can dial by tapping on the phone number on a name card. Magic Cap dials the number for you and shows you how long the call is taking. As we noted earlier, if you don't have a phone line connected, you can use the tones that come out of the communicator's speaker to dial a phone through its handset, although that trick doesn't work on all phones.

The commands in the lamp provide a few more handy goodies for the name file. Use the *find* command to search for names or stamps on name cards, *file* to store

information on memory cards, and *print* to get your name cards on paper. When you file or print, you can choose all the cards in the file or just the one you're looking at.

Summary

The name file is a great address book, and it also helps you communicate with anyone listed there. You can add people, companies, and information services, and you can create groups for people who share something in common (such as family, friends, co-workers). There are lots of typing shortcuts built into the name file, since you'll spend a lot of time there entering information. Automatic completion of words is based on information that Magic Cap learns from previous entries, but it causes no harm if it guesses wrong. Automatic capitalization is another subtle trick that makes entering names easier.

A name card can handle people with single names (Cher), two first names with the same last name (Bill and Hillary Clinton), and business names (Upstairs Company) and it alphabetizes them correctly. If you enter a company name that your name file already knows, it will offer that company's work address and phone when you enter new employees of that company. Anything that's automatically entered shows up as selected text with the typing point properly placed, so you can type over it without extra work.

Stamps help you add additional information, letting you enter multiple addresses, phone numbers, fax numbers, and electronic mail addresses. The name file is available throughout Magic Cap, and you can access or add a new card while using the datebook, or phone, or stationery. The newest card you add, or the last one you viewed in the name file, becomes the current contact, and that

name is highlighted in the name chooser window in any other scene. That name card remains the current contact when you are in the datebook (whom to schedule an appointment with), when you are using the phone (whom to call), or when you are composing mail (whom to address the message to).

You can log any communication (phone or mail) you have with anyone in your name file. You can get in touch with people who have mail addresses on other networks just by having their electronic address on their name card. If you receive mail from a client who's recently added a new fax number, the client's card in your name file will be updated. Similarly, any mail you send will have your name card attached, so the recipients of your messages can also be sure to have the most up-to-date information in their name files.

Chapter 6



Phone

It's Your Call

Most of us started using telephones when we were very young, so our telephone habits were learned early in life, and they're very hard to change. When I was growing up, my mother only called people when she actually had something to tell them, and she didn't have much use for people who spent their time calling just to chat about stuff like what she made for dinner last night. My father, a doctor, obviously spent a lot of time on the telephone and later had a pager so that he could always be reached when he was on call.

These habits of my parents are probably where I get my feelings about the phone: Mostly, I find phones obtrusive and more often than not, I use an answering machine to screen my calls. Although I don't use the phone frivolously, I still have telephones around for convenience. I have phones in four different rooms at home, a separate line for my computer and even a portable cellular phone that moves from car to car. All these phones are around not necessarily because I want to be reachable anywhere, but because I want to be able to use the phone at my convenience; I want to be in touch. Although Magic Cap focuses on electronic mail, it also includes advanced, flexible telephone features, and these features can keep you in touch wherever you are.

Connected or Not

Tap the phone on the desk to use Magic Cap's phone features. The phone lets you do different sets of things depending on which of these three situations you're in:

1. Your communicator isn't connected to a telephone line.
2. Your communicator is connected to a telephone line.
3. Your communicator is connected to a telephone line and has a handset attached, or there's a telephone connected to the same line as your communicator.

In the first case, you can use the Magic Cap phone as an automatic dialer, playing touch tones into the mouthpiece of a standard telephone's handset to place a call. In the second case, you can use Magic Cap for one-way conversations where you only have to listen, not talk. The third case lets you use your communicator as a full telephone. We'll go over each of these three cases in this chapter's scenarios.

Keypad

You've probably already used the phone on the desk if you followed the *Getting Started* lessons. You went through a sequence of actions to tell your communicator your location to ensure that it dials properly, and you might even have made a phone call (see Chapter 1 if you need to refresh your memory). Since that might have been one of the very first things you did when you got your communicator, you may not have explored it fully. Let's go back to the phone and explain all the buttons and actions in some depth.

When you touch the phone, it opens to a scene that has a telephone keypad, buttons to help with dialing and

saving phone numbers, a panel of blank speed-dial buttons, and a new set of buttons down the right side of the screen that controls your phone (see Figure 6-1). This is the keypad scene, and the top button on the right side, called *keypad*, is highlighted.

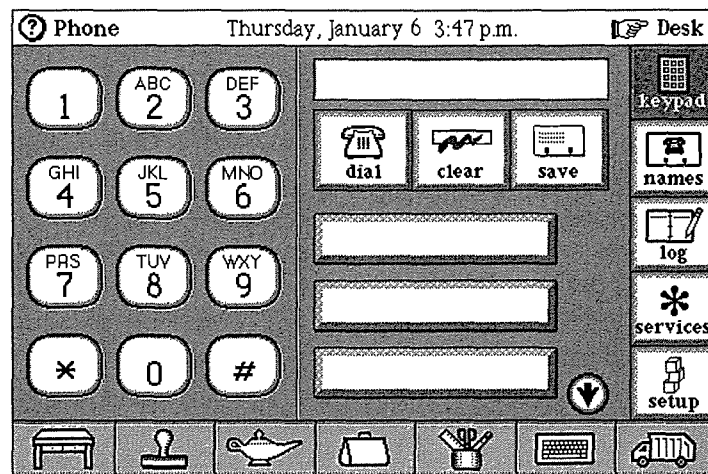


FIGURE 6-1. The phone's keypad

The most obvious thing to do with the phone is to make a call, as you did during the *Getting Started* lessons. Dialing the phone is easy—just press the digits on the keypad, and then touch *dial*. For flexibility, it also works the other way—touch *dial* to take the phone off the hook, and then dial the number.

If you have a communicator with an optional telephone handset, or there's a telephone connected to the same phone line, you can complete a call that lets you speak to the other party. Without the handset, you can hold the communicator up to the mouthpiece of a telephone so the tones will play and auto-dial the phone, although the

communicator volume must be loud and the phone you're using has to be responsive to auto-dialing.

Let's say you dial directory assistance to get the telephone number of a popular new Italian restaurant. Instead of trying to find a pen and paper to jot the number down, or even using the notebook in your communicator and then having to manually dial the number, you can use the phone to cut out several steps. Tap the number on the keypad while you're getting it from directory assistance. There's a display that shows you the number as you're entering it, and it even formats it for you as you're typing it, properly separating the area code and prefix. Because you're entering it directly into Magic Cap's phone, you can then dial the number just by touching *dial*.

If you want to keep the number to use later, you can touch *save*, the button that looks like a little name card, and you'll get to make a new name card. You can choose to make the new card for a person or company. When you type in the name, you'll see that Magic Cap has already entered the phone number for you. The next time you feel like having lasagna, you'll have the name card of the restaurant and its phone number.

You might notice that *dial* changes to *hang up* after the call is made, so if the only available dinner reservations are before 4 or after 11, you can hang up and call somewhere else. After you hang up, that versatile button then becomes *redial* until you enter a new telephone number.

Whenever you dial a number, Magic Cap makes a *Phone status* window that tells you what number was dialed, and it lets you adjust the volume, see the log of phone calls, or hang up. You're not stuck in the phone scene after you've started a call; you can do other

actions, like opening the datebook to make an appointment or opening the call's log entry to take notes, all while the phone call continues. When you switch scenes, the *Phone status* window closes, but you can see it again by touching the tiny telephone handset that appears at the top of the screen while the call continues. When you're ready to end the call, you can touch *hang up* in the *Phone status* window.

You can also dial by touching any one of the nine speed-dial buttons that you can program for your most frequent calls. To set a speed-dial button, touch the blank button to open it. You'll see a window (shown in Figure 6-2) that lets you label the button and offers you a couple of choices for how to enter the number.

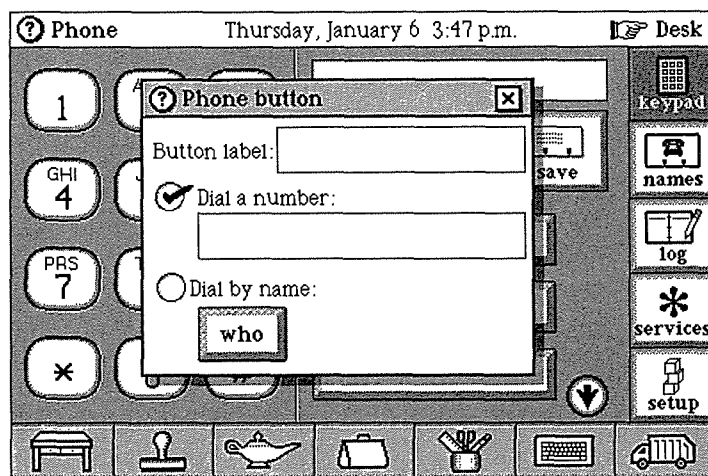


FIGURE 6-2. This window helps you program speed-dial buttons

The *Dial by name* choice is probably the most common one for most numbers. If you're a concerned parent as well as a full-time employee, you may want to have the

first button set for your home so you can check in with the nanny and make sure the little ones ate their lunches and took their naps. You can name the new button *Home*, check the *Dial by name* option, then touch *who* below it, which will produce a name chooser window that has a list of names. As in all name chooser windows, you can pick from the list, or you can touch *new* to make a new card and enter a name and phone number right then and there.

As you select the name you want (in this case, it will be your own), you will also see the phone number displayed above the *accept* button, as shown in Figure 6-3. If Magic Cap knows more than one phone number for the name shown, the numbers will appear in a choice box so you can scroll through and choose the right one. When you have the right combination of name and number, you can touch *accept* and you'll have a speed-dial button that will help you keep in touch with your kids, whether you're calling from your office or from a business trip in Boston.

You can also dial a number manually to set a speed-dial button. As soon as you touch the *Dial a number* box, the keyboard opens, already set to type numbers, and the typing point is placed so you can enter the number. When you're done entering the number, touch the x in the window to close it and the keyboard, and you'll see your new speed dial button set.

This is a handy feature for numbers you may call frequently that don't really have a person or company connected to them. An obvious speed-dial button you would set this way is 911, because it's unlikely you would want to enter a name card for that number. You might also want to set a speed-dial button this way for the local time and temperature phone numbers, or for a road conditions recording if you travel a lot.

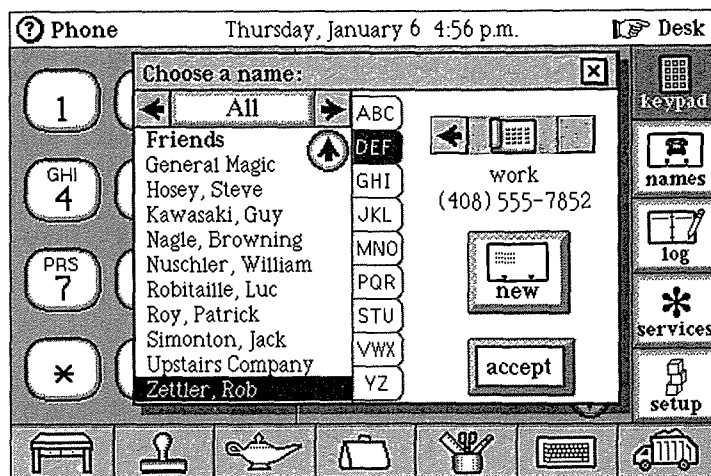


FIGURE 6-3. Speed dial's name chooser includes phone numbers

Once a speed-dial button is set, one simple touch on the button dials the number. You can reprogram a button by holding down the option key when you touch it. This tells Magic Cap that you want to change the button rather than use it to dial. Option-touching a programmed speed-dial button opens the window shown in Figure 6-2 to help you enter the new number.

Here's another Magic Cap shortcut: You don't have to label the button when you program it. If you touch a name card to get the phone number, and then close the window, the speed-dial button automatically takes its name from the name card, with the phone number's description added in parentheses. Similarly, if you program a button with just a number, the number will be used as the button's label.

Seeing Names and Logs

Because the phone is closely related to the name file, it makes sense to give you a quick way to view the name card list with all the phone numbers. When you touch *names*, a button with a picture of a telephone on a card, you see the familiar name chooser, but with one obvious difference: When you select a name from the list, you see all the phone numbers entered on that card with their accompanying stamps, as shown in Figure 6-4.

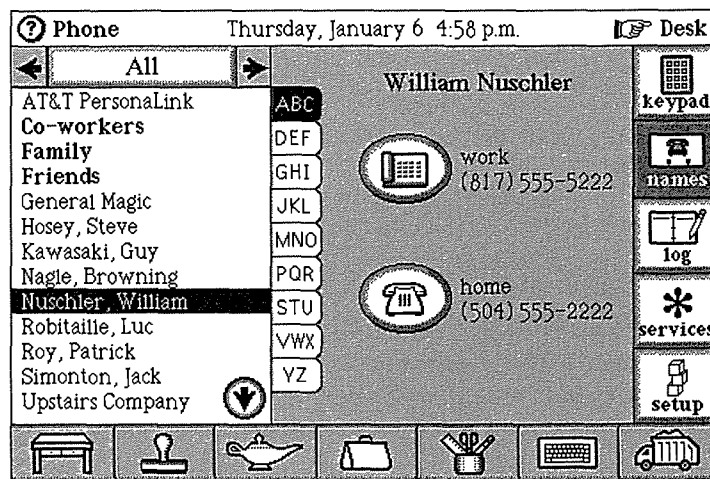


FIGURE 6-4. The phone's display of names and numbers

You can't open an individual name card from this index, but you can touch any phone stamp next to a name to dial the number. You might remember from Chapter 5 that when you're looking at a name card, you can touch the telephone stamps right on the name card to start dialing. This is exactly the same feature, except that you don't have to open the name file first.

The *log* button shows an index of the phone calls made from your communicator, as pictured in Figure 6-5. This can be especially useful if you contact a lot of clients by phone and you need to keep track of the calls for billing purposes. In fact, the first two rules that you can set for the phone involve logs. There's a rule that determines whether to make a log entry for each call, and another rule that automatically throws away old log entries after a certain amount of time has passed.

| ? Phone | | Friday, January 7 5:34 p.m. | | Desk |
|---------------|-----------|-----------------------------|----------------|----------|
| Who | When | Time | About | |
| Michael Stipe | 1:03 p.m. | 0:12:26 | document | keypad |
| Bill Berry | 9:33 a.m. | 0:07:23 | reconstruction | names |
| 415 555-5364 | 9:31 a.m. | 0:02:09 | 10 at 10 | log |
| Peter Buck | 9:30 a.m. | 0:00:22 | | services |
| Mike Mills | 9:27 a.m. | 0:02:30 | Rockville | setup |
| Kate Pierson | yesterday | 0:00:13 | | |
| Kate Pierson | yesterday | 0:00:06 | | |
| 404 555-2527 | yesterday | 0:01:53 | strobe light | |

FIGURE 6-5. The log of phone calls

The calls are logged according to the name or number called, depending on whether the number was chosen from the name card list or entered manually, and the log includes the time and duration of the phone call. If you look at the log on the same day you've made a call, you'll see the time that the call was started. If you look at the log anytime after that day, it will show the date of the call instead, making the log a much more useful tracking tool.

If you touch an individual entry in the log index, it zooms open to allow you to see the description of the call you wrote earlier, along with any notes you might have added regarding the call. You can also use stamps to mark the log entry, which you can then search for later. In consistent Magic Cap fashion, there are left and right arrows at the top of the log entry so you can move back and forth with ease. You'll get a log entry any time you call a number, whether you dial with the speed dial buttons, a manually entered number, or from a name in any of the phone's name lists. You'll also get a log entry when you dial a number by touching a phone number stamp while you're looking at a card in the name file.

You can have Magic Cap file your log entries in a folder in the file cabinet. You can file a log entry using the *file* command from the lamp, or by sliding an individual entry to the tote bag, opening the file cabinet, and then sliding the entry into a folder. If you spend a lot of your time on the phone with clients, you now have an easy way to track those minutes for billing purposes, because everyone wants a log.

Phone Rules

Let's explore the phone rules a little more. Besides the rules about making and keeping log entries, there are two rules about dealing with an incoming phone call. You probably won't be getting any phone calls unless you have a handset and a phone line connected to your communicator. (If you get an incoming phone call and you don't have a phone line connected, you're witnessing either a miracle or a spectacular bug.) You can set the rule that displays an announcement when any call is received, and you can choose a custom ring sound to alert you when a phone call comes in.

The last rule is vital for making sure that you can use your communicator in more than one place. As you take your communicator around, you'll connect and unplug it at different locations you've listed, which you might remember from the *Getting Started* lesson about telling the communicator where you're dialing from. The communicator needs to know where you are in order to dial calls correctly; if you call outside a number's area code, it must dial the area code first, of course.

To make this work properly, you'd have to remember to reset your location every time you connected in a different place. Of course, most people would forget to do this most of the time. To make this work better, a couple of dedicated engineers at General Magic thought it would be a good idea if the communicator could detect when you plugged into a phone line, so that whenever you connected, it would "feel the tickle" and remind you to tell it your location. After a lengthy struggle with communicator hardware, they were finally able to make this important feature work. If you leave the rule at its factory setting, the communicator will ask you to confirm your location each time it feels the tickle of a telephone line being plugged in. Figure 6-6 shows the message you see when you connect.

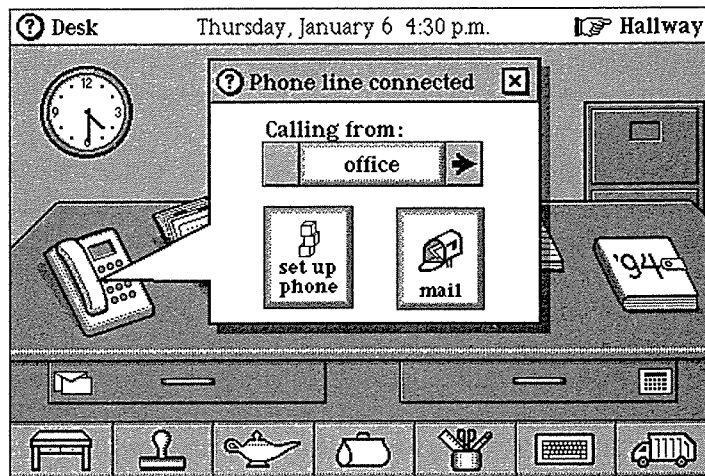


FIGURE 6-6. This window appears when you connect a phone line

Services

As we continue down the right side of the phone's screen, we see that the next button is called *services*. This takes you to a page that's intended to hold buttons to help you communicate with automated telephone response systems, such as voice mail, bank account balance inquiries, or stock quotations. However, the only thing you'll see when you touch this button is a page of text describing what you may be able to do soon with some third-party help.

Although Magic Cap doesn't come with any of these services built in, you can imagine what some of them might be. If your office has a voice-mail system, you know that you must press a specific sequence of numbers on the telephone keypad to get your messages. Because the communicator can play telephone tones, the communicator could be programmed to play the right tones when

you touch buttons; for example, you could use a button labeled *save* instead of having to remember to dial 76 (or was it 67?) to save a message. A savvy phone company might offer you Magic Cap controls for your voice mail system on a memory card. You could insert the memory card and the page of commands would hop onto the *services* page.

You might also see information providers from downtown Magic Cap offer you similar services. If you're registered with the stock broker downtown, it might offer you a custom-programmed page of buttons that would step through the tones necessary to check on your personal portfolio of stocks using an automated attendant system. Finally, you'd have a release from the tyranny of those annoying automated instructions to "press 47, followed by the star key now."

Setting Up

The phone's *setup* page is one of the least intuitive places in Magic Cap, in part because it's not a metaphor for anything you have to do in real life. The main purpose of the phone setup is to tell the communicator where you are, something you usually don't have to do when you use a telephone. You've already seen the phone setup as part of the *Getting Started* lessons, but if you're like many of us, you may have followed the lesson without fully understanding what was happening and why.

You'll probably take your communicator to work, bring it home, take it with you on a business trip, and then use it at the airport when your flight is delayed. Because you work in different locations, your communicator needs to know where it is in case you ask it to dial the phone to collect your mail, call someone, or send a fax. You wouldn't have to do any of this if the telephone network allowed

you to dial the area code you're in when you're calling another number in the same area. That way, you could always dial exactly the same digits to reach a number, no matter where you were (as long as you stayed in the same country).

Because the telephone network doesn't let you dial the area code when calling a local number, Magic Cap needs to know the area code you're in so it can dial out properly. That's why the gentle reminder to confirm where you are is so important.

When you first touch *setup*, you see the setup screen (shown in Figure 6-7) with the calling location you added in the *Getting Started* lessons. When you touch the stamper, a special phone drawer is opened, and you can choose from the images for home, work, and hotel. If you depend on your communicator at work, that might be the next stamp you add. After you touch the stamp for work, it hops onto the screen, a window for entering information opens, and the keyboard appears.

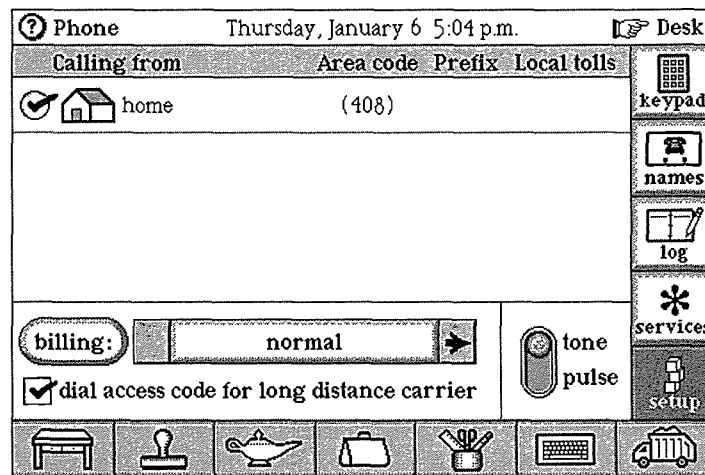


FIGURE 6-7. The phone's setup screen

The first step in the sequence is to type in the name of the location you're calling from. It already says *work*, but you can personalize it even more if you want, so we'll change it to say *office*. Next is a choice box with the country you're calling from, and as much as you'd like it to be Jamaica, you'll just keep it set to United States. Magic Cap then needs to know the area code you're calling from. Because many people live and work in different area codes these days, this may be the major difference between using your communicator at the office and at home.

Finally, you must tell your communicator if it needs to dial a prefix to get an outside line before calling. After you're done entering the location, you now have two items with check boxes that tell your communicator where it's calling from, as shown in Figure 6-8. When you connect to a phone line and the communicator asks you to confirm your location, you'll choose from these two options.

| Phone | | Thursday, January 6 5:03 p.m. | | Desk |
|--|-----------|-------------------------------|-------------|----------|
| Calling from | Area code | Prefix | Local tolls | |
| <input type="radio"/> home | (408) | | | keypad |
| <input checked="" type="radio"/> office | (415) | 9 | | names |
| | | | | log |
| | | | | services |
| billing: <input type="text" value="normal"/> | | tone pulse | | setup |
| <input checked="" type="checkbox"/> dial access code for long distance carrier | | | | |

FIGURE 6-8. Phone setup now has *home* and *work* options

The stamper has choices for *work*, *home*, and *hotel*, but because you can enter a new name for these location stamps, you can make as many as you need. If you ever need to take your communicator on a trip to the Netherlands, you can add a new stamp for *hotel*, and set the country when you fill out the location. Magic Cap already knows most international calling codes, so when you connect to a phone line in Eindhoven, every one of your numbers on name cards will use the right country code as a prefix for dialing numbers back in the United States.

The other important part of the phone setup is the *billing* button, which is where you tell the phone how you're paying for long-distance calls. It comes factory set to *normal*, which means that no credit card information will be entered and the call will just be billed automatically to the caller (or the recipient, if you call collect). If you use a calling card or other credit card to keep track of phone expenses, or you want to connect to your own long-distance carrier, the *billing* button can help you.

You can tap *billing* to specify how to direct the charges, and there is also a place to enter a long-distance access code, if you want your communicator to dial that before calling a number. Magic Cap comes with several choices for credit cards, and you can change or remove them, or add your own to personalize your communicator to match your billing needs.

Magic Cap also comes with the access code to AT&T's long-distance service built in, but it can be easily changed to access any other carrier instead. There's a check box for telling the phone whether to dial the access code for your long distance carrier. If you've entered a new access code or just left it at the factory-set code for AT&T, it makes sense to keep this feature on. It doesn't get in the way of any dialing, and if you need it, you won't have to

remember to keep turning it on and off. Finally, there's a switch to choose between tone and old-fashioned pulse dialing, if you happen to be plugged into a phone system that doesn't understand touch tones.

Summary

The main scene in the phone is the keypad. You can manually enter a phone number from the keypad and if you're connected to a phone line, the communicator can dial the phone for you with the touch of a button. The phone can help you even if you're not connected to a phone line or you don't have a telephone handset attached. Some models of communicators can be held up to the mouthpieces of some telephones, and the communicator can be used as an automatic dialer to initiate a call.

You can set numbers for nine speed-dial buttons, and you can either type in a number for these buttons or use the information already entered on your name cards to program them. The phone will let you save any manually entered number as a name card, without having to go to the name file. The names button is a fast way to see an index of all the names in your name file, and it displays the phone numbers entered on each card. You can quickly find any name and phone number this way, and you can start a call just by touching the stamp next to each number. When you make a call, you can move to another scene as the call continues.

The phone keeps a log of every phone call you make with your communicator, whether you touch a speed-dial button, manually enter a phone number, touch a phone number stamp from the names button in the phone, or touch a stamp in the name file itself. There is a log

made of the time the call was made, who or what number was called, how long it lasted, and any notes you might have made.

The phone has rules about automatically making log entries and how long to keep them before discarding them. You can file individual log entries in your file cabinet if you need to keep them longer than the rule's limit.

You can look forward to filling the *services* scene with controls from software developers and information providers. These services help you use your communicator to control automated systems like voice mail, stock quotes, or ticket ordering.

The phone setup is one of the first things you do when you're personalizing your communicator, and you find out why it's important to tell your communicator where you're calling from. You can add entries for the different places you might be when you plug your communicator into a phone line. The phone knows how to dial from different area codes and countries, and it even includes long-distance carrier access codes and calling card numbers if you want. All you need to do is tell your communicator where it is, and it takes care of the rest.

Chapter 7



Notebook

A Blank Book

When I started to write about Magic Cap, I had very little trouble writing about the datebook, because I could easily see how to use Magic Cap instead of a printed calendar. The electronic mail information was a little intimidating because there was so much to tell, but the features there are well-designed and very important to Magic Cap. I was comfortable with the name file, easily understanding how Magic Cap's features could meet my needs for an address book, especially one that helps so much in actually communicating with people.

In contrast, I was really skeptical when I started to think about how I might use the notebook. I don't carry around a notebook, and I really couldn't see why I'd want to have one right on my Magic Cap desk. I was sure I'd have a hard time writing about something I didn't need or want to use. After a little experimenting, though, I saw how I might use it, and I found that I liked using it. I love getting nice surprises like that.

Writing it Down

When you open the notebook, you see a blank sheet of paper (see Figure 7-1) and all the promise that it brings. You can use this page just as you would use plain paper, with a few differences because the page only exists inside a personal communicator and not in the physical world.

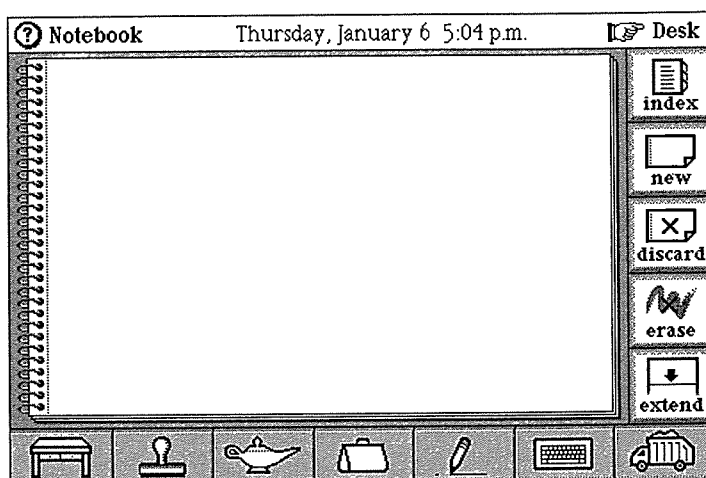


FIGURE 7-1. Blank notebook page

Magic Cap assumes that you'll want to write your notes rather than type them, so it automatically selects the pencil tool for you to write with. You can see which tool you're using by looking at the tool holder's space along the bottom of the screen. If you'd rather type, you can tap the keyboard to open the on-screen keyboard, but on plain paper, you'll probably want to jot down a quick note or make a simple drawing with your stylus.

You can choose pencils that have different points for writing, depending on how thick you want your lines. If

you're writing a note to remind yourself that you have to call your associate in Salt Lake City, you'll probably be content with the thin pencil that the notebook gives you when the new page opens. If you're drawing a map to your house for friends who are coming to dinner from out of town, you might use a thicker pencil to show busier roads.

Bucking a trend in personal electronics, Magic Cap communicators don't try to recognize your handwriting. The notebook doesn't try to interpret the notes you write, instead giving you this choice: You can use a pencil to convey the personal touch of handwriting, or you can use the keyboard to type in text that must be entered with precision. Using a stylus to write on a screen takes some practice—you have to get used to how fast to move and how hard to press. Once you get the hang of it, though, it's fun and easy.

Mapping the Way

Many folks are not great artists and admit to needing all the help they can get with drawing. Magic Cap provides tools for shapes and lines that will help your drawings look much more presentable. Let's assume you want to draw a map to your office for a visiting customer. Start by drawing the lines for Highway 280 and Highway 17 freehand with a pencil, since roads are rarely straight lines anyway. You can tap the tool holder to see a list of choices that includes lines and shapes; you'll use these shortly to help draw other things.

As you start drawing the freeway exit that leads to your office, you discover a mistake: You didn't leave enough room to write the details of finding your office after getting off the freeway. This gives you a chance to try out the erase button on the right side of the screen.

Tapping *erase* removes the marks on the page one at a time, the most recently added ones first. You can also pick the eraser from among the pencil tools and use your stylus or finger like an eraser, rubbing over just the marks you want to erase (see Figure 7-2).

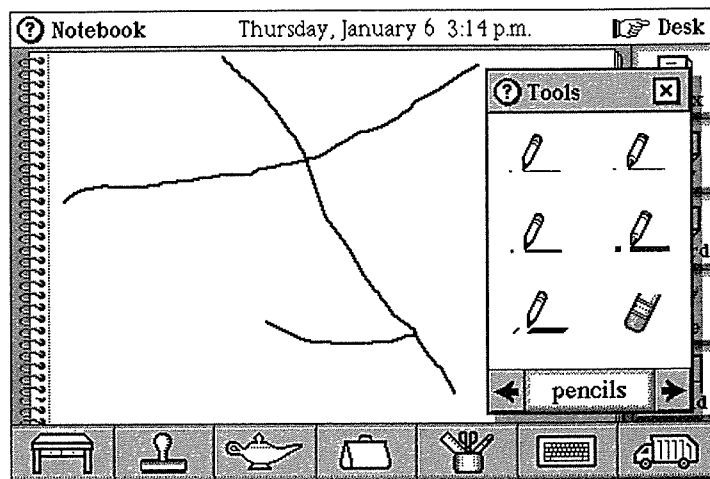


FIGURE 7-2. Part of the map

If you want, you can skip the erasing and tap *discard* to get rid of this page and start over again (and again, and again, if you want). The *discard* command tears the page out of the notebook and throws it away. To begin again, you can tap *new* and then choose the kind of paper for the new page from among the notebook's six kinds of stationery (see Figure 7-3).

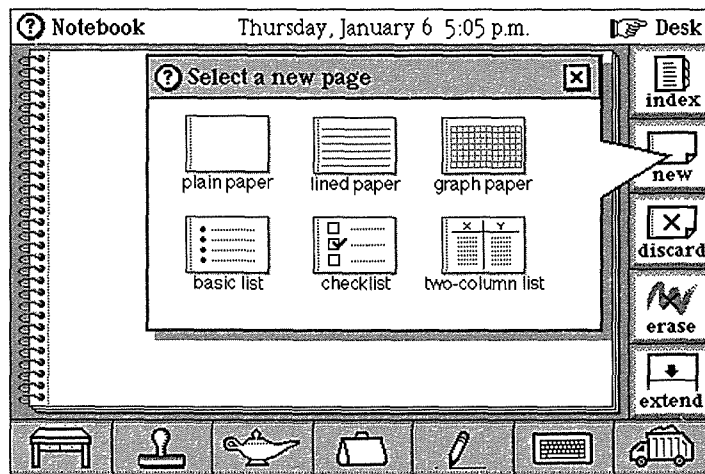


FIGURE 7-3. Notebook paper choices

If you're working on a notebook page and you want to start over, you don't have to throw the page away or pick a new one. Here's the shortcut for starting over on a page: If you hold down the option key and tap *erase*, all the pencil marks, stamps, lines, and other customizations on the page are erased at once, leaving you once again with a blank piece of paper. Think of it as recycling.

Lines and Words

If you're one of the many who can't draw a straight line unless you're using a ruler (and with me, even that's no guarantee), you'll appreciate Magic Cap's line tools. These tools let you draw straight lines of different thicknesses and at various angles. They're represented by lines in the tool holder, which you can use to draw both horizontal and vertical lines. These tools are different from the pencils because they guarantee straight lines. In fact,

there are line tools that produce two different kinds of lines: lines that can be drawn at any angle, and lines that always snap to the nearest 90-degree angle. You can see the line tools in Figure 7-4.

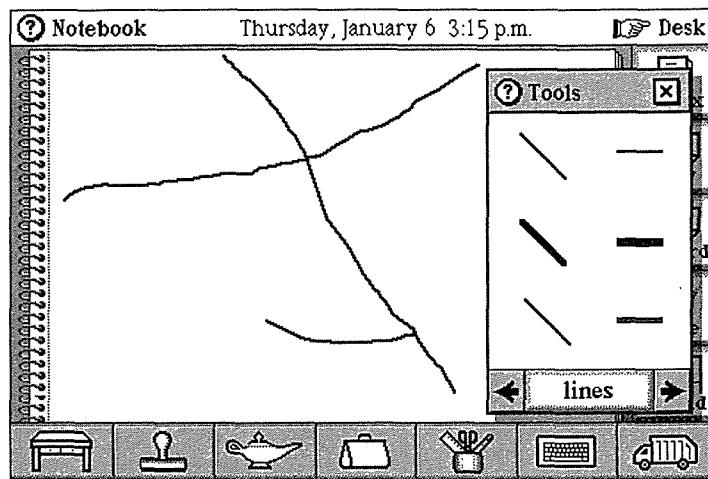


FIGURE 7-4. Line tools

After working on the map awhile, you might realize that you'd also like to give the directions in words; even with Magic Cap, drawing a legible map is kind of a chore. Now you can use the text tools in the tool holder to make a new text field on the notebook page. Tap the tool holder and then go to the text tools. Tap *plain* to make a text field that will show its contents with a simple line around it as a border, which will be ideal for the map's title.

When you choose the plain text field, its image hops into the tool holder, showing the tool that you're using. Then, when you tap the page, a new field is placed at the touch point, the keyboard opens, and you can start typing. When you're done, you can tap the tool holder again to make another text field. This time, you can pick

transparent, which will show the typed text with no border at all. Transparent text fields are ideal for written directions on a map that can blend in with the drawing.

If you want to dress up the map you're making, you might go for a *fancy* text field, which borders its words with a more elegant frame. Every time you pick a different text field, the text choices window disappears and the image hops into the tool holder. If you want to go back and edit your text after entering it, just tap directly on the words. As shown in Figure 7-5, Magic Cap opens the text selection window, which lets you retype the text, change its style, copy it, or remove it.

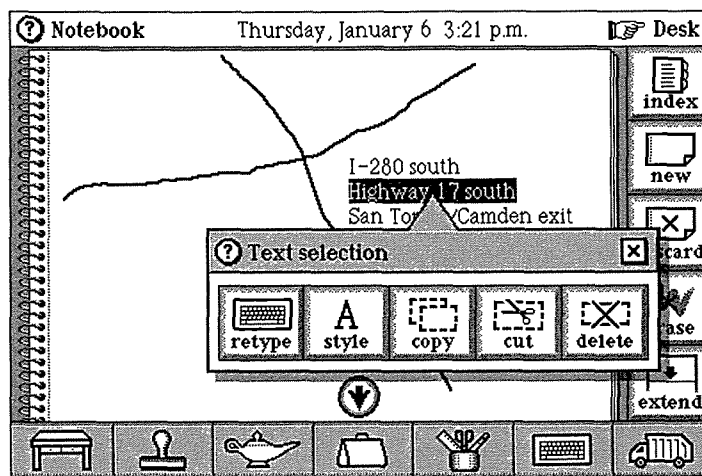


FIGURE 7-5. Selected text opens window

Arrangements

The last set of tools is for arranging items on a page. If you tap *move*, which has an image that looks something like a moving van without wheels but on closer inspection is a box with motion lines, you can slide any

object that you've placed on the page. This comes in handy when you realize the written directions cover up part of the freeway on the map.

With the *copy* tool, you can duplicate any item and slide it to another position on the page or stash it in the tote bag to put on another page. This can be useful for reusing pieces of a drawing, such as changing a map for someone coming from a different direction. You can even copy the text and then change it to make it applicable to the other person's starting point. You can use *stretch* to reshape any item you've put on the page, so when you figure out that the exit off the freeway doesn't really look the way you drew it, you can make the line longer or shorter instead of drawing it again.



Evolution. In the early development of Magic Cap, the *move*, *copy*, and *stretch* tools were far more important. Before there was a notebook or datebook, Magic Cap's features consisted mainly of construction tools for assembling components. In that world, the arranging tools were vital. In fact, they were so important that at one point, each of those tools was assigned its own physical key on the communicator's case. Eventually, the designers found a way to avoid the need for the arranging tools most of the time, and the dedicated keys were removed.

Send It Off

After you've created a notebook page by drawing or writing, you can actually use what you've made to communicate with someone. As long as you've gone to the trouble to draw the map, you should probably make sure

your guest sees it before coming. With just a touch of the lamp, you can send the message by tapping *fax*—there goes one excuse for arriving late at your office (see Figure 7-6).

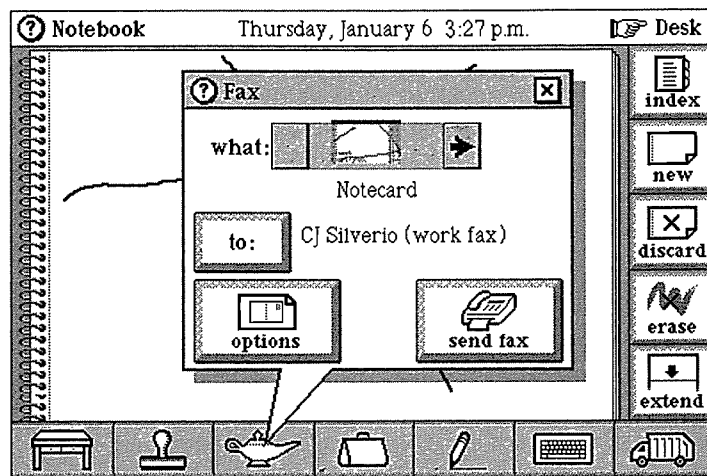


FIGURE 7-6. Preparing to fax the map

You can choose to send the page you just made or a picture of the whole screen, and the *options* button lets you decide whether to include a cover page and whether to have the fax come out sideways. You can tap *to:* and pick the addressee you want with the name chooser. If your addressee is already in the list, you can choose the name, and then tap *send fax* to get the fax going. If you haven't yet added the addressee to the name file, you can add a new name card and fill in the fax number in the same step, without ever leaving the notebook. (Of course, when you open the name file, the name you just entered will appear because it's the current contact, and you'll be able to add addresses and more phone numbers if you want to, but you probably already guessed that).

