T. Landauer, The Trouble with Computers: usefulness, usability, and productivity, The MIT Press, Cambridge, 1995, Chapter 6, pp. 141-168.

6

Usefulness and Usability

Usefulness

Upscale's department store installed a checkout machine intended for better inventory control and customer service. The screen showed a menu of items for sale. It started with a list of general categories: housewares, men's clothing, and so forth, each represented by a colorful picture, an icon. Choose an icon by touching it with your finger, and up comes a more detailed list—sport clothes, suits, shoes . . . each represented by another colorful icon. And on down to the individual item type. No longer did clerks need to know or enter the item type or number—so long as their interpretation of the icons and categories matched that of the designer. I once tried to buy some candles. Three clerks struggled for fifteen minutes to enter the sale. Candles were under vases under platters under mixers under chairs.[1]

It's easy for a computer to offer operations that don't help people. For example, computer schemes for finding books in libraries and for presenting them electronically make it harder to find the books and harder to read them. Later we'll see how dumb applications have been turned around by user-centered design. For now, let's look at some of the underlying problems.

Phase one computer applications perform the same task that a human might otherwise do. Programming for these applications demands near-perfect understanding of what the task is and how it can be accomplished. In tasks like bookkeeping and gun aiming there are well-defined

formulas by which the task can be performed, principles that humans would follow precisely if they could. These can be translated directly into computer programs. It has been said that the first industrial revolution replaced human and animal muscle with more efficient energy sources and that the information revolution is replacing human mental work with more efficient electronic processes. That's the way phase one applications works. Phase one computing has had two effects. First, it has pushed forward the replacement of human motor skills with more accurately controlled mechanical ones. Second, it has largely replaced humans in simple acts of arithmetic and filing. Arithmetic is an easily described skill and one at which no mathematical John Henry would stand a chance.

The second phase of computer application, helping people think, is surprisingly more difficult. Note how shallow are the components of writing assumed by word processors, how superficial the aspects of business decision making taken on by spreadsheets. The mental processes of composing memos and documents, of making medical and business decisions, of negotiating and persuading, of formulating plans, and communicating ideas will not soon be captured and imprisoned in a machine. No one really knows how humans do these mysterious things.[2] Humans have amazing memories from which details can be recalled in response to any hint about any part of the original experience. (What time did your childhood neighbors eat dinner on Christmas?) People have astounding visual and auditory pattern-recognizing ability, can easily identify one among a hundred thousand faces or words. In reading text, a literate adult can guess a missing from context half the . And these are paltry feats compared to what goes on in the mind during the comprehension, appreciation, or creation of a serious (or funny) verbal passage. No matter what you may have read in an airline magazine, we can't yet come close to these human abilities in any programmable process.

In services, the role of the computer is to help people help people. Thinking how to do this has proved more difficult than technologists expected. Upscale's worthless point-of-sales device is worse than most others perhaps, but so far not enough are good enough. Reservation and information retrieval systems help people do some important business

chores more easily. So do bar code inventory systems, income tax preparation programs, email, and several dozen more. But most department store service has gotten worse since computers; so has the handling of telephone inquiries and dozens more. Figuring out really good things to do with computers in service businesses is difficult.

Usability

I tried to change a character in the bibliography of this book to put in a French diacritical mark. After an hour of reading three manuals (for the text editor, the bibliography program, the bibliography program's enhancement package) and many experiments, I gave up. I hope the Acadèmie Français will forgive me.

A phase two system is unlikely to be useful unless it is easy to operate, although the opposite is possible: a system that is easy to operate but of no value. The division between usefulness and usability is sometimes fuzzy; one can't tell whether a system flunks for one reason or the other. Was the Upscale department store's menu a failure because it was hard to use or because it wasn't useful? Or was it useless because it was hard to use?

Nevertheless, usability is the key issue in thousands of cases. In a system in which you sometimes enter *K* to "keep" a program and in other cases enter *K* to "kill" the same program, there is a usability problem no matter how useful the program might otherwise be. Because different programs are written by different programmers at different times and marketed by different companies, such inconsistencies are rife. Even when companies like Apple, and more recently IBM and others, attempt to set up interface consistency standards, they don't always have the desired effect. Sometimes it's impossible to maintain consistency as the functions of a system multiply. Bruce Tognazinni, Apple's erstwhile human interface evangelist, has written, "The Macintosh promises consistency, and it is a promise that is dreadfully hard to keep" (Tognazzini 1992). The Macintosh user interface guidelines say the menu bar (a strip along the top of the screen from which available actions can be chosen by a mouse click) must always be visible. But here comes a program

intended to produce a slide show on the screen. Should we leave the menu bar visible on each slide? Of course not.

