Com:
compatible computers and communicators

Eric Hehner

Most of this essay is not new. Most of what I describe already exists. I just want to record the design choices I would make if I were designing computing and communicating devices.

Early cars (1895 to 1930) had a great variety of controls, such as levers and joysticks, until they settled on some standards (steering wheel, gas pedal on the right and brake pedal on the left) so that anyone can get into any car and know how to drive it. Computers and communications have necessarily gone through an experimental stage. It isn't always clear what ways of doing things are best until you try them for a while. Then you have to settle on some standards to make them usable without a lot of relearning. This essay is my standards preference.

I first saw and used windows (the kind on computers) in 1980 when I worked at Xerox PARC (Palo Alto Research Center), several years before Microsoft learned about it. We even discussed whether "windows" was the right word, since they weren't transparent. My first reaction was horror: I saw that they allowed a computer user to reproduce a messy, disorganized desk. Some people apparently like having a messy desk, with piles of papers haphazardly everywhere, on top of each other; windows should suit them well. But I am a neatnik. In this essay I present what I would like.

One Device, Many Sizes

I envision a series of devices, all pretty much the same except for size. Each one is called a com; that's for computer plus communicator. They all deserve the same name because they all work the same way, so com is also for compatible. There's the size that fits in your hand. We have been calling it a smartphone, but that's a poor name because phoning is just one of its capabilities, and not the most used capability. There's a slightly larger size that we've been calling a tablet. Both sizes have a touch screen, a microphone, a speaker, an orientation sensor, a camera, wifi, bluetooth, and cell communications. They are exactly the same device except for size. But at present, they don't work exactly the same way; even if you know how to use one, you may not know how to use the other. Different brands of smartphone and different brands of tablet don't work the same way. I would like them to work exactly the same way so I'll know how to work all of them just by learning to use one of them.

Moving up, there's a size that sits on your lap. We've been calling it a laptop computer. It has a keyboard and touch pad, so its screen may not be a touch screen. It may lack an orientation sensor. But otherwise it is the same device, and I would like it to work exactly the same way as the smaller devices. Moving up again, there's the size we've been calling a desk top computer. Then there are the sizes we call smart TVs. The smart TV size could have a touch screen, or it could have a keyboard and touch pad connected by bluetooth. It should be exactly the same device, and work in exactly the same way, as the smaller sizes.

There can be many sizes, distinguished by a measurement, for example, a 50cm com. Or the sizes can be given names, for example, a mini com. All sizes are the same device, except that larger devices that are not handheld do not have an orientation sensor, and some devices have a touch screen while others have a keyboard and touch pad.

In addition to the loudness and brightness controls, there are three special buttons or keys on a com. They are: the command key, the files key, and the landscape key.

Commands

Screen space is a scarce resource, and it is a shame to see it wasted on a clutter of menu items. On a com, there is no menu sitting permanently on the screen taking up space. In general, each program uses the entire screen. There is a command key, perhaps placed on both sides of the space key. Pressing the command key causes a menu of commands to pop up. When you select a command, either the command causes some action, or a submenu of commands pops up. Submenu commands can cause further submenus. Pressing the command key again returns to the screen as it was before you pressed the command key the previous time. Some commands have a letter or other character beside them; if so, the combination of command key and letter/character is a shortcut that causes the action without navigating the menus. You start the execution of a program by choosing its command, and you choose actions within a program by choosing commands on its submenu. New commands can be added to a (sub)menu, either by compiling a program or by receiving (importing) a program; commands can be moved (resorted); and commands can be deleted from a menu.

I made this suggestion at PARC in 1980 when screen space was smaller than it is today. But back then, personal computing had not yet become public, and there was a lot of concern for how to make computers seem friendly, not scary, to the public. It was decided that having the menu visible at all times would be reassuring, so my suggestion was not adopted. And that decision persists today by inertia.

Files

Files are the way you organize your documents. When the files key is pressed, the screen displays a list. There may be more items in the list than fit on the screen, in which case you scroll down to see the others. You can reorder the items. Each document and file has a name, a creation date and time, a last modification date and time, and a size. Files can contain documents and files. Selecting a file displays its documents and files. You can create new files and delete old files. The file structure is your one and only information organization; there isn't a separate mail file structure, music file structure, photo file structure, and so on. How you organize your files is up to you. Pressing the files key again returns to the screen as it was before you pressed the files key the previous time.