By faxing the map right away, your customer can get it before leaving work, and then you won't have to worry about playing telephone tag or dictating the directions over the phone.

A Birthday Card

A blank piece of paper just begs for a child's creativity. Let's say you're helping Jess, an imaginative eight-year-old who wants to use your communicator to make a birthday card for his grandmother (coincidentally, she just got her first personal communicator, too). The *shapes* tools in the tool holder let him pick from various kinds of shapes, including circles, stars, rectangles, and more (see Figure 7-7).

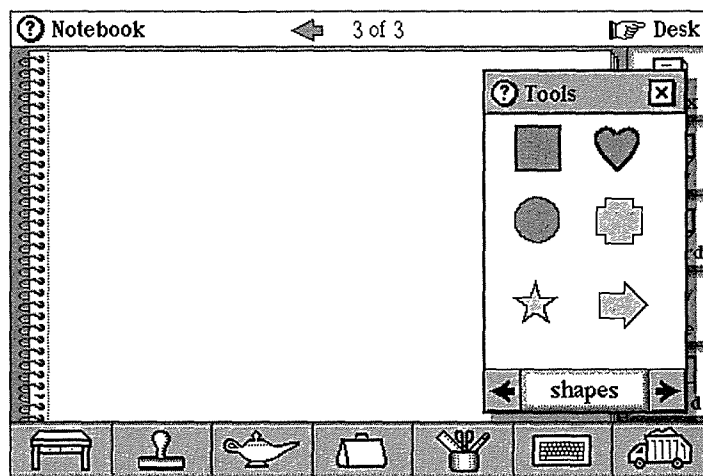


FIGURE 7-7. Some of Magic Cap's shapes

After selecting the heart by tapping it, the heart shape hops into the tool holder, and you can show your young artist how to draw hearts all over the page. We'll stick with hearts right now, but by option-tapping the heart in the tool holder spot, the shapes choices reappear, making the other shapes available.

After choosing the heart shape as the tool of choice, Jess can touch the screen wherever he wants to put the shape and then slide to stretch the shape to any size and proportion. When he makes an arrow shape, it points to the right or left depending on which way he slides. He can go to the text tools, and then choose a fancy text field and place it on the screen. As the field is placed, the keyboard appears, and he can type a happy birthday message.

Just a plain *Happy Birthday, Bubbie!* isn't good enough for an eight-year-old, of course. He wants to make the words appear bigger and bolder. To do this, he slides through the words, which selects them and opens the text selection window (see Figure 7-8). Then, he can change the type face (there's a sample that shows what each choice will look like), the type size (he likes 18 point), and the style (he'll choose from bold, italic, or underline). Naturally, he thinks he wants bold, italic, *and* underline, but ultimately he decides that bold by itself looks coolest.

Next, he chooses the pencil tool so he can sign his name. Third-graders are still working on spacing and he runs out of room for his elaborate signature. But Magic Cap saves him: he taps *extend*, and the page grows by about half. Now there's plenty of room to finish signing. He's ready to finish decorating the card, so he taps the stamper to see what's inside. With the stamper open, he can adorn the card with as many stamps as an eight-year-old can hope for (which is a lot).

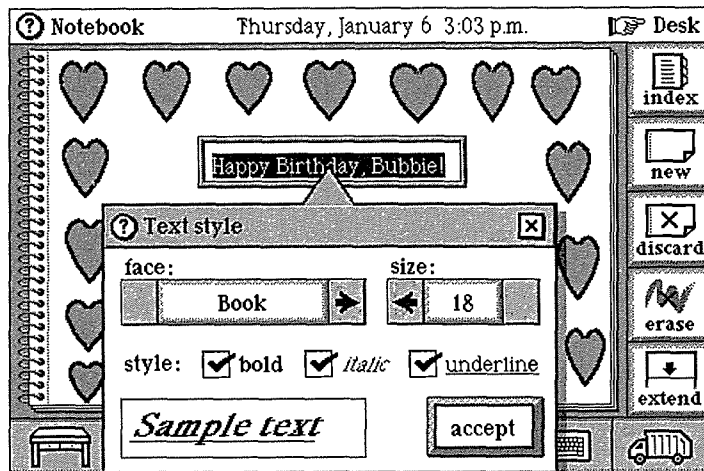


FIGURE 7-8. Eight-year-olds don't know from subtlety

Of course, Jess can't resist the animated birthday candle holding a piece of cake, and he drops a song stamp on the candle to make it play a catchy tune as it dances back and forth across the bottom of the card. At last, the card is ready to be sent. Jess touches the lamp, then *mail*, and then the *send* button. He chooses his grandmother's name from the list, and Magic Cap makes a new message for her and attaches the notebook page birthday card.

The card goes off with a tap of *send*, and the next time his grandmother checks her mail, she'll see a message from Jess. When she touches the notebook page that's attached, she'll see an original drawing from her grandson that moves and plays music. And she'll have a very, very happy birthday.

Notebook Index

As you look down the right side of the screen, you'll see the commands for pages in the notebook: *index*, *new*,

discard, *erase*, and *extend*. All of these are similar to commands in other parts of Magic Cap, but a couple of them have special twists just for the notebook.

As in other parts of Magic Cap, the top of the screen shows the total number of pages and which page is currently showing, as well as left and right arrows for flipping through the pages. While you're looking at any page in the notebook, you can tap *index* and you'll get a pictorial table of contents for the notebook. You'll see miniature versions of every page, as in Figure 7-9.

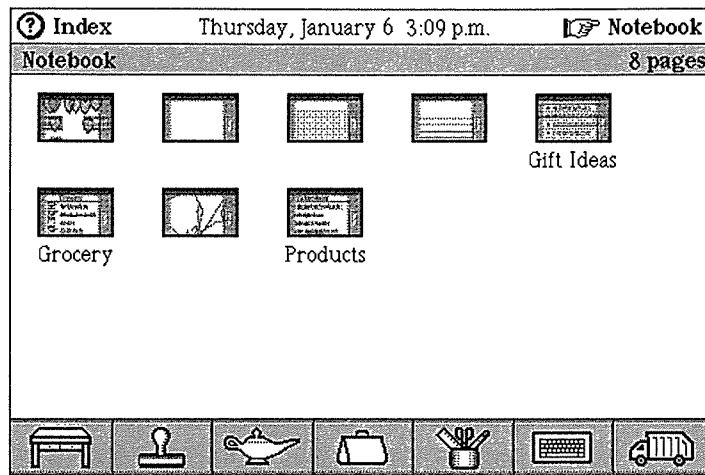


FIGURE 7-9. Notebook index shows miniature cards

You can use the index to get to any page that's displayed. Just tap on a miniature image of a page and you'll go there. There's more to do with the index. You can also slide these miniature pages to the tote bag and then attach them to messages as enclosures—they take the full-size version along, too. We'll look at an example of how you might do that.

While you're in a staff meeting, you might take notes for your upcoming trip to a trade show. Later, after the meeting ends, you can tap *index* and see miniature images of each page in the notebook. Slide the page of notes to the tote bag. You can also option-slide the image, which would ask Magic Cap to make a copy of the page to go into the tote bag, leaving the original in the notebook.

Then, you open the datebook and make a new multi-day appointment for the trade show trip. Once that's done, you can slide the miniature notebook page out of the tote bag and onto the business trip appointment. That's all you have to do to attach your notes to the appointment. By tapping the notebook page, you can take a look at the notes to make sure you have everything ready for the trip. By attaching it to the appointment for the business trip, you can find the notes the next time you have to travel on business. You can even edit the notes without leaving the datebook, thanks to smart integration of information in Magic Cap.



Consistency in the Interface. Consistency is a Magic Cap standard, and when you see things that look the same in Magic Cap, you can guess that they'll probably work the same way, too. In the name file or in box, you can option-tap the right arrow to see the last card, or option-tap the left arrow to see the first card. If you try the same trick in the notebook, you get the same result: option-tap the right arrow to flip to the last page in the notebook, and option-tap the left arrow to go back to page one.

Drawing on Paper

Not everybody likes to start out with a blank piece of paper staring them in the face. Some folks feel more comfortable with lined paper, and the notebook lets you work on paper that has lines on it, like school paper. If you type on lined paper, the words fit nicely between the lines. Of course, the words automatically wrap to the next line if they don't fit (and you might actually remember back when *that* was magic). Although the lines are kind of narrow for writing a lot of legible text, you might imagine a future version of Magic Cap that lets you choose between college rule and wide rule.

You might think of charts and graphs as business tools, but that eight-year-old who sent the birthday card to his grandmother probably makes a lot of little graphs in school. Students use graphs to help them visualize their estimates of things, like the total number of letters in the names of the students in the class. Before planting their garden, they need to make a line graph of the temperatures over a period of three school weeks to determine if the time is right to plant certain seeds.

The notebook provides two kinds of paper to help with this: lined paper, which has horizontal lines, and graph paper, which comes with a grid on it. Yes, you may have thought it was just a communicator, but it's also a notebook! It's a datebook! It's a learning tool! It slices and dices! But wait, there's more!

Let's start with a sheet of graph paper. To make a graph of temperatures, you start by choosing the horizontal line tool from the tool holder so you can make the axes. The horizontal axis needs to cover 15 lines for each day of the three school weeks. The vertical axis should probably have at least 20 lines (each line will be two degrees), in case you need to cover a spring cold spell.

When you try to put 20 vertical lines on a page, you'll quickly see that they won't fit. To get more space on the page, just tap *extend* and you'll get more lines at the bottom of the page. Go to the arranging tools and tap *stretch*, and you can extend the vertical axis to cover 20 lines. That just about runs to the bottom of the extended page, so you can tap *extend* again to get some room at the bottom for labels. The grid is complete now; you can see a screenful of it in Figure 7-10.

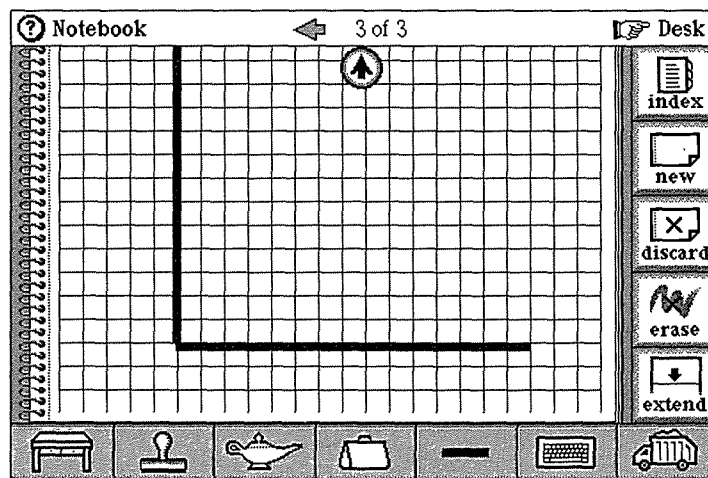


FIGURE 7-10. Temperature grid on graph paper

Now you can open the text tools and choose *transparent* to add labels for days under the horizontal axis. The vertical axis is next, and after that's done, you can start recording the temperatures using circles (from the shape tools). To start going in circles, tap the tool holder, get to shapes, and then tap the circle and see it hop into the tool holder. You can then touch the places on the screen that represent the temperatures for each of the 15 days. Your typical kid will really love this part; in fact, it'll take

great restraint to prevent going crazy touching the screen and putting circles everywhere.

Should you connect all the points on the graph with lines? If you're not sure, try it; you can erase them if you don't like them. Use the tool holder to choose a line tool, which draws straight lines at any angle. Then go ahead and connect the dots, another big hit with the eight-year-old meteorologist crowd. After you finish drawing the lines, your young collaborator may decide it looked better before you added the lines, so you can have a great time pressing *erase* over and over as you watch each line disappear while making the trash noise. After you're sure the graph is done, if you're connected to a printer you can tap the lamp and print the page. If you don't have a printer connected, you can get a paper copy by faxing the page to a fax machine in your home or office. You and your favorite eight-year-old will have learned the valuable skill of making a line graph, and the planting season will no doubt be a great success.

Gifts and Groceries

You'll find list-making paper available in the notebook. To get a page of lined paper with a circle marking each item, tap *new* and then *basic list*. If you like making lists, you might read through your favorite holiday catalog and use a basic list to enter the goodies that you'd really like to convince someone to get for you.

When you tap a line item on a list page, the keyboard opens so you can type its name. You can tap *return* to move to the next line to type in the next item. Once you've entered several items in the list, you can alphabetize them by using the *sort* button in the lamp. But you realize that alphabetizing them really doesn't convey the proper order of how much you want each one (hint, hint,

wink, wink), so you can rearrange the list to suit your priorities just by touching an item and sliding it to the position you want. When you do this, the item moves into the desired position and everything else automatically moves down a line.

If you decide to take something off your list (you realize that you've already got one, and one is plenty), you can throw it away by sliding it out of the list and into the trash. When you do, every other entry moves up, so there won't be a glaring empty line in the middle of your list.

If you want to change something, you can go back and edit an item just by touching that line. When you do that, Magic Cap opens the keyboard again and selects the text, giving you a chance to retype or add anything to the item. If you really want to push your luck, you can even tap *extend* to make room for more stuff on the page. See Figure 7-11 for the example list.

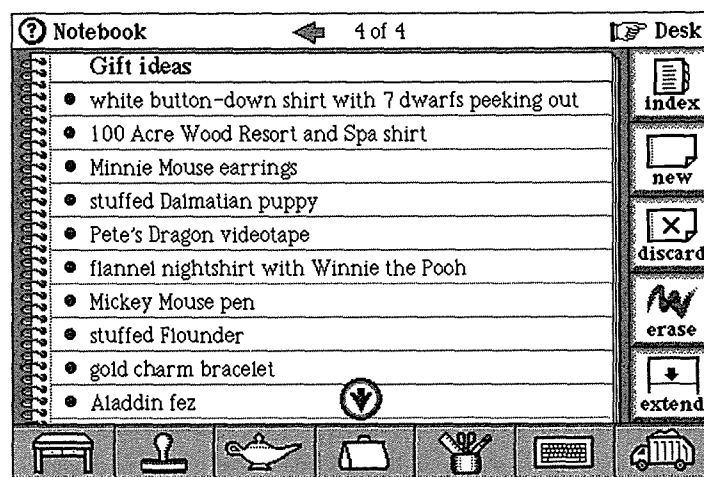


FIGURE 7-11. Gifts from favorite catalog

You might make other lists that are slightly more mundane, like what to buy on the weekly trip to the grocery store. The notebook provides checklist stationery, which, as the clever name implies, gives you lined paper and starts each line with a check box that you can tap to make a check mark. When you make a shopping list, you probably think of what you need to buy when someone in your family finishes the box/can/bag/bottle. It would be great if you could list the items according to the aisles in the store, but let's not get carried away.

If you list the items you need to pick up before leaving home, you can take your communicator to the store and when you actually take something off the shelf, you can tap the item's check box so you'll know the item is in the cart. When you get the lunch meat and cheese (they're in the same aisle), you can check each one individually even though they're separated by four other items, and then proceed to the next aisle for taco shells and seasonings. You can always extend the page if you need a bigger list. Figure 7-12 shows the example checklist.

If there are still some items left on your list after the shopping is done (let's say they were out of garlic bagels and the bananas were too ripe), you can slide the checked items into the trash, leaving the first two items on the checklist for your next shopping trip. Now that you've got just the two remaining items, you can do some fancy Magic Cap tricks to keep your shopping organized.

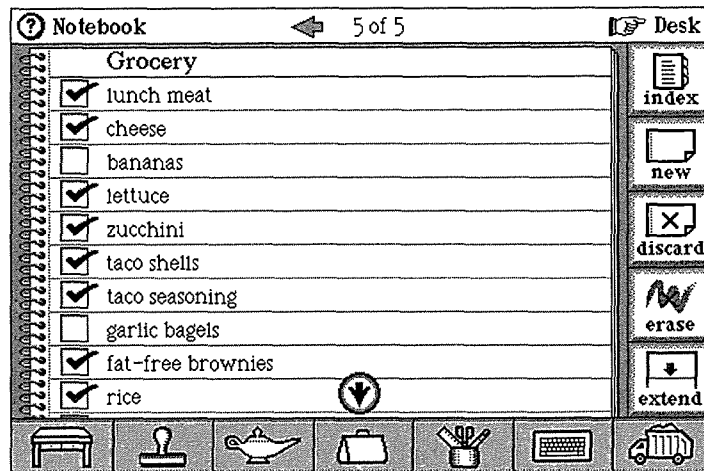



FIGURE 7-12. Grocery checklist

First, you can tap *index* to see the miniature images of your notebook pages. You locate the grocery checklist and slide it into the tote bag. Stepping back to the desk, you open the datebook and enter a *to do* appointment for going to the grocery store (a different one) tomorrow. Slide the miniature notebook page out of the tote bag onto the appointment and tap *save*. When you go to the grocery store tomorrow, you'll find the shopping list attached to that appointment. You can open and work with the checklist right in the datebook. You can check any of the boxes or throw away items. If you want, you can add other items to the list as you think of them.

 **Taking Snapshots.** You can make your own versions of those miniature images in any part of Magic Cap. If you have construction mode turned on (see Chapter 10 to learn how), you'll find a camera in the lamp that takes snapshots. These snapshots are reduced versions of

whatever scene you're in. You can drag the snapshot to other scenes; then, when you touch the snapshot, you'll be transported to the scene in the snapshot.

Because you made a *to do* appointment, it will appear on today's page in your datebook until you complete it. You can even take the list back to the notebook and reinsert it there, just as if you had a three-ring binder. You could also slide the image back into the tote bag and carry it back to the notebook. Because you've got your communicator with you all the time, you can add items to it whenever you think of them. That includes important work revelations that come to you at home as well as must-buy grocery items that you think of when you're at work.

Two-Column Lists

The Magic Cap notebook has another kind of list, which provides two columns of information. This list would be useful, for example, if you were brainstorming to make a list of your company's proposed new products and prices. Start by tapping *new*, then *two-column list* to make a new page. You can touch the top of the first column and rename it by typing *Products*. You rename the other column the same way, calling it *Prices*.

Each line starts with the same circle that appears on the basic list. When you tap an item, the keyboard opens so you can type the product's name. After you type an item, you touch *return* to move to the price column, where you can decide how much to price the product. Another tap on *return* and you're on the next line, ready for another product. You can see the two-column list in Figure 7-13.

| Products | Prices |
|------------------------|---------|
| • MLB stamps | \$15.95 |
| • NFL stamps | \$19.95 |
| • NBA stamps | \$15.95 |
| • NHL stamps | \$12.95 |
| • Floral Stationery | \$ 9.95 |
| • Animal Stationery | \$ 9.95 |
| • Quotation Stationery | \$ 9.95 |
| • Peanuts stamps | \$24.95 |
| • Disney stamps | \$24.95 |
| • Looney Tunes stamps | \$24.95 |

FIGURE 7-13. Two-column list of products and prices

You can slide a product to move it, and when you do, the corresponding price in the second column goes with it. You can tap *sort* in the lamp to alphabetize the products. When you alphabetize, the prices follow along with the products, of course. A handy feature in a future version of Magic Cap would let you sort the second column so you could see the products listed from the cheapest to the most expensive.

Customizing

One of the ways that Magic Cap helps you work is by letting you customize features that you use frequently. You can make your own kinds of appointments in the datebook, design your own stationery for sending messages, or add your own forms to the notebook. Let's make a new custom form now.

If you're an avid fan of a popular sports team, you might find yourself in the position of buying full season tickets but not attending all the games. If you're often in the position of trying to sell extra seats (where permitted by law, of course), you might like to have a quick and easy way to advertise the games you want to sell. The seat location and the price don't change from game to game, and the information about how to reach you also stays the same. The only thing that does change, then, is the date of the game and the opponent (for example, if you're a National League baseball fan who doesn't live in New York, you may have lots of games available against the Mets).

To make the new form, start by tapping *new*, and because the new form will be mostly text, tap *plain paper*. You can use the keyboard to type the information that's always the same. When you've entered that stuff, you can spruce up the page a bit by making a boldface headline to draw attention to it. You've already seen how to edit text: Slide across the sentence, which selects the text and opens the text selection window. Tap *style, bold*, size *18*, and then *accept*.

When you've finished designing the advertisement, you're ready to make it into a form. Tap *index* to see a miniature image of the page, and then slide it into the tote bag. Go back to the notebook view by tapping the step-back hand or *Notebook* in the upper-right corner. Tap *new* to see the *Select a new page* window, and then slide the miniature page out of the tote bag and into the window. The new page pops into place, and your newly designed form is added to the choices for new pages.

As a final touch, you can give your new form a descriptive name. Option-tap the keyboard image to open the keyboard with label maker. Type the new name (*Tix for sale*), slide the label right onto the new form, and the paper is renamed (see Figure 7-14). From now on, whenever you want to try to sell your tickets, you've already got the right form in your notebook. Let's hope they have a great season!

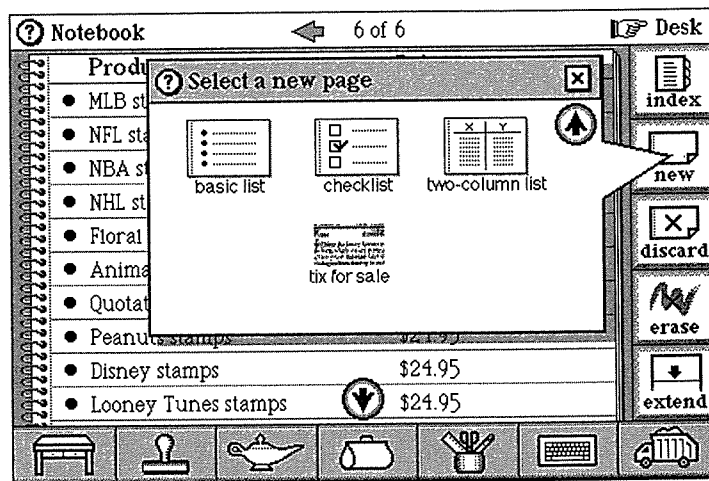


FIGURE 7-14. Custom form added to notebook

Summary

The notebook is the place for writing and drawing freehand, as well as typing notes that are not necessarily going to be seen by anyone else. The notebook paper doesn't try to interpret your handwriting, so you can jot down notes or type them. You can use built-in writing and drawing tools, like pencils and lines of different thicknesses and several different shapes.

The notebook pages can be plain paper, lists, lined paper, or graph paper, and you can easily extend any kind to make longer pages. You can stamp any kind of paper with anything you find in the stamp drawer. You can add music, sound effects, and animations to your notes. There's an erase button that removes marks you've made. There's also an eraser tool that you can use with your finger or stylus that simulates rubbing with the eraser at the end of a pencil.

You can use the keyboard to type anything on the notebook paper, and you can even specify the kinds of border (or no border) that encloses the text. There are tools to arrange the items on your paper: *stretch* lets you extend lines and shapes that have already been drawn, *copy* lets you duplicate things for use on other pages or elsewhere in Magic Cap, and *move* lets you slide objects around on the page.

There's an index in the notebook that shows miniature images of each page. You can move these miniature images around to use elsewhere. You can slide one into the tote bag, and then slide it out onto an appointment in your datebook. It remains an active, editable page, even inside the datebook. You can also attach a notebook page to a message that you want to send.

Because you can always use *mail* and *fax* in the lamp, you can touch *mail* when you're looking at a notebook page and it will shrink and hop onto a postcard, ready for you to address and send. You can also choose to fax it, with an option to add a cover page or to turn it sideways. To fully customize your notebook, you can design your own kind of paper and then add this new form to the choices that are available whenever you add a new page to your notebook.

Chapter 8



File Cabinet

Filing

I hate to file. Sure, I love to buy filing paraphernalia—folders, color-coded labels, storage boxes of all shapes and sizes. I have every intention of being able to find all the insurance records when I need them, keeping the frequent flier mileage statements where they belong, and knowing where to track down the Little League roster, the third-grade class list, and every other phone list I have. You know what they say about good intentions and a certain road, though. I hate to file.

Machines do not hate to file. In fact, that's one of the things they do best. People who in real life are allergic to manila are often not very thorough about filing things on their computers, either. For them, filing means selecting an image and dragging it into another image. It's hard to find the time to set up an efficient filing system, so everything winds up in folders named *Filed stuff #37* and so on.

Magic Cap's designers recognized that filing shouldn't have to take much time, thought, or effort. A good set of filing features should be flexible enough to accommodate various styles of filing behavior, from those who just want to stash things quickly to folks who set up elaborate systems for putting everything away. Well-designed filing

features should include built-in shortcuts and commands to file things automatically and have them end up where users expect them.

In the File Cabinet

Tucked behind the desk on the right side is the file cabinet. When you touch it, it opens to show you a drawer inside, which you can see in Figure 8-1. The drawer is filled with neat rows of folders, with the folders' left, right, and center index tabs lining up perfectly, another one of those things that never works out in the physical world. Magic Cap's file cabinet guesses that most of the things you'll file will be electronic mail messages that you've received or sent, so it has built in drawers for each of those two categories.

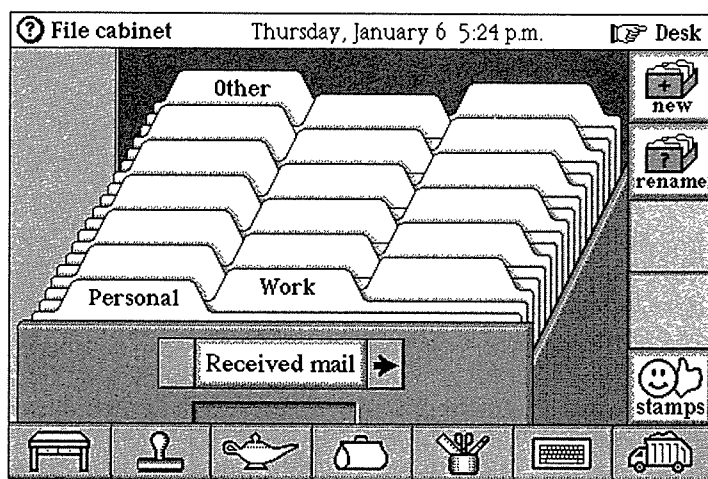


FIGURE 8-1. *Received mail drawer in file cabinet*

You can also create whole new drawers for more specific subjects, and every drawer has 19 folders that you can fill with information. Although you'll usually file messages here, you can actually use the file cabinet to hold onto several different kinds of items, including notebook pages, name cards, and old appointments.

Drawers in real file cabinets usually have a small paper label that shows what's kept inside, just above each drawer handle (a legendary story says that author L. Frank Baum spotted the bottom drawer of his file cabinet when trying to decide what to call the fantasy city in his children's story, and the Land of Oz got its name). In the Magic Cap file cabinet, that space also contains a choice box that you can use to choose which drawer you want to see.

The choice box has the familiar left and right arrows to show you its choices. Touching the label itself opens up a window that lists all the available drawers, and you can open the desired drawer just by touching any of the names on that list. If you add more drawers, you can take advantage of these consistent shortcuts: Option-touch the right arrow to see the last choice, or option-touch the left arrow to see the first choice.

Building a New Drawer

Let's say you're a real estate agent, and you want a drawer to keep notes about the different properties you have listed. Each house could have its own folder, which might contain some messages you sent to confirm some details of the property, messages you received from prospective buyers, and any notes you may have made about the features of that house. You may even want to keep copies of the name cards of prospective buyers in the folders.

If you needed more space for a real file cabinet in your office, you'd go and buy a new one. It's much easier to add another drawer to the Magic Cap file cabinet—just touch *new* in its familiar top spot on the right side of the screen. The keyboard opens, along with a window where you can type the name for the new drawer, which we'll call *Properties* (see Figure 8-2). *Properties* then pops in as the drawer's name, and it's also added to the choice box as the last choice.

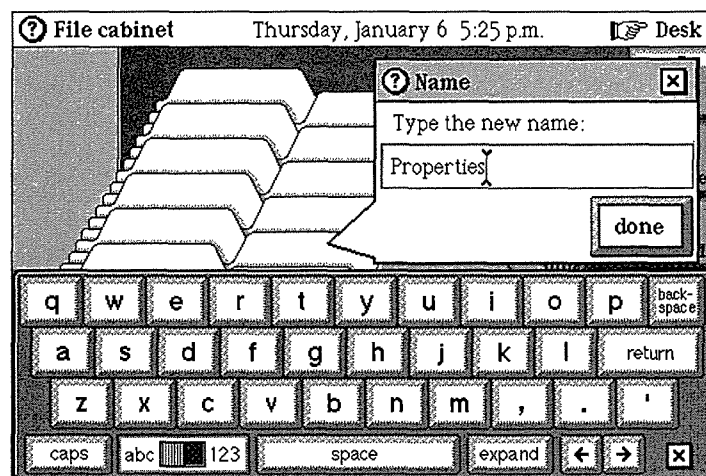


FIGURE 8-2. Making new *Properties* drawer

If you find that you've made a mistake in typing the name of the drawer, or if you want to change the name entirely, you can tap *rename* to get the keyboard and the naming window again. If you decide you want to get rid of the whole drawer and anything you may have filed there, you can tap *discard* and the drawer will be gone. You can get rid of any drawers you create, but the *Received mail* and *Sent mail* drawers are bolted in there permanently and can't be removed.

Once you've built the drawer, naming the folders inside it is also intuitively simple: You just touch the index tab at the top of a folder. When you do, the folder opens, the keyboard appears, and the typing point is positioned at the center of the index tab, as shown in Figure 8-3. Folder names are limited to the available width of a folder tab (about 10 characters), so you might just use the street name of each house in the example.

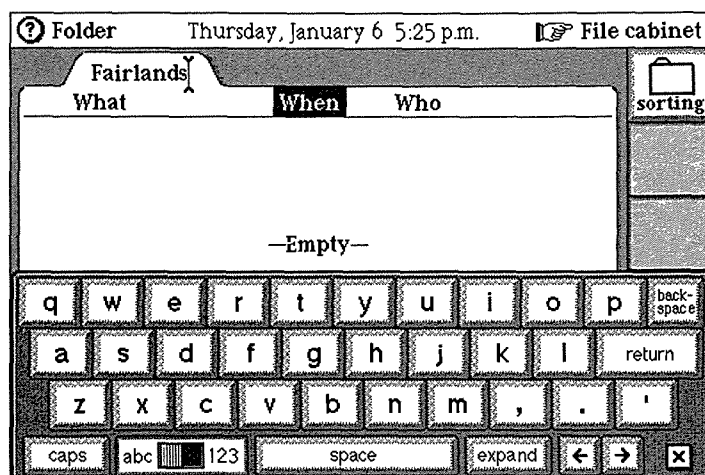


FIGURE 8-3. Typing a name for the folder

Because you get to decide which folders to name, you can put them in any order in the drawer. Once you've named the folders, you can reposition them by sliding them around in the drawer. If you've just recently created a *Fairlands* folder named after a street, you may want to move it so it's alphabetically in front of the *Hacienda* property folder, but behind the *Capri* townhouse folder.

As you're sliding *Fairlands*, you can see a ghostly outline of the folder and its label as it travels across the drawer, and you can even see the other folders move out

of the way as you slide past them. When you get to the desired spot, you can stop sliding as it snaps into place. Of course, you can put the folders in any order you're comfortable with—alphabetically, chronologically in order of when you listed the property, in order of asking price, or anything else you want. An interesting enhancement for a future version of Magic Cap would be to include a way to automatically alphabetize the folders.

Once you have items filed inside a folder, you can move the filed items from one place to another just by sliding them. Let's say that Mr. and Mrs. Rusanowsky are prospective home buyers and you have a copy of their name card in the Fairlands folder. If they're unimpressed with that property, you'd like to move their name card to another house you think they might like.

To move their name card, tap the folder tab to open the folder, choose the item you want to move (the name card) and slide it into the tote bag. Tap *File cabinet* to see all the drawers again, and then slide the name card into the folder for the house on another street, which should be in the same drawer. You'll see the ghost and label of the Rusanowskys' name card as you're moving it, and you'll hear it get sucked into the folder where you now want it. You could also slide it to the tote bag and then move it to a new drawer, or even slide it into the trash if they've finally decided to leave town altogether.

File folders also give you a shortcut that lets you pull items out without even opening the folders. While you're looking at a file cabinet drawer, touch a folder tab and slide; the first thing you filed in the drawer pops out and comes along wherever you drag it, and you see its image and label as you drag. You can then drop it somewhere else, such as in another folder, the tote bag, or the trash.

Moving Folders

You can move whole folders and their contents to different drawers. If the real estate business is slow, you might need to add a drawer called *Inactive*, for properties that haven't sold in several months. For example, if the folder for the Hacienda property is empty because you've had no interest from anyone, you might want to move it to the *Inactive* drawer.

To move the folder, start by sliding it into the tote bag, opening the other drawer, and then pulling the folder out of the bag and into the new drawer. As you slide the folder, you'll see the outline of the folder and its label moving along the screen and into the tote bag; you'll leave an empty, unnamed folder behind in its place. Next, use the choice box to open the *Inactive* drawer. Then you can slide the folder back out of the tote bag into the new drawer and drop it into whichever position you want.

If you're a successful real estate agent, you'll probably want to add another new drawer called *Sold*. After the Fairlands property sells, you may want to move that folder and its information to the new drawer. To move an empty folder between drawers, you only need to slide it in and out of the tote bag. Magic Cap requires that you hold down the option key when there are things in the folder so it can tell that you want to drag the whole folder and not just something inside.

After you've got the folder in the tote bag, you can switch to a different drawer or create a new one; then you can slide the folder out of the tote bag and into the *Sold* drawer, with all its contents intact. Magic Cap always shows you how many items are in a folder because each time you add or remove something, a number just to the left of the folder name changes to reflect how many items are filed there.

Sorting Incoming Mail

Because Magic Cap was designed around easy communication, the file cabinet was built to automatically file incoming and outgoing messages. In Chapter 2, we discussed rules for the in box and out box about filing messages with certain text, senders, or even stamps. By using these rules, you can have the in box file your incoming messages in folders you specify before you even read them.

If you're an account executive for a sporting goods company, you have to sell lots of different items to various retail stores. Because you're on the road a lot, it makes sense for your customers to send their orders to you via electronic mail, rather than take their chances reaching you by phone. One of your biggest customers, Barry B's Baseball World, is having a sale on catchers' equipment, so the manager needs to add to his regular order. Soccer Locker is running low on shin guards and will need more before your next visit. In addition to those orders, you get regular orders from Hockey Hut and What A Racket!, as well as sales reports and messages from the main office.

You can set up a folder for each store in the drawer for *Received mail*. Then, you can set rules for your in box to look at the senders of each message you receive and file it in the appropriate folder before you even open the messages. If you make a folder for each store, then messages from Barry B's will be filed automatically in that store's folder, and so on.

When you create folders for senders and direct your inbound mail to go there by setting the appropriate in box rules, your mail is presorted and it's already categorized when you read it. If you don't specify where to file

the messages from other senders, they'll stay in the in box until you read them or file them another way. For more information on using the in box rules to file messages, take another look at Chapter 2.

Filing Messages Manually

Some people feel a bit uneasy having their unread mail shuffled off to a folder as soon as it arrives in the in box. These folks like to work from the in box, knowing that messages get filed only after they've been read. If a message must be forwarded, or if it needs a reply, or even if it should be thrown away, they prefer to direct the action themselves instead of having instructions that are carried out by magic.

If you use your communicator to answer customer service questions, you might want to make sure that each message gets personal attention. You can leave the in box rules at their factory settings so that your messages will stay in the in box rather than being filed automatically before you open them. As you open a message and read it, you can choose to act on it then and there.

After you've handled the message by replying or forwarding, you'll want to file it. For the greatest control over filing, touch the file button on the right side of the screen, as shown in Figure 8-4. The *File* window has buttons that let you file a *copy* of the message or *move* the message itself somewhere. By the way, this is the same *File* window that appears when you tap the *file* button in the lamp; the in box gives you this more convenient *file* button because filing is such a common operation in the in box.

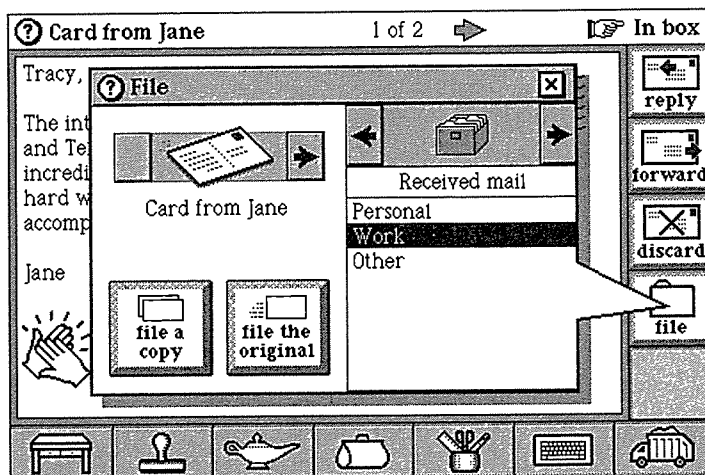


FIGURE 8-4. Filing a message that's in the in box

You can pick where you want to file the message by using a choice box that has the names of the file cabinet drawers and their folders. The choice box also offers to let you file the card in packages in main memory or on a memory card if one is installed in your communicator.

By using *file*, you can read through your customers' queries in whatever order you want, and then act on each one after you've read it and file it for future reference. If you use *file the original*, the message will vanish from your in box as it's filed away. By using *file a copy*, you can keep the message in your in box while you make a copy and file it.

If you're looking at the in box index, you can quickly file all the messages you see anywhere you choose, such as in a file cabinet folder or on a memory card package. To do this, tap the lamp to open it, and then tap the *file* button inside. You'll get the usual *File* window, except that the left side of the window, which tells you what you're filing, will indicate that you're about to file everything in the in box.



Humanizing the Interface. Filing the contents of the in box demonstrates how hard Magic Cap tries to be human and friendly. If there's only one item in the in box, the *File* window tells you the name of the message's sender. If there are three or more items, the window says how many there are, as in *All 5 messages*. The most conversational message appears when there are two items; in that case, the window says that it's about to file *Both messages*.

There's another manual way to file messages in the file cabinet. You can slide a message from the in box index to the tote bag, open the appropriate drawer in the file cabinet, and then slide the message out of the tote bag and into a folder. This is great for beginners who aren't comfortable with the *file* button or for filing messages that you may have left sitting around on the desk or somewhere else. Don't be sloppy!

A Filing System for Incoming Mail

As you've probably noticed by now, Magic Cap tries hard to be adaptable to several different ways of working. In keeping with that theme, you can use yet another way to file your mail. This technique lets you read your messages in the in box, and then file them automatically according to specifications that you've set up.

There are two steps involved in using this technique. First, you'll make the folders you want and teach those folders what kind of messages they should attract. Then, you can use the *file all* button on the right side of the in box index, just below the button for collecting mail. When you tap *file all*, you'll be taken back to your desk, a folder

will pop out of the file cabinet, and your messages will hop into the folders you set up.

Let's work through an example that uses this kind of filing. As a customer service representative for an office equipment retail store, you'll start by opening the drawer for *Received mail* and creating four folders, one for each line of products you support: copiers, fax machines, personal computers, and printers.

When you open a folder, the only button you'll see on the right side is labeled *sorting* (look back at Figure 8-3 to see this button when the folder is open). This is the button you'll use to tell the folder what kind of messages it should file. When you touch *sorting*, you'll see the *Sorting criteria* window (see Figure 8-5), which lets you set all kinds of instructions about what should be filed in this folder. The window has check boxes to turn each criterion on or off, as well as the appropriate choice boxes and text fields for typing.