Standards often force designers to use suboptimal designs; the best way to lay out the screen or structure the user system dialog is not the same for different jobs. For consistency, interface standards say that every computer action should be invoked by the same series of user actions: for example, first click a choice from the main menu bar, then pick one of the subsequently displayed submenu options, and then, perhaps, confirm or type in needed information—like a line thickness or a search term. But use this interface to control a steel-rolling plant and you'd have consistently crumpled metal; all those steps would take much too long. As Emerson might have observed, a foolish consistency is the hobgoblin of little computers.

Another problem is the incredibly rapid change in computer technology. Last year's hot program is this year's dog. Everybody is in a constant state of computer illiteracy. The situation cries out for simplicity, but the cry goes unheard. Instead we get ever greater complexity.

Such problems are not the sole property of PCs. Consider the control of advanced telephone services by a Touch-Tone pad. Transferring a call to another telephone or adding on a third participant can be useful, but remembering to do it by punching 23# is taxing. Thus the familiar, "if this doesn't work, call me back."

The list that follows describes some major phase two applications and contrasts their amazing feats and flaws. If you're not a computer maven, this survey should give you an idea of what a wonderful whiz the computer that's in trouble is.

Mathematics, Science, and Engineering Tools

Computers found their first major applications in doing calculations for scientists and engineers. These have continued to be major phase two applications, taking over the drudgery and increasing the accuracy of computations needed to derive predictions from physical theories and engineering models. Computers can also act as accurate and patient logicians, following a series of deductions through endless boring steps. These capabilities have been used to prove mathematical theorems that were long intractable, for example, the famous four-color conjecture that any two-dimensional map can have its regions distinctly colored by just

four colors without any two regions of the same color ever adjoining each other. To prove this one, a ghastly number of cases had to be enumerated and shown to conform. A computer programmed by a mathematician did it.

Engineers can pretend that a bizarrely shaped building is composed of millions of tiny polygons, each vigorously shoving its neighbors, and predict whether it will stand or fall. Power turbines and space shuttle shapes are tested before they're really tested. Simulation isn't cheap; it uses most of the multimillion dollar supercomputers in the world. Simulation isn't enough—NASA has just built a new wind tunnel—but it is a big help.

Data Storage and Retrieval

Businesses need to keep track of bills, accounts, inventories, facilities, personnel assignments, and facts about products. Databases can store information in extraordinarily compact forms and get it back in extremely short times. However, users have to pose questions in extremely cumbersome ways. The query languages in popular use require people to enter formal, logical expressions, the exact words in prescribed order, every bit of syntax and punctuation perfect.

The usability of query languages is so poor that the managers, executives, or salespeople who need the data are rarely able to ask the computer for it themselves. Instead, they ask a lookup specialist, a "systems analyst"—a new employment category needed since computers. I know an applications designer who has invented a wonderful artificial intelligence (AI) program that helps callers find the right employee to talk to at her company. It goes unused because the human effort to extract updates from the corporate database exceeds the resources of the electronic data processing department.

Interestingly, the standard query language, called SQL, was originally introduced to make searching easier, in expectation that end users would ask the computer questions by themselves. Previous schemes had been too difficult. But SQL also failed the usability test and did not reduce the need for technicians.

The latest episode is the development of "natural language" front ends. These AI programs accept questions in near-English (or near-French, etc.) and convert them into SQL. Of course, the necessary words must be added for each database. For example, you might be able to ask

about "trousers" or "slacks" but not "pants." And you have to avoid the ambiguities of normal English: "Tell me the salaries of managers who are black and women" may not give you what you want. And you must know what is in the database and what is not. The result is that while such systems have sold well, they need just as many technicians as in the past.

Text Storage and Retrieval

Computer storage and retrieval of text is a substantial industry. Bibliographic records—author, title, category—abstracts of journal articles in the sciences, summaries of legal cases, stock analysis reports, and much more, have been put online. Several large companies provide remote access to huge text databases; instead of libraries with stacks, they operate "disk farms." These services, widely used and handsomely paid for, are useful because they provide rapid physical access to information that people cannot afford to collect or keep on hand. However, the user's chance of finding all and only wanted items is slightly worse than with print on paper.