Landscape

Movable, resizable, overlapping windows make a messy and confusing workspace. Windows are too small and they get buried. There are no windows on a com. A document has the entire screen. When a document is too large to fit on a screen, as it usually is, scrolling reveals the rest of the document.

Imagine a landscape of rectangular fields. The landscape is how a com organizes currently active documents, each of which has its own field. At any moment, the screen is coincident with one of the fields; the other fields cannot currently be seen. There is a field to the left of the screen, a field above the screen, a field to the right of the screen, a field below the screen, a field diagonally above-right, and in all directions, and more fields beyond them. Different programs may be running in different fields, and different documents may be in different fields. You can move your screen to any field, or to say the same thing differently, you can move the landscape so that any field is showing on the screen.

When you press the landscape key, you see the landscape in miniature. Each field is just large enough to be recognizable. If that size does not allow the whole landscape to fit on the screen, the landscape can be scrolled in any direction. The fields can be moved around to change the order. New fields can be added and old ones deleted. A field can be selected to become the screen display.

In general, each field has one document. But sometimes there is a reason to subdivide a field into rectangular subfields, with a different document in each subfield, and each subfield is separately scrollable. The screen displays one field, which may be more than one subfield. Then the command key shows the menu of commands appropriate to the subfield where the cursor is located.

Gestures

On a touch screen, you can select something by touching it, and you can move it by dragging your finger from its location to where you want it. On a touch pad, moving a finger moves a small arrow, called the cursor, on the screen, and pressing the touch pad selects whatever is at the cursor. You can move something by moving the cursor to where it is, then pressing and holding the touch pad while dragging the cursor to where you want it, then releasing the touch pad. A single finger gesture can also be used for selecting a region of the screen or drawing a line.

On a touch screen or touch pad, two fingers are used for scrolling. You can scroll in any direction; the screen image moves in the same direction as your two fingers. Rotating two fingers around a central point rotates the screen image. Spreading two fingers enlarges the screen image (zooms in), and moving two fingers toward each other shrinks the screen image (zooms out).

On a touch screen or touch pad, three fingers are used for changing fields in the landscape. Moving three fingers to the left pulls the entire landscape to the left, so that the screen shows the field that was previously to the right of the screen. And similarly for any other direction.

Programs

I don't remember when the word “app” was invented. It is short for “application”, but what is being applied to what? Before the word “app” we used the word “program”; I see no need to replace “program” with “app”, and I shall continue to use the word “program”.

A program text is a document; you can edit it. You can also compile it. Compiling a program creates a new command. Selecting the command starts execution of the program. All programs should run on all coms (except that programs requiring an orientation sensor cannot run on large coms that don't have an orientation sensor).

One of the programs that runs on all coms is the programming system used to program all the programs that run on coms (including itself): ProTem (I named it that in 1987 when I first conceived it). The correspondence between ProTem and com concepts is not accidental: ProTem's dictionaries are a com's files; ProTem's programs are a com's programs; ProTem's data are a com's documents.

Documents

There aren't different kinds of documents, such as an email document, a text document, a video, and so on. There is just one kind of document for all purposes, and it can have in it any combination of text, diagrams, photos, videos, sounds, links, buttons, and perhaps other things. A website is just a document.

A document is either fluid or rigid. A fluid document can be edited and copied. A rigid document can neither be edited nor copied. A newly created document is fluid. When you copy a fluid document, you choose whether the copy is fluid or rigid. If you choose rigid, you must also choose a "uses" number, which is the number of times the document can be used (if it is music, that means the number of times it can be played; if it is a movie, that means the number of times it can be watched; if it is a text document, that means the number of times it can be read). The uses number can be infinity. If it is a finite number, each use decreases the uses number by one. When the uses number reaches zero, the document becomes useless, or perhaps it destroys itself.

When a fluid document is communicated to another com, the sender still has the original fluid document, and the receiver has a fluid copy. When a rigid document is communicated to another com, the sender no longer has it; only the receiver has it, just like a physical object. (With current technology, I don't think it is possible to make documents rigid, but I am describing my ideal.)