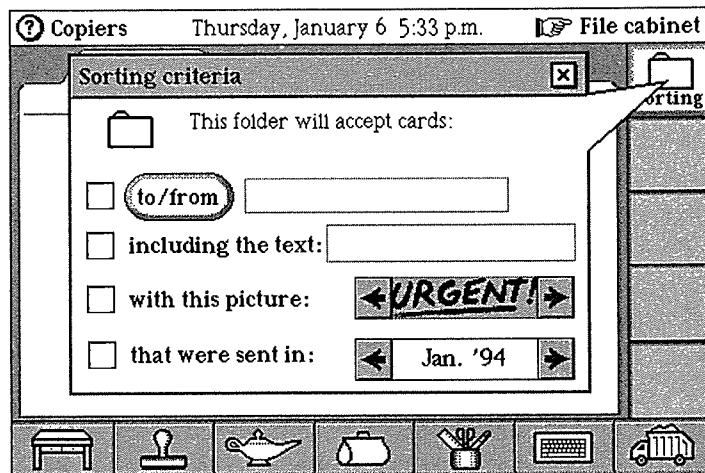


FIGURE 8-5. *Sorting criteria* window

You want to set up your new folders to hold messages about each of the four product lines. Tap the *Copiers* folder to open it, then tap *sorting* to set the criteria. You want this folder to hold any messages about copiers, so tap the check box that looks for particular words in the message, and then type *copier*.

You can repeat this process for each of your four folders. When you're done, the folders know how to accept the right messages. From now on, when you're looking at the in box index and you tap *file all*, any messages with the words you specified will automatically wind up in those folders without any further decisions by you. This shows how a little planning and preparation can go a long way!

If you want to set up a folder for messages from a specific customer, you can check the *to/from* box. When you do, the familiar name chooser will appear, and you can pick the name of the customer you want. When you get messages from that customer, tapping *file all* will put those messages in this folder.

Another sorting choice lets you file messages according to stamps that appear on them. You set this criterion with the choice box labeled *with this picture*. You can see the stamps by using the scroll arrows, or you can choose them by name from a list by touching the center of the choice box, as shown in Figure 8-6. Once you pick a stamp, messages with that stamp will be filed in this folder.

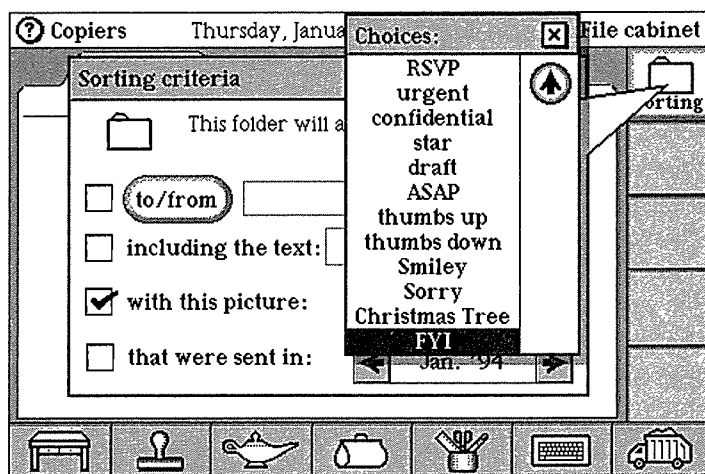


FIGURE 8-6. Choose a stamp by name

You can use the file-by-stamp feature to organize your incoming messages by stamping them yourself. For example, you might want to have a folder for mail you've received about new product procedures. After you read one of these messages, you can add an FYI stamp, even though the sender of the message didn't put that stamp on it. Then, when you touch *file all*, the FYI-stamped messages will jump into the folder you set up to accept messages with that stamp.

The final sorting criterion is the month and year the message was sent. You can use the check box's arrows to move forward and backward in time to pick the month you want. After you set this criterion, messages that were sent in the indicated month will be attracted to that folder when you *file all*.

What happens if you turn on more than one check box for a folder? The folder criteria work like this. If any of them is true for a message that you file, the message goes into that folder. For example, if you set up a folder to

accept messages with the RSVP stamp or with the text *your party*, any messages with either that stamp or that text (or both) will wind up in the folder.

You can customize the set of stamps that's used for folder criteria. In the file cabinet scene, tap *stamps* to see the stamps that are available for folder criteria. You can add any new ones if you want, or you can remove stamps that you're sure you'll never use as filing criteria.

What if more than one folder wants a message? If you file a message that can match the criteria set up by more than one folder, the first folder in order in the drawer that wants it gets it. This lets you get really fancy and arrange your folders in a sort of hierarchy, with the most important ones getting first shot at messages.

If you haven't specified any rules when you tap *file all*, or if you have messages that don't respond to the criteria for any folders, your messages will be filed in a folder called *Other* in the *Received mail* drawer. Once they're there, you can move them into other folders whenever you want.

If you have stray messages sitting around, you can take advantage of the folder criteria you set up. Just slide these lost messages to the file cabinet on the desk. If the message was one you received, it'll go into the *Received mail* drawer; otherwise, it's filed in *Sent mail*. Inside the appropriate drawer, the message is filed according to the folder criteria you set up.

Filing Outgoing Mail

You can use the same folder criteria technique to file copies of your outgoing messages. For outgoing mail, the folder criteria work together with the out box rules. The out box rules let you file messages containing a specific word or certain stamps (urgent, confidential, or low

priority), and they also let you choose to file into places other than the *Sent mail* drawer, such as another drawer or the tote bag. You can get filing options by setting the criteria for folders in the *Sent mail* drawer, but these out box rules and the folder criteria basically perform the same function.

One important note: To make sure that copies of all outgoing messages get filed, the out box rule *When any other message is sent, file it in the file cabinet* must be turned on (it's turned on at the factory). With this rule turned on, copies of all outgoing messages will wind up somewhere in the *Sent mail* drawer.

Let's say you want to order a jacket from Hendler's Emporium, a well-known store in downtown Magic Cap. You go downtown and pick out just the item you want to order. Hendler's has an electronic order form that you fill out and send directly to the store. You can order the purple suede jacket in size 10, and then mail the message to them. Because you have a folder set up for Hendler's in the *Sent mail* drawer of your file cabinet, you'll be able to keep a copy of your order, which includes the date and time stamped on the postmark.

Each time you send in an order, your copy of the order form is filed automatically in the folder. If your jacket isn't delivered in the promised two weeks, you have a copy of the order to refer to. If you hadn't set up a folder just for this store, a copy of the sent mail would have been filed in the *Other* folder in the *Sent mail* drawer.

Although the file folder criteria were created mainly for filing messages, they actually work for other items that you can file, including name cards and notebook pages. If you slide name cards or notebook pages to the file cabinet on the desk, they'll be filed in the *Sent mail* drawer according to the folders' criteria there.

The file cabinet has just one rule—how long to keep the items filed before trashing them. The rule is factory-set to throw out items after one month, but you can change the time setting. If you never want to throw anything away, you can turn the rule off completely. After all, who knows when you might need some note you wrote and sent four years ago?

Keeping things forever would be nice, but your communicator has limited memory, so you'll probably want to let the file cabinet throw things away after some time has elapsed. For those essential items that you want to be sure to keep around forever, you can mark with a Save stamp. Items marked that way will never get tossed by the file cabinet, even if their time has come.

Storeroom

Magic Cap includes a software version of extra closet space. It's called the storeroom, and you can find it by going down the hall, third door on your left. As you might expect from its name, the storeroom is the place where you manage the memory and items in your communicator. You can work with the communicator's main memory, any memory cards that are inserted, or a personal computer that's connected.

When you touch the storeroom door, it opens to reveal a directory, a personal computer set up on a table, and a shelving unit filled with boxes, some open and some closed, as you can see in Figure 8-7. The boxes on the shelves represent software packages, which are collections of objects that work inside your communicator.

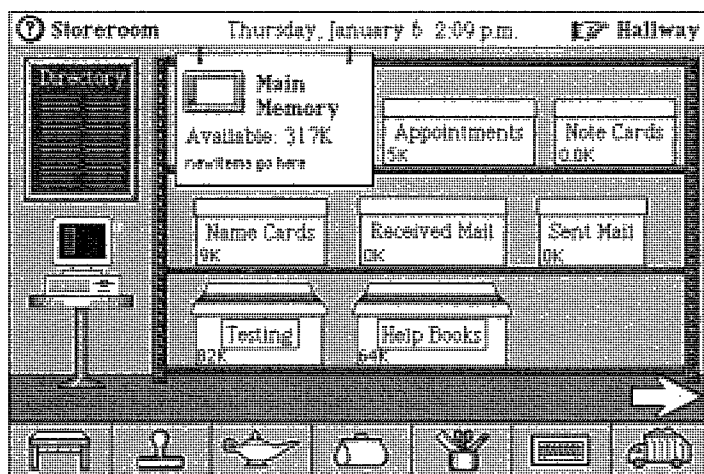


FIGURE 8-7. The storeroom shows software packages on shelves

The sign hanging in front of the shelves tells you what those shelves represent: main memory, a memory card, or a personal computer cabled to your communicator. You'll also see that each package is labeled with its contents and size.

There is a package for appointments, which, as you might guess, contains the contents of your datebook. Any package can be opened by touching it, and it hops off the shelf and opens to tell you what's inside. Like the hallway and downtown, the storeroom is wide enough to take up more than one screenful of information. You can tap the arrow on the floor to see more shelves off to the right, one for each memory card in the communicator.

The main memory shelves contain packages that represent important sets of information inside the communicator: appointments, note cards (pages in the notebook),

name cards, received mail, and sent mail. When you touch a package, it jumps off the shelf, zooms open, and tells you a little bit about itself. Figure 8-8 is an example, showing you the inside of the appointments package.

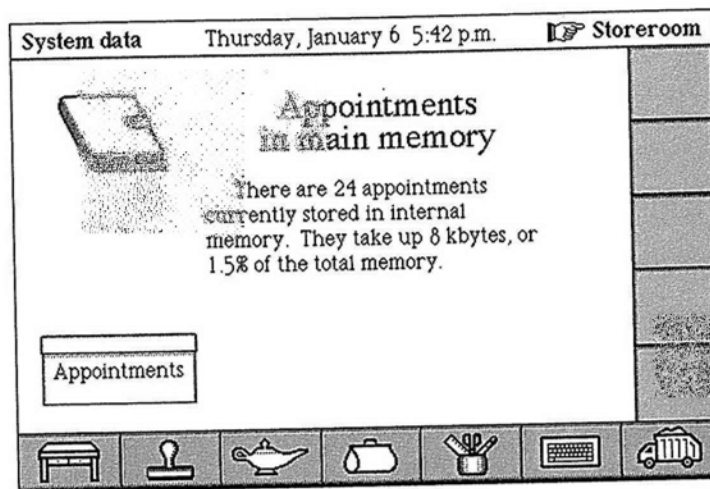



FIGURE 8-8. Information about the appointments package

 **Please Touch.** Magic Cap users get used to the idea that they can touch items to make something happen. To support this, Magic Cap's designers tried to be sure that something happens (and something predictable) when you touch familiar items. As an example, the datebook that appears inside the appointments package isn't just an image, but the actual working model. If you touch this datebook, which is inside a box in the storeroom down the hall from your desk, it opens your datebook just as if you had gone back to the desk and touched the datebook there.

The bottom main memory shelf has an open package called *Help Books*. This package has the set of library books that came from the factory: *Getting Started*, *Secrets*, *Basics*, and others. This package looks open to show that it's *unpacked*, that is, its contents are visible and available for use. Magic Cap encourages you to support your local library, so you can check the books out at the library, which is just down the hall.

You might notice that the image for the *Help Books* package is different from the others. That's because the *Appointments*, *Note Cards*, *Name Cards*, *Received Mail*, and *Sent Mail* are special—they're system packages that always represent their specific kind of information inside the communicator. The *Help Books* and any packages you buy from software companies appear as boxes that can be open (unpacked and ready for use) or closed (packed up). You can pack up a package to make its contents unavailable, which will free some memory in the communicator, and then unpack it later when you need it. You can also get rid of it completely by sliding it to the trash, which will free a lot of memory, but you won't have the package any more.

Opening Packages

We looked at the inside of the appointments system package; now we'll open a standard package. Touch the *Help Books* package to open it (see Figure 8-9). This one tells who made it and when, and also provides some buttons along the right side. The first button, *pack up*, closes the package and puts its contents away, as discussed earlier. After you pack up the package, the button turns into an *unpack* command.

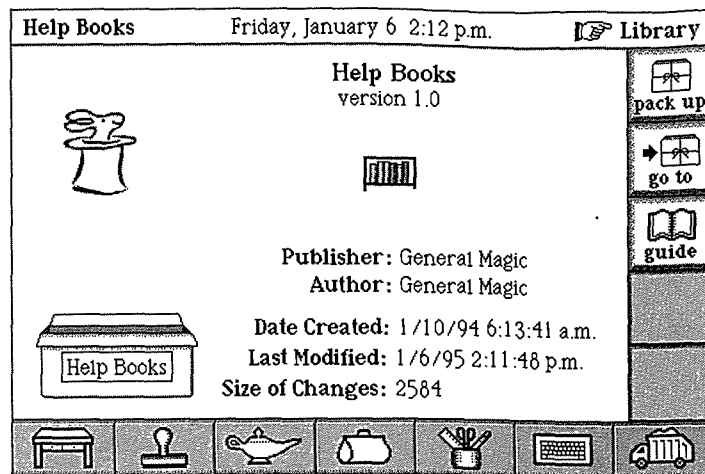


FIGURE 8-9. Inside the Help Books package

Every package installs its items in the appropriate place in the Magic Cap world. Books are shelved in the library, games go in the game room, stationery forms go into the stationery drawer in the desk, and there are dozens of other places that can hold items from packages. You can tap *go to* if you want to be transported to the scene that holds the package's contents; for example, tapping *go to* from the *Help Books* takes you to the library.

Many packages come with some kind of built-in documentation. You can tap *guide* to see the documentation for these packages. The people who provide your software packages are interested in hearing from you, and because this is a communicator, it's easy to get in touch with them. Many packages include a *respond* button that creates a new message addressed to the package's publisher, giving you an easy way of registering for upgrades or complaining about something.

Memory Cards

Every Magic Cap communicator lets you store packages on memory cards, which are credit card-size storage devices that you can use to increase the amount of stuff your communicator can work with. They're known technically as PCMCIA cards, an acronym only a computer could love.

When you insert a memory card, a new shelving unit appears to represent it. A sign hangs in front of the shelves with the card's name, as pictured in Figure 8-10. You can use the shelves to move packages from main memory to the memory card, just like moving unused items into a trunk in the attic. You can use this feature to archive old appointments or little-used name cards by sliding the appropriate box to the memory card.

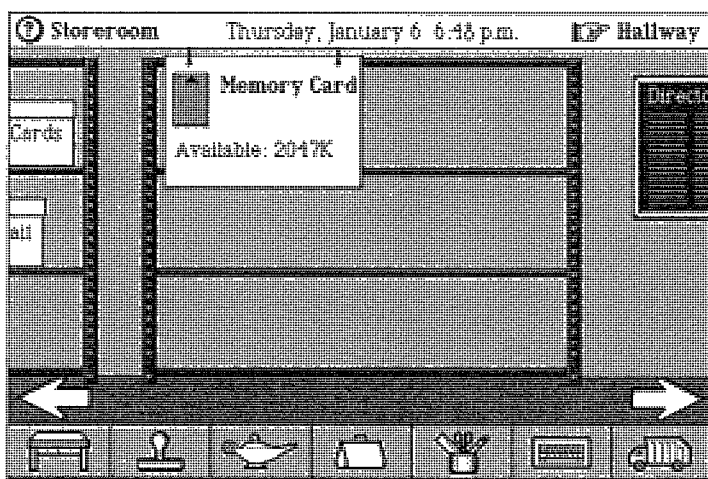


FIGURE 8-10. Shelves for memory card

Items on the memory card are very much alive and accessible. When you insert a memory card, you can unpack its packages, which installs the packages' contents in their proper places. So, if you have a memory card with special name cards or old appointments, they'll appear in the name file or datebook when you insert the card.

Extending Main Memory with a Memory Card

You can extend your communicator's main memory by setting up a memory card as the place where all new items go. Here's a scenario: As an architect, you use the notebook frequently for sketches, and you may have to keep them for quite a while. It might be easiest to set up a memory card for your notebook, so every time you add another page, it automatically saves onto the memory card.

To do this, tap the memory card's sign. It zooms open to let you change its name or check the *new items* box. Tap the check box to turn it on, and then close the window. The sign now says *new items go here*, and whenever you make a new page in the notebook, everything will work as it usually does, except that the new page will be stored on the memory card.

When you have a memory card that's getting all the new items, it will also store any new name cards or appointments that you make. As long as the memory card is inserted, these items will act just as they would if they were in main memory—notebook pages appear in the notebook, appointments show up in the datebook, and so on. But as soon as you remove the card, anything that's stored on it will vanish.

To help you remember what's on the memory card, a tiny image of a memory card appears at the top left part of the screen for any item that's stored there. Figure 8-11 shows a notebook page from a memory card as an example. If you decide that the item is too important to keep on a memory card, you can tap the tiny memory card and Magic Cap will copy the item into main memory.



FIGURE 8-11. Tiny image at top shows important notebook page is on memory card

The storeroom has a few other details to keep you up to date on what's happening inside your communicator. There's a directory on the wall that lists the shelves you have. You'll also see a sign at the far right that shows you how much of your communicator's main memory is being used.

Backing Up and Connecting to a Personal Computer

The storeroom includes buttons in the lamp for backing up all your information to a memory card. Open the lamp, and then tap *back up* to save all your precious data. Then, if disaster strikes, you can insert the memory card that has your backup, and then tap *restore* in the lamp—it will all come back to you.

When you add an optional software package and cable, Magic Cap can connect to a personal computer. With this software installed on your personal computer, you can tap the computer in the storeroom to connect. Once you're connected, you can use the linked computer to back up your packages or as a source for installing new packages. You can also print using a printer connected to the computer.

Summary

The Magic Cap file cabinet was designed to behave just like file drawers in the physical world. There are standard drawers for *Sent mail* and *Received mail*, but you can make new drawers at the touch of a button. There are folders in each drawer that you can easily name. You can move folders around inside each drawer, and you can also move a folder with its contents to another drawer. You can file items manually by sliding them into the tote bag and then back out into a specified folder, or you can use the *file* command available in the lamp. There are also rules for filing in both the in box and out box scenes.

Each folder can be set with a variety of sorting criteria, which enables it to collect messages automatically. You can have incoming messages filed before you've read them based on who sent them, or on a certain word or stamp included in the message. You can file your mail

according to when it was sent. You can also file the messages individually from the in box after you've read them.

The same sorting criteria can be used to file mail you've sent, too. Even if you don't specifically set a folder for mail, if you keep the factory-set rule turned on for filing all outgoing messages, copies of them will wind up in your file cabinet in a folder called *Other*. All folders tell the number of items inside them by showing a small number next to the name.

The main storage space in Magic Cap is the storeroom. It is set up with shelves containing packages that represent important items in your communicator. Each package is an active representative of its contents; you can open the *Appointments* package in the storeroom and see information about the appointments in your datebook. You can insert memory cards, which make new shelves for packages, to store information. Memory cards can also be used to extend your communicator's main memory. You can insert one and set it to store all newly created items so that as long as it's there, your communicator has that much more memory. As soon as you remove the card, though, that information is no longer accessible.

The storeroom scene also includes buttons in the lamp to back up your communicator's information to a memory card, or even to a personal computer if you have additional accessories.

Chapter 9



Other Features

Details and Loose Ends

When I'm working, there are lots of little things that I need that aren't directly part of the work I'm doing. Supplies and tools, like a dictionary and thesaurus, are essential, plus some sticky notes and a pen. The environment is important, like having music to fill the background and having a glass of water or can of cream soda. And a stapler. And maybe some paper clips, and the three-hole punch. And my calendar. And the clock. And...well, you get the idea.

I may not need to use all those things on any particular day, but if I do, I don't want to interrupt my work to go dig them up someplace. I like to have easy access to the things I might need right where I need them, even if I never do need them. There are lots of these kinds of accessories designed into Magic Cap. There are also storage spaces that stay empty until some farsighted software developer creates a package to fill a need, like an accounting package to live in your desk drawer or games for the game room, or a whole travel agency downtown.

Some of these details are vital, like the control panels and the library. There are others that you might never use, such as the calculator. You might never notice the clock after setting it once because it just blends into the Magic Cap world. Depending on your interests, it's

possible that you could walk right past the game room in the hallway without bothering to peek inside. Magic Cap's designers worked to make sure that these extra features wait patiently until you need them.

Time and Date

A clock is typically an indispensable feature in almost any environment. Hanging prominently on the wall behind your desk, the clock is a nice analog one that not only shows the current time, but it also has controls to set the time and date and even to show you different time zones.

As you might guess, you can set the clock by touching it. When you touch the clock, it zooms up close to show you a calendar, a bigger analog clock with a digital time display below it, and four buttons on the right edge to show you other parts of the clock (see Figure 9-1). The top button, *display*, is already highlighted to tell you that this scene simply displays the current time and today's date. This scene is strictly for looking; to set the clock, you touch *set time*, and to change the date, you use *set date*.



Up to date. For years, computer engineers have been putting the time and date together. After all, they know that the time and date are really the same thing mathematically—the time is just a way to talk about the date in finer detail. But Magic Cap's designers observed that most people think of the time and date as two separate things. To match this expectation, Magic Cap always explicitly refers to the time and date separately, as in the *Getting Started* lesson named *Set the time and date*.

Touch *set time* when you want to change the time of the clock inside your communicator. It will look like the scene in Figure 9-2.

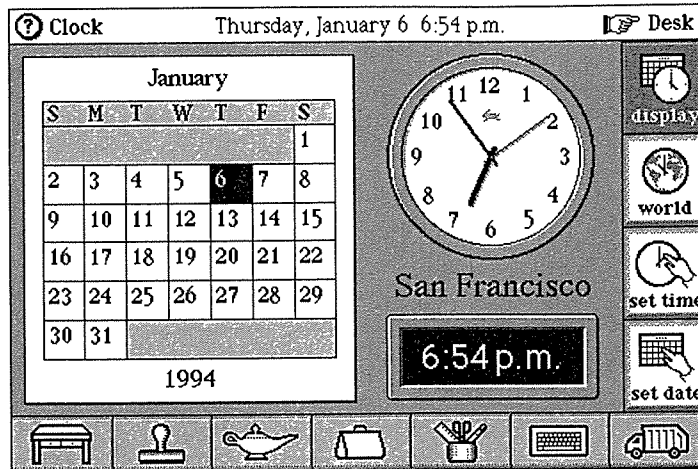


FIGURE 9-1. Clock scene shows date and time

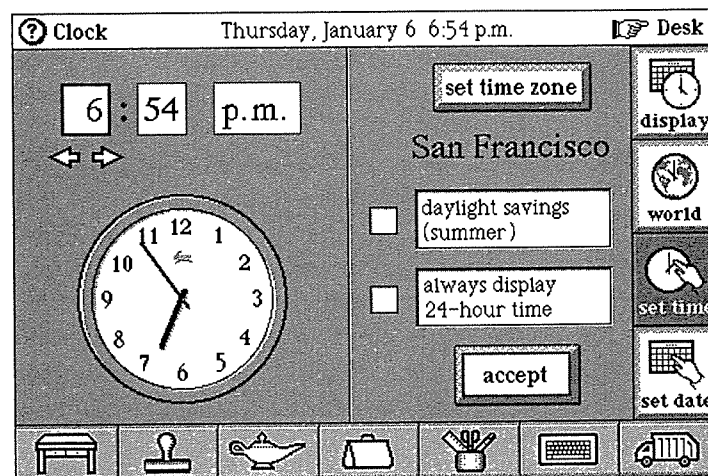


FIGURE 9-2. Set the time and indicate your time zone

The left side of the scene shows an analog clock. You can just slide the hands of the clock to set the time. A digital display above the analog clock has arrows that let you change the hours and minutes with more precision. The right side of the scene has a button at the top that lets you tell Magic Cap what time zone you're in by picking a city in your time zone. If you tap this button, you'll see a list of prominent international cities. Pick your city, or if the city you're in isn't listed, choose one that's in your time zone. You may have to settle for another city in your time zone instead of your exact location (sorry, Seattle).

Two check boxes on the right side of the scene complete your time-keeping options. If your city is observing Daylight Savings Time right now, tap the check box to set it. There's a check box you can use to always display 24-hour time in digital clocks and in text if that's how you want it. When you're done, tap *accept*, and the new time is set and the scene switches back to the clock display. In practice, you'll really have no need to set the time unless your communicator is brand new or all your batteries died and you need to start over again. However, when you travel, you can use the *set time* feature to choose the time zone you're in and the clock will change accordingly.

The *set date* button displays the calendar for the month, with today's date highlighted, as shown in Figure 9-3. This calendar looks just like the ones in other parts of Magic Cap, with arrows at the top to set the month and arrows at the bottom to change the year. To set the date, just use the arrows to pick the right month and year, then touch today's date on the calendar. Magic Cap knows about the international date line, so setting your new time zone when you travel will also take care of changing the date, if necessary. Unless you have fatal battery problems or a time machine, you won't need to reset the date.

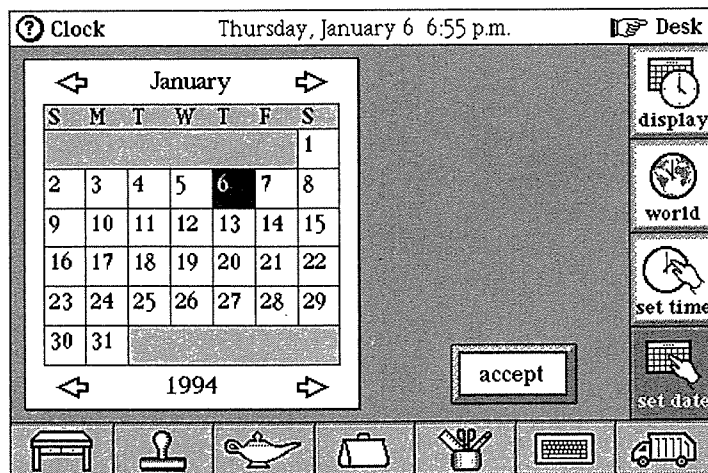


FIGURE 9-3. Setting today's date

The fourth button in this scene is the intriguing *world*, and if you look very closely at Figure 9-1, you'll see that the button has a picture of clock hands on a globe. This button shows you Magic Cap's world clock, which lets you answer that age-old question, "If it's 3:10 P.M. in Iowa, what time is it in Tokyo?" (Give yourself 25 bonus points if you knew that the answer is 6:10 A.M. the following day.)

The world clock, pictured in Figure 9-4, shows a map of the world with time zones drawn in, along with panels showing the time and day in four cities. The cities are also highlighted by points on the map itself. The world clock lets you see what time it is in four different places at once.

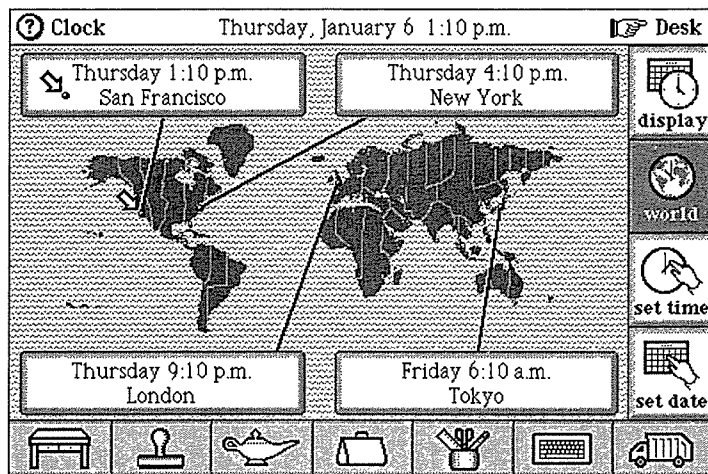


FIGURE 9-4. World clock shows the time in four cities

If you're not fond of the cities shown in the world clock, you can change them. Touch any of the time panels to select a different city from Magic Cap's list. It's the same list that you saw when choosing your time zone in the *set time* screen. (This is one of the few lists in Magic Cap that doesn't allow you to add your own entries.)

The map includes a funky arrow pointing to the city you chose for your time zone, kind of like those maps at the mall with a reassuring "you are here." If you set a panel to show a different city, the time in that panel changes to reflect the new city you chose. You can also select a city by holding down the option key while you touch a point on the map. The city nearest your touch goes into one of the panels.

Once you've selected a new city from the list, there's a check box you should set if that city is observing daylight savings time. When you touch *accept*, the panel and time are changed to that new location. By the way, you're not going crazy if you thought you saw something swimming in the waves of the South Pacific.

Desk Drawers and the Calculator

The desk includes two drawers for keeping important items you may need. The left drawer holds your stationery, easy enough to get to, but stashed away out of sight until you need it. The drawer on the right has room for miscellaneous desk accessories, including the calculator that comes with every Magic Cap communicator. The calculator is the only standard item that Magic Cap puts in the desk accessories drawer, but you might imagine this drawer also holding maps of major cities or sets of income tax forms.

The calculator, like other Magic Cap tools, gives you choices about how it can best meet your needs. Figure 9-5 shows you what the calculator looks like. With the choice box set to *Paper Tape*, the calculator prints all its work on a built-in paper tape. You can see your calculations as you enter them, which is great if there are a lot of numbers, as the tape holds 100 lines of information.

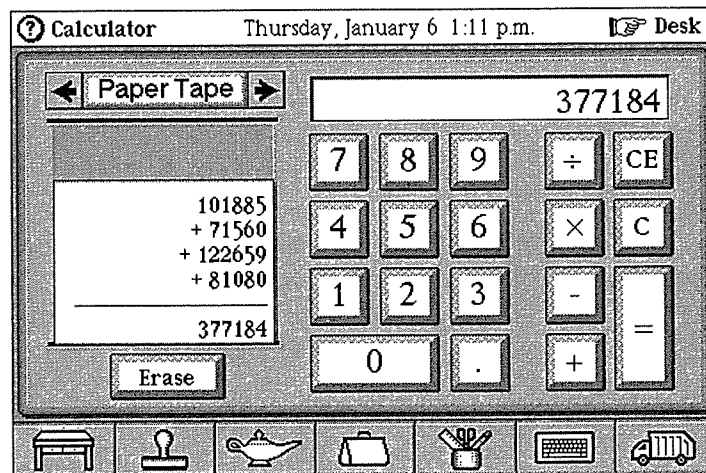


FIGURE 9-5. Magic Cap's calculator has a paper tape

If you hold down the option key, you can tear the results off the tape as a text coupon and put them in the tote bag. If you need to attach the calculations to a message to your boss about project expenses, you could create the new message and then slide the tape out of the tote bag to your message.

When you slide the coupon out of the tote bag, it automatically reproduces all the text from the tape, so your boss can see exactly how you calculated the expenses. You could also slide the paper tape out of the tote bag and onto your notes that are attached to the budget meeting scheduled in your datebook. You might also use the numbers when you slide them out of the tote bag onto a blank notebook page to include in your presentation at the budget meeting. You get the idea.



Copycat. You might remember that Magic Cap often uses option-slide to copy things, such as items in the tote bag, pages in the notebook index, or addresses on a name card. It works here, too; option-slide moves a copy of your calculations from the paper tape in the calculator. If you're going to use the same calculations in other places, option-slide the coupon out of the tote bag to move a copy of the tape. The original paper tape remains in the calculator until you press the button to erase it.

If you don't need the paper tape, you can set the calculator to Basic, which replaces the paper tape with several buttons. One particularly useful button on the basic calculator helps you figure tips. You'll find a place to set the tip percentage. Touch the plus sign to move up from 15% if you're a big tipper, or the minus sign if the service wasn't great. When the lunch bill comes, you can enter the total bill, adjust the percentage, and then hit the *tip*

button to quickly calculate the tip. Of course, you won't need this if there's someone at your table who can figure percentages in his or her head, which happens often in Silicon Valley where I live.

The third choice for the calculator is *Scientific*, which provides keys that engineers and scientists would expect to find on a scientific calculator, hence the descriptive name. For people who understand why there's a switch to choose between radians and degrees, along with buttons for arc, sine, cosine, and tangent, no further explanation is necessary about the power of the scientific calculator. For the rest of us who don't understand those things, no further explanation is necessary, either.

The Hallway and Controls

When you're not working at the desk in your communicator, you'll use the hallway as the corridor to get you between your desk and other interesting places, including downtown. The hallway includes several useful places behind its doors, as well as some whimsy. Magic Cap's designers obviously remember what was said about all work and no play.

When you're at the desk, you can get to the hallway by touching the step-back hand in the upper-right corner of the screen (see Figure 9-6). The directory on the wall opens with a touch. The directory isn't just a map of the rooms in your hallway—you can touch anything in the list, and you'll zip down the hall and watch the door open for you. You can get there in more leisurely fashion by touching the arrows on the floor and tapping the door yourself.

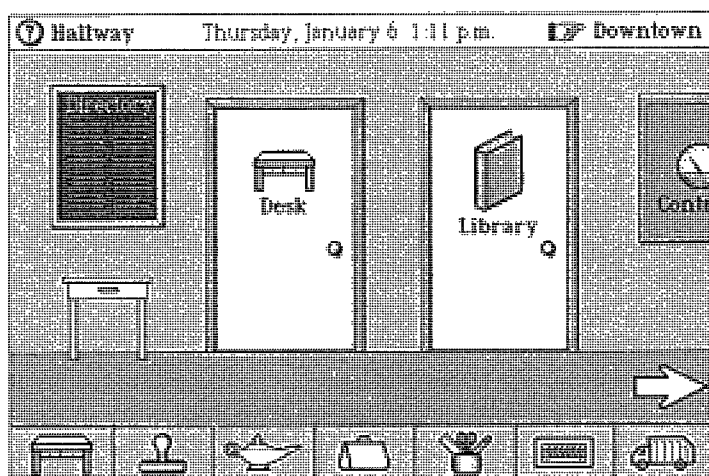


FIGURE 9-6. The hallway shows how to get to other rooms

Going through the hallway's first door takes you to back to your desk; of course, you can always touch the desk button at the lower-left corner of the screen to get to the desk. Touch the step-back hand while in the hallway to go directly downtown. Touch the arrow on the floor to move farther along down the hall so you can get to more rooms.

Along with the doors, there's a panel on the wall that leads to Magic Cap's controls. The control panel is the place where you can set preferences for the way your communicator behaves in certain situations. Inside the control panel are buttons that open scenes containing settings for different parts of your communicator, as you can see in Figure 9-7. You can also hold down the option key and touch the lamp from any scene in Magic Cap as a shortcut to the control panel. The control panel includes buttons for several sets of controls, including *general*, *screen*, *sound*, *power*, *privacy*, and *signature*.

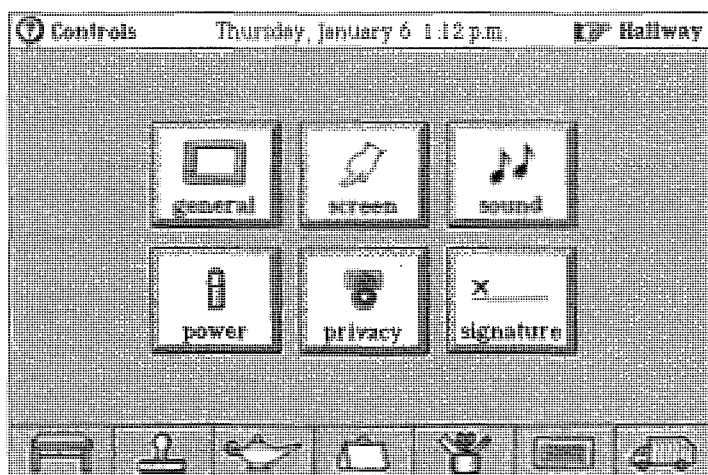


FIGURE 9-7. Buttons that lead to various control panels

The *general* panel includes settings to display items at the top of the screen in every scene. There's a check box to display the battery level of your communicator. You'll also find check boxes for displaying today's date and the current time at the top of the screen.

In addition to the three check boxes that let you show things at the top of the screen, there are three more check boxes for miscellaneous options. You can have your communicator ask for confirmation before you do permanent or destructive things, such as discarding a name card or setting the current user. If you are pretty sure about your Magic Cap expertise, you might turn off the warnings and take your chances. Of course, even if you fail to heed the warnings, you can usually pull recently discarded items from the trash.

Another check box lets you hear the phone dialing when you make a call. If you like to hear the number being dialed and the modems screaming at each other while the connection is being made, you can keep this

option on. The third option you can set in the general control panel is whether construction mode is turned on or off. With construction mode turned on, you have access to warehouses full of fun stuff in the magic hat, and you get the tinker tool for going behind the scenes and changing the behavior of items. For lots more in-depth information on Magic Cap's version of home improvement, you can peek ahead to Chapter 10.

The *screen* button only lets you adjust one thing—your touch screen. When you use this control panel, you'll repeat the initial screen setup that asks you to touch a target that hops around the screen. Use this control panel if your communicator seems to be feeling your touches in the wrong place.

Magic Cap uses sound effects as an integral part of the user interface; the sound effects were designed to make you feel more confident about the actions you're taking. These sound effects, along with the ability to play digitized songs, are why there's a control panel dedicated to sound. Some talented music-meisters worked on Magic Cap's sounds, including musician/composer/producer/interactive artist Todd Rundgren.

When you touch the *sound* button, which is decorated by musical notes, you see a screenful of sound effects along with the names of the actions that play the sounds. For example, there are sounds for a door opening and closing, a window going away, or a switch flipping. You can put other sounds here to change what you hear when you use Magic Cap. There's also a sliding volume control that goes from silent to almost obnoxious.

The *power* control panel gives you a graphic display of how much juice is left in your main and backup batteries, which is why there's an image of a battery on the button. This is also where you instruct your communicator how

long to wait before shutting off to save power. There's a check box that lets you decide whether the communicator should automatically shut off even if it's plugged in. Keeping this check box turned on will speed up the battery recharging time on some communicators.

The *privacy* button is aptly represented by a padlock. This control panel lets you set a security password for your communicator. You get to decide whether you need to protect your information from a nosy co-worker or a fearless eight-year-old. If you decide to go for it, there's a button to set your secret password on a telephone-style keypad. There's also a choice box that lets you designate how often you want your communicator to check security. You can instruct it to ask for a password every time it turns on, once an hour, once a day, or never.

The final control panel, *signature*, is certainly the most personal. This button takes you to a scene that lets you add your signature to Magic Cap (see Figure 9-8). Arrows point out the signature lines, one for just your first name and one for your full name, and you also see a smaller box that shows you a reduced view of your signature. Each line has an erase button so you can keep trying until you get it the way you want. After you sign your name, you'll find stamps available in the stamper with your personal signatures, available for your use on messages or anywhere else.

The control panel is always just a few steps away from your desk, and there will be times when you'll need to change a setting. Most of the time, the control panel will be the kind of place that's nice to visit, but you wouldn't want to have to live there.

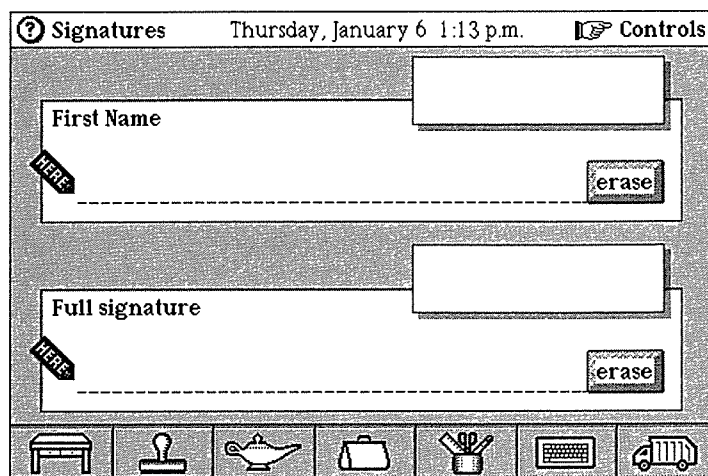


FIGURE 9-8. Adding your signatures

Library

Some futurists claim that computers and interactive materials will render printed books virtually obsolete. On the other side of the argument, traditionalists say that the powerful experience of reading a book can't easily be replaced by new technology. Magic Cap tries to balance these two views with the library, which you can visit by going down the hallway (see Figure 9-9).