Like the quantitative databases, these require arcane logical queries. Here is an example:

ABS = (information AND retrieval AND NOT about [range = 3]) AND (text OR data) AND (NOT ((quantitative AND data) OR (numbers OR numerical OR statistics) OR information AND NOT text))

This query is intended to find abstracts about information retrieval, rejecting articles that only discuss databases for numbers. (There are two fatal errors in this query. It wouldn't parse—that is, the computer would reject it because the syntax prevents unambiguous interpretation—and if the syntax were corrected, it would return nothing because the logic is faulty. There are more problems as well. Try to find them.) Such queries make use of Boolean expressions, named after the English mathematician George Boole who invented them in 1847. The skilled user can find all articles that contain the words *common* and *cold* within eight words of each other and do not contain the word *ice* or *snow* and were either written before 1900 or in the interval between 1946 and 1953, and had as one of their authors a person whose middle initial is G. Fundamental theory proves that any text item can be distinguished from any other by statements of this kind and that machines can find the matching items with great efficiency.

Unfortunately the design of these languages was not influenced by the way ordinary people think or by the way people ordinarily use words (Furnas et al. 1987). Most designers in this field have been proudly and steadfastly ignorant of such soft stuff. However, these factors defeat success. Boolean-speak is so difficult, and in practice unreliable, that yet another new occupation, information specialist, has grown up. Not only an expense, this means that questions are translated by someone who always knows less than you do about what you want.

Word Processing and Text Editing

The earliest text editors were invented to help programmers correct programs. At first they just let you march through the program line by line, deleting bad lines and adding new ones. Later editors embodied crafty schemes such as "regular expressions" in which a search for /^[a-Z]..[o,ou]/ will find any word beginning with a capital letter at the beginning of a line in which the fourth letter is an *o* or the fourth and fifth letters are *ou* (in case you're looking for either American or English spelling of *colour*). The UNIX version of such a text editor in an early form, intended for use by secretaries, had a user manual containing the following:

> The global command **g** is used to execute one or more ed commands on all those lines in the buffer that match some specified string. For example
>
> **g/spellng/p**
>
> prints all lines that contain spellng.

Several dozen different commands, each with a nonobvious name (Furnas et al. 1987; Landauer, Galotti, and Hartwell 1983), were listed in a manual. There was no way to look up how to do something without knowing the command name. I was given the dubious honor of evaluating the system for use by legal secretaries. I predicted that they would have trouble; the programmers are still mad at me.

Most modern text editing systems are easier to operate than these early ones, but some are many times as complex and difficult and can be mastered only with a great deal of practice by people with high intelligence

and nimble hands. The former have become widespread tools of office workers and the latter important implements for advanced programmers.

Current text editors provide many techniques to correct text, to format it in a variety of type fonts and styles, to rearrange and modify it, to check its spelling and even, crudely, its punctuation, and suggest, again crudely, synonyms or alternative phrases. These features add great power, the characteristic cherished most by developers and gurus, but make them more complex for the ordinary user without necessarily improving the everyday product or speeding the process. The online thesaurus, for example, is tempting but not very good. The most important deficiency is that none of these facilities begins to touch the difficult parts of the composition of memos, letters, and documents, which is just that—their composition. The hard, time-consuming, labor-expensive part, because it is done by high-priced people, is the collection, ordering, and expression of ideas, arguments, descriptions, facts, narratives, and explanations. More help with these would be more useful.

Before we leave text editors, a comment on usability. Documents made with them should have fewer typos. It doesn't always work out that way. The very facilities of easy deleting, inserting, cutting, and pasting that make some kinds of errors easier to correct make others easier to commit. You can tell when a document has been produced with a text editor. It has telltale flaws: missing lines, extra words—as if the typist were stuttering, or like a speaker starting phrases over in the middle—disagreements in number, person, tense. They happen because the computer lets the user make partial changes without typing whole sentences or paragraphs over.

What's going on here here? Computerers are. are wonderful.. **Maybe,** if you're Like me flew years black, the very title of these

book, would did puzzled you, or andm mad. you made. Trouble? rouble indeed! There *must were errors in the meas*urements to productivity .Computerers are fantastic, awe awesome. You love they, I lovethem. They're sell like love the m, I livethem, They're selling like hotcakes:. Úmaybe a little slower that

they used to,, ...but still their the the bigger biggest bright,spit spot on with of by modern perseverance. Everyone has wants one. Everyone has wants one.Everyone uses wants they, one way or anther. The*y'e* hare, They're everywhere.. Everyone has wants one.

Every accountant, author, secretary, scientist, business person, engineer secretary, scientist, business person, engineer has to has havetoo two one. They're. making new millionaraires—no billionaraires—all over the plates ..

Trouble, what trouble?

Many professional writers are devoted to their word processors, claiming much greater productivity, and I am loathe to discount their testimonies. Conceivably, the 30 percent gains we chalked up earlier are especially important to them; perhaps they extract more value than does the average office worker or laboratory letter writer.