To see the desirability of the rigid attribute, consider the following story. A group of musicians create a musical document, editing it, sending it to each other, until they are satisfied with it, and offer it for sale. When someone buys it, the musicians make a rigid copy of it, setting the uses number according to what the buyer is paying, and send this rigid copy to the buyer. The musicians made the copy rigid for two reasons. The document is the music the way they want it heard, so they don't want the buyer to edit it. The musicians also don't want the buyer to make copies to give or sell, or to get more uses than they have paid for. The buyer can send it to a friend with the remaining number of uses, but then the buyer doesn't still have it.

With current computers, when you send something by email, you are sending a copy; you still have what you sent. The proposal here is that when you send a rigid document, you don't still have what you sent. To keep what you are sending, the document has to be fluid.

You don't "save" a document; saving is automatic. If you don't like the changes you have made, you can "undo" them. There are various levels of "undo", including "undo session"; see ProTem.

Communications

You may have several coms, perhaps of different sizes, for your personal use. These coms can be linked so that any actions performed on one com are automatically reflected on the others. If an action is performed on one com while a linked com is turned off, the result of the action will be reflected on the linked com when it is turned on. Obviously this requires the linked coms to communicate with each other, but since the user does not have to do anything to cause the communication (after the initial linking), we don't call it "communication"; we call it "linked action". We reserve the word "communicate" for coms that are not linked.

Right now, if I want to send you a message, I can write it on paper and send it by the post office, or I can phone you, or I can text you, or I can fax you, or I can email you, or I can send you a Facebook message, or I can send you a Twitter message, or initiate a Zoom meeting, or several other possibilities. This is not convenience; it's often a meaningless choice for the sender, and trouble for a receiver who has to continually check all possibilities. In my ideal world, there is a single space for communications, with a single addressing scheme. Our present internet addressing scheme name@domain , in which the name cannot include spaces and certain other characters, and the domain might include subdomains, was designed for easy implementation. I would prefer a free choice of name, allowing characters from any alphabet, and spaces. There is no need for domains and subdomains; there can be a global registration mechanism. So that these free form addresses

will be well defined and can be recognized, they are enclosed in brackets 《 》 that are reserved for this one purpose only. An address like this replaces the current internet addresses, telephone numbers, twitter names, twitter handles, facebook names, and so on.

There are two options for communication; live communication, and recorded communication. These two options might alternatively be called synchronous communication and asynchronous communication. For short, I will call them a "call" and a "message".

Recorded communication, a message, means sending a document or program or file to one or more recipients. A document may be just sound, or just video, or just text, or just photos, or any combination of these and any other things a document may contain. The recipient(s) do(es) not need to respond at the time the message is sent; the message is there for the recipient(s) to receive if and when they want it; that's the meaning of "asynchronous". This is what we currently call emailing or texting.

Live communication, a call, consists of sound and/or picture. It is what we currently call a "phone call" or a "video call". There may be one or more callees. Each callee must decide, at the time of the call, whether or not to "answer the call"; that's the meaning of "synchronous". The caller and each callee who answers can decide individually whether to include their own sound (mute off or on) and their own picture (video on or off), and these decisions can be changed at any time. Perhaps the caller also has the ability to turn off sound and video of the callees.

If you attempt a live communication with someone (you call them) and they do not answer, you are not then offered the opportunity to leave a recorded message. That's because you can always send them a recorded message. You can send them a recorded message without attempting a live communication, or during a live communication, or after a live communication. Sending a recorded message is independent of attempting a live communication.

A communication attempt (call or message) costs the sender 1¢ for each recipient (whether or not the recipient chooses to receive). I intend this price to be cheap enough that it doesn't discourage normal communications, but expensive enough to make phishing and scamming unprofitable. An additional amount based on the amount of information sent/received should be charged, though I don't know what proportion should be charged to the sender and what to the receiver, and together these charges should pay for the communication infrastructure. (These charges are independent of any charges negotiated between sender and receiver.)

Conclusion

This essay is far from complete. There are uses and issues that I have not discussed. One use is money transfer: com-to-com communication provides all that is necessary for all money transactions (see Money and Taxes). One issue is security: a com should work only for a set of authorized users. The purpose is to protect privacy, and prevent theft of a com by making it useless to unauthorized people. To do this it must detect the identity of a user by fingerprint or face recognition or some other biometric input. I have tried to describe a framework and some principles, but there is much more to say.

What I like best may not be what other people like best. What people say they like may not be what works best. That's why washing machines have a dazzling array of selling features: in the store they look great, but at home you find that most of them aren't useful. All I can do is say what I like and think is best.