The library is filled with books that really look and act like the books you're used to; it even has a card catalog that maintains an inventory of the books there. Of course, these are electronic replicas of books, with no paper to tear and no jackets to lose, only information-filled pages. Magic Cap takes advantage of the fact that books provide a familiar way to browse information by having tips for using your communicator written in volumes you can read in the library.

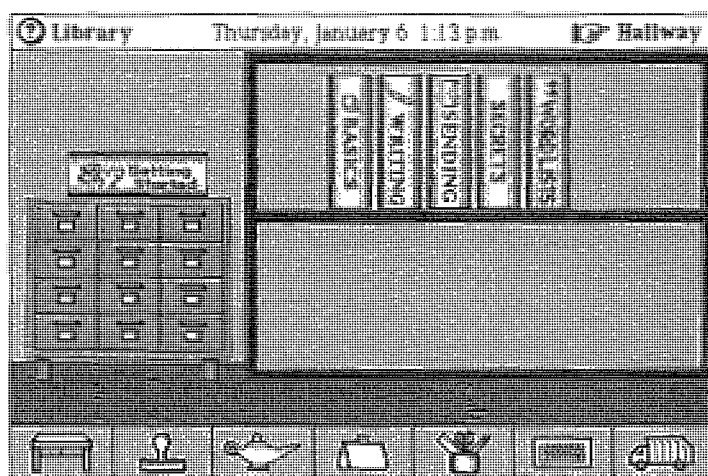



FIGURE 9-9. Your own personal library has room for many books

Back in Chapter 1 we looked at the *Getting Started* lessons that help you navigate through Magic Cap's setup procedures. You can find all those lessons in the *Getting Started* book in the library. When you open that volume, you see book pages that are not only literately written for a beginning user, but also buttons on each page that take you to the necessary place in Magic Cap and step you through each action to set up and teach you about your communicator.

 **Synchronicity.** If you're more savvy and you prefer to skip the lessons and set things up yourself, the *Getting Started* book knows how to update itself when you complete an action. For example, if you bypass the lessons and figure out that you can personalize your communicator by making a new name card for yourself, the corresponding page in *Getting Started* now tells you that your name card has already been set up, and it removes the step-by-step lesson button.

Touching the folded-down corner at the top of a page turns to the next or previous page of the book, and you shortcut fans can hold down option while touching the corner to move to the book's first or last page. Even if you passed all the *Getting Started* lessons when you first started using your communicator, the book stays on the library shelf for future reference in case you need a refresher course.

Books

Getting Started is not the only important reference in your library. There's a book called *Basics* that goes over some of the general features of Magic Cap. *Basics* provides a thorough introduction to your communicator's features, working together with the user manual to help you get the most out of your communicator (and depending on which model of communicator you have, the user manual and *Basics* may even have been written by the same person).

Each time you open a book in the library, there are three buttons that appear on the right side of the screen, as you can see in the *Basics* screen shown in Figure 9-10. The first one is *shelve*, which puts the book away for you when you're done. The second button, *contents*, turns the pages to show you the table of contents. As you might expect, the contents page doesn't just show you what topics are in the book—it actually helps you get to them. When you touch a chapter name or page number on the contents page, the book flips directly to the page you want.

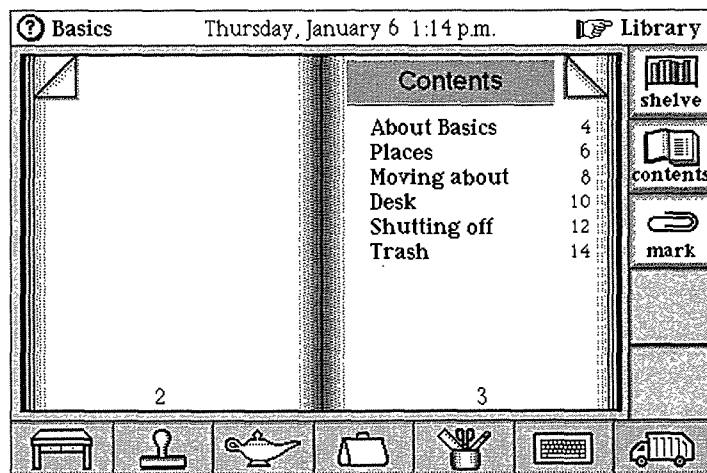


FIGURE 9-10. *Basics* book and its buttons on the right

The *mark* button, the third one down on the right, shows an image of a paper clip, and that tells you what it does: It clips a page for quick reference. It's similar to the physical world's neon yellow highlight pen, marking a page with a tiny paper clip on the page's entry in the table of contents and on the page itself. Unlike the yellow highlight marks, the paper clip mark is easily removed by touching the button again; it changes to *unmark* if the page is already marked. Also, these marks don't have that funny marker smell.

The library also includes a book that contains helpful tips on creating and editing messages. There's another book that includes some generic instructions on sending messages. *Secrets* is a fun book that contains a few examples of the "power user" features of Magic Cap. You don't need to know any of these tricks or shortcuts to effectively and efficiently use your communicator, but they're in there if you want to read about them.

Word Lists

The other standard reference in your library is *Word Lists*. This volume contains all the words your communicator uses for the automatic word-completion feature we saw in Chapter 5. For example, this is where you'll find the list of first names that Magic Cap can guess from to save you typing when you're entering a name card. There are lists of cities, states, and countries that are also used for auto-completing words.

The word lists can be built automatically or manually. You'll typically add words automatically: When you add new information to name cards, the words are added to the proper word list. Because you entered your name card, your last name was added to the Last Names word list. When you enter a name card for a family member with the same last name, the automatic completion feature will guess at the last name because you already taught it yours, as you may remember from our examples about Susan and Mark Anthony and their friend Hans Anderson in Chapter 5.

Every time you add a name card, the entry you type for first name, last name, company name, job title, city, state, and country will be added to the appropriate word list if it's not already there. This is how Magic Cap can remember a long name, like the law firm of Jacksons, Monk, and Rowe, so you only have to type it in once even though you'll enter several name cards for contacts you have there. Magic Cap communicators really do learn from you, and you don't have to teach them the lessons over and over again.

You can also add new words manually by opening the *Word Lists* book and touching the *add* button that each chapter includes. There's also a *change* button to correct spelling mistakes and a *remove* button to reclaim memory by getting rid of entries that you won't use.

Let's say you're a doctor and you need to use your communicator to jot down some notes about a patient when you're at home. Let's also say that you don't have the best handwriting, so you might want to type the note. Typing is tough on a communicator's small on-screen keyboard.

One of the tricks Magic Cap uses to reduce typing is a system for automatically expanding abbreviations. You can add abbreviations that you use often to the word lists, and then after you've typed them, you can expand them by touching the keyboard's *expand* key after typing the abbreviation. If you have to write notes about hospital patients, you might want to add an abbreviation to the word list. Adding abbreviations is straightforward: turn to the *Abbreviations* page, touch *add*, and then type in the abbreviation followed by the words it represents.

You might enter MMC to stand for Mercy Medical Center. Then, whether you're adding a name card for someone at the hospital or sending a message to another doctor who will be seeing your patients today, you can type *mmc* (upper- or lowercase are treated the same way), then touch *expand*, and you've cut your keystrokes from 20 to 4. You can enter as many abbreviations as you want to minimize your typing and take maximum advantage of your communicator.

Play, Art, and Work

As you go down the hallway, you probably noticed the *Game room* door with a playing card on it. This room was added to give you a break once in a while and provide a place for you to keep games. When you open the game room door, you'll see shelves with space for games and other fun packages. There's also one of those cool cat clocks on the wall that has the cat's swinging tail as

the pendulum, a subtle reminder of how much time is passing while you're playing games.

Magic Cap has a coin in the game room that you can flip by touching. Heads is the General Magic bunny in his hat, and tails is the rabbit's rear end. You can look forward to games from third-party developers to fill the shelves of the game room, and it'll be especially cool to see interactive games that use beaming or other kinds of communication so several people can play. Until then, the game room's coin will be handy when you have to choose where to go for lunch.

There are a couple of other things you will see in the hallway that aren't really essential, but they are appealing and fun (see Figure 9-11). There's a painting hanging on the wall. By touching the painting, you can switch between images, from Picasso to Matisse and back to Picasso. If you hold down the option key and touch the painting, you can have the name of the artist displayed below the painting for your less-cultured associates.

You can even have the picture change automatically every day, or add your own images to the rotation. While Magic Cap includes just these two masterpieces, a savvy developer could produce collections of image art for you to hang on your wall, courtesy of a memory card or an image gallery downtown.

Near the end of the hallway is a plaque that displays the General Magic logo. When you touch the plaque, you'll see a list of all the dedicated people who brought you Magic Cap and Telescript, plus some required copyright and legal stuff. You might not ever touch this sign, but you can remember that it's a credit to the people who are proud of their work and are thrilled to put their names on it.

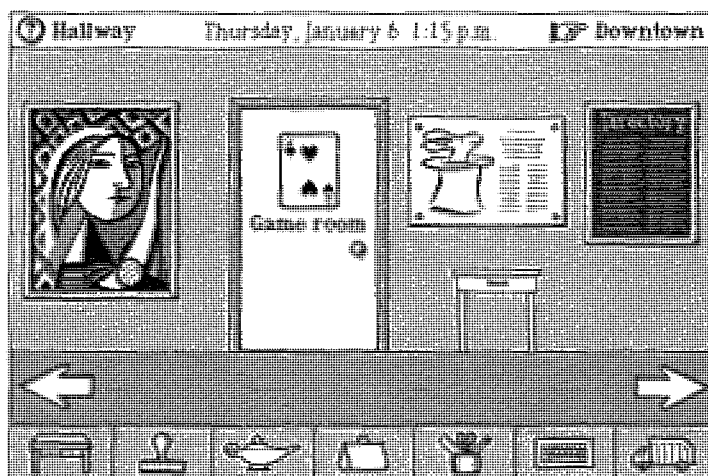


FIGURE 9-11. The hallway has a painting and a credits plaque

Summary

Magic Cap tries very hard to simulate a comfortable working environment. Many of the tools you'll use often are placed in, on, and around your desk. You can walk down the hallway to find other useful places, including rooms, control panels, and tables with drawers that open. You can stroll downtown, visiting stores that offer merchandise and services.

All of these places and lots of others have been designed to accept additional packages and services from third-party developers. For example, you might buy a package of stamps with licensed logos of Major League Baseball teams. When you install the package, it will automatically add another drawer to the stamper. You might buy a memory card with a check-writing package that knows how to move into your desk drawer when you

unpack it, or you might receive an electronic signup form for a news information service that builds a building downtown that you can visit to request new topics.

Every Magic Cap communicator includes a clock and a calendar. You'll set them once when you first set up your communicator, and you can reset them if you ever need to. The clock also includes a scene that shows the time in four international cities of your choice.

You can use the calculator in the desk drawer to perform simple computations, put numbers on a tape that you can move into a message or elsewhere, or handle scientific calculations.

The hallway has control panels that let you set preferences for how your communicator works. There are settings that let you display the date, time, and battery strength at the top of the screen all the time. Other controls turn on construction mode, extra warnings, and phone-dialing sounds. The control panels also help you adjust the touch screen, choose sound effects, determine whether the communicator should shut off automatically when not in use, set a password, and sign your name.

Magic Cap has a library that holds books about how to use the communicator. Some books have special features, like built-in lessons or lists of words that affect typing. Software developers can add more books with all kinds of information.

For people who need a diversion close at hand, there's a game room with a flipping coin and space for other fun packages. Finally, the hallway includes a magic painting that changes its images and a plaque listing the names of the people who created Magic Cap and Telescript.

Chapter 10



Construction

Get Your Hard Hat

I like Magic Cap just the way it is. What's not to like? The desk looks fine, I like the colors and sounds, and the tools for drawing are OK with me. Magic Cap's designers worked hard to include everything they believed would be useful to the average communicator owner, maybe someone like me. However, the designers wanted Magic Cap to have the power of a higher level of customization. While most of us would be horrified at moving objects that aren't supposed to move, some people want to be able to redecorate their scenes. These people like adding their own sounds to buttons, or putting together their own check boxes and buttons.

For these tinkerers who want to customize Magic Cap using more than just stamps, there's construction mode. If you enjoy a little power-using once in a while, this chapter may be helpful for you, especially if you figure out how it can save you some time.

Construction Mode

You can start doing construction by flipping on the *construction mode* check box in the general control panel. Turning on construction mode affects what you see in every scene. The first thing you'll notice is that the

233

stamper image on the bottom of the screen changes into an upside-down top hat, reminiscent of the hats used by magicians and their rabbits. When you tap the magic hat, a window opens to show you all the categories of stuff inside (see Figure 10-1).

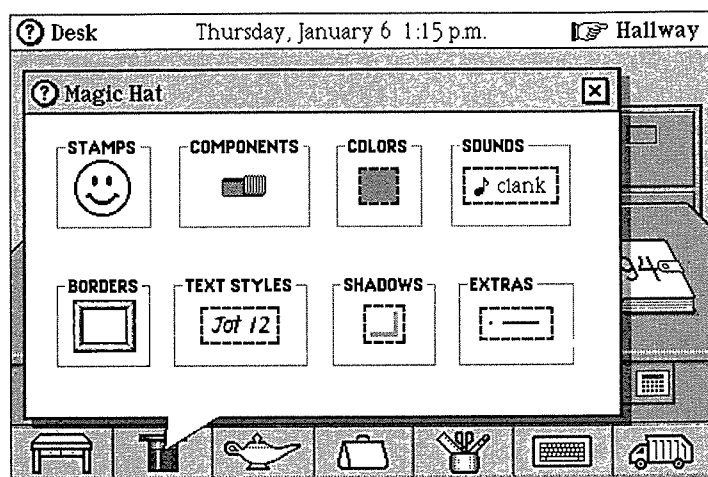


FIGURE 10-1. The magic hat shows its categories of items

When you tap a category's image, the screen changes to show you more of that category's contents. As you can see, stamps are just one category inside the hat; the happy face stamp represents the stamper and all its drawers, and they're still available just as they were without construction mode.

The second effect of construction mode is that you'll get access to more tools in every scene. For example, every scene gets to use the *move*, *copy*, and *stretch* tools, which are normally available only in certain scenes, such as the notebook, where you usually do lots of drawing.

When you're doing construction, you can use these tools with many more items, which is why they're available in every scene.

The third effect of construction mode is to add commands in the lamp. Various scenes will add their own commands to the lamp in construction mode; for example, because you can rearrange items on the desk in construction mode, you'll also get a *clean up* command for the desk. The only lamp command that's added for every scene is *snap*, which takes a magic snapshot of the current scene. You can move a snapshot elsewhere and use it to go back to the scene instantly, or slide things into the snapshot to send them to the scene it pictures.

In addition to producing the magic hat and adding tools and commands, scenes can be affected by construction mode in various custom ways. For example, when you're in construction mode the control panel adds dialing controls that let you set fine details like whether to speed up or slow down the dialing.



Trying it Out. You can get a preview of construction mode without actually flipping the switch. When you open the stamper, hold down the option key while you touch the stamps window's title bar. The magic hat window replaces the stamps window and you see what's in Figure 10-1. The effect is only temporary; after you make a selection and close the window, and then tap the stamper again to open it, it will be just stamps again. If you're going to use more than one effect from the magic hat, you might as well turn on construction mode from the control panel.

Custom Forms

Let's say it's your turn to be in charge of ordering lunch for the departmental staff meeting. Twelve people in your department depend on their communicators because they're out on the road quite a bit, so you know that sending an electronic message is the fastest way to reach them. You also know, though, that just because you send the message doesn't mean they'll take the time to write a complete message in reply.

To make it easier on them and you, you create a message that includes choice boxes so they can select their preferences for lunch and then easily send the message back with those choices. Start by touching the postcard on the desk and addressing it to the group *The Department*, which you created earlier. You type the message confirming the date, time, and location of the meeting, and then you ask them to respond with their choices for lunch.

Now for the fun part. With construction mode turned on, touch the magic hat to open it. In the magic hat, touch *components* to see the parts inside. The drawers are like those in the stamper; you want to open the one labeled *choices*. You touch it and see the page of choice boxes, meters, and sliding controls, as shown in Figure 10-2. Touch the choice box; by keeping your finger on it, you can stamp one out and slide it right into the position you want on the postcard.

If you option-touch the magic hat again, you get right back to the same drawer, where you can add two more choice boxes (this is a long meeting). Now you've got the message written and three choice boxes with meaningless choices. The next step is to make the choices mean something.

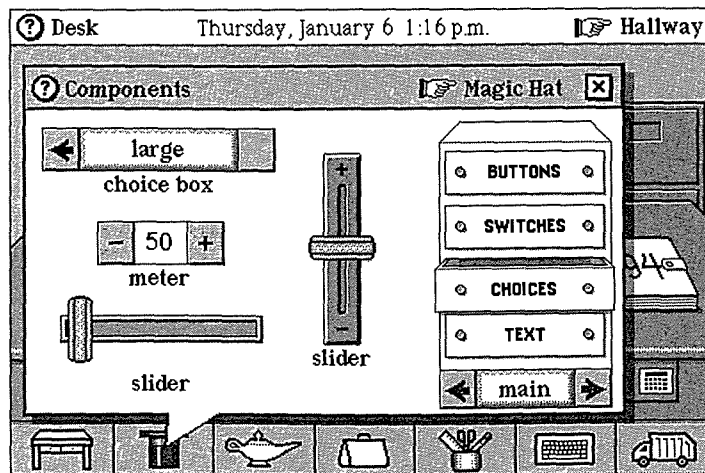


FIGURE 10-2. Choice boxes available in the magic hat

To start doing this, option-touch the keyboard to get to the label maker. Type *sandwich*, and then tear it off by sliding it over to the top choice box and dropping it right on the choice box's label. You'll hear the slurping sound that means the choice box has accepted the new coupon. The choice box is now called *sandwich*.

Go back to the label maker (option-touch the keyboard) to type the choices. Type these words and press *return* between each choice: *turkey*, *vegetarian*, *ham & Swiss*. Tear the coupon off, slide it over, and drop it right on the middle of the choice box. With a slurp, the choice box now has three new choices. Repeat the same steps for the middle choice box for salad and the bottom choice box for dessert. The result is shown in Figure 10-3.

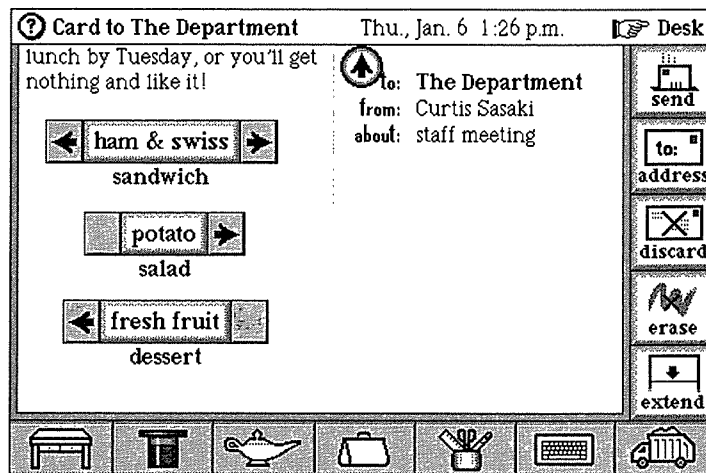


FIGURE 10-3. The message and its custom choice boxes

You've just custom-made three choice boxes that Magic Cap's designers couldn't possibly have programmed for you, and you did it with a minimum amount of work. Now you can send off the message and wait for replies. Each recipient can make choices on the message, and then touch *forward* to send it back to you. (Remember that forwarded messages have the original message attached.) Before the end of the day, you've received all 12 lunch orders for the staff meeting, and you didn't even have to leave your communicator's desk.

A Magic Invitation

The *components* drawers in the magic hat are full of lots of interesting things to play with besides choice boxes. In the *switches* drawer, you can use various switches and check boxes to make your messages more interactive. Let's try out a check box example.

Chapter 4 showed how you can get Magic Cap's datebook to automatically invite people to a meeting.

This time, we'll make a party invitation, and then use electronic mail to invite people manually. In this example, you want to throw a surprise birthday party for your spouse. Because it's a surprise, you have to avoid having people call your house with an RSVP. Here's a perfect way to use your communicator to not only send the invitations, but also get back the responses. Not all of your friends have Magic Cap communicators yet, so the check box won't work for everyone, but for the ones who do, responding to an invitation is a snap.

After you've designed the invitation, you can go to the magic hat *components* and choose a check box. By touching it and not letting up on your touch, you can slide it right onto the invitation. You option-touch the keyboard, type *I'll be there!* on the label maker, tear it off, and then slide it over and drop it on the check box to rename it. The invitation is ready to send off to your friends.

When your invitation is received, all the guest will have to do is touch the box to make the check mark, (or not change it at all if the answer is no) and then forward it back to you. The forwarded message has a copy of the original message attached, so when you receive the message back, you can see from the check box whether or not the sender can come. The party is more likely to stay a surprise because you are the only person receiving the replies.

Customizing Sounds and Images

We've seen the kinds of things you can do with items from the *components* drawers in the magic hat. Now, let's look at some of the other things construction mode can help you do.

Back in Chapter 7, we saw how a creative kid could make a birthday card with sound and animation. We

stamped a cartoon onto a notebook page, then got another stamp that played a song, and dropped the song stamp onto the animation, in effect adding a soundtrack to the cartoon. In construction mode, we can expand on the idea of adding sound effects to actions.

One of the sounds heard most often in Magic Cap is the click that's made when you touch something to open it or tap a button along the bottom of the screen. You've decided you're getting a little bored with that sound and you want to take advantage of some of the other sounds Magic Cap plays. The magic hat has a set of drawers labeled *sounds* filled with sound coupons that you can drop anywhere to replace the factory-set sounds.

Touch the magic hat to open it, and then touch *sounds*. Inside is a drawer called *instruments*, which is where you decide to begin your customizing. You touch the *clarinet* stamp, and then slide it off onto the desk button in the lower-left corner. Every time you touch the desk button from now on, you'll hear a quick blast on the clarinet instead of the old click. Like a mad composer, you continue on across the screen: the stamper gets a piano note, the lamp plays like a guitar, the tote bag becomes a trumpet, the tool holder sounds like the string section of an orchestra, the keyboard gets the snare, and the trash truck sounds like a crashing cymbal when you throw something away. Talk about your *personal* communicator.

There are other items in the magic hat that affect images. The coupons for colors, borders, text styles, and shadows all change the way Magic Cap items look. You can use these tools to vary the text styles that the keyboard types, change the coloring or shadowing of an object, or choose different kinds of borders for the things that use them.

Magic Hat Extras

The final category in the magic hat is called *extras*, and it offers all the goodies that didn't fit anywhere else, which means it includes some of the highest-level customizing features in Magic Cap. When you touch *extras*, you'll see drawers that let you change tools in the tool holder. You can add various kinds of line styles or shape types to the tool holder just by touching the appropriate coupon and sliding it into the tote bag. Then, when you open the specific set of tools in the tool holder, slide the coupon into that window; it becomes a choice you can select the next time you're using the tools.

Another interesting drawer in *extras* is the one called *properties*. When you touch this drawer to open it, you'll see coupons that affect the way items appear on the screen (see Figure 10-4). You can choose coupons to hide items or show everything that's been hidden. There are coupons that layer objects by bringing them closer to the screen or sending them to the back. Two opposing coupons set an item so you can either throw it away or keep it from being thrown away. Other coupons rotate objects 90 degrees or flip them 180 degrees.

The *flip* and *rotate* coupons work on shapes. You could slide a *rotate* coupon into the tote bag, choose an arrow tool, and draw an arrow on a blank notebook page. As we saw in Chapter 7, you can draw an arrow pointing left or right, depending on which way you slide your finger as you're drawing it. You can now take the rotate coupon out of the tote bag, drop it onto your sideways arrow, and suddenly it points up or down. This could be useful the next time you draw a map.

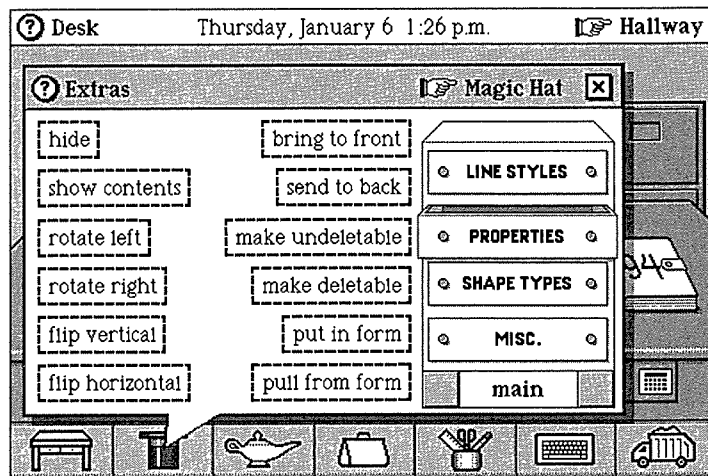


FIGURE 10-4. The *properties* drawer in magic hat *extras*

Tinkering

Now that you've seen some of the uses for construction mode, let's take a closer look at the ultimate construction item, the *tinker* tool. This tool, which looks like a wrench, appears in the *authoring* tool set along with *move*, *copy*, and *stretch*. With construction mode turned on, touch the tool holder, and then touch the wrench. It hops into the tool holder and becomes the active tool.

Tinkering lets you see inside objects and change things that otherwise wouldn't be changeable. When you touch an item with the wrench, you'll see a window that tells you something about that item in terms of coupons and settings. Let's go to the painting in the hallway and tinker with it. Your goal is to take it off the wall in the hallway and move it behind the desk so you can see it more often. This may take quite a few steps, but for a person who likes to tinker, getting there is most of the fun.

The first thing to do is choose the *move* tool and try sliding the painting into the tote bag. It doesn't work; the factory setting for the painting prevents you from moving it. Let's try tinkering with it to see what might be wrong. With construction mode turned on, touch the tool holder, go to the *authoring* tools, and then touch the wrench. It hops into the tool holder, becoming the active tool.

Now let's open up the painting and look behind the scenes. Touch the painting; instead of changing the art, as normally happens when you touch it, you see a window that shows some interesting stuff about the painting, as shown in Figure 10-5. There are coupons for the border and the sound; some controls for the painting's label; and check boxes that indicate whether it can be moved, copied, or stretched.

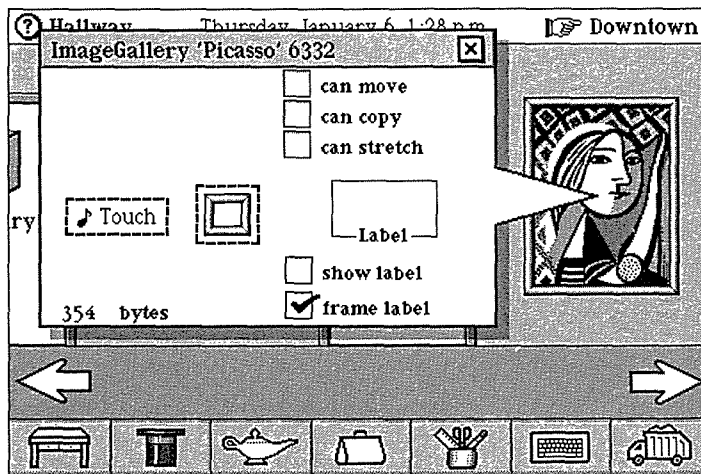


FIGURE 10-5. The painting's tinkering window tells all about it

This is pretty interesting: The *can move* check box is turned off. Could that be what's preventing us from moving the painting? Touch the *can move* box to check it, and then close the window. Go back to the tool holder to get the *move* tool and try sliding the painting into the tote bag. It works! Just like in a magic show, the whole painting fits comfortably in that little tote bag.

Now tap the desk button to go back to the desk. To make room for rehangng the painting on the wall behind the desk, you have to move the in box and the out box over a bit so they don't cover up the image. You get to tinker with two more things, using the *move* tool to move them one at a time.

When there's enough room on the wall, you can schlepp the painting out of the tote bag to its new home behind your desk. Uh, oh. It's too big, and it covers up half of your notebook. Let's try using a coupon from the magic hat's *extras* to send it to the back so it gets tucked in behind your desk. Open the magic hat, touch *extras*, and open the *properties* drawer. Touch *send to back*, slide it onto the painting, and—presto! The Picasso hangs behind your desk. If you touch it, it switches to the Matisse, just as when it was hanging in the hallway (see Figure 10-6). Looks like you could use a smaller desk, though.

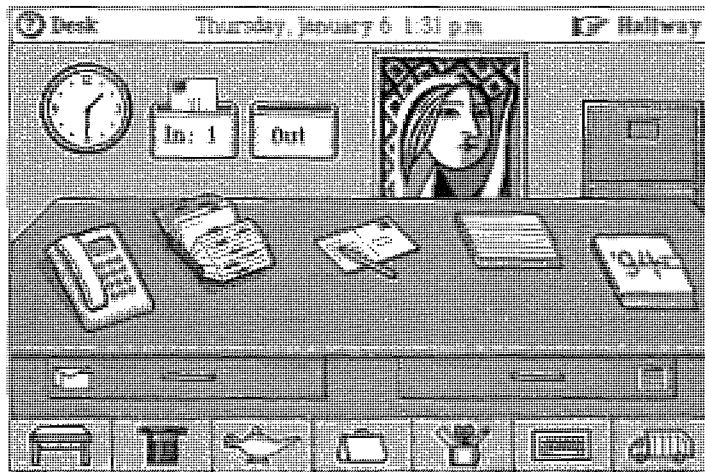


FIGURE 10-6. The painting has been rehung behind the desk

Another Tinkering Project

As long as we're redecorating, let's change the image on the library door. You could try stamping a happy face on the door, but if you did, the stamp would just end up going inside the room. You need to get a coupon that has the image of the happy face stamp, and then replace the image on the door with your custom-selected image. Okay, tinkers, start your wrenches.

The first thing you do is get the stamp that has the face image. As long as you're going to be changing the library door, you might as well just put it on the floor in the hallway (see Figure 10-7). Now, get the wrench from the tool holder, and then touch the stamp again to see its secrets. Slide the stamp's image coupon out of the window and onto the library door.

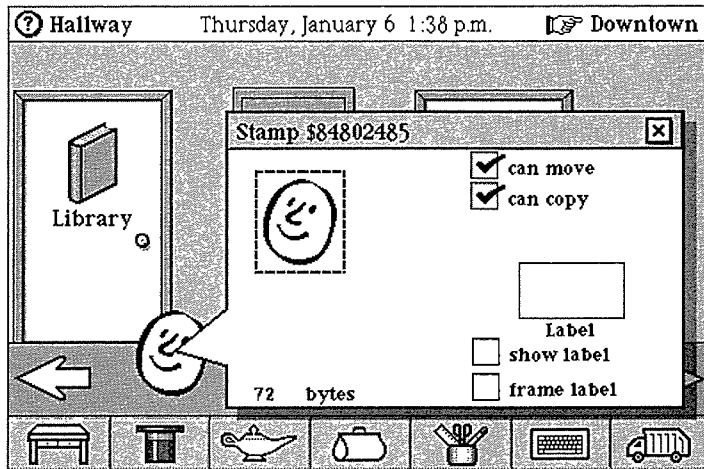


FIGURE 10-7. Tinkering with the happy face stamp

You'll hear the slurp sound, see the visual highlighting effect, and the next thing you know, your library door has a smiling face. Because you should always clean up when you're done, you can throw away the stamp you stole the image coupon from. You're now ready for your next real task, or even more tinkering, if you prefer.

You can look forward to bright third-party developers using tinkering and construction to whip up some impressive accessories and packages for your Magic Cap communicator. Maybe you've even got some ideas of your own about what you'd like to see. Until those accessories are available, you can use construction tools yourself to spruce up and personalize your communicator. You've probably guessed by now that there's no end to the messing around you can do customizing your communicator with construction mode and tinkering.



There's More to Life Than This. For serious package developers, General Magic offers a powerful computer-based development kit. This kit uses object-oriented programming tools along with Magic Cap's construction features to help programmers build software packages.

Summary

You turn construction mode on with a check box in the general control panel, or temporarily by option-tapping the stamp window's title bar. When you're in construction mode, the stamper image turns into a magic hat and contains not only stamps, but also many other pages of goodies that you can choose from to customize Magic Cap. Construction mode affects what you see in every scene: It adds more tools to the tool holder, more commands to the lamp, and more controls to the control panel. Construction mode lets you use the *move*, *copy*, and *stretch* tools in every scene.

Construction mode gives you components to add to your messages to make them more useful. You can add choice boxes, check boxes, and switches, and you can customize them. You can add custom sounds to any button or other item. You can get special coupons that rearrange the order and features of items on the screen.

Another part of construction is the *tinker* tool, a wrench that's one of the *authoring* tools when you're in construction mode. The wrench lets you look inside items and see their behavior and settings, including their image and sound coupons and check boxes for allowing moving, copying, and stretching.

You can make something movable by tinkering with it, and then use the move tool to put it somewhere else. You can take a stamp's image coupon and use it to

customize some other item's image. You can use the construction and tinker tools to change factory settings so that you can make your personal communicator really personal.

Finally

Writing this book was both intriguing and intimidating. The software was so deeply integrated that every time I started an explanation, several other aspects also had to be tied in. Did I mention something in a scenario in Chapter 2 that I didn't explain in depth until Chapter 9? How could I set up practical examples when the software was still changing?

As I used the many features and experimented with different examples so that I could write about them, I found myself getting more and more excited about Magic Cap and how useful it would be in real life. I actually began thinking in terms of having a Magic Cap communicator with me all the time, and how I could use it to schedule meetings, keep addresses handy, jot down notes, and, most important, communicate with people easily and conveniently. Writing this book reaffirmed for me how Magic Cap will change my life, and I tried to share that sense of excitement as I wrote. Magic Cap has great potential to expand and enrich the way people communicate.