It is also possible, however, that they are unduly impressed with the benefits and too easily overlook the costs. One professional writer, Steven Levy, author of *Artificial Life: The Quest for a New Creation,* stated the case nicely. In a column called "The Iconoclast" in *Macworld* magazine, he first reviewed Loveman's discouraging analyses of productivity, reporting that as a fan of word processing he vigorously resisted the conclusion. But then he thinks again. He notes that despite the new technology, he and all the other writers he knows still work all the time. He recalls just having spent most of his morning installing and exploring a new minor release, 5.0 to 5.1, of his favorite editor.

"Back in the old days when I toiled on a typewriter, I never spent a whole morning installing a new ribbon. Nor did I subscribe to Remington World *and* IBM Selectric User. *I did not attend the Smith-Corona Expo two times a year. I did not scan the stores for the proper cables to affix to my typewriter, or purchase books that instructed me on how to get more use from my liquid White-Out" (Levy,* Macworld, *March 1993).*

Spreadsheets

Spreadsheets are the second most commercially successful application for PCs. With a spreadsheet program, you set up a table of data, for example, a set of budget categories and dates, with the amounts for each in the

cells. You can change an entry, and the machine will automatically recalculate the sums of each row and column. More sophisticated applications are legion. A typical one calculates the consequences of alternative investment instruments: yield, present value, long-term performance, best- and worst-case outcomes. Financial experiments like this are difficult to perform by hand but useful to clarify people's thinking.

Do spreadsheet programs make users almost perfectly accurate? No. I have seen data suggesting that in common use, over 40 percent of spreadsheets contain at least one error. The computer doesn't add or subtract wrong, but people can't figure out how to get it to do what they want, or they use its power to do foolish or careless things that the programs are helpless to prevent.

Shortly before the publication deadline for this book, I wanted to revise a figure, one I'd done a year ago. After an hour's searching I still could not find a version of the spreadsheet file that the current graph program could work with. I had to start from scratch. (Experts will be tempted to think me disorganized and stupid. That's my point.)

Spreadsheet programs can tempt users into endless puttering—changing this, trying that—of little value. Additional time is stolen by housekeeping: filing results, making disk copies, finding the disks and files later, organizing and cleaning up overloaded file systems, upgrading programs, learning and teaching how to do new operations, keeping the hardware working. Bookstores stock fat volumes about spreadsheet programs, and community colleges offer courses in their use. My first reaction to such books and courses is always that the systems for which they are intended are not well enough designed. The proper goal is that computer systems be so easy to use that people can master them on their own in a short time.

Graphics Programs

Computer chips cannot be made without the systems that let engineers build, see, and test imaginary circuits on their computer screens. The engineer moves and copies parts, or touches a mouse to the screen to

apply an imaginary voltage and make an imaginary meter reading. These systems are extremely valuable.

The popular variety of graphics programs, ones like MacDraw and Cricket, allow ordinary people to do on their personal computers what drafting and art departments used to do by hand. There are no data available on the work time or quality of the figures produced by these amateur draftspeople, though. The programs are certainly popular. Where I work, highly educated scientists and engineers spend days producing graphics of much lower quality than drafting department standards. Before every conference, I find Ph.D.s in on weekends running back and forth from their offices to the printer. It appears that people who are unable to execute pretty pictures with pen and paper find it gratifying to try with a computer. This is an important example of a phase two application success, a computer aid that allows people to do things that they could not before. With greater ease and quality, it could be an economic gain as well as a source of personal satisfaction.

"Last year I spent more than fifty hours trying to draw my house in a 3-D graphics program. There is not a wall in that drawing that is properly connected to any other wall. The roof kind of floats here and there, depending on the angle. I eventually abandoned the application and built a model of the house out of construction paper."(Tognazzini 1992, 189).

Desktop Publishing

Desktop publishing means doing type composition, page layout, and laser printing in addition to typing and copyediting on a PC. Desktop publishing has made small-scale publishing ventures, particularly newsletters, spring up like mushrooms after rain. Whether there is a genuine economic advantage over traditional publishing techniques is not so clear. Instead, it could be another case in which person A's time is traded for person B's, in which flexibility is gained but advantages of specialization and scale are lost. Most major book publishers and magazines have not turned to these techniques.[3] A high-placed editor of a major publishing house recently expressed amazement that so many of his authors

turned in copy carefully prettified with such systems. The first thing he does is have the manuscripts retyped for editing according to house standards, discarding all the fancy typography and layout painstakingly created by the author.[4]

A report on desktop publishing in the public relations business, a heavy service industry user, turned up considerable enthusiasm but some additional problems as well (Gordon 1989). According to one informant, "It allows us to do things very professionally and save time. With new business proposals, it helps us generate income, although I'm not sure exactly how I'd quantify that." In previous years, the glitz of a new business proposal was not as important, he explained. Now expectations are much higher. Another said, "Because you're making changes electronically, you have to proof more carefully—watch out for dropped letters. It adds more time at the late stages."

Several users said the number of changes requested by clients had soared because of the perception that changes were easy to make. Nitpicking was rampant, four or five revisions standard. Because the software is so complex, many companies created new specialists for the job. Other respondents thought the quality that resulted from amateur designers using such systems was unacceptable: "It looks like junk," one said. Said another: "I created a newsletter and then wasted days doing layouts without even realizing how much time I was spending. People have so much fun making pretty pictures, they forget they're there to work."

Artificial Intelligence and Expert Systems

Everyone has heard about artificial intelligence. Indeed, the field's leading researchers agree that everyone has heard too much about artificial intelligence. Early workers phenomenally underestimated the difficulty of mimicking human mental powers and confidently expected early sweeping successes. In the 1960s it was predicted that within ten years computers would convert ordinary speech and handwriting to print, comprehend and compose natural language, drive trucks, do housework, and tutor students better than professors could. Thirty years later many proponents see no reason to change these predictions; they still expect them within ten years.

The present stance of the still-confident is that they underestimated the amount of commonsense knowledge used in real life, and that once this is added to AI programs all will be well. A large project under Doug Lenat at MCC has been trying to do that. When it started in 1986, Lenat thought about a million facts (or rules, as the building blocks of knowledge are called in AI) would do the trick. A few years later he upped the number to 2 million, a few years later to 4 million. When last heard from, he was sure his system would be ready to act human around 1997, and by 2004 would have all the 20 to 40 million facts it really needs (Goldsmith 1994).

It is unfortunate that artificial intelligence has been the subject of so much excessive zeal, for some of its products have become, after all, significantly smart and useful. Special-purpose chess machines can sometimes outplay some grand masters, isolated words and handwritten characters can be recognized under favorable conditions with fair accuracy, artificial visual control can guide smart bombs with devastating effectiveness. (Full natural language processing continues to be more difficult than we expected—too much knowledge needed, truck driving is far too complex in both vision and control, tutoring involves mysteries yet to be cracked).

The most important commercial spinoffs of research in artificial intelligence are known as expert systems or "knowledge-based systems." These programs try to put the knowledge of a human expert into a computer routine. Expert systems have been energetically promoted over the last decade and reputedly have found thousands of applications in a variety of different industries (Feigenbaum, McCorduck, and Nii 1988). Digital Equipment Corp. (DEC) was an early user of the technology. One program could choose and list all the pieces needed for a custom installation of one of DEC's computer systems. When first introduced, it reportedly reduced the time and expertise needed for such planning and cut down on delivery goofs and retrofitting excursions. In 1989, it was reported that DEC was using fifteen people and $2.5 million per year updating and maintaining expert systems (Enslow 1989). This represents both the utility of the programs and one of their emerging deficiencies. The systems tended to get out of date rapidly as products and business changed and fell into disuse unless revamped.

Expert system reporting is remarkably prone to the "expected benefit" syndrome: "TI expects the system to reduce cost overruns and preparation expenses by an average of $2 million a year" (Enslow 1989). A 1988 promotional book by one of the fathers of expert systems, Edward Feigenbaum, and two colleagues, begins by telling us it will survey hundreds of successful systems in use, but closer reading reveals that all but a handful are research prototypes being studied in academic labs, not commercially deployed systems.

Point-of-Sale Systems

My wife stopped by the video store. The clerk typed her name into the computer and announced that she owed $27.50 for late charges. She objected, pointing out that the computer had this huge fine posted against a tardiness of one day. The clerk said, "Oh, the computer's got one of those ghosts. Sometimes it just gets confused and keeps issuing these charges. No one knows where they come from."

Perhaps the most commonly met phase two computer applications are the machines that process grocery checkouts. These are quite amazing gadgets. The box or can is passed over a scanner that automatically identifies the product: the computer looks up its current price, rings it up on the register, prints the product name and price on a tape, and stores the data for later use. At least that happens if the clerk is skillful, the scanner glass clean, and all is working well. The usefulness and usability issues are fascinating because of the number of different players involved. Presumably the major recipient of added utility is the store management, which is supposed to get accurate up-to-the minute inventory information, fewer checkout errors, and less theft; save money by avoiding individual price labeling; and obtain valuable marketing information. Of course, it was thought, all of these advantages could be passed along to the customers in the form of lower prices, a more flexible and popular mix of products, speedier and more accurate checkout.