Index

- Actions
 - reversing, 72
 - sounds, 4
- Addressees
 - adding, 43
 - multiple, 43
 - replacing, 43
 - types, 43–44
- Addresses, 9
 - name cards, 120, 122–123
- Addressing commands
 - window, 44
- Alarm+ button, 104–108
- Animation, 3
- Anniversaries, 81–82
- Announcement window, 62–63
- Appointments
 - alarms, 109–110
 - customizing, 108–110
 - datebook, 80–85
 - multi-day, 107
 - overlapping, 101–102
 - priority, 106–108
 - recurring, 83–85
 - status, 106–108
- Automatic word-completion lists, 228–229
- Basics book, 226
- Bassett hound and searching, 71
- Batteries
 - displaying status, 222–223
 - installing, 3
 - viewing level, 68
- Birthdays, 80–81
- Books
 - Basics, 226
 - creating and editing messages, 227
 - Getting Started, 225–226
 - marking, 227
 - power-user features, 227
 - putting away, 226
 - sending messages, 227
 - table of contents, 226
- Bottom of screen lesson, 14
- Business letter
 - addressing, 31–32
 - customizing, 44–45
 - making it standard, 46
 - salutation, 30
 - sender's name, 30
 - sending, 31, 33
- Business trips, 97–98
- Buttons, 68–74
 - alarm+, 104–108
 - contact, 70
 - desk, 68–69, 220
 - fax, 70
 - file, 71
 - find, 71
 - garbage truck, 73–74
 - infrared beam, 70
 - lamp, 70–72
 - mail, 70
 - option key and, 74
 - print, 71–72
 - revert, 72
 - stamper, 69
 - toolholder, 72–73
 - tote bag, 72
- Calculator
 - attaching tape to message, 218
 - figuring tips, 218–219
 - paper tape, 217–218
 - scientific, 219
- Calendar, 214
- Choice boxes, 65
 - customizing messages, 236–238
 - status, 106
- Clock, 212–214
 - daylight savings time, 214
 - setting, 212–214
 - time zone, 214
- Clock scene, 15, 61
- Confirmation
 - announcement window, 63
- Construction mode, 75, 233–246
 - adding lamp commands, 235
 - categories within, 234
 - custom forms, 236–239
 - magic hat, 234
 - on/off, 222
 - previewing, 235
 - snapshots, 178–179
 - sound and image customization, 239–240
 - sound drawers, 240
 - tinker tool, 242–246
 - tools for scenes, 234–235
- Control panel
 - general panel, 221–222
 - power panel, 222–223
 - privacy panel, 223
 - screen panel, 222
 - signature panel, 223
 - sound effects, 222
- Coupons, 16

Current scene, 61
 Custom forms, 236–239

 Date, 68, 212–216
 setting, 11–13, 214
 Datebook, 77–110
 adding/removing
 options, 82–83
 anniversaries, 81–82
 attaching notebook
 page, 172
 birthdays, 80–81
 business trips, 97–98
 changing images, 102–
 103
 customizing, 102–103
 daylight savings time
 and, 79–80
 entering appointments,
 80–85
 general purpose
 meeting, 103
 holidays, 79
 meeting integration, 93
 multiple-day events,
 97–101
 on-screen
 announcement, 109
 overlapping
 appointments, 101–
 102
 phases of the moon, 80
 recurring
 appointments, 83–
 85
 reminders, 104–106
 rule customization,
 108–110
 scheduling meetings,
 85–93
 to do lists, 93–97
 vacations, 98–101
 views, 78–79
 Daylight savings time,
 214
 datebook, 79–80
 Delivery choice, 36–40
 Desk, vii–ix, 5
 drawers, 217
 returning to, 12
 Desk button, 68–69, 220

 Directory, 219
 Drawings, 161–163
 discarding, 162
 erasing, 161–163
 tools, 72, 73

 E-mail drawer, 131–
 132
 Electronic mail, 19–54
 addresses, 131–133
 announcing in-coming,
 52–53
 arriving, 50–53
 business letter, 30–33
 customizing messages,
 42–47
 delivering messages,
 36–41
 directory lookup, 38–
 40
 future, 55–56
 gateways, 132–133
 PersonaLink, 21–22
 replying and
 forwarding, 28–29
 responses from
 meeting attendees,
 92
 sending messages, 24–
 28
 setting rules before
 receiving, 53
 signing up for
 services, 20–23
 sorting, 53
 staying in touch on the
 road, 30–33
 store-and-forward
 capability, 54
 Expand key, 73
 Extended keyboard, 73

 Fax button, 70
 Faxes, 40–41
 graphs, 175
 notebook, 167–168
 File button, 71
 File cabinet, 185–209
 adding drawers, 187–
 188
 choice box, 187
 customizing stamps,
 199
 drawers, 186–188
 file-by-stamp
 procedure, 197–198
 filing messages after
 reading, 195–199
 filing outgoing mail,
 199–200
 folders, 186–187, 189–
 190
 manually filing
 messages, 193–195
 moving folders, 191
 moving items between
 folders, 190
 sorting in-coming
 mail, 192–193
 storeroom, 201–209
 File command, 71, 137
 Find command, 71, 137
 Forwarded messages,
 238

 Game room
 storing games, 229
 Garbage truck, 73–74
 Gateways, 21, 37–38,
 132–133
 General panel
 construction mode on/
 off, 222
 controlling scene
 display, 221
 hearing phone while
 dialing, 221–222
 Get signup form button,
 22
 Getting Started, 4
 adding your name, 9
 address and phone
 number, 9–10
 desk, 5
 entering information, 6
 fine-tuning screen
 alignment, 3
 Getting Started book,
 7–10, 18, 225, 226
 lessons, 6–18
 on-screen keyboard, 8–
 10

personalization, 5–6, 9
 required setup, 8–11
 setting time and date, 12–13
 set up dialing, 10
 touch sensitivity, 3
 Graphs, 173–175
 faxing, 175
 labeling, 174
 points, 174–175
 printing, 175
 Groups
 adding members, 134–135
 name cards, 133–137
 sending messages to, 134–135
 stamped, 135–137
 Hallway, 219–230
 control panel, 220–223
 desk, 220
 directory, 219
 game room, 229–230
 library, 224–229
 listing Magic Cap's creators, 230
 painting, 230
 step-back hand, 219
 Help Books package, 204
 Images, customization, 239, 240
 In box, 50–53
 announcing in-coming mail, 52–53
 filing messages, 194–195
 reading messages then filing, 195–199
 sorting in-coming mail, 53, 192–193
 Information
 searching for, 71
 writing down, 160–161
 Information network
 offers, 13–14
 Infrared beaming, 41, 70
 Interface, 59
 Keyboard. *See* on-screen keyboard
 Label maker, 47, 73, 76, 182, 237, 239
 Lamp, 70–72
 adding buttons, 72
 adding commands, 235
 Lessons, 6–18
 adding signature, 15–16
 bottom of screen, 14
 date and time, 11–13
 finishing, 17–18
 getting started, 9–14
 keyboard, 15
 library, 225–226
 making phone calls, 16–17
 quitting, 7
 sending messages, 16
 signing up for PersonaLink service, 13
 top of screen, 14
 Library
 books, 224–227
 lessons, 225–226
 Word Lists, 228–229
 Lists, 175–180
 deleting items, 176
 editing, 176
 naming, 175
 rearranging, 176
 sorting, 175
 two-column, 179–180
 Magic Cap
 consistency, 65, 172
 development kit, 247
 navigation, x, 60–65
 personal communication and, xiv
 Magic hat, 234
 drawers, 236
 Mail button, 70
 Meetings
 adding names, 87–88
 electronic mail responses, 92
 integration, 93
 inviting people, 89–91
 postcards, 89–90
 prioritizing, 88
 responses from attendees, 91–92
 scheduling, 85–93
 selecting attendees, 86–88
 sending messages, 90–91
 Memory cards, 206–208
 archiving data, 206–207
 extending main memory, 207–208
 viewing what is stored there, 208
 Messages
 addressees, 43–44
 addressing, 26
 arriving, 50–53
 book on creating and editing, 227
 choosing tools, 42–43
 clearing, 52
 creation, 25–26
 customizing, 42–47
 delivering, 36–41
 faxes, 41
 filing, 51–52, 195–200
 filing rules, 49, 50
 forwarding, 28–29, 54, 238
 groups, 134–135
 in box, 50–53
 infrared beaming, 41
 logging, 49
 mailing, 27–28
 manually filing, 193–195
 offers from information networks, 13–14
 out box, 47–50
 replying, 28–29
 return to sender, 39
 sending, 24–28
 sending lesson, 16
 signing, 27
 sorting, 51, 196–200

- stationery, 42
- summary, 52
- typing, 26
- what happens after
 - they are sent, 49–50
- when to send, 48–49
- writing, 26
- Move tool, 243–244
- Multiple-day events, 97–101, 107
- Name card scene, 15
- Name cards, 80, 113–138
 - adding, 129–131
 - adding addresses and telephone numbers, 120, 122–123
 - alphabetizing, 126
 - automatic completion of words, 116–120
 - automatically dialing phone, 148
 - certified slash, 130
 - changing, 123–124
 - commands, 123
 - communicating with, 114
 - companies, 131
 - current contact, 127–129
 - customizing, 137–138
 - deleting, 125
 - dialing telephone, 137
 - duplicating, 124
 - entering, 115–123
 - first and last names, 121, 126–127
 - groups, 34, 133–137
 - locating, 125
 - log button, 125
 - moving around in, 137
 - new command, 123
 - notebook, 167
 - on-line services, 131–133
 - searching, 137
 - stamped groups, 135–137
 - storing information, 137–138
 - tabs, 125–126
 - viewing, 125–126, 148
- Name chooser window, 25, 62, 87, 146
- tabs, 87
- Navigation, 60–65
- Notebook, 159–182
 - arranging items on page, 165–166
 - attaching pages as message enclosures, 171–172
 - birthday cards, 168–170
 - custom forms, 180–182
 - customizing text, 169
 - drawings, 161–163
 - duplicating items, 166
 - erasing, 163
 - faxes, 167–168
 - graphs, 173–175
 - index, 170–172
 - line tools, 163–164
 - lined paper, 173
 - lists, 175–180
 - name cards, 167
 - pencils, 160–161
 - reshaping items, 166
 - sending pages, 166–168, 170
 - stationery, 162
 - text, 164–165
 - writing, 160–161
- On-line services
 - addresses, 131–133
 - gateways, 21
 - Internet, 133
 - name cards, 131–133
 - signing up, 22–23
- On-screen keyboard, 8–9
 - capitalization, 10
 - changing from letters to numbers, 73
 - extended, 73
 - lesson, 15
 - opening, 73
 - smart guesses, 10
- Option key, 2, 35, 43–47, 57, 67, 73–74, 80–81, 84, 88, 124–125, 135, 137, 147, 163, 169, 172, 182, 187, 191, 216, 218, 220, 226, 230, 235–237, 239, 247
- buttons and, 74
- Out box
 - filing messages from, 199–200
 - what happens after messages are sent, 49–50
 - when to send messages, 48–49
- Packages, 202
 - closing, 204
 - documentation, 205
 - information about, 203
 - installing items from, 205
 - memory cards, 206–208
 - opening, 204
- Passwords, setting, 223
- PCMCIA cards. *See* memory cards
- Personal Computer, backing up data, 209
- Personal letters, 45
- PersonaLink, vii, 21
 - delivery choices, 39
 - electronic mail, 21–22
 - going through the switchboard, 36
 - multiple message matches, 39–40
 - paging you, 54–55
 - signing up for, 13
- PersonaLink (trying directory) stamp, 39–40
- Phones, 16–17, 141–157
 - access codes for long distance carrier, 156
 - adding stamps, 156
 - area codes, 154
 - automated response systems, 152

- automatically dialing, 114, 142
- billing information, 156
- connected or not, 142
- dialing, 137, 142–148
- filing log entries, 150
- hanging up, 144–145
- hearing while dialing, 221–222
- incoming calls, 150
- keypad, 142–143
- location, 155
- logging calls, 149–150
- manually dialing, 146
- numbers, 9, 17, 120, 122–123, 144
- prefixes necessary, 155
- services button, 152–153
- setting up, 153–157
- speed-dial button, 146–147
- stamps, 154
- status window, 17, 145
- telling communicator where dialing from, 151
- tone or pulse, 157
- viewing call description, 150
- Postage stamp, 32, 36–37
- Postcards, 44
 - meetings, 89–90
 - urgent, 45
 - vs. letters, 33
- Power panel, 222–223
- Power-user features, 227
- Print button, 71–72
- Privacy panel, 223
- Reminders in datebook, 104–106
- Reply button, 28–29
- Required setup
 - address, 9–10
 - dialing locations, 10–11
 - name, 9
 - phone number, 9–10
 - Set up dialing, 10
- Return to sender message, 39
- Revert button, 72
- Save stamp, 201
- Scenes, 15, 60–65
 - additional tools, 234–235
 - adjusting objects, 244
 - help with, 67
 - information about, 67
 - message, 62
 - moving objects between, 72, 242–244
 - name card, 15
 - postcard, 15
 - redecorating, 245–246
 - stepping back, 67
- Screen
 - adjusting, 222
 - bottom, 14, 67–74
 - buttons, 68–74
 - changing appearance, 241
 - current scene, 15
 - date and time, 68
 - fine-tuning alignment, 3
 - information about, 5
 - top, 14, 67–68
- Screen panel, 222
- Shapes, flipping and rotating, 241
- Signature, 15–16
 - entering, 223
- Signature panel, 223
- Smart envelopes, xi
- Smart integration, 65
- Smart messages, xi
- Snapshots, 178–179
- Software packages, 201
- Sorting criteria window, 196
- Sound, 4, 66–67
 - appointments, 109–110
 - customization, 222, 239–240
 - in-coming mail, 52–53
- Stamped groups, 135–137
- Stamper, 10–11, 69
- Stamps
 - electronic mail, 132
 - phones, 122
 - save, 201
 - signature, 16
 - urgent, 45
- Stationery, 42, 44–45
 - customizing, 46–47
 - notebook, 162
- Stationery drawer, 46
- Status choice box, 106
- Step-back hand, 219
- Stepping back, 67
- Storeroom, 201–209
 - connecting to personal computer, 209
 - directory, 208
 - Help Books package, 204
 - packages, 202–208
 - software packages, 201
- Stylus, 4
- Summary button, 52
- Telephone. *See* phone
- Telescript, xi–xiii, 20–2
 - engines, 22
 - smart messages, xi
- Text
 - customizing, 169
 - notebook, 164–165
- Text tools
 - arranging items on page, 165–166
 - duplicating items, 166
- Throwing away items, 73–74
- Time, 68, 212–216
 - setting, 11–13
- Tinker tool, 242–246
- To do lists, 93–97
 - adding tasks, 94
 - deadlines for tasks, 94
 - finished tasks, 96
 - reminder notes, 95
 - task time-limit, 97
- Toolholder, 72–73
 - changing tools, 241

line tools, 163–164
 shapes tools, 168–169
 text tools, 164–165
 Tools, customizing, 42–43
 Top of screen lesson, 14
 Tote bag, 72
 Touch, 2, 4
 setting sensitivity, 3
 stylus vs. fingers, 4
 Typing messages, 26
 Urgent stamp, 45
 Visual effects, 66–67
 Windows, 15, 60–65
 accept button, 64
 addressing commands, 44
 announcement, 62–63
 closing, 64–65
 confirmation
 announcements, 63
 fax, 41
 file, 193
 information about, 5
 name chooser, 62
 phone status, 17, 145
 sorting criteria, 196
 Word Lists book, 117, 228–229
 adding/removing words, 228
 automatically expanding abbreviations, 229
 World Clock, 215–216
 Writing
 messages, 26
 tools, 72–73

General Computing

Presenting MAGIC CAP™

The new generation of personal intelligent communicators from leading companies such as Sony and Motorola allow you to communicate electronically with undreamed-of ease. At the heart of these hand-held devices is Magic Cap™, General Magic's revolutionary software platform designed by a talented team headed by Bill Atkinson and Andy Hertzfeld, the brains behind Macintosh® system software.

Presenting Magic Cap introduces the capabilities of this innovative system by stepping through Magic Cap's screens, showing how it works, and suggesting ways to use it. The book describes the power of smart messaging to customize the way you handle electronic mail, a name card file that learns how to keep track of your contacts, a datebook that performs like a faithful assistant, and countless other abilities.

Magic Cap has the usefulness and simplicity to attract the total novice, combined with the sophistication and intelligence to enthuse even the most jaded computer professional. *Presenting Magic Cap* shows how to take full advantage of the hidden power of Magic Cap.

Barbara Knaster is a freelance writer who was one of the first testers of Magic Cap. She lives in Campbell, California.

Cover design by Grand Design/Boston
The Magic Cap logo is a trademark of General Magic



Addison-Wesley
Publishing Company

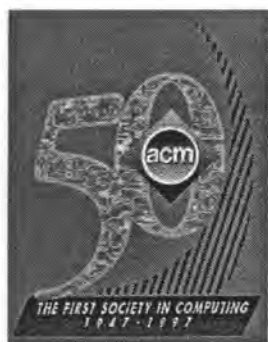


ISBN 0-201-40740-X

\$16.95 US
\$21.95 CANADA



1997 International Conference on Intelligent User Interfaces



Editors

Johanna Moore, Papers & Program Chair

Ernest Edmonds, Panels

Angel Puerta, Papers & Debate

Sponsored by:

ACM SIGART - Special Interest Group on Artificial Intelligence

ACM SIGCHI - Special Interest Group on Computer-Human Interaction

IUI97 January 6-9, 1997 - Orlando, Florida, USA

The Association for Computing Machinery
1515 Broadway
New York, N.Y. 10036

Copyright © 1997 by the Association for Computing Machinery, Inc.(ACM). Permission to make digital or hard copies of all of this work for personal or classroom use is granted without fee provided that the copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, to republish, to post on servers or to redistribute to lists requires prior specific permission and/or a fee. Request permission to republish from: Publications Dept. ACM, Inc. Fax +1 (212) 869-0481 or <permissions@acm.org>
For other copying of articles that carry a code at the bottom of the first or last page, copying is permitted provided that the per-copy fee indicated in the code is paid through the Copyright Clearance Center, 222 Rosewood Drive, Danvers, MA 01923.

ACM ISBN: 0-89791-839-8

Additional copies may be ordered prepaid from:

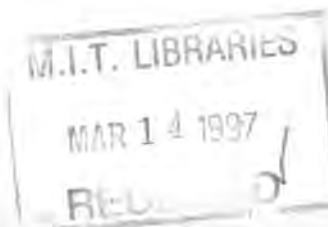
ACM Order Department
PO Box 12114
Church Street Station
New York NY 10257

ACM European Service Center
108 Cowley Road
Oxford OX 4 1JF UK
Phone: +44-1-865-382338
Fax: +44-1-865-3811338
E-mail: acm_europe@acm.org
URL: <http://www.acm.org>

Phone: 1-800-342-6626
(USA and Canada)
+1-212-944-1318
Fax: +1-21-944-1318
E-mail: acmpubs@acm.org

ACM Order Number: 608970

Printed in the U.S.A.



Multimodal User Interfaces in the Open Agent Architecture

Douglas B. Moran

Adam J. Cheyer

Luc E. Julia

David L. Martin

SRI International

333 Ravenswood Avenue

Menlo Park CA 94025 USA

+1 415 859 6486

{moran,cheyer,julia,martin}@ai.sri.com

Sangkyu Park

Artificial Intelligence Section

Electronics and Telecommunications

Research Institute (ETRI)

161 Kajong-Dong

Yusong-Gu, Taejon 305-350 KOREA

+82 42 860 5641

skpark@com.etri.re.kr

ABSTRACT

The design and development of the Open Agent Architecture (OAA)¹ system has focused on providing access to agent-based applications through an intelligent, cooperative, distributed, and multimodal agent-based user interfaces. The current multimodal interface supports a mix of spoken language, handwriting and gesture, and is adaptable to the user's preferences, resources and environment. Only the primary user interface agents need run on the local computer, thereby simplifying the task of using a range of applications from a variety of platforms, especially low-powered computers such as Personal Digital Assistants (PDAs). An important consideration in the design of the OAA was to facilitate mix-and-match: to facilitate the reuse of agents in new and unanticipated applications, and to support rapid prototyping by facilitating the replacement of agents by better versions.

The utility of the agents and tools developed as part of this ongoing research project has been demonstrated by their use as infrastructure in unrelated projects.

Keywords: agent architecture, multimodal, speech, gesture, handwriting, natural language

INTRODUCTION

A major component of our research on multiagent systems is in the user interface to large communities of agents. We have developed agent-based multimodal user interfaces using the same agent architecture used to build the back ends of these applications. We describe these interfaces and the larger architecture, and outline some of the applications that have been built using this architecture and interface agents.

Permission to make digital/hard copies of all or part of this material for personal or classroom use is granted without fee provided that the copies are not made or distributed for profit or commercial advantage, the copyright notice, the title of the publication and its date appear, and notice is given that copyright is by permission of the ACM, Inc. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires specific permission and/or fee.

IUI 97, Orlando Florida USA

© 1997 ACM 0-89791-839-8/96/01 ..\$3.50

OVERVIEW OF OPEN AGENT ARCHITECTURE

The Open Agent Architecture (OAA) is a multiagent system that focuses on supporting the creation of applications from agents that were *not* designed to work together, thereby facilitating the wider *reuse* of the expertise embodied by an agent. Part of this focus is the user interface to these applications, which can be viewed as supporting the access of human agents to the automated agents. Key attributes of the OAA are

- *Open:* The OAA supports agents written in multiple languages and on multiple platforms. Currently supported languages are C, Prolog, Lisp, Java, Microsoft's Visual Basic and Borland's Delphi. Currently supported platforms are PCs (Windows 3.1 and 95), Sun Workstations (Solaris 1.1 and 2.x) and SGIs.
- *Distributed:* The agents that compose an application can run on multiple platforms.
- *Extensible:* Agents can be added to the system while it is running, and their capabilities will become immediately available to the rest of the agents. Similarly, agents can be dynamically removed from the system (intentionally or not).
- *Mobile:* OAA-based applications can be run from a lightweight portable computer (or PDA) because only the user interface agents need run on the portable. They provide the user with access to a range of agents running on other platforms.
- *Collaborative:* The user interface is implemented with agents, and thus the user appears to be just another agent to the automated agents. This greatly simplifies creating systems where multiple humans and automated agents cooperate.
- *Multiple Modalities:* The user interface supports handwriting, gesture and spoken language in addition to the traditional graphical user interface modalities.

- *Multimodal Interaction*: Users can enter commands with a mix of modalities, for example, a spoken command in which the object to be acted on is identified by a pen gesture (or other graphical pointing operation).

The OAA has been influenced by work being done as part of DARPA's I3 (Intelligent Integration of Information) program (<http://isx.com/pub/I3>) and Knowledge Sharing Effort (<http://www-ksl.stanford.edu/knowledge-sharing/>) [13].

THE USER INTERFACE

The User Interface Agent

The user interface is implemented with a set of agents that have at their logical center an agent called the *User Interface (UI) Agent*. The User Interface Agent manages the various modalities and applies additional interpretation to those inputs as needed. Our current system supports speech, handwriting and pen-based gestures in addition to the conventional keyboard and mouse inputs. When speech input is detected, the UI Agent sends a command to the Speech Recognition agent to process the audio input and to return the corresponding text. Three modes are supported for speech input: *open microphone*, *push-to-talk*, and *click-to-start-talking*. Spoken and handwritten inputs can be treated as either raw text, or interpreted by a natural language understanding agent.

There are two basic styles of user interface. The first style parallels the traditional graphical user interface (GUI) for an application: The user selects an application and is presented with a window that has been designed for the application implemented by that agent and that is composed of the familiar GUI-style items. In this style interface, the application is typically implemented as a primary agent, with which the user interacts, and a number of supporting agents that are used by the primary agent, and whose existence is hidden from the user. When text entry is needed, the user may use handwriting or speech instead of the keyboard, and the pen may be used as an alternative to the mouse. Because the UI Agent handles all the alternate modalities, the applications are isolated from the details of which modalities are being used. This simplifies the design of the applications, and simplifies adding new modalities.

In the second basic style of interface, not only is there no primary agent, the individual agents are largely invisible to the user, and the user's requests may involve the cooperative actions of multiple agents. In the systems we have implemented, this interface is based on natural language (for example, English), and is entered with either speech or handwriting. When the UI Agent detects speech or pen-based input, it invokes a speech recognition agent or handwriting recognition agent, and sends the text returned by that agent to a natural language understanding agent, which produces a *logical form* representation of the user's request. This logical

form is then passed to a *Facilitator* agent, which identifies the subtasks and delegates them to the appropriate application agents. For example, in our *Map-based Tourist Information* application for the city of San Francisco, the user can ask for the distance between a hotel and sightseeing destination. The locations of the two places are in different databases, which are managed by different agents, and the distance calculation is performed by yet another agent.

These two basic styles of interfaces can be combined in a single interface. In our *Office Assistant* application, the user is presented with a user interface based on the Rooms metaphor and is able to access conventional applications such as e-mail, calendar, and databases in the familiar manner. In addition there is a subwindow for spoken or written natural language commands that can involve multiple agents.

A major focus of our research is multimodal inputs, typically a mix of gesture/pointing with spoken or handwritten language. The UI agent manages the interpretation of the individual modalities and passes the results to a *Modality Coordination* agent, which returns the composite query, which is then passed to the Facilitator agent for delegation to the appropriate application agents (described in subsequent sections).

Speech Recognition

We have used different speech recognition systems, substituting to meet different criteria. We use research systems developed by another laboratory in our organization (<http://www-speech.sri.com/>) [3] and by a commercial spin-off from that laboratory.² We are currently evaluating other speech recognizers, and will create agents to interface to their application programming interfaces (APIs) if they satisfy the requirements for new applications being considered.

Natural Language Understanding

A major advantage of using an agent-based architecture is that it provides simple mix-and-match for the components. In developing systems, we have used three different natural language (NL) systems: a simple one, based on Prolog DCG (Definite Clause Grammar), then an intermediate one, based on CHAT [16], and finally, our most capable research system GEMINI [6, 7]. The ability to trivially substitute one natural language agent for another has been very useful in rapid prototyping of systems. The DCG-based agent is used during the early stages of development because grammars are easily written and modified. Writing grammars for the more sophisticated NL agents requires more effort, but provides better coverage of the language that real users are likely to use, and hence we typically delay upgrading to the more sophisticated agents until the application crosses certain thresholds of maturity and usage.

¹Open Agent Architecture and OAA are trademarks of SRI International. Other brand names and product names herein are trademarks and registered trademarks of their respective holders.

²Nuance Corporation (formerly Corona Corp.), Building 110, 333 Ravenswood Avenue, Menlo Park, CA 94025 (domain: coronacorp.com)

Pen Input

We have found that including a pen in the user interface has several significant advantages. First, the gestures that users employ with a pen-based system are substantially richer than those employed by other pointing and tracking systems (e.g., a mouse). Second, handwriting is an important adjunct to spoken language. Speech recognizers (including humans) can have problems with unfamiliar words (e.g., new names). Users can use the pen to correct misspelled words, or may even anticipate the problem and switch from speaking to handwriting. Third, our personal experience is that when a person who has been using a speech-and-gesture interface faces an environment where speech is inappropriate, replacing speech with handwriting is more natural.

Using 2D gestures in the human-computer interaction holds promise for recreating the pen-and-paper situation where the user is able to quickly express visual ideas while she or he is using another modality such as speech. However, to successfully attain a high level of human-computer cooperation, the interpretation of on-line data must be accurate and fast enough to give rapid and correct feedback to the user.

The gestures-recognition engine used in our application is fully described in [9] as the early recognition process. There is no constraint on the number of strokes. The latest evaluations gave better than 96% accuracy, and the recognition was performed in less than half a second on a PC 486/50, satisfying what we judge is required in terms of quality and speed.

In most applications, this engine shares pen data with a handwriting recognizer. The use of the same medium to handle two different modalities is a source of ambiguities that are solved by a competition between both recognizers in order to determine whether the user wrote (a sentence or a command) or produced a gesture. A remaining problem is to solve a mixed input (the user draws and writes in the same set of strokes).

The main strength of the gestures recognition engine is its adaptability and reusability. It allows the developer to easily define the set of gestures according to the application. Each gesture is actually described with a set of parameters such as the number of directions, a broken segment, and so forth. Adding a new gesture consists of finding the description for each parameter. If a conflict appears with an existing object, the discrimination is done by creating a new parameter. For a given application, as few as four parameters are typically required to describe and discriminate the set of gestures.

We can use any handwriting recognizer compatible with Microsoft's PenWindows.³

Modality Coordination Agent

Our interface supports a rich set of interactions between natural language (spoken, written, or typed) and gesturing (e.g., pointing, circling)—much richer than that seen in the *put-*

that-there systems. Deictic words (e.g., *this*, *them*, *here*) can be used to refer to many classes of objects, and also can be used to refer to either individuals or collections of individuals.

The Modality Coordination (MC) agent is responsible for combining the inputs in the different modalities to produce a single meaning that matches the user's intention. It is responsible for resolving references, for filling in missing information for an incoming request, and for resolving ambiguities by using contexts, equivalence or redundancy.

Taking into account contexts implies establishing a hierarchy of rules between them. The importance of each context and the hierarchy may vary during a single session. In the actual system, missing information is extracted from the dialogue context (no graphical context or interaction context).

When the user says "*Show me the photo of this hotel*" and simultaneously points with the pen to a hotel, the MC agent resolves references based on that gesture. If no hotel is explicitly indicated, the MC agent searches the conversation context for an appropriate reference (for example, the hotel may have been selected by a gesture in the previous command). If there is no selected hotel in the current context, the MC Agent will wait a certain amount of time (currently 2 to 3 seconds) before asking the user to identify the hotel intended. This short delay is designed to accommodate different synchronizations of speech and gesture: different users (or a single user in different circumstances) may point before, during or just after speaking.

In another example, the user says "*Show me the distance from the hotel to here*" while pointing at a destination. The previous queries have resulted in a single hotel being focused upon, and the MC agent resolves "*the hotel*" from this context.⁴ The gesture provides the MC agent with the referent of "*here*". Processing the resulting query may involve multiple agents, for example, the location of hotels and sightseeing destinations may well be in a different databases, and these locations may be expressed in different formats, requiring another agent to resolve the differences and then compute the distance.

Flexible Sets of Modalities

The OAA allows the user maximum flexibility in what modalities will be used. Sometimes, the user will be on a computer that does not support the full range of modalities (e.g., no pen or handwriting recognition). Sometimes, the user's environment limits the choice of modalities, for example, spoken commands are inappropriate in a meeting where someone else is speaking, whereas in a moving vehicle, speech is likely to be more reliable than handwriting. And sometimes, the user's choice of modalities is influenced by the data being entered [14].

With this flexibility, the telephone has become our low-end user interface to the system. For example, we can use the

³Our preferred recognizer is *Handwriter for Windows* from Communication Intelligence Corp (CIC) of Redwood City, CA.

⁴User feedback about which items are in focus (contextually) is provided by graphically highlighting them.

telephone to check on our appointments, and we use the telephone to notify us of the arrival and content of important e-mail when we are away from our computers.

This flexibility has also proven quite advantageous in accommodating hardware failure. For example, moving the PC for one demonstration of the system shook loose a connection on the video card. The UI agent detected that no monitor was present, and used the text-to-speech agent to generate the output that was normally displayed graphically.

In another project's demonstration (CommandTalk), the designated computer was nonfunctional, and an underpowered computer had to be substituted. Using the OAA's innate capabilities, the application's components were distributed to other computers on the net. However, the application had been designed and tested using the microphone on the local computer, and the substitute had none. The solution was to add the Telephone agent that had been created for other applications: it automatically replaced the microphone as the input to the speech recognizer.

Learning the System

One of the well-known problems with systems that utilize natural language is in communicating to the user what can and cannot be said. A good solution to this is an open research problem. Our approach has been to use the design of the GUI to help illustrate what can be said: All the simple operations can also be invoked through traditional GUI items, such as menus, that cover much of the vocabulary.

OAA AGENTS

Overview

OAA agents communicate with each other in a high-level logical language called the Interagent Communication Language (ICL). ICL is similar in style and functionality to the Knowledge Query and Manipulation Language (KQML) of the DARPA Knowledge Sharing Effort. The differences are a result of our focus on the user interface: ICL was designed to be compatible with the output of our natural language understanding systems, thereby simplifying transforming a user's query or command into one that can be handled by the automated agents.

We have developed an initial set of tools (the Agent Development Toolkit) to assist in the creation of agents [11]. These tools guide the developer through the process, and automatically generate code templates from specifications (in the style of various commercial CASE tools). These tools are implemented as OAA agents, so they can interact with, and build upon, existing agents. The common agent support routines have been packaged as libraries, with coordinated libraries for the various languages that we support.⁵

These tools support building both entirely new agents and creating agents from existing applications, including legacy systems. These latter agents are called *wrappers* (or *transducers*); they convert between ICL and the application's API

(or other interface if there is no API).

The Facilitator Agent

In the OAA framework, the *Facilitator* agents play a key role. When an agent is added to the application, it registers its capabilities with the Facilitator. Part of this registration is the natural language vocabulary that can be used to talk about the tasks that the agent can perform. When an agent needs work done by other agents within the application, it sends a request to the Facilitator, which then delegates it to an agent, or agents, that have registered that they can handle the needed tasks. The ability of the Facilitator to handle complex requests from agents is an important attribute of the OAA design. The goal is to minimize the information and assumptions that the developer must embed in an agent, thereby making it easier to reuse agents in disparate applications.

The OAA supports direct communication between application agents, but this has not been heavily utilized in our implementations because our focus has been on aspects of applications in which the role of the Facilitator is crucial. First, we are interested in user interfaces that support interactions with the broader community of agents, and the Facilitator is key to handling complex queries. The Facilitator (and supporting agents) handle the translation of the user's model of the task into the system model (analogous to how natural language interfaces to databases handle transforming the user's model into the database's schemas). Second, the Facilitator simplifies reusing agents in new applications. If a community of agents is assembled using agents acquired from other communities, those agents cannot be assumed to all make atomic requests that can be handled by other agents: simple requests in one application may be implemented by a combination of agents in another application. The Facilitator is responsible for decomposing complex requests and translating the terminology used. This translation is typically handled by delegating it to another agent.

In the OAA, the Facilitator is a potential bottleneck if there is a high volume of communication between the agents. Our focus has been on supporting a natural user interface to a very large community of intelligent agents, and this environment produces relatively low volume through the Facilitator. In the CommandTalk application (discussed later), the multiagent system is actually partitioned into two communities: the user interface and the simulator. The simulator has very high volume interaction and a carefully crafted communication channel and appears as a single agent to the Facilitator and the user interface agents.

Triggers

In an increasing variety of conventional applications, users can set *triggers* (also called monitors, daemons or watchdogs) to take specific action when an event occurs. However, the possible actions are limited to those provided in

⁵A release of a version of this software is planned. The announcement will appear on <http://www.ai.sri.com/~oaa/>.

that application. The OAA supports triggers in which both the condition and action parts of a request can cover the full range of functionality represented by the agents dynamically connected to the network.

In a practical real-world example, one of the authors successfully used agent triggers to find a new home. The local rental housing market is very tight, with all desirable offerings being taken immediately. Thus, you need to be among the first to respond to a new listing. Several of the local newspapers provide on-line versions of their advertisements before the printed versions are available, but there is considerable variability in when they actually become accessible. To automatically check for suitable candidates, the author made the following request to the agent system: *"When a house for rent is available in Menlo Park for less than 1800 dollars, notify me immediately."* This natural language request installed a trigger on an agent knowledgeable about the domain of World Wide Web sources for house rental listings. At regular intervals, the agent instructs a Web retrieval agent to scan data from three on-line newspaper databases. When an advertisement meeting the specified criteria is detected, a request is sent to the Facilitator for a notify action to be delegated to the appropriate other agents.

The notify action involves a complex series of interactions between several agents, coordinated by the Notify and Facilitator agents. For example, if the user is in a meeting in a conference room, the Notify agent first determines his current location by checking his calendar (if no listing is found, the default location is his office, which is found from another database). The Notify agent then requests contact information for the conference room, and finds only a telephone number. Subsequent requests create a spoken version of the advertisement and retrieve the user's confirmation password. When all required information is collected, the Facilitator contacts the Telephone agent with a request to dial the telephone, ask for the user, confirm his identity with password (entered by TouchTone), and finally play the message. Other media, including FAX, e-mail and pager, can be considered by the Notify agent if agents for handling these services happen to be connected to the network.

DISTRIBUTED SYSTEMS

Multiple Platforms

The OAA applications that we have implemented run on a variety of platforms, and the exact location of individual agents is easily changed. We currently support PCs (Windows 3.1 and 95) and Sun and SGI workstations. Our primary user interface platform is the PC, partly because it currently offers better support for pen-based computing and partly because of our emphasis on providing user interfaces on lightweight computers (portable PCs and PDAs in near future). PCs also have the advantage of mass-market GUI-building packages such as Visual Basic and Delphi. A lesser version of the user interface has been implemented under X for UNIX workstations.

Even when the UI is on a PC, some of the agents in the UI package are running elsewhere. Our preferred speech recognizer requires a UNIX workstation, and our natural language agents and Modality Coordination agent have been written for UNIX systems.

Mobile Computing

We view mobile computing not only as people moving about with portable computers using wireless communication, but also people moving *between* computers. Today's user may have a workstation in his office, a personal computer at home, and a portable or PDA for meetings. In addition, when the user meets with management, colleagues and customers ("customers" in the broad sense of the people who require his services), their computers may be different platforms. From each of these environments, the user should be able to access his data and run his applications.

The OAA facilitates supporting multiple platforms because only the primary user interface agents need to be running on the local computer, thereby simplifying the problem of porting to new platforms and modality devices. Also, since only a minimal set of agents need to be run locally, lightweight computers (portables, PDA, and older systems) have the resources needed to be able to utilize heavyweight, resource-hungry applications.

COLLABORATION

One of the major advantages of having an agent-based interface to a multiagent application is that it greatly simplifies the interactions between the user and the application: application agents may interact with a human in the same way they interact with any other agent.

This advantage is readily seen when building collaborative systems. Perhaps the simplest form of collaboration is to allow users to share input and output to each other's applications. This form of cooperation is inherent in the design of the OAA: it facilitates the interoperation of software developed by distributed communities, especially disparate user communities (different platforms, different conventions).

We are currently integrating more sophisticated styles of collaboration into the OAA framework, using the synchronous collaborative technology [5] built by another group within our organization. In the resulting systems, humans can communicate with agents, agents can work with other automated agents, and humans can interact in realtime with other humans users.

APPLICATIONS AND REUSE

Two applications, the *Office Assistant* and *Map-based Tourist Information* have been the primary experimental environments for this research project. The agent architecture and the specific agents developed on this research project have proved to be so useful that they are being used by an expanding set of other projects within our organization. These other internal projects are helping us improve the documen-

tation and packaging of our toolkits and libraries, and we are hoping to release a version in the near future.

Some of the projects adopting the OAA have been motivated by the availability of various agents, especially the user interface agents. Some projects have gone further and used the OAA to integrate the major software components being developed on those projects.

Office Assistant

The OAA has been used as the framework for a number of applications in several domain areas. In the first OAA-based system, a multifunctional “office assistant”, fourteen autonomous agents provide information retrieval and communication services for a group of coworkers in a networked computing environment ([4]). This system makes use of a multimodal user interface running on a pen-enabled portable PC, and allows for the use of a telephone to give spoken commands to the system. Services are provided by agents running on UNIX workstations, many of which were created by providing agent wrappers for legacy applications.

In a typical scenario, agents with expertise in e-mail processing, text-to-speech translation, notification planning, calendar and database access, and telephone control cooperate to find a user and alert him or her of an important message. The office assistant system provides a compelling demonstration of how new services can arise from the synergistic combination of the capabilities of components that were originally intended to operate in isolation. In addition, as described earlier, it demonstrates the combination of two basic styles of user interaction — one that directly involves a particular agent as the primary point of contact, and one that anonymously delegates requests across a collection of agents — in a way that allows the user to switch freely between the two.

In the interface for this system, the initial screen portrays an office, in which familiar objects are associated with the appropriate functionality, as provided by some agent. For instance, clicking on a wall clock brings up a dialogue that allows one to interact with the calendar agent (that is, browsing and editing one’s appointments). In this style of interaction, even though the calendar agent may call on other agents in responding to some request, it has primary responsibility, in that all requests through that dialogue are handled by it.

The alternative style of interaction is one in which the user might speak “*Where will I be at 2:00 this afternoon?*”. In this case, the delegation of the request to the appropriate agents — which is done by the User Interface agent in concert with a Facilitator agent — reflects a style that is less direct and more anonymous.

Map-based Tourist Information

In a number of domains, access to information can very naturally be organized around a map-based interface. In creating such interfaces for several different systems, we have found

the agent-based approach to multimodality to be extremely useful. In these systems, all the components share a common interface—the map—and the fact that there are many agents is entirely invisible to the user.

One example is a map-based system to provide tourist information about San Francisco. Requests expressed in a variety of modalities can control the scrolling and zoom level of the map, retrieve information about locations and distances, display hotels or attractions meeting a user’s preferences, or present detailed information in a variety of media about particular hotels or attractions. Where appropriate, this information is derived and updated regularly from WWW sources.

Map-based interfaces provide a rich setting in which to explore the coordination of gesture with speech and traditional GUI modalities. The tourist information system accommodates the use of a variety of familiar pen gestures, such as circling objects or regions, drawing arrows, X’ing positions or objects, and striking out objects. Depending on context and timing considerations, requests can be derived from single gestures, multiple gestures interpreted together, spoken or handwritten input, point-and-click, or some combination of these operations.

For example, an arrow drawn across a map from right to left (which itself is recognized from two or three pen strokes) is interpreted as a request to scroll the map. The same effect may be achieved by speaking “*scroll left*”. Display of hotels can be obtained by writing or speaking “*Show hotels*”, or, perhaps, “*Show hotels with a pool*”. The distance between two objects or locations may be obtained by circling, X’ing, or clicking on each of them, and then drawing a straight line between them. Alternatively, one can speak “*Show the distance from here to here*”, while selecting two locations, or one can write “*distance*” either before or after selecting two objects.

This system, and the organization of the input recognition agents, is described in detail in [2]. A related system is described in [15].

CommandTalk

CommandTalk, a system in quite a different domain than tourism, was able to make use of the same approach to the map-based integration of speech with other modalities.⁶ In the CommandTalk system, currently installed at the Marine Corps Air Ground Combat Center at Twentynine Palms, CA, a collection of OAA-enabled agents provides a spoken-English interface to a map-based simulation of armed forces [12]. CommandTalk has proven useful in providing realism to scenarios used in training military commanders. The simulator is roughly 500,000 lines of code that was provided to the interface developers. Within 2 weeks of receiving the simulator code, they were able to demonstrate a spoken language interface to the basic functionality of the package by creating an agent interface to that portion of the simulator’s

⁶In the case of CommandTalk, gesture has not yet been a factor, but there has been an emphasis on the comprehensive use of speech, in combination with traditional GUI modalities.

functionality and then adapting the existing user interface agents to that domain. After the early prototype had demonstrated the utility of the concept, a more extensive analysis was conducted of the task and the commands used, and more capable prototypes were developed. One of the significant enhancements was the replacement of our simplest natural language agent (DCG-based) with our most sophisticated (based on GEMINI [6, 7]).

Summarization of Conversation

A system that summarizes conversations provided a novel opportunity to use two instances of a speech recognition agent, in conjunction with a single instance of a text processing agent ([10]). In this system, MIMI, two Japanese speakers engage in a conversation, such as, for example, an inquiry about room availability at a hotel. Each speaker is on a separate microphone, and each microphone feeds into a separate speech recognition agent. The output streams of these agents are both fed into a text processing agent, adapted especially for this task. Following the completion of the conversation, the text processing agent is able to print out a summary of what was discussed and agreed upon.

In constructing this system, as with CommandTalk, the ability to reuse and reconfigure preexisting user interface agents, in conjunction with newly created agents, afforded a significant savings in system construction time. The English-language speech recognizer was replaced with a Japanese-language version, and the natural language understanding agent that generated commands to the rest of the system was replaced by an agent that analyzed and stored the summary of the conversation.

Air Travel Information System

Web-based interfaces can readily be integrated into an agent-based system. At the same time that the agent system benefits from the universal accessibility of a Web interface, the HTML paradigm is extended and strengthened by the use of persistent interface agents to maintain the state of a sequence of interactions.

In one such system, user interface agents have been used to provide a Web/telephone interface to a spoken language Air Travel Information System (ATIS) [1]. In addition to speech recognition and natural language understanding agents, this system involves a telephone control agent, a response generation agent, and a User Interface agent. The initial version was based on HTML. The current version uses Java to provide more incremental feedback to the user.⁷

Multi-robot Control

SRI's family of mobile robots have been integrated as agents within the OAA framework. As such, robots may access, and be accessed by, existing OAA services, including corporate databases, text-to-speech generation, and telephone interfaces. In the Robot Competition at the 1996 AAAI con-

ference, OAA's capabilities were used by the SRI team to coordinate the activities of three robots. SRI won the Office Navigation task, completing it much faster than any of the other competitors (who were using only single robots) [8].