Let us consider the various parties: employees, management, and customers. The checkout job does not look very much different; where clerks no longer do much price entry, they need new skills to make scanners obey and often have to memorize extensive lists of product numbers to key into the system for bagels and broccoli and other weighed or counted items.

A twelve-month observational case study from England reveals some details about what sometimes actually happens (Cutler and Rowe 1990). The scene is a recently upgraded branch of one of England's largest, most modern chains of supermarkets. Store managers reported that since the scanners were introduced, they were able to divert more employees to bagging on busy days, reducing checkout time, pleasing customers, increasing throughput, and decreasing crowding in the parking lot. Staffing records, however, showed a net *increase* in total hours worked and a shift of employment into operating, maintaining, and managing the technology. There was no before-and-after stopwatching of checkout speed, but both observers and cashiers believed there had been no change. When everything went smoothly, scanned entry was fast. But when bar codes were damaged or produce codes unremembered, times could go very much higher than with traditional registers. One response was to pass difficult items through uncharged. Entry errors were supposed to disappear but sometimes increased instead. Employee morale went down, partly due to the physical discomfort of the new checkout setup—a consultant found poor ergonomics in the station design—and partly due to the shift in jobs.

The actual effect on inventory control and stocking was also unexpected. The system produced more detailed records of items sold—brand X toilet paper and brand Z washing powder separately versus just "household goods." The problem was that, for efficiency's sake, goods were still received from the central supply depot in large lots, say a box from Peter Piper's Paper Products containing twenty cases of toilet paper and thirty of paper towels. Thus, the detailed numbers had little advantage over the traditional method of noticing that brand X toilet paper is getting low.

How else do these systems affect customer service? For one thing, they produce itemized receipts. Customers reportedly like these, and they appear to be necessary; rapid, repeated passes over the scanner to get the required beep often causes double charging that customers don't detect from the quickly vanishing item display. (The English investigators don't speculate as to whether items uncharged even out double entries.) Bar

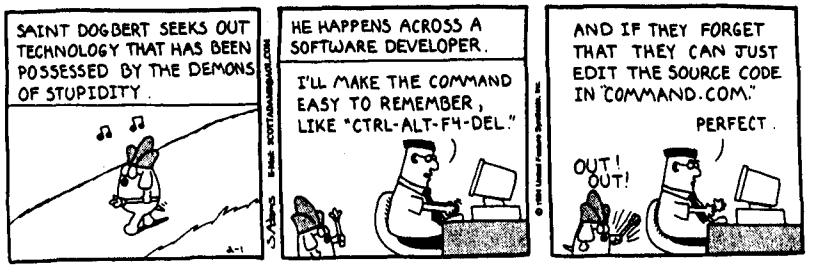
codes also make it possible to forgo individual price labels on items; the computer looks them up from the item code, and prices are posted only on store shelves, not on the products. This makes it easier for the store to change prices quickly. From the customer's point of view, however, once the package is in the grocery cart, only memory can help until the item has gone through the scanner and been posted on the display and receipt. A calculator is needed to estimate the current total value of the cart as you shop and a notepad to compare the price of hot sauces in the condiments aisle with those in foreign foods. If you want to keep track at home (e.g., to notice that breakfast cereal prices sometimes increase twice in as many months) you have to search receipts instead of looking in the pantry. (Ah, here's a use for your database program: key in a description of each grocery item and its price.)

My friend Dave remarks that supermarkets these days seem to be out of items much more frequently. He blames it on "not quite in time" inventory control made possible by IT.

Transaction Systems

Computers stand beside clerks who deal with customers by telephone or mail. The burgeoning mail order business has deftly exploited these systems to manage inventory, buying, marketing, and order processing. Using computers, a small staff can run a high-volume operation. Similarly, the processing of payments for everything from monthly utilities to mutual fund investments is handled by systems in which the input clerk does nothing more than enter the amount of payment; the rest comes from the machine-readable portion of the return stub or is fetched from the bowels of the computer.

From management's point of view, these systems save clerical salaries. From both worker and customer's point of view, it depends on how they are implemented. Some phone-order companies give the entire responsibility for an order to a single agent, the person who answers the telephone. With the help of the computer, this person knows about inventory, is knowledgeable about the product, and interacts with the customer from beginning to end. If there is a problem, the single agent can resolve it. Other times, the functions are divided up; one person takes



DILBERT reprinted by permission of UFS, Inc.

requests, another complaints, someone else knows about inventory, and another about customer account status. Salespeople in the former case probably have interesting jobs, ones in which they effectively handle most of the business. In the latter case, the job is as deadly as bobbin loading in a nineteenth-century textile mill.

As a customer in the former case, you are awed and delighted when told that although the store is out of your size sweater in blue you can have one in green or another style, delivered Tuesday, and, by the way, that the correction you requested for the returned sushi calendar has been made, and thank you, Mr. Landauer—pronounced correctly. Or, in the latter case, you can be bumped from ignorant agent to ignorant agent, eventually to be sent the wrong item at the wrong address; you can write on your bill repeatedly, enclose explanatory letters, call thrice and get nowhere, be charged monthly with interest added and collection threats appended. With computers to help, as they say, it all depends.

Desirability: A Critical Combination of Usefulness and Usability

"The application 'unknown' has suddenly quit because an error of type 15 has occurred."

Usefulness and usability, essential as they are, do not quite say it all. A technology will be of little benefit if few use it. When, following a time-honored premise of industrial management, word processors were used to centralize document preparation, typists lost their knowledge of context and jargon, often producing nonsense output, and author and typist

could no longer fix problems by quick chats. Departments that needed typing didn't like the situation, and most word processing centers were soon abandoned (Johnson and Rice 1987).

A desirability failure that spoils employees' work lives is especially serious. Centralized word processing had this evil defect as well. Typists and their supervisors usually hated the new style of work—its separation from the real consumer, its social isolation from the life of the office, its artificial work flow control procedures. Many bookkeeping applications have been even worse offenders. They separate the most routine, mechanical parts of the work and assign them to back office employees doing physically stressful and mentally deadening jobs. Computerized workplaces all too easily can become modern sweatshops, modern only in the look of the equipment. Such use of technology is morally repugnant. It is also economically stupid; workers who enjoy their work do it better, stay longer in the face of temptation, and contribute ideas and enthusiasm to the evolution of enhanced productivity.

Other Troubles

"In the world of the computer, we calmly accept limitations we would tolerate nowhere else. The typical monitor is no substitute for the printed page, so we squint at fuzzy, ugly characters we would find utterly repugnant in print. Tiny, blurry, jerky windows of video display that even an inveterate couch potato would scorn are hailed as breakthroughs merely because a computer displays them." Stephen Manes, *New York Times,* March 29, 1994, B7.

Technical Limitations

Most troubles come from not knowing what functions the computer ought to perform or from our inability to program them to do what we really would want—to help me liven up this section, for example. The problem is rarely lack of computing power. Nevertheless, there are still troubles from this source. For example, although startling special effects for motion pictures are now being created with computers, fully detailed and shaded simulations of realistically prancing bears and dancing people are still a long way off. Somewhat closer to the office, computer screens are not yet up to snuff. People read anywhere from 5 percent to 30 percent slower from computer screens than from printed pages. The problem is the graininess of the display. The characters on a computer screen are made up of little dots of light and dark, just as they are in offset printing. If there are enough tiny dots, as there are in offset, the human eye can't tell the difference. But that takes at least 100,000 dots (or pixels, as they are called on computers) per square inch of surface. Ordinary computer screens have less than one-fifth that many. In addition, the overall size of computers screens is still too small. You don't get nearly as much area to scan by just moving your eyes and head as you do with a full-sized newspaper or with papers spread out on a desk.

Except for very limited uses, you can't talk to a computer. When you type to it, you have to use special language. We don't know how much these limitations reduce usefulness and usability. It is not clear, for example, for what and how much speech recognition would help. Would you want to play a violin or draw a graph by oral instructions?

Unreliability

On March 13, 1993, the Electronic Data System network connecting 5,200 ATMs all across the United States failed because a snowstorm in New Jersey disabled a computer. It took two weeks to restore service and much longer to untangle the accounting mess created by temporary arrangements and transfers of services to other networks (Jones 1993).

One of the most annoying habits of computers is unreliability. How many times have you been told a service is unavailable because the "computer is down"? Compare this with the number of times you've been told you can't get gasoline because the pumps are broken or you can't have your breakfast because the toaster's not working just now. If cars were nonfunctional as often and unexpectedly as computers, we'd probably give them up. I'm constantly amazed at our forbearance.

Computers fail for many reasons. Only a minority of the failures are due to hardware, though disk drives in particular are quite delicate. More common are software glitches. The programs that run the computer, the system software, are extremely complex. They can't anticipate all the combinations of actions and loads put on them, and thus are often surprised and flummoxed into a kind of mental paralysis. They hang, and

the user has to start them up again from some point before the confusion began.