The multimodal map application was minorly modified to provide monitoring and control of the robots as they navigate a building. The screen displays a blueprint-style map of the area in which the robots operate, and the positions of the movable objects (robots and the objects that they can manipulate) are updated in realtime. Although the input modalities are the same as the earlier application (Map-based Tourist Information), there are noticeable differences. First, the inputs are predominantly commands, instead of information retrieval (queries). Second, some pen gestures mean different things: for example, with the robots, an arrow is used to indicate orientation ("*Robot one, face this direction*") or direction ("*Move this way*").

Emergency Response System

Another system for which a map-based interface has been useful is a prototype system of pen-based mobile computing units for use in the field by teams responding to a disaster such as an earthquake. In this system, a database of maps is available on each mobile unit (to avoid having to download sizable bitmaps), but information about specific locations and structures is stored in a centralized set of databases. This information can be retrieved and/or updated as appropriate by each mobile unit. The centralized database server also receives updates from hospitals and clinics as to their status, capacity, and patients being treated.

For example, as a response team learns the condition of the streets and structures in its region, it is able to record this information on the map-based interface, using point-and-click in combination with handwriting or typing, and then upload the data to the central databases. When a street or structure is found to be unsafe, that information can be relayed to all mobile units.

In the case in which an injured person is found, the system allows for the entry of some basic facts about the injury. Following that, an agent operating on the central server makes a determination of what hospital or clinic would be most appropriate for the person, based on current status reports, and this recommendation is then returned to the response team.

This system has both Japanese-language and English-language interfaces.

CONCLUSIONS

The OAA has proven to be useful in constructing sophisticated systems because it provides the flexibility to combine applications that were not originally envisioned as a package. The OAA differs from much of the other research on distributed agents in its focus on providing multimodal user interfaces to systems assembled from disparate agents. This focus results in a tradeoff which is a major limitation of this

⁷It may be accessed at <http://www-speech.sri.com/demos/atis.html>

architecture: while the Facilitator agent is key to cooperation between independently developed agents, it is a potential bottleneck in systems where agents need high-volume, low-delay interactions (discussed in *The Facilitator Agent*). In one existing application (and one under consideration), a composite approach has provided a viable solution for this limitation.

ACKNOWLEDGMENTS

This paper is based on work that was supported in part by a contract to SRI from the Electronics and Telecommunications Research Institute (Korea). Philip R. Cohen (now at the Oregon Graduate Institute) was project leader until August 1994, and was responsible for many of the design decisions in the systems described here. Any opinions expressed in this paper are strictly those of the authors.

REFERENCES

- 1 Harry Bratt, John Dowding, and Kate Hunicke-Smith. The SRI telephone-based ATIS system. In *Proc. of the ARPA Spoken Language System Technology Workshop*, Austin, Texas, January 1995. Also <http://www.ai.sri.com/natural-language/projects/arpa-sls/apps.html>.
- 2 Adam Cheyer and Luc Julia. Multimodal maps: An agent-based approach. In *Proc. of the International Conference on Cooperative Multimodal Communication (CMC/95)*, Eindhoven, The Netherlands, May 1995. Also <http://www.ai.sri.com/~oaa/> + "Bibliography".
- 3 Michael Cohen, Hy Murveit, Jared Bernstein, Patti Price, and Mitchel Weintraub. The DECIPHER speech recognition system. In *IEEE ICASSP*, pages 77–80, 1990.
- 4 Philip R. Cohen, Adam J. Cheyer, Michelle Wang, and Soon Cheol Baeg. An open agent architecture. In O. Etzioni, editor, *Proc. of the AAAI Spring Symposium Series on Software Agents*, pages 1–8, Stanford, California, March 1994. American Association for Artificial Intelligence.
- 5 Earl Craighill, Martin Fong, Keith Skinner, Ruth Lang, and Kathryn Gruenefeldt. SCOOT: An object-oriented toolkit for multimedia collaboration. In *Proc. of the ACM MULTIMEDIA 94 Conference*, pages 41–49, San Francisco, CA, October 1994. Also <http://www.std.sri.com/public/ftp/ACE/Papers/SCOOT94.ps.Z>.
- 6 John Dowding, J. Mark Gawron, Douglas Appelt, John Bear, Lynn Cherny, Robert Moore, and Douglas Moran. GEMINI: A natural language system for spoken-language understanding. In *Proc. of the 31st Annual Meeting of the Association for Computational Linguistics*, pages 54–61, Ohio State University, Columbus, Ohio, 22–26 June 1993.
- 7 John Dowding, Robert Moore, Francois Andry, and Douglas Moran. Interleaving syntax and semantics in an efficient bottom-up parser. In *Proc. of the 32nd Annual Meeting of the Association for Computational Linguistics*, pages 110–116, New Mexico State University, Las Cruces, New Mexico, 27 June – 1 July 1994.
- 8 Didier Guzzoni, Adam Cheyer, Luc Julia, and Kurt Konolige. Report on the SRI Pioneer Robot Team at AAAI-96 Robotics Competition and Exhibition (tentative title). To appear in *AI Magazine*, Winter 1996 or Spring 1997.
- 9 Luc Julia and Claudie Faure. Pattern recognition and beautification for a pen based interface. In *ICDAR'95*, pages 58–63, Montreal, Canada, 1995.
- 10 Megumi Kameyama, Goh Kawai, and Isao Arima. A real-time system for summarizing human-human spontaneous spoken dialogues. In *Proc. of the Fourth International Conference on Spoken Language Processing (ICSLP-96)*, October 1996. Also <http://www.ai.sri.com/~megumi/> + "My publications".
- 11 David L. Martin, Adam J. Cheyer, and Gowang-Lo Lee. Agent development tools for the Open Agent Architecture. In *Proc. of the First International Conference on the Practical Application of Intelligent Agents and Multi-Agent Technology*, pages 387–404, London, April 1996. The Practical Application Company Ltd.
- 12 Robert Moore, John Dowding, Harry Bratt, J. Mark Gawron, Yonael Gorf, and Adam Cheyer. Commandtalk: A spoken-language interface for battlefield simulation. Technical report, Artificial Intelligence Center, SRI International, 21 June 1996. Also <http://www.ai.sri.com/natural-language/projects/arpa-sls/apps.html>.
- 13 R. Neches, R. Fikes, T. Finin, T. Gruber, R. Patil, T. Senator, and W. Swartout. Enabling technology for knowledge sharing. *AI Magazine*, 12(3), 1991.
- 14 S. L. Oviatt. Pen/voice: Complementary multimodal communication. In *Proc. of Speech Tech'92*, pages 238–241, 1992.
- 15 S. L. Oviatt. Multimodal interfaces for dynamic interactive maps. In *Proc. of CHI 96*, Vancouver, Canada, 1996. Assoc. for Computing Machinery.
- 16 F. C. N. Pereira. *Logic for Natural Language Analysis*. PhD thesis, U. of Edinburgh, 1983.

Multimodal Maps: An Agent-based Approach

Abstract

The SRI Multimedia Interfaces Group is currently building a series of prototype map-based applications that accept handwritten, verbal and gestural requests issued in a synergistic fashion. These applications are distinguished by comparatively rich natural language capabilities, access to existing data sources including the World Wide Web, and a mobile handheld interface. To implement the described application, a hierarchical distributed network of heterogeneous software agents was augmented by appropriate functionality for developing synergistic multimodal applications.

Features

- *Multimodal Input*: A user interacts with the system through handwritten, gestural, vocal, and direct manipulation commands.
- *Multimedial Output*: Responses are provided by the system through textual, graphical, audio and video media.
- *Agent Interaction*: User requests are processed by a distributed community of software agents, which automate tasks such as natural language processing, interaction with planners or simulators, and accessing data from remote datasources (Prolog DBs, X.500 DBs, WWW, etc.)

Demos

- To see a demonstration (including video) of the multimodal map interface, [click here](#).
- To see a picture representing the agents used in the multimodal map demonstration, [click here](#).

Papers

- *Multimodal Maps: An Agent-based Approach*. With Luc Julia. International Conference on Cooperative Multimodal Communication (CMC/95), 24-26 May 1995, Eindhoven, The Netherlands. [PostScript: 171K](#)
- *A Multimodal Computer-augmented Interface for Distributed Applications*. With Luc Julia. HCII'95, July, 1995, Tokyo.

Related Work

- The [Open Agent Architecture \(tm\)](#) is a general purpose framework for developing multi-agent systems.
 - A set of [Agent Development Tools](#) is being developed to simplify the task of creating new multimodal agent-based applications.
-

Created by [Adam Cheyer](#), 19 February 1995

[Next](#) [Up](#) [Previous](#)Next: [Introduction](#)

Multimodal Maps: An Agent-based Approach

Adam CHEYER and Luc JULIA
SRI International
333 Ravenswood Ave
Menlo Park, CA 94025 - USA

Abstract:

In this paper, we discuss how multiple input modalities may be combined to produce more natural user interfaces. To illustrate this technique, we present a prototype map-based application for a travel planning domain. The application is distinguished by a synergistic combination of handwriting, gesture and speech modalities; access to existing data sources including the World Wide Web; and a mobile handheld interface. To implement the described application, a distributed network of heterogeneous software agents was augmented by appropriate functionality for developing synergistic multimodal applications.

-
- [Introduction](#)
 - [Natural Input](#)
 - [Input Modalities](#)
 - [Combination of Modalities](#)
 - [A Multimodal Map Application](#)
 - [Approach](#)
 - [Building Blocks](#)
 - [Open Agent Architecture](#)
 - [TAPAGE](#)
 - [Synthesis](#)
 - [CONCLUSIONS](#)
 - [Acknowledgements](#)
 - [References](#)
 - [About this document ...](#)
-

Adam Cheyer

Mon Aug 12 15:07:21 PDT 1996

[Next](#) [Up](#) [Previous](#)

Next: [Input Modalities](#) **Up:** [Multimodal Maps: An Agent-based](#) **Previous:** [Introduction](#)

Natural Input

- [Input Modalities](#)
 - [Combination of Modalities](#)
-

Adam Cheyer

Mon Aug 12 15:07:21 PDT 1996

[Next](#) [Up](#) [Previous](#)Next: [Combination of Modalities](#) Up: [Natural Input](#) Previous: [Natural Input](#)

Input Modalities

Direct manipulation interface technologies are currently the most widely used techniques for creating user interfaces. Through the use of menus and a graphical user interface, users are presented with sets of discrete actions and the objects on which to perform them. Pointing devices such as a mouse facilitate selection of an object or action, and drag and drop techniques allow items to be moved or combined with other entities or actions.

With the addition of electronic pen devices, gestural drawings add a new dimension to direct manipulation interfaces. Gestures allow users to communicate a surprisingly wide range of meaningful requests with a few simple strokes. Research has shown that multiple gestures can be combined to form dialog, with rules of temporal grouping overriding temporal sequencing [\[23\]](#). Gestural commands are particularly applicable to graphical or editing type tasks.

Direct manipulation interactions possess many desirable qualities: communication is generally fast and concise; input techniques are easy to learn and remember; the user has a good idea about what can be accomplished, as the visual presentation of the available actions is generally easily accessible. However, direct manipulation suffers from limitations when trying to access or describe entities which are not or can not be visualized by the user.

Limitations of direct manipulation style interfaces can be addressed by another interface technology, that of natural language interfaces. Natural language interfaces excel in describing entities that are not currently displayed on the monitor, in specifying temporal relations between entities or actions, and in identifying members of sets. These strengths are exactly the weaknesses of direct manipulation interfaces, and concurrently, the weaknesses of natural language interfaces (ambiguity, conceptual coverage, etc.) can be overcome by the strengths of direct manipulation [\[6\]](#).

Natural language content can be entered through different input modalities, including typing, handwriting, and speech. It is important to note that, while the same textual content can be provided by the three modalities, each modality has widely varying properties.

- Spoken language is the modality used first and foremost in human-human interactive problem solving [\[4\]](#). Speech is an extremely fast medium, several times faster than typing or handwriting. In addition, speech input contains content that is not present in other forms of natural language input, such as prosody, tone and characteristics of the speaker (age, sex, accent).
- Typing is the most common way of entering information into a computer, because it is reasonably fast, very accurate, and requires no computational resources.
- Handwriting has been shown to be useful for certain types of tasks, such as performing numerical calculations and manipulating names which are difficult to pronounce [\[18\]](#), [\[20\]](#). Because of its relatively slow production rate, handwriting may induce users to produce different types of input than is generated by spoken language; abbreviations, symbols and non-grammatical patterns may be expected to be more prevalent amid written input.

[Next](#) [Up](#) [Previous](#)

Next: [Combination of Modalities](#) **Up:** [Natural Input](#) **Previous:** [Natural Input](#)

Adam Cheyer

Mon Aug 12 15:07:21 PDT 1996

[Next](#) [Up](#) [Previous](#)**Next:** [A Multimodal Map Application](#) **Up:** [Natural Input](#) **Previous:** [Input Modalities](#)

Combination of Modalities

As noted in the previous section, direct manipulation and natural language seem to be very complementary modalities. It is therefore not surprising that a number of multimodal systems combine the two.

Notable among such systems is the Cohen's Shoptalk system [\[6\]](#), a prototype manufacturing and decision-support system that aids in tasks such as quality assurance monitoring, and production scheduling. The natural language module of Shoptalk is based on the Chat-85 natural language system [\[26\]](#) and is particularly good at handling time, tense, and temporal reasoning.

A number of systems have focused on combining the speed of speech with the reference provided by direct manipulation of a mouse pointer. Such systems include the XTRA system [\[1\]](#), CUBRICON [\[15\]](#), the PAC-Amodeus model [\[16\]](#), and TAPAGE [\[9\]](#), [\[12\]](#).

XTRA and CUBRICON are both systems that combine complex spoken input with mouse clicks, using several knowledge sources for reference identification. CUBRICON's domain is a map-based task, making it similar to the application developed in this paper. However, the two are different in that CUBRICON can only use direct manipulation to indicate a specific item, whereas our system produces a richer mixing of modalities by adding both gestural and written language as input modalities.

PAC-Amodeus systems such as VoicePaint and Notebook allow the user to synergistically combine vocal or mouse-click commands when interacting with notes or graphical objects. However, due in part to the selected domains, the natural language input is very simple, generally of the style "Insert a note here."

TAPAGE is another system that allows true synergistic combination of spoken input with direct manipulation. Like PAC-Amodeus, TAPAGE's domain provides only simple linguistic input. However, TAPAGE uses a pen-based interface instead of a mouse, allowing gestural commands. TAPAGE, selected as one of the "building blocks" for our map application, will be described more in detail in section 4.2.

Other pertinent work regarding the simultaneous combination of handgestures and gaze can be found in [\[2\]](#), [\[13\]](#).

[Next](#) [Up](#) [Previous](#)**Next:** [A Multimodal Map Application](#) **Up:** [Natural Input](#) **Previous:** [Input Modalities](#)

Adam Cheyer

Mon Aug 12 15:07:21 PDT 1996

[Next](#) [Up](#) [Previous](#)

Next: [Approach](#) Up: [Multimodal Maps: An Agent-based](#) Previous: [Combination of Modalities](#)

A Multimodal Map Application

In this section, we will describe a prototype map-based application for a travel planning domain. In order to provide the most natural user interface possible, the system permits the user to simultaneously combine direct manipulation, gestural drawings, handwritten, typed and spoken natural language. When designing the architecture for the system, other criteria were considered as well:

- The user interface must be light and fast enough to run on a handheld PDA while able to access applications and data that may require a more powerful machine.
- Existing commercial or research natural language and speech recognition systems should be used.
- Through the multimodal interface, a user must be able to transparently access a wide variety of data sources, including information stored in HTML format on the World Wide Web.



Figure 1: Multimodal Application for Travel Planning

The map functionality, interface design, and classes of input data of the system presented here is based on a design by Oviatt and Cohen, used by them in a wizard-of-oz simulation system designed to explore complex interactions of modalities [\[19\]](#). The agent-based architecture used to realize Oviatt and Cohen's design is new, as is its application to travel planning.

As illustrated in Figure 1, the user is presented with a pen sensitive map display on which drawn gestures and handwritten natural language statements may be combined with spoken input. As opposed to a static paper map, the location, resolution, and content presented by the map change,

according to the requests of the user. Objects of interest, such as restaurants, movie theaters, hotels, tourist sites, municipal buildings, etc. are displayed as icons. The user may ask the map to perform various actions. For example :

- *distance calculation* : e.g. ``How far is the hotel from Fisherman's Wharf?"
- *object location* : e.g. ``Where is the nearest post office?"
- *filtering* : e.g. ``Display the French restaurants within 1 mile of this hotel."
- *information retrieval* : e.g. ``Show me all available information about Alcatraz."

The application also makes use of multimodal (multimedia) output as well as input: video, text, sound and voice can all be combined when presenting an answer to a query.

During input, requests can be entered using gestures (Figure 2), handwriting, voice, or a combination of pen and voice. For instance, in order to calculate the distance between two points on the map, a command may be issued using the following:

- *gesture*, by simply drawing a line between the two points of interest.
- *voice*, by speaking ``What is the distance from the post office to the hotel?"
- *handwriting*, by writing ``dist p.o. to hotel?"
- *synergistic combination of pen and voice*, by speaking ``What is the distance from here to this hotel?" while simultaneously indicating the specified locations by pointing or circling.

Notice that in our example of synergistic combination of pen and voice, the arguments to the verb ``distance" can be specified before, at the same time, or shortly after the vocalization of the request to calculate the distance. If a user's request is ambiguous or underspecified, the system will wait several seconds and then issue a prompt requesting additional information.

The user interface runs on pen-equipped PC's or a Dauphin handheld PDA ([7]) using either a microphone or a telephone for voice input. The interface is connected either by modem or ethernet to a server machine which will manage database access, natural language processing and speech recognition for the application. The result is a mobile system that provides a synergistic pen/voice interface to remote databases.

In general, the speed of the system is quite acceptable. For gestural commands, which are handled locally on the user interface machine, a response is produced in less than one second. For handwritten commands, the time to recognize the handwriting, process the English query, access a database and begin to display the results on the user interface is less than three seconds (assuming an ethernet connection, and good network and database response). Solutions to verbal commands are displayed in three to five seconds after the end of speech has been detected; partial feedback indicating the current status of the speech recognition is provided earlier.

[Next](#) [Up](#) [Previous](#)

Next: [Building Blocks](#) **Up:** [Multimodal Maps: An Agent-based](#) **Previous:** [A Multimodal Map Application](#)

Approach

In order to implement the application described in the previous section, we chose to augment a proven agent- based architecture with functionalities developed for a synergistically multimodal application. The result is a flexible methodology for designing and implementing distributed multimodal applications.

-
- [Building Blocks](#)
 - [Open Agent Architecture](#)
 - [TAPAGE](#)
 - [Synthesis](#)
-

Adam Cheyer

Mon Aug 12 15:07:21 PDT 1996

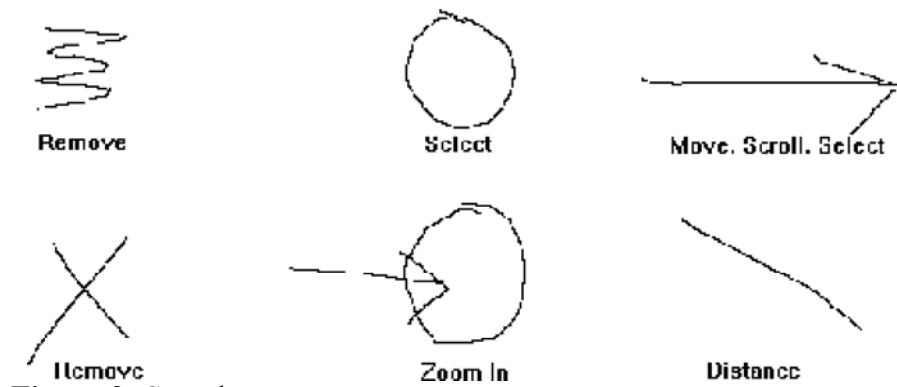


Figure 2: Sample gestures

[Next](#) [Up](#) [Previous](#)

Next: [Approach](#) **Up:** [Multimodal Maps: An Agent-based](#) **Previous:** [Combination of Modalities](#)

Adam Cheyer

Mon Aug 12 15:07:21 PDT 1996

[Next](#) [Up](#) [Previous](#)

Next: [Open Agent Architecture](#) Up: [Approach](#) Previous: [Approach](#)

Building Blocks

- [Open Agent Architecture](#)
 - [TAPAGE](#)
-

Adam Cheyer

Mon Aug 12 15:07:21 PDT 1996

[Next](#) [Up](#) [Previous](#)Next: [TAPAGE](#) Up: [Building Blocks](#) Previous: [Building Blocks](#)

Open Agent Architecture

The Open Agent Architecture (OAA) [\[5\]](#) provides a framework for coordinating a society of agents which interact to solve problems for the user. Through the use of agents, the OAA provides distributed access to commercial applications, such as mail systems, calendar programs, databases, etc.

The Open Agent Architecture possesses several properties which make it a good candidate for our needs:

- An Interagent Communication Language (ICL) and Query Protocol have been developed, allowing agents to communicate among themselves. Agents can run on different platforms and be implemented in a variety of programming languages.
- Several natural language systems have been integrated into the OAA which convert English into the Interagent Communication Language. In addition, a speech recognition agent has been developed to provide transparent access to the Corona speech recognition system.
- The agent architecture has been used to provide natural language and agent access to various heterogeneous data and knowledge sources.
- Agent interaction is very fine-grained. The architecture was designed so that a number of agents can work together, when appropriate in parallel, to produce fast responses to queries.

The architecture for the OAA, based loosely on Schwartz's FLiPSiDE system [\[24\]](#), uses a hierarchical configuration where client agents connect to a "facilitator" server. Facilitators provide content-based message routing, global data management, and process coordination for their set of connected agents. Facilitators can, in turn, be connected as clients of other facilitators. Each facilitator records the published functionality of their sub-agents, and when queries arrive in Interagent Communication Language form, they are responsible for breaking apart any complex queries and for distributing goals to the appropriate agents. An agent solving a goal may require supporting information and the agent architecture provides numerous means of requesting data from other agents or from the user.

Among the assortment of agent architectures, the Open Agent Architecture can be most closely compared to work by the ARPA knowledge sharing community [\[10\]](#). The OAA's query protocol, Interagent Communication Language and Facilitator mechanisms have similar instantiations in the SHADE project, in the form of KQML, KIF and various independent capability matchmakers. Other agent architectures, such as General Magic's Telescript [\[11\]](#), MASCOS [\[21\]](#), or the CORBA distributed object approach [\[17\]](#) do not provide as fully developed mechanisms for interagent communication and delegation.

The Open Agent Architecture provides capability for accessing distributed knowledge sources through natural language and voice, but it is lacking integration with a synergistic multimodal interface.

[Next](#) [Up](#) [Previous](#)Next: [TAPAGE](#) Up: [Building Blocks](#) Previous: [Building Blocks](#)

Adam Cheyer

Mon Aug 12 15:07:21 PDT 1996

[Next](#) [Up](#) [Previous](#)Next: [Synthesis](#) Up: [Building Blocks](#) Previous: [Open Agent Architecture](#)

TAPAGE

TAPAGE (edition de Tableaux par la Parole et la Geste) is a synergistic pen/voice system for designing and correcting tables.

To capture signals emitted during a user's interaction, TAPAGE integrates a set of modality agents, each responsible for a very specialized kind of signal [9]. The modality agents are connected to an "interpret agent" which is responsible for combining the inputs across all modalities to form a valid command for the application. The interpret agent receives filtered results from the modality agents, sorts the information into the correct fields, performs type-checking on the arguments, and prompts the user for any missing information, according to the model of the interaction. The interpret agent is also responsible for merging the data streams sent by the modality agents, and for resolving ambiguities among them, based on its knowledge of the application's internal state. Another function of the interpret agent is to produce reflexes: reflexes are actions output at the interface level without involving the functional core of the application.

The TAPAGE system can accept multimodal input, but it is not a distributed system; its functional core is fixed. In TAPAGE, the set of linguistic input is limited to a *verb object argument* format.

Adam Cheyer

Mon Aug 12 15:07:21 PDT 1996

[Next](#) [Up](#) [Previous](#)

Next: [CONCLUSIONS](#) Up: [Approach](#) Previous: [TAPAGE](#)

Synthesis

In the Open Agent Architecture, agents are distributed entities that can run on different machines, and communicate together to solve a task for the user. In TAPAGE, agents are used to provide streams of input to a central interpret process, responsible for merging incoming data. A generalization of these two types of agents could be :

Macro Agents: contain some knowledge and ability to reason about a domain, and can answer or make queries to other macro agents using the Interagent Communication Language.

Micro Agents: are responsible for handling a single input or output data stream, either filtering the signal to or from a hierarchically superior "interpret" agent.

The network architecture that we used was hierarchical at two resolutions - micro agents are connected to a superior macro agent, and macro agents are connected in turn to a facilitator agent. In both cases, a server is responsible for the supervision of its client sub-agents.

In order to describe our implementation, we will first give a description of each agent used in our application and then illustrate the flow of communication among agents produced by a user's request.

Speech Recognition (SR) Agent: The SR agent provides a mapping from the Interagent Communication Language to the API for the Decipher (Corona) speech recognition system [\[4\]](#), a large vocabulary, continuous speech, speaker independent recognizer based on Hidden Markov Model technology. This macro agent is also responsible for supervising a child micro agent whose task is to control the speech data stream. The SR agent can provide feedback to an interface agent about the current status and progress of the micro agent (e.g. "listening", "end of speech detected", etc.) This agent is written in C.

Natural Language (NL) Parser Agent: translates English expressions into the Interagent Communication Language (ICL). For a more complete description of the ICL, see [\[5\]](#). The NL agent we selected for our application is the simplest of those integrated into the OAA. It is written in Prolog using Definite Clause Grammars, and supports a distributed vocabulary; each agent dynamically adds word definitions as it connects to the network. A current project is underway to integrate the Gemini natural language system [\[8\]](#), a robust bottom up parser and semantic interpreter specifically designed for use in Spoken Language Understanding projects.

Database Agents: Database agents can reside at local or remote locations and can be grouped hierarchically according to content. Micro agents can be connected to database agents to monitor relevant positions or events in real time. In our travel planning application, database agents provide maps for each city, as well as icons, vocabulary and information about available hotels, restaurants, movies, theaters, municipal buildings and tourist attractions. Three types of databases were used: Prolog databases, X.500 hierarchical databases, and data loaded automatically by scanning HTML pages from the World Wide Web (WWW). In one instance, a local newspaper provides weekly updates to its Mosaic-accessible list of current movie times and reviews, as well as adding several new restaurant reviews to a growing collection; this information is extracted by an HTML reading database agent and made accessible to the agent architecture. Descriptions and addresses of new

restaurants are presented to the user on request, and the user can choose to add them to the permanent database by specifying positional coordinates on the map (eg. "add this new restaurant here"), information lacking in the WWW database.

Reference Resolution Agent: This agent is responsible for merging requests arriving in parallel from different modalities, and for controlling interactions between the user interface agent, database agents and modality agents. In this implementation, the reference resolution agent is domain specific: knowledge is encoded as to what actions must be performed to resolve each possible type of ICL request in its particular domain. For a given ICL logical form, the agent can verify argument types, supply default values, and resolve argument references. Some argument references are descriptive ("How far is it to the hotel on Emerson Street?"); in this case, a domain agent will try to resolve the definite reference by sending database agent requests. Other references, particularly when contextual or deictic, are resolved by the user interface agent ("What are the rates for this hotel?"). Once arguments to a query have been resolved, this agent coordinates the actions and calculations necessary to produce the result of the request.

Interface Agent: This macro agent is responsible for managing what is currently being displayed to the user, and for accepting the user's multimodal input. The Interface Agent also coordinates client modality agents and resolves ambiguities among them: handwriting and gestures are interpreted locally by micro agents and combined with results from the speech recognition agent, running on a remote speech server. The handwriting micro-agent interfaces with the Microsoft PenWindows API and accesses a handwriting recognizer by CIC Corporation. The gesture micro-agent accesses recognition algorithms developed for TAPAGE.

An important task for the interface agent is to record which objects of each type are currently salient, in order to resolve contextual references such as "the hotel" or "where I was before." Deictic references are resolved by gestural or direct manipulation commands. If no such indication is currently specified, the user interface agent waits long enough to give the user an opportunity to supply the value, and then prompts the user for it.

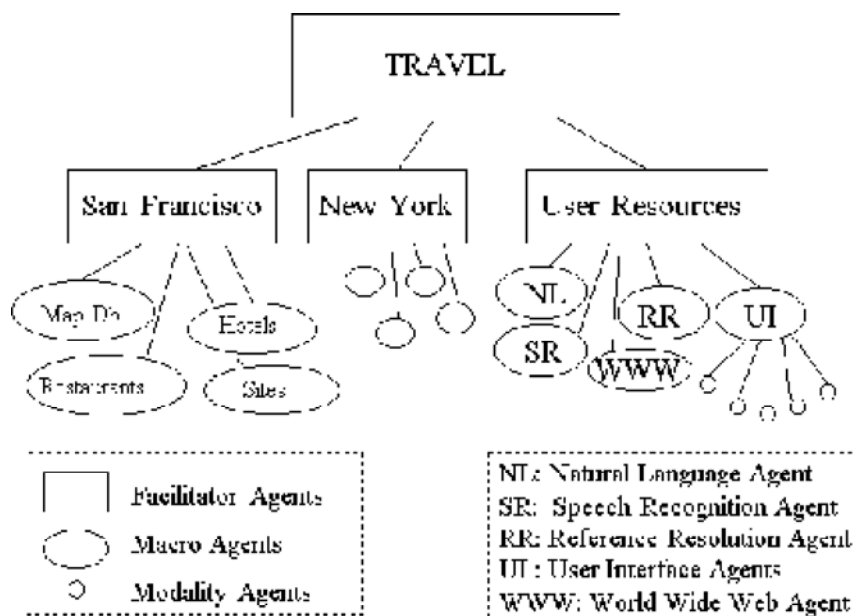


Figure 3: Agent Architecture for Map Application

We shall now give an example of the distributed interaction of agents for a specific query. In the following example, all communication among agents passes transparently through a facilitator agent in an undirected fashion; this process is left out of the description for brevity.

1. A user speaks: ``How far is the restaurant from this hotel?''
2. The speech recognition agent monitors the status and results from its micro agent, sending feedback received by the user interface agent. When the string is recognized, a translation is requested.
3. The English request is received by the NL agent and translated into ICL form.
4. The reference resolution agent (RR) receives the ICL distance request containing one definite and one deictic reference and asks for resolution of these references.
5. The interface agent uses contextual structures to find what ``the restaurant" refers to, and waits for the user to make a gesture indicating ``the hotel", issuing prompts if necessary.
6. When the references have been resolved, the domain agent (RR) sends database requests asking for the coordinates of the items in question. It then calculates the distance according to the scale of the currently displayed map, and requests the user interface to produce output displaying the result of the calculation.

[Next](#) [Up](#) [Previous](#)

Next: [CONCLUSIONS](#) **Up:** [Approach](#) **Previous:** [TAPAGE](#)

Adam Cheyer

Mon Aug 12 15:07:21 PDT 1996

[Next](#) [Up](#) [Previous](#)**Next:** [Acknowledgements](#) **Up:** [Multimodal Maps: An Agent-based](#) **Previous:** [Synthesis](#)

CONCLUSIONS

By augmenting an existing agent-based architecture with concepts necessary for synergistic multimodal input, we were able to rapidly develop a map-based application for a travel planning task. The resulting application has met our initial requirements: a mobile, synergistic pen/voice interface providing good natural language access to heterogeneous distributed knowledge sources. The approach used was general and should provide a means for developing synergistic multimodal applications for other domains.

The system described here is one of the first that accepts commands made of synergistic combinations of spoken language, handwriting and gestural input. This fusion of modalities can produce more complex interactions than in many systems and the prototype application will serve as a testbed for acquiring a deeper understanding of multimodal input.

In the near future, we will continue to verify and extend our approach by building other multimodal applications. We are interested in generalizing the methodology further; work has already begun on an agent-building tool which will simplify and automate many of the details of developing new agents and domains.

Adam Cheyer

Mon Aug 12 15:07:21 PDT 1996

[Next](#) [Up](#) [Previous](#)Next: [References](#) Up: [Multimodal Maps: An Agent-based](#) Previous: [CONCLUSIONS](#)

Acknowledgements

The work reported here would not have been possible without the inspiration of Sharon Oviatt and Phil Cohen under whose direction we worked for a year on a project (NSF Grant No. IRI-9213472) in which the combination of modalities contained in the interface presented here was crystallized and studied via simulations. Neither they nor their sponsors, of course, are responsible for the work presented here.

Adam Cheyer

Mon Aug 12 15:07:21 PDT 1996

[Next](#) [Up](#) [Previous](#)

Next: [About this document](#) **Up:** [Multimodal Maps: An Agent-based](#) **Previous:** [Acknowledgements](#)

References

- 1 Allegayer, J, Jansen-Winkel, R., Reddig, C. and Reithinger, N. ``Bidirectional use of knowledge in the multi-modal NL access system XTRA". In Proceedings of IJCAI-89, Detroit, pp. 1492-1497.
- 2 Bolt, R. ``Put that there: Voice and Gesture at the Graphic Interface". Computer Graphics, 14 (3), 1980, pp. 262-270.
- 3 Bellik, Y. and Teil, D. ``Les types de multimodalites", In Proc. IIM'92 (Paris), pp. 22-28.
- 4 Cohen, M., Murveit, H., Bernstein, J., Price, P., Weintraub, M., ``The DECIPHER Speech Recognition System". 1990 IEEE ICASSP, pp. 77-80.
- 5 Cohen, P.R., Cheyer, A., Wang, M. and Baeg, S.C. ``An Open Agent Architecture". In Proc. AAAI'94 - SA (Stanford), pp. 1-8.
- 6 Cohen, P. ``The role of natural language in a multimodal interface". Proceedings of UIST'92, 143-149.
- 7 Dauphin DTR-1 User's Manual, Dauphin Technology, Inc. 337 E. Butterfield Rd., Suite 900, Lombard, Ill 60148.
- 8 Dowding, J., Gawron, J.M., Appelt, D., Bear, J., Cherny, L., Moore, B. and Moran D., ``Gemini: A natural language system for spoken-language understanding", Technical Note 527, AI Center, SRI International, 333 Ravenswood Ave., Menlo Park, CA 94025, April 1993.
- 9 Faure, C. and Julia, L. ``An Agent-Based Architecture for a Multimodal Interface". In Proc. AAAI'94 - IM4S (Stanford), pp. 82-86.
- 10 Genesereth, M. and Singh, N.P. ``A knowledge sharing approach to software interoperation". Computer Science Department, Stanford University, unpublished ms., 1994.
- 11 General Magic, Inc., ``Telescript Product Documentation", 1995.

- 12 Julia, L. and Faure, C. "A Multimodal Interface for Incremental Graphic Document Design". HCI International '93, Orlando.
- 13 Koons, D.B., Sparrell, C.J., and Thorisson, K.R. "Integrating Simultaneous Input from Speech, Gaze and Hand Gestures". In *Intelligent Multimedia Interfaces*, Edited by Mark Maybury, Menlo Park, CA, AAAI Press, 1993.
- 14 Maybury, M.T. (ed.), *Intelligent Multimedia Interfaces*, AAAI Press/MIT Press: Menlo Park, Ca, 1993.
- 15 Neal, J.G., and Shapiro, S.C. "Intelligent Multi-media Interface Technology". In *Intelligent User Interfaces*, Edited by J. Sullivan and S. Tyler, Addison-Wesley Pub. Co., Reading, MA, 1991.
- 16 Nigay, L. and Coutaz, J. "A Design Space for Multimodal Systems: Concurrent Processing and Data Fusion". In Proc. InterCHI'93 (Amsterdam), ACM Press, pp. 172-178.
- 17 Object Management Group, "The Common Object Request Broker: Architecture and Specification", OMG Document Number 91.12.1, December 1991.
- 18 Oviatt, S. "Toward Empirically-Based Design of Multimodal Dialogue Systems". In Proc. AAAI'94 - IM4S (Stanford), pp. 30-36.
- 19 Oviatt, S. "Multimodal Interfaces for Dynamic Interactive Maps". In Proc. CHI '96, (Vancouver), pp. 95-102.
- 20 Oviatt, S. and Olsen, E. "Integration Themes in Multimodal Human-Computer Interaction". Proceedings of ICSLP'94, Yokohama, pp. 551-554.
- 21 Park, S.K., Choi J.M., Myeong-Wuk J., Lee G.L., and Lim Y.H. "MASCOS : A Multi-Agent System as the Computer Secretary". Submitted for publication.
- 22 Pfaff, G. and Ten Hagen, P.J.W. *Seeheim workshop on User Interface Management Systems* (Berlin), Springer- Verlag.
- 23 Rhyne J. "Dialogue Management for Gestural Interfaces". Computer Graphics, 21(2), 1987, pp. 137-142.
- 24

Schwartz, D.G. ``Cooperating heterogeneous systems: A blackboard-based meta approach". Technical Report 93-112, Center for Automation and Intelligent Systems Research, Case Western Reserve University, Cleveland Ohio, April 1993. Unpublished PhD. thesis.

25

Sullivan, J. and Tyler, S. (eds.), *Intelligent User Interfaces*, Addison-Wesley Pub. Co., Reading, MA, 1991.

26

Warren, D. and Pereira, F., ``An Efficient Easily Adaptable System for Interpreting Natural Language Queries", in *American Journal of Computational Linguistics*, 8(3), 1982, pp. 110-123.

27

Wauchope, K., ``Eucalyptus: Integrating Natural Language with a Graphical User Interface." Naval Research Laboratory Technical Report NRL/FR/5510-94-9711, in press, 1994.

Adam Cheyer

Mon Aug 12 15:07:21 PDT 1996

[Next](#) [Up](#) [Previous](#)

Up: [Multimodal Maps: An Agent-based](#) Previous: [References](#)

About this document ...

Multimodal Maps: An Agent-based Approach

This document was generated using the [LaTeX2HTML](#) translator Version 96.1-g (June 11, 1996)
Copyright © 1993, 1994, 1995, 1996, [Nikos Drakos](#), Computer Based Learning Unit, University of
Leeds.

The command line arguments were:

latex2html mmap.tex.

The translation was initiated by Adam Cheyer on Mon Aug 12 15:07:21 PDT 1996

Adam Cheyer

Mon Aug 12 15:07:21 PDT 1996

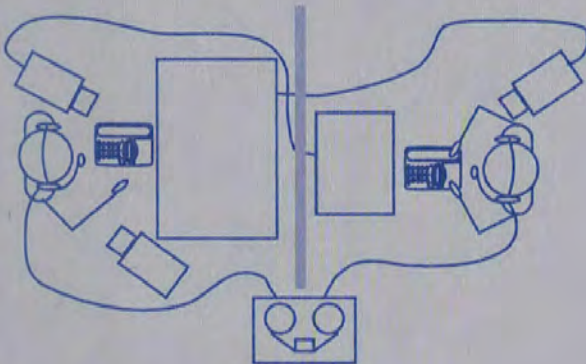
Lecture Notes in Artificial Intelligence 1374

Subseries of Lecture Notes in Computer Science

Harry Bunt Robbert-Jan Beun
Tijn Borghuis (Eds.)

Multimodal Human-Computer Communication

Systems, Techniques, and Experiments



Springer

Lecture Notes in Artificial Intelligence 1374

Subseries of Lecture Notes in Computer Science

Edited by J. G. Carbonell and J. Siekmann

Lecture Notes in Computer Science

Edited by G. Goos, J. Hartmanis and J. van Leeuwen

Springer

Berlin

Heidelberg

New York

Barcelona

Budapest

Hong Kong

London

Milan

Paris

Santa Clara

Singapore

Tokyo

Harry Bunt Robbert-Jan Beun
Tijn Borghuis (Eds.)

Multimodal Human-Computer Communication

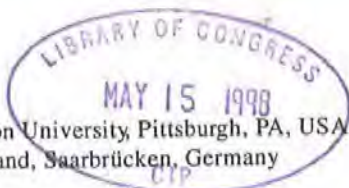
Systems, Techniques,
and Experiments



Springer

Series Editors

Jaime G. Carbonell, Carnegie Mellon University, Pittsburgh, PA, USA
Jörg Siekmann, University of Saarland, Saarbrücken, Germany



Volume Editors

Harry Bunt

Tilburg University

Warandelaan 2, 5000 LE Tilburg, The Netherlands

E-mail: bunt@kub.nl

Robbert-Jan Beun

Center for Research on User-System Interaction (IPO)

P.O. Box 513, 5600 MB Eindhoven, The Netherlands

E-mail: rjbeun@ipo.tue.nl

Tijn Borghuis

Eindhoven University of Technology

P.O. Box 513, 5600 MB Eindhoven, The Netherlands

E-mail: tijn@win.tue.nl

Cataloging-in-Publication Data applied for

Die Deutsche Bibliothek - CIP-Einheitsaufnahme

Multimodal human computer communication : systems, techniques, and experiments / Harry Bunt ... (ed.). - Berlin ; Heidelberg ; New York ; Barcelona ; Budapest ; Hong Kong ; London ; Milan ; Paris ; Santa Clara ; Singapore ; Tokyo : Springer, 1998

(Lecture notes in computer science ; Vol. 1374 : Lecture notes in artificial intelligence)

ISBN 3-540-64380-X

CR Subject Classification (1991): I.2, H.5.1-2, I.3.6, D.2.2, K.4.2

ISSN 0302-9743

ISBN 3-540-64380-X Springer-Verlag Berlin Heidelberg New York

This work is subject to copyright. All rights are reserved, whether the whole or part of the material is concerned, specifically the rights of translation, reprinting, re-use of illustrations, recitation, broadcasting, reproduction on microfilms or in any other way, and storage in data banks. Duplication of this publication or parts thereof is permitted only under the provisions of the German Copyright Law of September 9, 1965, in its current version, and permission for use must always be obtained from Springer-Verlag. Violations are liable for prosecution under the German Copyright Law.

© Springer-Verlag Berlin Heidelberg 1998

Printed in Germany

Typesetting: Camera ready by author

SPIN 10631926

06/3142 - 5 4 3 2 1 0

Printed on acid-free paper

98-3460

Preface

This volume contains revised versions of seventeen selected papers from the First International Conference on Cooperative Multimodal Communication (CMC/95), held in Eindhoven, the Netherlands, in May 1995. This was the first conference in a series, of which the second one was held in Tilburg, The Netherlands, in January 1998. Three of these papers were presented by invited speakers; those by Mark Maybury, Bonnie Webber, and Kent Wittenburg. From the submitted papers that were accepted by the CMC/95 program committee, thirteen were selected for publication in this volume, after revision.

We thank the program committee for their excellent and timely feedback to authors of submitted papers, and at a later stage for advising on the contents of this volume and for providing additional suggestions for improving the selected contributions. The program committee consisted of Norman Badler, Harry Bunt, Jeroen Groenendijk, Walther von Hahn, Dieter Huber, Hans Kamp, John Lee, Joseph Mariani, Mark Maybury, Paul Mc Kevitt, Rob Nederpelt, Kees van Overveld, Ray Perrault, Donia Scott, Wolfgang Wahlster, Bonnie Webber, and Kent Wittenburg. We thank the Royal Dutch Academy of Sciences (KNAW) and the Organization for Cooperation among Universities in Brabant (SOBU) for their grants that made the conference possible.

January 1998

Harry Bunt
Robbert-Jan Beun
Tijn Borghuis

Table of Contents

| | |
|--|---|
| Issues in Multimodal Human-Computer Communication <i>Harry Bunt</i> | 1 |
|--|---|

Part I: Systems

| | |
|---|-----|
| Towards Cooperative Multimedia Interaction <i>Mark T. Maybury</i> | 13 |
| Multimodal Cooperation with the DenK System <i>Harry Bunt, René Ahn, Robbert-Jan Beun, Tijn Borghuis and Kees van Overveld</i> | 39 |
| Synthesizing Cooperative Conversation <i>Catherine Pelachaud, Justine Cassell, Norman Badler, Mark Steedman, Scott Prevost and Matthew Stone</i> | 68 |
| Instructing Animated Agents: Viewing Language in Behavioral Terms <i>Bonnie Webber</i> | 89 |
| Modeling and Processing of Oral and Tactile Activities in the GEORAL System <i>Jacques Siroux, Marc Guyomard, Franck Multon and Christophe Remondeau</i> | 101 |
| Multimodal Maps: An Agent-Based Approach <i>Adam Cheyer and Luc Julia</i> | 111 |
| Using Cooperative Agents to Plan Multimodal Presentations <i>Yi Han and Ingrid Zukerman</i> | 122 |

Part II: Techniques

| | |
|--|-----|
| Developing Multimodal Interfaces: A Theoretical Framework and Guided Propagation Networks <i>Jean-Claude Martin, Remko Veldman and Dominique Bérroule</i> | 158 |
|--|-----|

VIII

| | |
|--|-----|
| Cooperation between Reactive 3D Objects and a Multimodal X Window Kernel for CAD | 188 |
| <i>Patrick Bourdot, Mike Krus and Rachid Gherbi</i> | |
| A Multimedia Interface for Circuit Board Assembly | 213 |
| <i>Fergal McCaffery, Michael McTear and Maureen Murphy</i> | |
| Visual Language Parsing: If I Had a Hammer... | 231 |
| <i>Kent Wittenburg</i> | |
| Anaphora in Multimodal Discourse | 250 |
| <i>John Lee and Keith Stenning</i> | |

Part III: Experiments

| | |
|--|-----|
| Speakers' Responses to Requests for Repetition in a Multimedia Language Processing Environment | 264 |
| <i>Laurel Fais, Kyung-ho Loke-Kim and Young-Duk Park</i> | |
| Object Reference in Task-Oriented Keyboard Dialogues | 279 |
| <i>Anita Cremers</i> | |
| Referent Identification Requests in Multi-Modal Dialogs | 294 |
| <i>Tsuneaki Kato and Yukiko I. Nakano</i> | |
| Studies into Full Integration of Language and Action | 312 |
| <i>Carla Huls and Edwin Bos</i> | |
| The Role of Multimodal Communication in Cooperation: The Cases of Air Traffic Control | 326 |
| <i>Marie-Christine Bressolle, Bruno Pavard and Marcel Leroux</i> | |
| Author Index | 345 |

Multimodal Maps: An Agent-Based Approach

Adam Cheyer and Luc Julia

SRI International
333 Ravenswood Ave
Menlo Park, CA 94025 - USA

Abstract. In this paper, we discuss how multiple input modalities may be combined to produce more natural user interfaces. To illustrate this technique, we present a prototype map-based application for a travel planning domain. The application is distinguished by a synergistic combination of handwriting, gesture and speech modalities; access to existing data sources including the World Wide Web; and a mobile handheld interface. To implement the described application, a hierarchical distributed network of heterogeneous software agents was augmented by appropriate functionality for developing synergistic multimodal applications.

1 Introduction

As computer systems become more powerful and complex, efforts to make computer interfaces more simple and natural become increasingly important. Natural interfaces should be designed to facilitate communication in ways people are already accustomed to using. Such interfaces allow users to concentrate on the tasks they are trying to accomplish, not worry about what they must do to control the interface.

In this paper, we begin by discussing what input modalities humans are comfortable using when interacting with computers, and how these modalities should best be combined in order to produce natural interfaces. In Sect. 3, we present a prototype map-based application for the travel planning domain which uses a synergistic combination of several input modalities. Section 4 describes the agent-based approach we used to implement the application and the work on which it is based. In Sect. 5, we summarize our conclusions and future directions.

2 Natural Input

2.1 Input Modalities

Direct manipulation interface technologies are currently the most widely used techniques for creating user interfaces. Through the use of menus and a graphical user interface, users are presented with sets of discrete actions and the objects on which to perform them. Pointing devices such as a mouse facilitate selection

of an object or action, and drag and drop techniques allow items to be moved or combined with other entities or actions.

With the addition of electronic pen devices, gestural drawings add a new dimension direct manipulation interfaces. Gestures allow users to communicate a surprisingly wide range of meaningful requests with a few simple strokes. Research has shown that multiple gestures can be combined to form dialog, with rules of temporal grouping overriding temporal sequencing (Rhyne, 1987). Gestural commands are particularly applicable to graphical or editing type tasks.

Direct manipulation interactions possess many desirable qualities: communication is generally fast and concise; input techniques are easy to learn and remember; the user has a good idea about what can be accomplished, as the visual presentation of the available actions is generally easily accessible. However, direct manipulation suffers from limitations when trying to access or describe entities which are not or can not be visualized by the user.

Limitations of direct manipulation style interfaces can be addressed by another interface technology, that of natural language interfaces. Natural language interfaces excel in describing entities that are not currently displayed on the monitor, in specifying temporal relations between entities or actions, and in identifying members of sets. These strengths are exactly the weaknesses of direct manipulation interfaces, and concurrently, the weaknesses of natural language interfaces (ambiguity, conceptual coverage, etc.) can be overcome by the strengths of direct manipulation.

Natural language content can be entered through different input modalities, including typing, handwriting, and speech. It is important to note that, while the same textual content can be provided by the three modalities, each modality has widely varying properties.

- Spoken language is the modality used first and foremost in human-human interactive problem solving (Cohen et al., 1990). Speech is an extremely fast medium, several times faster than typing or handwriting. In addition, speech input contains content that is not present in other forms of natural language input, such as prosody, tone and characteristics of the speaker (age, sex, accent).
- Typing is the most common way of entering information into a computer, because it is reasonably fast, very accurate, and requires no computational resources.
- Handwriting has been shown to be useful for certain types of tasks, such as performing numerical calculations and manipulating names which are difficult to pronounce (Oviatt, 1994; Oviatt and Olson, 1994). Because of its relatively slow production rate, handwriting may induce users to produce different types of input than is generated by spoken language; abbreviations, symbols and non-grammatical patterns may be expected to be more prevalent amid written input.

2.2 Combination of Modalities

As noted in the previous section, direct manipulation and natural language seem to be very complementary modalities. It is therefore not surprising that a number of multimodal systems combine the two.

Notable among such systems is the Cohen's Shoptalk system (Cohen, 1992), a prototype manufacturing and decision-support system that aids in tasks such as quality assurance monitoring, and production scheduling. The natural language module of Shoptalk is based on the Chat-85 natural language system (Warren and Perreira, 1982) and is particularly good at handling time, tense, and temporal reasoning.

A number of systems have focused on combining the speed of speech with the reference provided by direct manipulation of a mouse pointer. Such systems include the XTRA system (Allegayer et al, 1989), CUBRICON (Neal and Shapiro, 1991), the PAC-Amodeus model (Nigay and Coutaz, 1993), and TAPAGE (Faure and Julia, 1994).

XTRA and CUBRICON are both systems that combine complex spoken input with mouse clicks, using several knowledge sources for reference identification. CUBRICON's domain is a map-based task, making it similar to the application developed in this paper. However, the two are different in that CUBRICON can only use direct manipulation to indicate a specific item, whereas our system produces a richer mixing of modalities by adding both gestural and written language as input modalities.

The PAC-Amodeus systems such as VoicePaint and Notebook allow the user to synergistically combine vocal or mouse-click commands when interacting with notes or graphical objects. However, due to the selected domains, the natural language input is very simple, generally of the style "Insert a note here".

TAPAGE is another system that allows true synergistic combination of spoken input with direct manipulation. Like PAC-Amodeus, TAPAGE's domain provides only simple linguistic input. However, TAPAGE uses a pen-based interface instead of a mouse, allowing gestural commands. TAPAGE, selected as a building block for our map application, will be described more in detail in Sect. 4.2.

Other interesting work regarding the simultaneous combination of handgestures and gaze can be found in Bolt (1980) and Koons, Sparrell and Thorisson (1993).

3 A Multimodal Map Application

In this section, we will describe a prototype map-based application for a travel planning domain. In order to provide the most natural user interface possible, the system permits the user to simultaneously combine direct manipulation, gestural drawings, handwritten, typed and spoken natural language. When designing the system, other criteria were considered as well:



Fig. 1. Multimodal application for travel planning

- The user interface must be light and fast enough to run on a handheld PDA while able to access applications and data that may require a more powerful machine.
- Existing commercial or research natural language and speech recognition systems should be used.
- Through the multimodal interface, a user must be able to transparently access a wide variety of data sources, including information stored in HTML form on the World Wide Web.

As illustrated in Fig. 1, the user is presented with a pen sensitive map display on which drawn gestures and written natural language statements may be combined with spoken input. As opposed to a static paper map, the location, resolution, and content presented by the map change, according to the requests of the user. Objects of interest, such as restaurants, movie theaters, hotels, tourist sites, municipal buildings, etc. are displayed as icons. The user may ask the map to perform various actions. For example :

- *distance calculation* : e.g. "How far is the hotel from Fisherman's Wharf?"
- *object location* : e.g. "Where is the nearest post office?"
- *filtering* : e.g. "Display the French restaurants within 1 mile of this hotel."
- *information retrieval* : e.g. "Show me all available information about Alcatraz."

The application also makes use of multimodal (multimedia) output as well as input: video, text, sound and voice can all be combined when presenting an answer to a query.

During input, requests can be entered using gestures (see Fig. 2 for sample gestures), handwriting, voice, or a combination of pen and voice. For instance, in order to calculate the distance between two points on the map, a command may be issued using the following:

- *gesture*, by simply drawing a line between the two points of interest.
- *voice*, by speaking "What is the distance from the post office to the hotel?".
- *handwriting*, by writing "dist p.o. to hotel?"
- *synergistic combination of pen and voice*, by speaking "What is the distance from here to this hotel?" while simultaneously indicating the specified locations by pointing or circling.

Notice that in our example of synergistic combination of pen and voice, the arguments to the verb "distance" can be specified before, at the same time, or shortly after the vocalization of the request to calculate the distance. If a user's request is ambiguous or underspecified, the system will wait several seconds and then issue a prompt requesting additional information.

The user interface runs on pen-equipped PC's or a Dauphin handheld PDA (Dauphin, DTR-1 User's Manual) using either a microphone or a telephone for voice input. The interface is connected either by modem or ethernet to a server machine which will manage database access, natural language processing and speech recognition for the application. The result is a mobile system that provides a synergistic pen/voice interface to remote databases.

In general, the speed of the system is quite acceptable. For gestural commands, which are handled locally on the user interface machine, a response is produced in less than one second. For handwritten commands, the time to recognize the handwriting, process the English query, access a database and begin to display the results on the user interface is less than three seconds (assuming an ethernet connection, and good network and database response). Solutions to verbal commands are displayed in three to five seconds after the end of speech has been detected; partial feedback indicating the current status of the speech recognition is provided earlier.

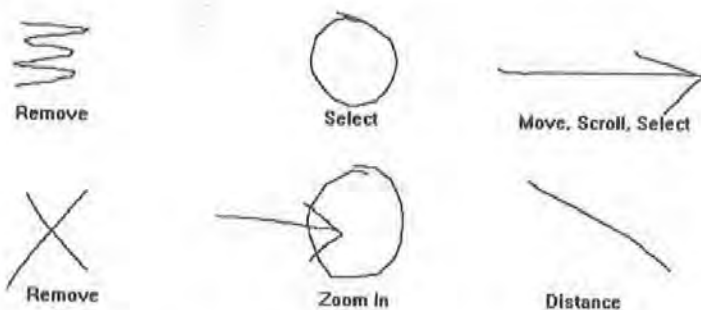


Fig. 2. Sample gestures

4 Approach

In order to implement the application described in the previous section, we chose to augment a proven agent-based architecture with functionalities developed for a synergistically multimodal application. The result is a flexible methodology for designing and implementing distributed multimodal applications.

4.1 Building Blocks

Open Agent Architecture. The Open Agent Architecture (OAA) (Cohen et al., 1994) provides a framework for coordinating a society of agents which interact to solve problems for the user. Through the use of agents, the OAA provides distributed access to commercial applications, such as mail systems, calendar programs, databases, etc.

The Open Agent Architecture possesses several properties which make it a good candidate for our needs:

- An Interagent Communication Language (ICL) and Query Protocol have been developed, allowing agents to communicate among themselves. Agents can run on different platforms and be implemented in a variety of programming languages.
- Several natural language systems have been integrated into the OAA which convert English into the Interagent Communication Language. In addition, a speech recognition agent has been developed to provide transparent access to the Corona speech recognition system.
- The agent architecture has been used to provide natural language and agent access to various heterogeneous data and knowledge sources.
- Agent interaction is very fine-grained. The architecture was designed so that a number of agents can work together, when appropriate in parallel, to produce fast responses to queries.

The architecture for the OAA, based loosely on Schwartz's FLiPSiDE system (Schwartz, 1993), uses a hierarchical configuration where client agents connect to a "facilitator" server. Facilitators provide content-based message routing, global data management, and process coordination for their set of connected agents. Facilitators can, in turn, be connected as clients of other facilitators. Each facilitator records the published functionality of their sub-agents, and when queries arrive in Interagent Communication Language form, they are responsible for breaking apart any complex queries and for distributing goals to the appropriate agents. An agent solving a goal may require supporting information and the agent architecture provides numerous means of requesting data from other agents or from the user.

Among the assortment of agent architectures, the Open Agent Architecture can be most closely compared to work by the ARPA knowledge sharing community (Genesereth and Singh, 1994). The OAA's query protocol, Interagent Communication Language and Facilitator mechanisms have similar instantiations in

the SHADE project, in the form of KQML, KIF and various independent capability matchmakers. Other agent architectures, such as General Magic's Telescript (General Magic, 1995), MASCOS (Park et al, submitted), or the CORBA distributed object approach (Object Management Group, 1991) do not provide as fully developed mechanisms for interagent communication and delegation.

The Open Agent Architecture provides capability for accessing distributed knowledge sources through natural language and voice, but it is lacking integration with a synergistic multimodal interface.

TAPAGE. TAPAGE (edition de Tableaux par la Parole et la Geste) is a synergistic pen/voice system for designing and correcting tables.

To capture signals emitted during a user's interaction, TAPAGE integrates a set of modality agents, each responsible for a very specialized kind of signal (Faure and Julia, 1994). The modality agents are connected to an 'interpret agent' which is responsible for combining the inputs across all modalities to form a valid command for the application. The interpret agent receives filtered results from the modality agents, sorts the information into the correct fields, performs type-checking on the arguments, and prompts the user for any missing information, according to the model of the interaction. The interpret agent is also responsible for merging the data streams sent by the modality agents, and for resolving ambiguities among them, based on its knowledge of the application's internal state. Another function of the interpret agent is to produce reflexes: reflexes are actions output at the interface level without involving the functional core of the application.

The TAPAGE system can accept multimodal input, but it is not a distributed system; its functional core is fixed. In TAPAGE, the set of linguistic input is limited to a *verb object argument* format.

4.2 Synthesis

In the Open Agent Architecture, agents are distributed entities that can run on different machines, and communicate together to solve a task for the user. In TAPAGE, agents are used to provide streams of input to a central interpret process, responsible for merging incoming data. A generalization of these two types of agents could be:

Macro Agents: contain some knowledge and ability to reason about a domain, and can answer or make queries to other macro agents using the Interagent Communication Language.

Micro Agents: are responsible for handling a single input or output data stream, either filtering the signal to or from a hierarchically superior 'interpret' agent.

The network architecture that we used was hierarchical at two resolutions: micro agents are connected to a superior macro agent, and macro agents are connected in turn to a facilitator agent. In both cases, a server is responsible for the supervision of its client sub-agents.

In order to describe our implementation, we will first give a description of each agent used in our application and then illustrate the flow of communication among agents produced by a user's request.

Speech Recognition (SR) Agent: The SR agent provides a mapping from the Interagent Communication Language to the API for the Decipher (Corona) speech recognition system (Cohen et al., 1990), a continuous speech speaker independent recognizer based on Hidden Markov Model technology. This macro agent is also responsible for supervising a child micro agent whose task is to control the speech data stream. The SR agent can provide feedback to an interface agent about the current status and progress of the micro agent (e.g. "listening", "end of speech detected", etc.) This agent is written in C.

Natural Language (NL) Parser Agent: translates English expressions into the Interagent Communication Language (ICL). For a more complete description of the ICL, see Cohen et al. (Cohen et al., 1994). The NL agent we selected for our application is the simplest of those integrated into the OAA. It is written in Prolog using Definite Clause Grammars, and supports a distributed vocabulary; each agent dynamically adds word definitions as it connects to the network. A current project is underway to integrate the Gemini natural language system (Cohen et al., 1990), a robust bottom up parser and semantic interpreter specifically designed for use in Spoken Language Understanding projects.

Database Agents: Database agents can reside at local or remote locations and can be grouped hierarchically according to content. Micro agents can be connected to database agents to monitor relevant positions or events in real time. In our travel planning application, database agents provide maps for each city, as well as icons, vocabulary and information about available hotels, restaurants, movies, theaters, municipal buildings and tourist attractions. Three types of databases were used: Prolog databases, X.500 hierarchical databases, and data loaded automatically by scanning HTML pages from the World Wide Web (WWW). In one instance, a local newspaper provides weekly updates to its Mosaic-accessible list of current movie times and reviews, as well as adding several new restaurant reviews to a growing collection; this information is extracted by an HTML reading database agent and made accessible to the agent architecture. Descriptions and addresses of new restaurants are presented to the user on request, and the user can choose to add them to the permanent database by specifying positional coordinates on the map (e.g. "add this new restaurant here"), information lacking in the WWW database.

Reference Resolution Agent: This agent is responsible for merging requests arriving in parallel from different modalities, and for controlling interactions between the user interface agent, database agents and modality agents. In this implementation, the reference resolution agent is domain specific: knowledge is encoded as to what actions must be performed to resolve each possible type of ICL request in its particular domain. For a given ICL logical form, the agent can verify argument types, supply default values, and resolve argument references. Some argument references are descriptive ("How far is it to the hotel on Emerson Street?"); in this case, a domain agent will try to resolve the definite reference by

sending database agent requests. Other references, particularly when contextual or deictic, are resolved by the user interface agent ("What are the rates for this hotel?"). Once arguments to a query have been resolved, this agent coordinates the actions and calculations necessary to produce the result of the request.

Interface Agent: This macro agent is responsible for managing what is currently being displayed to the user, and for accepting the user's multimodal input. The Interface Agent also coordinates client modality agents and resolves ambiguities among them: handwriting and gestures are interpreted locally by micro agents and combined with results from the speech recognition agent, running on a remote speech server. The handwriting micro-agent interfaces with the Microsoft PenWindows API and accesses a handwriting recognizer by CIC Corporation. The gesture micro-agent accesses recognition algorithms developed for TAPAGE.

An important task for the interface agent is to record which objects of each type are currently salient, in order to resolve contextual references such as "the hotel" or "where I was before." Deictic references are resolved by gestural or direct manipulation commands. If no such indication is currently specified, the user interface agent waits long enough to give the user an opportunity to supply the value, and then prompts the user for it.

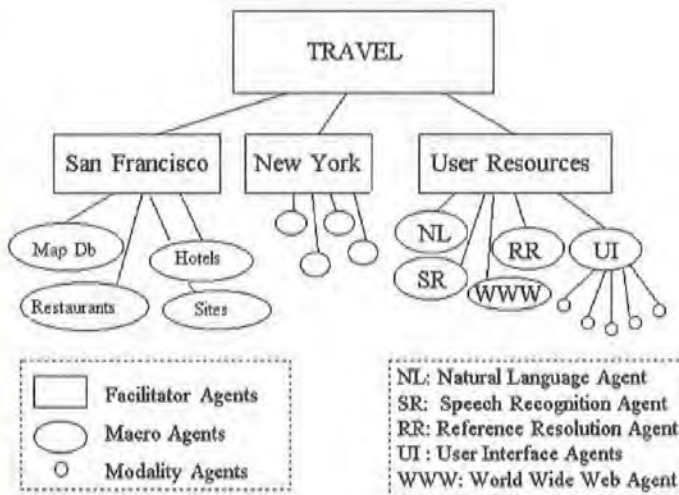


Fig. 3. Agent Architecture for Map Application

We shall now give an example of the distributed interaction of agents for a specific query. In the following example, all communication among agents passes

transparently through a facilitator agent in an undirected fashion; this process is left out of the description for brevity.

1. A user speaks: "How far is the restaurant from this hotel?"
2. The speech recognition agent monitors the status and results from its micro agent, sending feedback received by the user interface agent. When the string is recognized, a translation is requested.
3. The English request is received by the NL agent and translated into ICL form.
4. The reference resolution agent (RR) receives the ICL distance request containing one definite and one deictic reference and asks for resolution of these references.
5. The interface agent uses contextual structures to find what "the restaurant" refers to, and waits for the user to make a gesture indicating "the hotel", issuing prompts if necessary.
6. When the references have been resolved, the domain agent (RR) sends database requests asking for the coordinates of the items in question. It then calculates the distance according to the scale of the currently displayed map, and requests the user interface to produce output displaying the result of the calculation.

5 Conclusions

By augmenting an existing agent-based architecture with concepts necessary for synergistic multimodal input, we were able to rapidly develop a map-based application for a travel planning task. The resulting application has met our initial requirements: a mobile, synergistic pen/voice interface providing good natural language access to heterogeneous distributed knowledge sources. The approach used was general and should provide a for developing synergistic multimodal applications for other domains.

The system described here is one of the first that accepts commands made of synergistic combinations of spoken language, handwriting and gestural input. This fusion of modalities can produce more complex interactions than in many systems and the prototype application will serve as a testbed for acquiring a better understanding of multimodal input.

In the near future, we will continue to verify and extend our approach by building other multimodal applications. We are interested in generalizing the methodology even further; work has already begun on an agent-building tool which will simplify and automate many of the details of developing new agents and domains.

References

- Allegayer, J., Jansen-Winkel, R., Reddig, C. and Reithinger, N. (1989) Bidirectional use of knowledge in the multi-modal NL access system XTRA. In *Proceedings of IJCAI-89*, Detroit, pp. 1492-1497.

- Bolt, R. (1980) Put that there: Voice and Gesture at the Graphic Interface, *Computer Graphics*, 14(3), pp. 262-270.
- Cohen, M., Murveit, H., Bernstein, J., Price, P., and Weintraub, M. (1990) The DE-CIPHER Speech Recognition System. In *1990 IEEE ICASSP*, pp. 77-80.
- Cohen, P. (1992) The role of natural language in a multimodal interface. In *Proceedings of UIST'92*, pp. 143-149.
- Cohen, P.R., Cheyer, A., Wang, M. and Baeg, S.C. (1994) An Open Agent Architecture. In *Proceedings AAAI'94 - SA*, Stanford, pp. 1-8.
- Dauphin DTR-1 User's Manual, Dauphin Technology, Inc., Lombard, Ill 60148.
- Faure, C. and Julia, L. (1994) An Agent-Based Architecture for a Multimodal Interface. In *Proceedings AAAI'94 - IM4S*, Stanford, pp. 82-86.
- Genesereth, M. and Singh, N.P. (1994) *A knowledge sharing approach to software inter-operation*, unpublished manuscript, Computer Science Department, Stanford University.
- Telescript Product Documentation (1995), General Magic Inc.
- Koons, D.B., Sparrell, C.J., and Thorisson, K.R. (1993) Integrating Simultaneous Input from Speech, Gaze and Hand Gestures. In *Intelligent Multimedia Interfaces*, Maybury, M.T. (ed.), Menlo Park: AAAI Press/MIT Press.
- Maybury, M.T. (ed.) (1993) *Intelligent Multimedia Interfaces*, Menlo Park: AAAI Press/MIT Press.
- Neal, J.G., and Shapiro, S.C. (1991) Intelligent Multi-media Interface Technology. In *Intelligent User Interfaces*, Sullivan, J.W. and Tyler, S.W. (eds.), Reading: Addison-Wesley Pub. Co., pp. 11-43.
- Nigay, L. and Coutaz, J. (1993) A Design Space for Multimodal Systems: Concurrent Processing and Data Fusion. In *Proceedings InterCHI'93*, Amsterdam, ACM Press, pp. 172-178.
- Object Management Group (1991) *The Common Object Request Broker: Architecture and Specification*, OMG Document Number 91.12.1.
- Oviatt, S. (1994) Toward Empirically-Based Design of Multimodal Dialogue Systems. In *Proceedings of AAAI'94 - IM4S*, Stanford, pp. 30-36.
- Oviatt, S. and Olsen, E. (1994) Integration Themes in Multimodal Human-Computer Interaction. In *Proceedings of ICSLP'94*, Yokohama, pp. 551-554.
- Park, S.K., Choi J.M., Myeong-Wuk J., Lee G.L., and Lim Y.H. (submitted for publication), *MASCOS : A Multi-Agent System as the Computer Secretary*.
- Rhyne J. (1987) Dialogue Management for Gestural Interfaces, *Computer Graphics*, 21(2), pp. 137-142.
- Schwartz, D.G. (1993) *Cooperating heterogeneous systems: A blackboard-based meta approach*, Technical Report 93-112, Center for Automation and Intelligent Systems Research, Case Western Reserve University, Cleveland Ohio, (unpublished PhD. thesis).
- Sullivan, J. and Tyler, S. (eds.) (1991) *Intelligent User Interfaces*, Reading: Addison-Wesley Pub. Co.
- Warren, D. and Pereira, F. (1982) An Efficient Easily Adaptable System for Interpreting Natural Language Queries, *American Journal of Computational Linguistics*, 8(3), pp. 110-123.

STATE PROBLEMS IN PROGRAMMING HUMAN-CONTROLLED DEVICES

Joseph A. Konstan
Department of Computer Science
University of Minnesota
Minneapolis, MN 55455

ABSTRACT

Many consumer goods are complicated enough to benefit from programmed control. Today's home electronics devices support a wide range of options and controls. At the same time, personal digital assistants and programmable remote controls are now capable of learning and generating control sequences to control a wide range of devices. Unfortunately, most device interfaces are designed for interactive human control rather than programmed control.

This paper analyzes state-based obstacles to programming devices designed for interactive human control. It develops a theory of *stalelock*, a condition in which a control program is unable to synchronize with the state machine underlying the controlled device. The paper also presents design strategies to avoid stalelock and applies these strategies to the home audio/video and telephone autodialer domains.

KEYWORDS: Device interface, programmable remote control, automata, user/machine systems, audio/video control, telephone autodialers.

INTRODUCTION

Many consumer goods are complicated enough to benefit from programmed control. Today's home electronics devices support a wide range of options and controls.

At the same time, the emergence of personal digital assistants has created new possibilities for programmed device control. Basic PDA's can dial stored phone numbers. More advanced ones can also send messages to computers or facsimile machines, control remote devices using tone dialing, or even store and play back infrared control sequences such as are used for controlling televisions and other consumer audio/video devices.

Unfortunately most devices are not designed for programmed control. Consumer electronics devices can easily be controlled by a human with an infrared remote control, but only because the human can observe the state of the device and act accordingly. Programmed control units lack state awareness, and accordingly are generally unable to achieve even simple goals.

This paper discusses state awareness problems in building programmable controllers for devices with interfaces designed for interactive use. The next section describes in detail the problems involved with programming a controller for home audio/video equipment. It also formalizes the problems by defining *stalelock*—a condition that inhibits programmability. The following sections present theoretical results to show that state-awareness problems are fundamental and present design strategies for avoiding stalelock. The last section discusses the results and shows how they apply to a different control problem—the use of automatic telephone dialers-- and presents other related obstacles to building programmable controls for consumer electronics devices.

THE PROBLEM

"How can this 'universal' remote be programmed to enter 'home theater' mode?" A learning universal remote control unit controls multiple devices (e.g., televisions, video cassette recorders, stereo systems, etc.) by learning the infrared commands from those devices' individual controls. Many units come with sequence programming features to allow the user to define complex operations that are invoked by a single button. A typical goal is to define a "home theater" button that turns the television to a designated channel (typically channel 3 or 4 in the U.S.), turns the VCR on and sets it to display onto the TV, selects VCR input on the stereo system, and sets a moderate volume on the stereo (and no volume on the TV). All together, this

mode would control three devices directly and perhaps others indirectly (selecting VCR input turns off the CD and cassette players on some models of receiver) to provide the type of home entertainment so often advertised!

Unfortunately, there is one problem involved in programming the home theater button: in most cases, it cannot be done. In fact, there are many simpler operations that cannot be programmed. This is not due to the incompetence of the programmer, nor due to the weakness of the universal remote control device. Rather, the problem is one of poor programmability in the devices being controlled.

To illustrate the fundamental problem in programming remote-controllable devices, examine the simplest operation that cannot be programmed on many TV sets. There is no way to program a button to turn on the TV. There is a power button, but it operates as a toggle. It will turn on the TV if it is already off, and it will turn it off if it is already on. This implementation usually works well when a live human is operating the remote control (at worst, an error can be easily corrected), but a program has no way of knowing whether the TV is on or off. For the home theater button, the user would have to either define two buttons, one when starting with the TV off and another when starting with it on, or assume the TV is always on (or off) before entering theater mode (and therefore having to remember to make it so before pressing the magic key).

The problem, simply stated, is that many TVs, VCRs, and other devices have internal states that are not always known to the remote control. Controllable devices are simple finite state machines and remote control units generate command tokens that trigger transitions between states. Figure 1, for example, shows the simplest useful device—one which has two states, on and off. This state machine could support

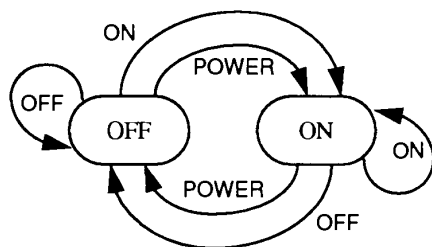
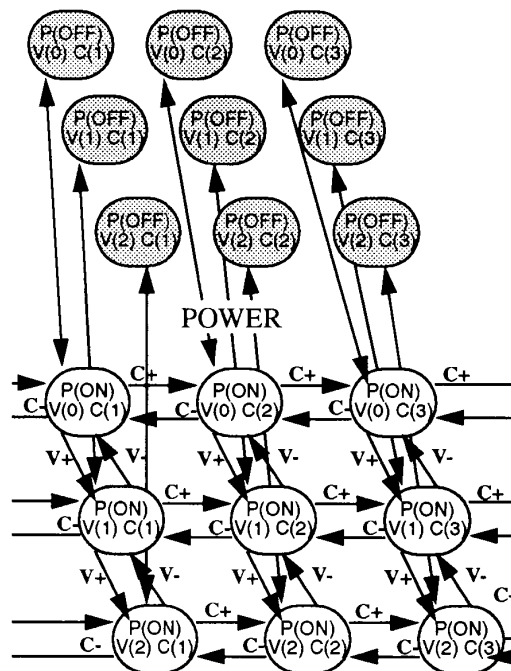


Figure 1. TV Power State Machine

three useful command tokens, ON, OFF, and POWER

(a toggle). With ON and OFF, the machine is completely programmable remotely, since commands can force it into a specific state regardless of the state in which the device starts. With POWER only, the machine is not remotely programmable if the starting state is not known. Interestingly, a machine with POWER and either ON or OFF is completely programmable (e.g., with POWER and OFF, OFF turns the machine off, OFF followed by POWER turns it ON) as long as excess intermediate states do not matter.

An actual home entertainment system has many more controls and many more states. Most of these states are made visible to a live user (or can be made visible by using a display function), though many of them are hidden until a transition moves the machine into a more visible state. Figure 2 shows a simplified model of a typical TV with states corresponding to 3 volume levels, three channels, and power and with transitions for volume up and down, channel up and down, and power toggle (adding direct channel access would render the diagram completely unreadable; an alternative representation for complex state machines is presented in [3]). The states shown in grey are “hidden.” They can-



note: unmarked transitions remain in same state

Figure 2. Simple TV State Diagram

not be distinguished merely by looking at the device. They are not the same state, however, since each encodes a different volume/channel pair.

The problem of remote programmability is the problem of creating a sequence of command tokens (i.e., and input string) that always leave the device(s) being controlled in the same desired state. (several analogous problems are discussed below).

In general, absolute access is easy to program and toggles are hard to program. In an n-dimensional state space, absolute access can be used along each dimension to reach the desired state. Relative access can only be programmed if there are fixed ends (e.g., a minimum and maximum volume) after which the command token causes the device to remain in the same state. Any level can be attained by first attaining a fixed level (e.g., by transmitting a number of VOLUME DOWN tokens larger than the number of volume levels) and then moving to the destination level through relative commands (e.g., VOLUME UP). Relative access in a cycle, including the two-element toggle, cannot generally be programmed.

In practice, device state machines are more complicated than this simple model suggests. Even the basic TV state machine shown in figure 2 has an asymmetry in it. When the power is off, channel and volume command tokens do not cause transitions to other states. Accordingly, any program must first turn the power on and then use volume and channel commands.

More complicated devices have still more complicated state machines. A typical VCR has several toggles with unusual interactions: the POWER toggle, the TV/VCR toggle, the TIMER toggle, and various PLAY/STOP, RECORD/STOP, and PAUSE/SLOW toggles. Figure 3 shows the interaction pattern for the first three of these states in a particular model, leaving out other states and transitions such as channel selection and play/record/stop/pause. Programming the VCR requires a nearly complete knowledge of the state machine, and some programs are still not possible because toggles such as PAUSE and SLOW are time- and context-dependent.

Problem Summary

Most devices designed for human control model state machines with internal state and externally defined command tokens. The user selects command tokens based on the externally visible state attributes of the machine. Programmable control, however, is more difficult because the start state of the machine is

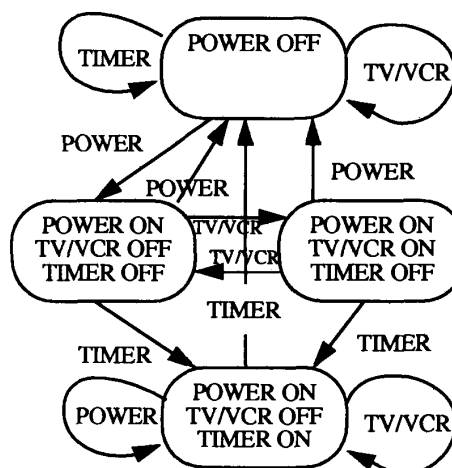


Figure 3. VCR State Interaction

unknown. Accordingly, many such devices cannot be programmed to reach a specific state.

The term *statelock* is used to refer to this lack of programmability. There are four conditions for statelock:

- Controlled devices must have internal state.
- The remote control program cannot determine the state of the device.
- There is no fixed string of command tokens to bring the device to a known state.
- The device state can change without the program being aware.

The first three conditions were discussed above.

Even with these conditions, however, it is possible to synchronize the device and the remote control program if the program is always made aware of any commands. A programmable remote control for a television, for example, could be synchronized with the television at a certain channel, volume, power status, etc. This possibility leads to the fourth condition which states that external agents can change the device state without the program's knowledge (e.g., a user can control the device manually or the device state can be altered by environmental conditions such as power failure).

It is also useful to include an assumption that any state can be reached from any other through a sequence of command tokens. Without this assumption, statelock could occur simply by leaving the device in a state from which the goal state is inaccessible. For practical

purposes, this assumption is true for almost any consumer device. Some notable exceptions are discussed at the end of this paper.

The four conditions of statelock are formulated so as to allow programmability to be established by nullifying a single condition. The first condition is fundamental to both the home audio/video example and the telephone example discussed below, and is likely to be true for any interesting device. The other three conditions, however, can be avoided. The next section presents prior theoretical work on determining whether a given device can be forced into a known state with a fixed command string. The following section discusses device and command set design options that avoid statelock by nullifying each of the last three conditions.

THEORETICAL BACKGROUND

While little theoretical research has been done on remote control applications themselves, results in automata theory and its applications can be applied to the remote control problem. The second and third conditions of statelock (i.e., unknown state and no fixed string to lead to a known state) correspond to the distinguishing sequence and synchronizing sequence problems for finite state machines [2].

The distinguishing sequence problem seeks either a string of tokens that generates a different output for each initial state in a finite state machine. Not every finite state machine has a preset distinguishing sequence (and it is a PSPACE-complete problem to find a preset distinguishing sequence). Adaptive distinguishing sequences, which change the input string based on output, can be found in polynomial time and have a bounded length of $O(n^2)$ [6]. Remote control devices cannot generally take advantage of the machines that have distinguishing sequences, even when the sequences are known, since they are not capable of observing and analyzing output.

Synchronizing sequences are strings of input tokens that take a finite state machine to a specific state regardless of the initial state. Any machine with a synchronizing sequence can be programmed by first using the synchronizing sequence to reach the known state and then sending a command string to reach the goal state from the known state.

Not all finite state machines have synchronizing sequences. Further, those synchronizing sequences that exist have a length bounded by $O(n^3)$ which is impractical for use with most current consumer electronic

devices (e.g., a mid-range television typically has at least 3200 states: 80 channels * 10 volume levels * 2 power states * 2 mute states). Because of this impracticality, the third condition for statelock can be extended to state that there is no short fixed string of command tokens to bring the device to a known state. "Short" can be defined by context as the number of command tokens that can be transmitted, received, and processed in a suitable time interval (e.g., three to five seconds if a user activates the option, perhaps a minute if the option is timer-activated).

Since finite state machine theory cannot nullify the second and third conditions of statelock, and since it does not address the first and fourth conditions, it is necessary to design more restrictive interfaces or automata to ensure programmability.

DESIGN STRATEGIES

This section presents four design strategies for ensuring programmability of remote controlled devices by avoiding statelock.

Ask and Ye Shall Know

One way to avoid statelock is to allow the remote control program to determine the state of the controlled device. Some high-end video products (specifically frame addressable video disk and video cassette players) provide a two-way communication link (generally serial RS232 communications) between the device and a controlling computer. The command set includes state queries (e.g., what frame is displayed, what is the play/pause/stop status, etc.) that generate replies. Remote control programs operate by querying the device state and sending appropriate commands to reach the goal state.

A full communication interface not only solves the statelock problem but also has other programming benefits. The remote control device can include conditional execution (e.g., eject only if there is a tape or disk loaded) and can be given access to any information available in the device itself.

Full communication interfaces, however, are much more expensive and complicated to implement. They require a two-way communication link between the remote control and the controlled device. Wire links are economical, but they limit portability and mobility. Worse yet, the programming complexity requires that either the remote control or the program itself accurately model the state machine of the controlled device in order to determine the correct sequence of com-

mands needed to reach the goal state.

Riding the Bus

Many consumer-level electronics devices are designed to work together with other devices made by the same manufacturer and to share a single remote control. They communicate through a command bus that broadcasts all significant state-changing actions. One company's consumer audio components, for example, communicate through a wired bus system to ensure that active devices are switched through the receiver and other devices are inactive (i.e., selecting PLAY, either manually or through a remote control, switches the receiver input and also selects STOP on other devices).

Programmable remote control devices could be designed to monitor this command bus. In doing so, they could prevent statelock by preventing the device from changing state without the program being aware.

Unfortunately, there are three major obstacles preventing widespread use of bus-monitoring remote controls: 1) the majority of devices do not yet support command busses, 2) even devices that support a command bus only broadcast commands of interest to other components—volume control and other “local” commands are not broadcast, and 3) bus-monitoring requires the same complexity and wire interface as full communication without presenting any significant long-term advantages.

Taking Control

Centralized control interfaces are a more practical alternative for ensuring that the remote control program is aware of all state changes and thereby avoiding statelock. Centralized control interfaces force all device commands to be routed to the remote control first, and then relayed to the controlled device. Several vendors sell components for assembling these interfaces for home automation. The typical kit includes infrared transmitters and receptors along with computer hardware and software to control the devices. User actions transmit signals to the computer program which then formulates an implementation and transmits commands to the individual devices. Interference between user-computer and computer-device communications is avoided either by physically isolating the infrared receptors of the devices or by using different frequencies and patterns. Users also must be prevented from using any manual controls on the device itself.

Centralized control has several advantages. First, it can be implemented to control any device that supports

remote-control access. No communication interface or command bus is needed. The remote control system merely learns the commands that the device understands and uses them to control it. Second, centralized control separates human input from programmed control. Since the system requires a general-purpose computer anyway, it is more practical to develop a high-level programming environment. Third, users can use conventional remote control devices for interactive control simplifying the user interface for new users. Finally, centralized control is a well understood model in home automation and can be easily integrated into a home automation system.

Centralized control also has several disadvantages. First, there are situations in which centralized control loses synchronization with the devices. Even when user inputs are reliably directed to the central controller, certain state changes occur directly at the device level. Power failures, for example, tend to force state changes (even when the device is merely unplugged for a few minutes). Similarly, tape and disk player/recorders have physical state corresponding to the presence of (and perhaps writeability of) a tape or disk.

Second, centralized control is inherently non-portable. Centralized control also requires expensive hardware and extensive software and removes the customary manual command access method with which most users are familiar.

Accordingly, centralized control is best suited for environments where home automation is a design priority. Centralized controllers can use two-way communication when available and also provide interfaces to telephone input, sensors, and other home automation components.

Starting From Scratch

The final solution requires redesigning the command set and state machines used in remote controlled devices. It has already been shown that it is neither possible nor feasible to find a reset sequence for the state machine associated with an arbitrary device. It is possible, however, to design machines with easily-accessible reset sequences and more regular state machines. Doing so nullifies the third condition for statelock.

There are three approaches to this solution. First, one can simply define a ground state and implement a RESET command that always causes a transition to the ground state. On a television set, for example, the

RESET might turn the set to channel 2 with the volume at minimum, the mute inactive, and the power off.

A second approach is to redesign the command set to avoid cycles in relative commands. Toggles would be replaced by two commands (i.e., ON and OFF). Larger cycles could either be replaced with absolute access (e.g., entering a channel number numerically) or by removing the cycle (e.g., not wrapping around between the highest and lowest channel). Finally, commands would be available regardless of state (e.g., channels can be changed when the TV is off or when the VCR is playing).

A final alternative is to provide a separate command set for programming from the normal remote control command set. In this way, manufacturers can maintain compatibility with their existing remote control units and user interface while allowing programmable access through a variant command set. A television manufacturer, for example, could provide only a POWER button on the device remote control but provide specifications for ON and OFF controls (e.g., through bar code) so programmable controls could use those commands. This alternative allows the human factors engineers to separate the best human interface from the programming interface.

DISCUSSION

These strategies are specifically designed to avoid statelock and thereby improve programmability to address the "home theater" example. In the long run, audio/video equipment will be designed for easy integration with home automation systems. Accordingly, full communication interfaces should become more popular even at the home consumer level. Since full-communication also provides the greatest flexibility and state knowledge (i.e., it can even detect the presence of tapes or disks), it is the preferred long-term solution to this specific problem. In the meantime, centralized control is still appealing for users willing to install complete home automation systems and manufacturers should take steps to provide a more flexible programming command set for users unwilling to do so.

Different problems have different solutions, however, and this section presents a related, but different problem: the use of portable telephone autodialers. After the discussion of autodialers, other non-statelock obstacles to programming devices designed for interactive control are introduced.

Another Example: Telephone Autodialers

The problem with programming home audio/video equipment is not limited to televisions and VCR's. Telephone autodialers face the same problem when trying to define a dialing sequence to reach a certain number from a wide range of telephones. It seems that dialing a telephone is a simple task, and one that is quite amenable to automation. Each telephone line, however, has different state information that affects how an input patterns (i.e., a sequence of digits or tones) is interpreted.

"How can this PDA (personal digital assistant) be programmed to call me at the office" is the analogous lament for autodialers. The goal task as again defined as designating a button (or command) that dials a specific phone number from anywhere that the PDA happens to be. The dialer should also enable calling line identification (a service that allows the recipient to know where the call is coming from) and disable call waiting (a service that allows a second call to interrupt and time-multiplex with the first). The dialer does not know from which phone it is placing the call. Once again, this task is often impossible when the PDA is moved to different phone within a building, let alone the nation or world. Here is an abbreviated list of the obstacles faced:

- Different access codes. Office phones often require "9" or "8" for outside calls.
- Feature codes vary (though most of the U.S. has standardized on "*70" for cancel call waiting and "*67" for enabling calling line ID).
- Feature codes are state-based. Most phone companies treat "*70" as an error if the line doesn't have call waiting. Some treat it as a toggle to enable call waiting for one call (for a charge). Similarly, "*67" is often an error in areas without calling line ID and is usually implemented as a toggle.
- Toll vs. local call prefixes. In an effort to avoid dialing errors and unexpected phone bills, many local companies require that long distance calls be dialed with a leading "1" *and* that local calls be dialed without a leading "1." A dialer would need a rate table, possibly based on the phone line's calling plan, to determine how to correctly dial the call.
- International dialing prefixes. The "+" refers to the local countries international dialing access code which varies from country to country.

International portability is out of the question today. Too many countries have phone systems that cannot support advanced features and that may not even sup-

port direct dialing of international calls. Within the United States (and within the North American nations sharing country code 1) and within any other country there should be easier portability. As with home audio/video remote controls, the fundamental problem is that phone lines carry state that cannot be queried by a programmed dialer and that the available command set does not allow the programmed dialer to set the line state with non-toggle commands.

Autodialers and statelock. Phone lines suffer the same statelock problems as audio/video equipment. The phone line has state (i.e., it has features, a local calling area, access codes, etc.). The state cannot be determined by the dialer device (i.e., there is no way for an autodialer to ask a phone line what prefix is needed for an outside line or whether call waiting is enabled). There is no fixed sequence of tones that will bring the line to a known state because of toggles (such as “*67”) and error conditions. And, state changes can occur without the dialer being aware, either when the dialer is connected to a new phone line or, less commonly, when a service order or service change alters the line state. Autodialers work fairly well on known lines because they can be made aware of line state and any state changes. They work poorly in a mobile setting because line state varies from line to line (and even the user often is unaware of the state).

Solutions. The full-communication solution addresses the state problems facing autodialers. Many telephone line states can be determined with YES/NO questions (e.g., is call-waiting enabled? is the number xxx-yyy local?). Some others might require tone responses that could be simply recorded and played back (e.g., what is the international access code? what is the outside-line access code?).

Bus-monitoring is not a useful solution for telephone autodialers. The major state changes occur when moving the dialer between phone lines, and these changes are not broadcast in any form. Similarly, centralized control offers little help both because there is no easy way for the centralized controller to obtain the relevant information and because centralized controllers are not portable and therefore could not be contained in an autodialer. Calling card speed dialing, a popular approach to automatic dialing, is based somewhat on centralized control. Calling card users still must figure out how to reach the central computer (i.e., with the appropriate access codes and toll-free number) before the “automated” part is available. Also, no provision is made for properly billing local calls that are made through calling card speed dialing.

The remaining alternative is redesigning the command interface. A RESET command is one appealing way to ensure a short synchronizing sequence. For a telephone line, RESET could turn off all optional line features (e.g., call waiting) and pre-dial the needed access codes (though feature codes and toll dialing restrictions would still be problems).

A more comprehensive solution would involve redesigning the command set to support an autodialer standard. Here is a rough design for such a command set:

- A prefix code (e.g., “*00#”) is defined to indicate that an autodialer is placing the call.
- Following the prefix code is a set of standard feature codes that are absolute (rather than toggle) and never fail (e.g., cancel call waiting is ignored if call waiting is off or not available).
- Next is a universal-form telephone number. Within the U.S., Canada, and other country-code 1 regions, for example, this is a ten digit number. International numbers would be identified by a prefix as is currently done.
- Phone companies would need to recognize the special prefix codes and would have to support a mode where local/toll calling restrictions are eliminated.
- Private phone systems (PBX’s and Centrex) would have to interpret the prefix code to include the needed “outside access” codes along with the feature codes. They should also store the entire number before initiating the call so they can determine whether the call is within the local group (or available by low-cost tie lines).

As with audio/video, there are different long- and short-term solutions. Digital telephony (e.g., ISDN) will provide the full-communication system in the long term. In the short term, a RESET command would be most practical, though a special autodialer command set would provide a more thorough solution. In all likelihood, neither will be adopted since ISDN is already being deployed, albeit slowly in some areas.

Analysis. The same statelock problems occur in telephone autodialing as in audio/video control. The major differences are: 1) state changes occur due to mobility rather than manual manipulation, and 2) the command tokens and state machines themselves change. Since the second difference can be viewed as a different start state in a larger state machine, it can be reduced to the problem that no synchronizing sequence exists. The same design approaches work in both cases, though the best designs differ due to the different details of the two problems.

Other Obstacles to Programmable Controls

Statelock is not the only obstacle to programmable control of devices designed for human control. Even when statelock is prevented, errors can occur if communications are unreliable. Error correction is generally not provided for human-controlled interfaces because feedback can be used to determine that the error occurred and the user can recover manually (e.g., a missed volume control command can be repeated, an misdialled number can be redialed after hanging up, etc.). Automated controllers cannot usually receive the feedback and accordingly have fewer recovery options available. A phone autodialer may be able to detect a connection and retry the number if no connection is reached in a specified interval. A television remote control generally can do very little unless the command set is based on absolute commands (which can be repeated). In the long run, full-communication interfaces (e.g., serial command connections or ISDN) will provide a mechanism for feedback and error handling. Until then, error handling will be necessarily limited.

A second obstacle is the existence of dangerous states. A VCR should never pass through a recording state on its way from one state to another since recording will overwrite data. Similarly, tape position should properly be considered as a component of state which makes fast forward and rewind operations dangerous since they are not generally invertible. Eject operations are similarly not invertible. Dangerous states can almost always be avoided in a sensible design, but designs should be validated to ensure that dangerous states are not inadvertently entered.

A third obstacle is the real-time nature of these controls. Telephone lines have time-outs and other time restrictions. Audio/video control programs often need to operate against real time constraints (e.g., to record broadcast programs). Accordingly, long term solutions must support real-time components, perhaps through integration with home automation systems.

RELATED WORK ON PROGRAMMED REMOTE CONTROL

Several automation projects have had to deal with problems in programmed remote control. The Home Bus system [4] placed all controls on a central bus, and provided video switching to control television output [1]. It did not, however, incorporate full communication between ordinary devices or otherwise address the problems of statelock.

The IHS system [5] supported a bus onto which infra-

red remote controls were attached. It avoided statelock through centralized control, though no formal mechanism existed to prevent users from operating devices manually.

Ironically, much of the work towards making remote controls more user friendly has the side effect of impeding programmability. A menu-driven remote control [7], for example may simplify select for users controlling devices interactively (since they need use only a simple joystick), but it also requires either a complicated graphical programming system or another command set to support programmability.

CONCLUSIONS

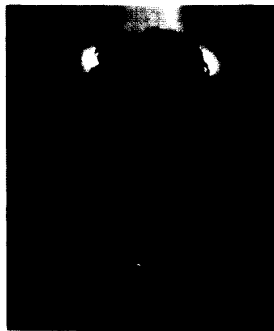
Programmable remote control of human-controlled faces many obstacles. The first, and most critical, obstacle is statelock, the inability to maintain synchronization between the control program and the state machine underlying the device being controlled. Statelock can be avoided either by making the remote control aware of machine state (i.e., through a query system or by monitoring state changes from a fixed starting state) or by ensuring that a short command sequence can bring the device to a known state (i.e., a reset command or a well-constructed command set).

In the long term, more extensive communication will be supported between devices and their controllers. Statelock can always be avoided if this communication is available. In the short term, however, the problem of statelock can best be addressed by designing a better command set. Absolute access to states (or sets of states) is better than relative access (e.g., it is better to enter a channel number than to be forced to use channel up and down buttons) and within relative access cycles should be eliminated (including toggles, which are two state cycles). When a modified command set conflicts with other human factors in interactive use (e.g., ease of learning, ease of use, etc.), it can be implemented as an addition to the command set (even a hidden addition) to preserve the original command set.

Whichever approach they take, developers of devices that can be controlled remotely and through automated systems must address the conflicts between their human interfaces and the ability to program their devices.

REFERENCES

1. Inoue, M., et. al. A home automation system. *IEEE Transactions on Consumer Electronics* (August 1985).
2. Kohavi, Z. *Switching and Finite Automata Theory*. McGraw Hill, 1970.
3. Kuuluvainen, I., et. al. A compact finite state machine representation for user interfaces and small embedded reactive systems. *IEEE Transactions on Consumer Electronics* (August 1991).
4. Murata, M., et. al. A proposal for standardization of Home Bus system for home automation. *IEEE Transactions on Consumer Electronics* (November 1983).
5. Tritton, J. Interactive Home Systems (IHS)—An overview. *IEEE Transactions on Consumer Electronics* (August 1988).
6. Yannakakis, M. Testing finite state machines (extended abstract). *Proceedings of the Twenty Third annual ACM Symposium on Theory of Computing* (May 1991).
7. Zeisel, G., et. al. An interactive menu-driven remote control unit for TV receivers and VC recorders. *IEEE Transactions on Consumer Electronics* (August 1988).



Joseph A. Konstan received an A.B. (1987) in Computer Science from Harvard College and an M.S. (1990) and Ph.D. (1993) in Computer Science from the University of California, Berkeley. He is currently an Assistant Professor in the Computer Science Department of the University of Minnesota.

He conducts research on human-computer interaction, user interface toolkits and frameworks, and multimedia systems. He is a member of the IEEE and ACM.

UNITED STATES PATENT AND TRADEMARK OFFICE

BEFORE THE PATENT TRIAL AND APPEAL BOARD

GOOGLE LLC
Petitioner

v.

IPA TECHNOLOGIES INC.
Patent Owner

Patent No. 6,757,718

POWER OF ATTORNEY FOR PETITIONER

Pursuant to 37 C.F.R. § 42.10(b), Google LLC hereby revokes any previous powers of attorney given in this proceeding and hereby appoints the practitioners associated with Paul Hastings LLP, Customer Number 36,183, including Naveen Modi, Daniel Zeilberger, and Arvind Jairam as its attorneys to transact all business before the Patent Trial and Appeal Board of the United States Patent & Trademark Office in connection with all *inter partes* review proceedings involving U.S. Patent No. 6,757,718. Counsel's contact and service information is provided below:

| Lead Counsel |
|--|
| Naveen Modi (Reg. No. 46,224) Paul Hastings LLP 875 15 th Street NW Washington, DC 20005 Telephone: (202) 551-1990 Facsimile: (202) 551-0490 E-mail: PH-Google-IPA-IPR@paulhastings.com |
| Back-Up Counsel |
| Daniel Zeilberger (Reg. No. 65,349) Paul Hastings LLP 875 15 th Street NW Washington, DC 20005 Telephone: (202) 551-1993 Facsimile: (202) 551-0493 E-mail: PH-Google-IPA-IPR@paulhastings.com |
| Back-Up Counsel |
| Arvind Jairam (Reg. No. 62,759) Paul Hastings LLP 875 15 th Street NW Washington, DC 20005 Telephone: (202) 551-1887 Facsimile: (202) 551-0387 E-mail: PH-Google-IPA-IPR@paulhastings.com |

Dated: December 21, 2017

By:

A handwritten signature in black ink, appearing to be 'Renny Hwang', written over a horizontal line.

Renny Hwang
Director, Litigation

CERTIFICATE OF SERVICE

I hereby certify that on January 12, 2018, I caused a true and correct copy of the foregoing Power of Attorney for Petitioner to be served via express mail on the Patent Owner at the following correspondence address of record as listed on PAIR:

THOMASON, MOSER & PATTERSON, LLP
595 SHREWSBURY AVENUE
SUITE 100
SHREWSBURY, NJ 07702

By: /Naveen Modi/
Naveen Modi (Reg. No. 46,224)

UNITED STATES PATENT AND TRADEMARK OFFICE

BEFORE THE PATENT TRIAL AND APPEAL BOARD

GOOGLE LLC

Petitioner

v.

IPA TECHNOLOGIES INC.

Patent Owner

Case No. IPR2018-00476

U.S. Patent No. 6,757,718

FILED: JUNE 30, 2000

ISSUED: JUNE 10, 2004

INVENTORS: CHRISTINE HALVERSON ET AL.

TITLE: MOBILE NAVIGATION OF NETWORK-BASED ELECTRONIC
INFORMATION USING SPOKEN INPUT

PATENT OWNER'S MANDATORY NOTICES
37 C.F.R. 42.8(a)(2)

Pursuant to 37 C.F.R. § 42.8(a)(2), Patent Owner submits the following mandatory notices:

(1) Real Party-in-interest

The real party-in-interest is the Patent Owner, IPA Technologies Inc., which is a wholly owned subsidiary of Wi-LAN Technologies Inc. (a Delaware corporation), which is a wholly owned subsidiary of Wi-LAN Inc. (a Canadian corporation), which is a wholly owned subsidiary of Quarterhill Inc. (a Canadian corporation publicly traded on the TSX and NASDAQ).

(2) Related matters

Pursuant to 37 C.F.R. § 42.8(b)(2), Patent Owner submits that the '718 patent is involved in the following proceedings:

DISH Network Corporation, et al. v. IPA Technologies Inc., IPR2018-00351 (PTAB); *Google LLC v. IPA Technologies Inc.*, IPR2018-00476 (PTAB); *IPA Technologies, Inc. v. DISH Network Corporation, et al.* No. 1:16-CV-01170 (D. Del.); *IPA Technologies Inc. v. NVIDIA Corporation.*, No. 1-17-cv-00287 (D. Del.); *IPA Technologies Inc. v. Sony Electronics Inc., et al.*, No. 1-17-cv-00055 (D. Del.); *IPA Technologies Inc. v. Amazon.com, Inc. et al.*, No. 1-16-cv-01266 (D. Del.).

(3) Lead and back-up counsel

Patent Owner provides the following designation and service information for lead and back-up counsel. 37 C.F.R. § 42.8(b)(3) and (b)(4). Please direct all correspondence regarding this proceeding to lead and back-up counsel at

their respective email addresses listed below. 37 C.F.R. § 42.8(b)(4).

| LEAD COUNSEL | BACK-UP COUNSEL |
|--|---|
| Steven W. Hartsell SKIERMONT DERBY LLP 1601 Elm Street, Suite 4400 Dallas, Texas 75201 Tel: (214) 978-6600 Fax: (214) 978-6601 IPA_SDTeam@skiermontderby.com Registration No: 58,788 | Alexander E. Gasser SKIERMONT DERBY LLP 1601 Elm Street, Suite 4400 Dallas, Texas 75201 Tel: (214) 978-6600 Fax: (214) 978-6601 IPA_SDTeam@skiermontderby.com Registration No: 48,760 |
| | Sarah E. Spires SKIERMONT DERBY LLP 1601 Elm Street, Suite 4400 Dallas, Texas 75201 Tel: (214) 978-6600 Fax: (214) 978-6601 IPA_SDTeam@skiermontderby.com Registration No: 61,501 |

January 26, 2018

Respectfully Submitted,

/Steven W. Hartsell/

Lead Counsel for Patent Owner

Reg. No. 58,788

SKIERMONT DERBY LLP

1601 Elm Street, Suite 4400

Dallas, Texas 75201

P: 214-978-6600/F: 214-978-6601

CERTIFICATE OF SERVICE

The undersigned hereby certifies that a copy of the foregoing Patent Owner's Mandatory Notices were served on January 26, 2018, by delivering a copy via electronic mail to the attorneys of record for the Petitioners as follows:

Naveen Modi
Daniel Zeilberger
Arvind Jairam
PAUL HASTINGS LLP
PH-Google-IPA-IPR@paulhastings.com

Dated: January 26, 2018

Respectfully submitted,

/Steven W. Hartsell/

Lead Counsel for Patent Owner

Reg. No. 58,788

SKIERMONT DERBY LLP

1601 Elm Street, Suite 4400

Dallas, Texas 75201

P: 214-978-6600/F: 214-978-6601

UNITED STATES PATENT AND TRADEMARK OFFICE

BEFORE THE PATENT TRIAL AND APPEAL BOARD

GOOGLE LLC

Petitioner

v.

IPA TECHNOLOGIES INC.

Patent Owner

Case No. IPR2018-00476

U.S. Patent No. 6,757,718

FILED: JUNE 30, 2000

ISSUED: JUNE 10, 2004

INVENTORS: CHRISTINE HALVERSON ET AL.

TITLE: MOBILE NAVIGATION OF NETWORK-BASED ELECTRONIC
INFORMATION USING SPOKEN INPUT

POWER OF ATTORNEY

Pursuant to 37 C.F.R. § 42.10(b), the Patent Owner of U.S. Patent No. 6,757,718, IPA Technologies Inc., hereby appoints the counsel identified below as its attorneys to transact all business in the United States Patent & Trademark Office associated with this *Inter Partes* review of U.S. Patent No. 6,757,718:

| LEAD COUNSEL | BACK-UP COUNSEL |
|--|--|
| Steven W. Hartsell SKIERMONT DERBY LLP 1601 Elm Street, Suite 4400 Dallas, Texas 75201 Tel: (214) 978-6600 Fax: (214) 978-6601 IPA_SDTeam@skiermontderby.com Reg. No. 58,788 | Sarah E. Spires SKIERMONT DERBY LLP 1601 Elm Street, Suite 4400 Dallas, Texas 75201 Tel: (214) 978-6600 Fax: (214) 978-6601 IPA_SDTeam@skiermontderby.com) Reg. No. 61,501 |
| | Alexander E. Gasser SKIERMONT DERBY LLP 1601 Elm Street, Suite 4400 Dallas, Texas 75201 Tel: (214) 978-6600 Fax: (214) 978-6601 IPA_SDTeam@skiermontderby.com Reg. No. 48,760 |
| | Paul J. Skiermont SKIERMONT DERBY LLP 1601 Elm Street, Suite 4400 Dallas, Texas 75201 Tel: (214) 978-6600 Fax: (214) 978-6601 IPA_SDTeam@skiermontderby.com (<i>pro hac vice</i> application to be submitted) |
| | Sadaf R. Abdullah SKIERMONT DERBY LLP 1601 Elm Street, Suite 4400 Dallas, Texas 75201 Tel: (214) 978-6600 Fax: (214) 978-6601 IPA_SDTeam@skiermontderby.com |

| | |
|--|---|
| | (<i>pro hac vice</i> application to be submitted) |
| | Mieke K. Malmberg SKIERMONT DERBY LLP 800 Wilshire Boulevard, Suite 1450 Los Angeles, CA 90017 Tel: (213) 788-4500 Fax: (213) 788-4545 IPA_SDTeam@skiermontderby.com (<i>pro hac vice</i> application to be submitted) |

The individual signing below has the authority to execute this document
on behalf of Patent Owner, IPA Technologies Inc.

SIGNATUR : Michael Zhang

NAME: Michael Zhang

TITLE: Director, Business Development

DATE: January 26, 2018

CERTIFICATE OF SERVICE

The undersigned hereby certifies that a copy of the foregoing Patent Owner's Power of Attorney was served on January 26, 2018, by delivering a copy via electronic mail to the attorneys of record for the Petitioners as follows:

Naveen Modi
Daniel Zeilberger
Arvind Jairam
PAUL HASTINGS LLP
PH-Google-IPA-IPR@paulhastings.com

Dated: January 26, 2018

Respectfully submitted,

/Steven W. Hartsell/

Counsel for Patent Owner

SKIERMONT DERBY LLP

1601 Elm Street, Suite 4400

Dallas, Texas 75201

P: 214-978-6600/F: 214-978-6601

UNITED STATES PATENT AND TRADEMARK OFFICE

BEFORE THE PATENT TRIAL AND APPEAL BOARD

GOOGLE LLC,
Petitioner,

v.

IPA TECHNOLOGIES INC.,
Patent Owner.

Case IPR2018-00476
Patent 6,757,718

Mailed: February 6, 2018

Before Tamara Henderson, *Trial Paralegal*.

NOTICE OF FILING DATE ACCORDED TO PETITION
AND
TIME FOR FILING PATENT OWNER PRELIMINARY RESPONSE

The petition for *inter partes* review filed in the above proceeding has been accorded the filing date of January 12, 2018.

Patent Owner may file a preliminary response to the petition no later than three months from the date of this notice. The preliminary response is

limited to setting forth the reasons why the requested review should not be instituted. Patent Owner may also file an election to waive the preliminary response to expedite the proceeding. For more information, please consult the Office Patent Trial Practice Guide, 77 Fed. Reg. 48756 (Aug. 14, 2012), which is available on the Board Web site at <http://www.uspto.gov/PTAB>.

Patent Owner is advised of the requirement to submit mandatory notice information under 37 C.F.R. § 42.8(a)(2) within 21 days of service of the petition.

The parties are encouraged to use the heading on the first page of this Notice for all future filings in the proceeding.

The parties are advised that under 37 C.F.R. § 42.10(c), recognition of counsel *pro hac vice* requires a showing of good cause. The parties are authorized to file motions for *pro hac vice* admission under 37 C.F.R. § 42.10(c). Such motions shall be filed in accordance with the “Order -- Authorizing Motion *for Pro Hac Vice* Admission” in Case IPR2013-00639, Paper 7, a copy of which is available on the Board Web site under “Representative Orders, Decisions, and Notices.”

The parties are reminded that unless otherwise permitted by 37 C.F.R. § 42.6(b)(2), all filings in this proceeding must be made electronically in Patent Trial and Appeal Board End to End (PTAB E2E), accessible from the Board Web site at <http://www.uspto.gov/PTAB>. To file documents, users must first register with PTAB E2E. Information regarding how to register with and use PTAB E2E is available at the Board Web site.

If there are any questions pertaining to this notice, please contact Tamara Henderson at 571-272-6439 or the Patent Trial and Appeal Board at 571-272-7822.

Case IPR2018-00476

Patent No. 6,757,718

PETITIONER:

Naveen Modi

Daniel Zeilberger

Arvind Jairam

PAUL HASTINGS LLP

PH-Google-IPA-IPR@paulhastings.com

PATENT OWNER:

Steven W. Hartsell

Alexander E. Gasser

Sarah E. Spires

SKIERMONT DERBY LLP

IPA_SDTeam@skiermontderby.com

**NOTICE CONCERNING ALTERNATIVE DISPUTE RESOLUTION
(ADR)**

The Patent Trial and Appeal Board (PTAB) strongly encourages parties who are considering settlement to consider alternative dispute resolution as a means of settling the issues that may be raised in an AIA trial proceeding. Many AIA trials are settled prior to a Final Written Decision. Those considering settlement may wish to consider alternative dispute resolution techniques early in a proceeding to produce a quicker, mutually agreeable resolution of a dispute or to at least narrow the scope of matters in dispute. Alternative dispute resolution has the potential to save parties time and money.

Many non-profit organizations, both inside and outside the intellectual property field, offer alternative dispute resolution services. Listed below are the names and addresses of several such organizations. The listings are provided for the convenience of parties involved in cases before the PTAB; the PTAB does not sponsor or endorse any particular organization's alternative dispute resolution services. In addition, consideration may be given to utilizing independent alternative dispute resolution firms. Such firms may be located through a standard keyword Internet search.

| CPR INSTITUTE FOR DISPUTE RESOLUTION | AMERICAN INTELLECTUAL PROPERTY LAW ASSOCIATION (AIPLA) | AMERICAN ARBITRATION ASSOCIATION (AAA) | WORLD INTELLECTUAL PROPERTY ORGANIZATION (WIPO) | AMERICAN BAR ASSOCIATION (ABA) |
|---|---|---|--|--|
| Telephone: (212) 949-6490 | Telephone: (703) 415-0780 | Telephone: (212) 484-3266 | Telephone: 41 22 338 9111 | Telephone : (202) 662-1000 |
| Fax: (212) 949-8859 | Fax: (703) 415-0786 | Fax: (212) 307-4387 | Fax: 41 22 733 5428 | N/A |
| 575 Lexington Ave | 241 18th Street, South, Suite 700 | 140 West 51st Street | 34, chemin des Colombettes | 1050 Connecticut Ave, NW |
| New York, NY 10022 | Arlington, VA 22202 | New York, NY 10020 | CH-1211 Geneva 20, Switzerland | Washington, D.C. 20036 |
| www.cpradr.org | www.aipla.org | www.adr.org | www.wipo.int | www.americanbar.org |

If parties to an AIA trial proceeding consider using alternative dispute resolution, the PTAB would like to know whether the parties ultimately decided to engage in alternative dispute resolution and the reasons why or why not. If the parties actually engage in alternative dispute resolution, the PTAB

would be interested to learn what mechanism (e.g., arbitration, mediation, etc.) was used and the general result. Such a statement from the parties is not required but would be helpful to the PTAB in assessing the value of alternative dispute resolution to parties involved in AIA trial proceedings. To report an experience with ADR, please forward a summary of the particulars to the following email address: PTAB_ADR_Comments@uspto.gov

File History Content Report

The following content is missing from the original file history record obtained from the United States Patent and Trademark Office. No additional information is available.

Document Date - 2018-02-20

Document Title - USPTO Communication Re: Change of Address

EIC2100

Search Results

Feedback Form (Optional)



Summary of the information you

The search results are presented in your report. You can click on any of the search results or comments to view the full text of the document. If you have any comments or suggestions, please contact Microsoft@Microsoft.com or call 1-800-485-2048.

Online Feedback Form (Optional) 303-881-1130
or call 1-800-485-2048

Optional Results Feedback Form

Do you agree with the results of the search?

- ☐ Yes, I agree with the results of the search.
- ☐ No, I disagree with the results of the search.
- ☐ I am not sure.
- ☐ I did not find what I was looking for.
- ☐ I found what I was looking for, but I am not sure if it is the best result.
- ☐ I found what I was looking for, but I am not sure if it is the most relevant result.
- ☐ I found what I was looking for, but I am not sure if it is the most useful result.

Why did you choose this answer?

☐ I did not find what I was looking for.

☐ I found what I was looking for, but I am not sure if it is the best result.

Other comments (optional) _____

Thank you for your feedback.

☐ I am satisfied with the results of the search.

☐ I am not satisfied with the results of the search.

Other Comments _____

Microsoft Corporation, 1000 Microsoft Way, Redmond, WA 98073

COD SHEET FOR CONTINUING DATA

| Line | Code | Serial No. | Filing Date | Status | Document No. | Issue Date |
|------|------|------------|-------------|--------|--------------|------------|
| 104 | 71 | 09/524,095 | 3/13/90 | | | |
| 105 | 82 | 09/225,198 | 1/5/99 | | | |
| 106 | 68 | 60/124,720 | 3/17/99 | | | |
| 107 | 68 | 60/124,719 | 3/17/99 | | | |
| 108 | 68 | 60/124,718 | 3/17/99 | | | |
| 109 | | | | | | |
| 110 | | | | | | |
| 111 | | | | | | |
| 112 | | | | | | |
| 113 | | | | | | |
| 114 | | | | | | |
| 115 | | | | | | |
| 116 | | | | | | |
| 117 | | | | | | |

Condition and Status Codes for Continuing Data

CONDITION CODE:

- 71 Continuation of application No.
- 81 which is a continuation of application No.
- 91 and a continuation of application No.

- 72 Continuation-in-part of application No.
- 82 which is a continuation-in-part of application No.
- 75 and a continuation-in-part of application No.

- 74 Division of application No.
- 84 which is a division of application No.
- 76 and a division of application No.

- 86 , said application No.
- 89 Application No.
- 90 and application No.
- 92 each

- 65 filed as application No.
- 66 substitute for application No.
- 68 Provisional application No.

STATUS CODE

- 01 Patent No.
- 03 abandoned
- 04 SIR No.

NOTE I: When the code 86 and 92 are used, they must be followed by 81, 82 or 84 - condition beginning with "which is"

NOTE II: Codes 71, 72 and 74 may be used only on the first line; one of them must be used on the first line in regular continuing data. 66 or 68 may be used on the first line in Substitute or Provisional cases. Remember, however, that if there is a Provisional and other continuing data, the Provisional is always listed last.

PATENT APPLICATION FEE DETERMINATION RECORD

Effective December 29, 1999

Application or Docket Number

CLAIMS AS FILED - PART I

| FOR | (Column 1) NUMBER FILED | (Column 2) NUMBER EXTRA |
|----------------------------------|----------------------------|----------------------------|
| BASIC FEE | | |
| TOTAL CLAIMS | 27 minus 20 = | 7 |
| INDEPENDENT CLAIMS | 3 minus 3 = | 0 |
| MULTIPLE DEPENDENT CLAIM PRESENT | | |

* If the difference in column 1 is less than zero, enter "0" in column 2

CLAIMS AS AMENDED - PART II

| | (Column 1) CLAIMS REMAINING AFTER AMENDMENT | | (Column 2) HIGHEST NUMBER PREVIOUSLY PAID FOR | (Column 3) PRESENT EXTRA |
|--|---|-------|---|--------------------------------|
| AMENDMENT A | 27 | Minus | 27 | = |
| Total | 27 | Minus | 27 | = |
| Independent | 3 | Minus | 3 | = |
| FIRST PRESENTATION OF MULTIPLE DEPENDENT CLAIM | | | | |

| | (Column 1) CLAIMS REMAINING AFTER AMENDMENT | | (Column 2) HIGHEST NUMBER PREVIOUSLY PAID FOR | (Column 3) PRESENT EXTRA |
|--|---|-------|---|--------------------------------|
| AMENDMENT B | 27 | Minus | 27 | = |
| Total | 27 | Minus | 27 | = |
| Independent | 3 | Minus | 3 | = |
| FIRST PRESENTATION OF MULTIPLE DEPENDENT CLAIM | | | | |

| | (Column 1) CLAIMS REMAINING AFTER AMENDMENT | | (Column 2) HIGHEST NUMBER PREVIOUSLY PAID FOR | (Column 3) PRESENT EXTRA |
|--|---|-------|---|--------------------------------|
| AMENDMENT C | | Minus | | = |
| Total | | Minus | | = |
| Independent | | Minus | | = |
| FIRST PRESENTATION OF MULTIPLE DEPENDENT CLAIM | | | | |

* If the entry in column 1 is less than the entry in column 2, write "0" in column 3.

** If the "Highest Number Previously Paid For" IN THIS SPACE is less than 20, enter "20."

*** If the "Highest Number Previously Paid For" IN THIS SPACE is less than 3, enter "3."

The "Highest Number Previously Paid For" (Total or Independent) is the highest number found in the appropriate box in column 2.

SMALL ENTITY
TYPE ☐

OR
OTHER THAN
SMALL ENTITY

| RATE | FEE |
|--------|--------|
| | 345.00 |
| X\$ 9= | 63 |
| X39= | / |
| +130= | / |
| TOTAL | 408 |

| RATE | FEE |
|--------|--------|
| | 690.00 |
| X\$18= | |
| X78= | |
| +260= | |
| TOTAL | |

SMALL ENTITY

OR
OTHER THAN
SMALL ENTITY

| RATE | ADDI- TIONAL FEE |
|---------------------|------------------------|
| X\$ 9= | |
| X39= | |
| +130= | |
| TOTAL ADDIT. FEE | |

| RATE | ADDI- TIONAL FEE |
|---------------------|------------------------|
| X\$18= | |
| X78= | |
| +260= | |
| TOTAL ADDIT. FEE | |

| RATE | ADDI- TIONAL FEE |
|---------------------|------------------------|
| X\$ 9= | |
| X39= | |
| +130= | |
| TOTAL ADDIT. FEE | |

| RATE | ADDI- TIONAL FEE |
|---------------------|------------------------|
| X\$18= | |
| X78= | |
| +260= | |
| TOTAL ADDIT. FEE | |

| RATE | ADDI- TIONAL FEE |
|---------------------|------------------------|
| X\$ 9= | |
| X39= | |
| +130= | |
| TOTAL ADDIT. FEE | |

| RATE | ADDI- TIONAL FEE |
|---------------------|------------------------|
| X\$18= | |
| X78= | |
| +260= | |
| TOTAL ADDIT. FEE | |