Perhaps the most common cause of failure is one that doesn't even get blamed on the computer, though it should: user error. One of the chief virtues of a computer is the flexibility with which it can be asked to do so many things. The unfortunate consequence is that the user can ask it to do things the user really doesn't want. The ability to make unlimited changes with a text editor rests on the fact that the text is stored in an evanescent form. It can go through scores of partial revisions. The user has to keep track of which version is current, "saving" new work at appropriate intervals and deciding whether to keep or obliterate the prior version at each change. It is painfully easy to get confused and wipe out things you wanted to keep. A significant fraction of the times we are told the computer is down results from some such operator error. As I will say again and again in many ways, "operator errors" are almost always the system's fault.

Incompatibility

There is yet another aspect of the complexity problem: computers and their application programs are often designed in staggeringly incompatible manners. Many programs ostensibly for use on the same machine don't function the same way or together. In many places the machines themselves are isolated, not connected together, so that work done in one place can't be conveniently transferred anywhere else. As yet, there is not nearly sufficient connectivity between computers. Documents created in paper can be almost universally shared between people and organizations, not so for electronic versions. It is not uncommon for the same business document or data to be typed separately into three different computers.

And . . .

There are also some widely discussed potential dangers and difficulties of the use of computers, both as physical objects—where the case appears quite flimsy—and as social or work instruments—where more worry is justified. Computers do not emit x-rays or microwave radiation in measurable quantities, despite some early concerns that they might. They do

generate some low-frequency electromagnetic energy, as do all other electrical appliances, especially television sets. Although there is no credible evidence of harmful effects, still, almost nothing is known about the long-term consequences. On the other hand, working at a computer terminal for long periods of time can strain eyes, backs, shoulders, and wrists. Resulting fatigue and other painful or disabling symptoms can affect both usefulness and usability and need to be dealt with. Better design of furniture and work practices—for example, arranging frequent short breaks to move around a bit and look at something other than the screen—can help a great deal.

Last, and most, work and social effects of computers are not all always beneficial. IT can usurp jobs. It can diminish satisfaction or remuneration. It makes possible new kinds of covert surveillance and piecework pacing. Computers can deskill and demean work, despite their potential to enrich and improve it.

Well-Known "Successes" and "Failures"

We can get another view of the situation by looking at some major applications of computers—some that are widely regarded as stunning successes, some as catastrophic failures. Three of the best-known success stories are automatic teller machines, airline reservation systems, and the Macintosh user interface.

ATMs Again

ATMs have spread like wild money. They terrorized CitiBank's competitors in New York until other banks banded together and built similar systems. Meanwhile, the head of computing at Chase Manhattan Bank, Elaine Bond, said that ATMs added to the cost of the banks doing business. They are "a convenience that someone has to pay for." It would be nice if computers could offer services that are more desirable than what they replace and at the same time cheaper and more profitable, but perhaps that would be asking them to take us to the moon.

The appeal of ATMs to those who use them is incontestable. They are useful. They improve the convenience of a common chore; users can get cash in many more places, in several other countries, at any time of the

day on almost any day, usually without standing in line. And they are usable. The ideal of usability often put forward by human-computer interaction specialists is the "walk up and use" interface. That means the user needs no instruction other than what is on the screen, and the interface results in successful, error-free, satisfying service. Many ATMs have nearly achieved this ideal. Except when the machine is out of order or money, you rarely hear a patron swearing. In some ways, ATMs are even easier to use than a human teller; the patron never asks "To whom should I make the check out?" "Should I sign it on both front and back?" "What is today's date?"

The computer seems to have been used to good effect in most ATMs. The screen presents needed information in a simple, straightforward way, and entry is by a small set of well-labeled buttons corresponding to a few obvious options. The interaction protocol—the series of steps taken by the customer and the machine—is short and sweet. How did it get so good? It's an interesting story.

Machines with the basic functionality—able do the database operations, log the ledger entries, count out bills, issue records—were available from the early 1970s, but they failed to catch on. Only after an intensive design-test-redesign effort to improve the usability of the interface, conducted by a group of human factors specialists at CitiBank, was public acceptance won. I have heard it said that it "was just a matter of ergonomics," as if that were a minor, easy detail. It was neither minor nor easy; it was critical and difficult. What was most difficult about it was realizing that it must be done.

Interestingly, not all imitators have managed to get it right. New interfaces have been introduced and new features added, often, apparently, without benefit of user-centered design methods. One type features a narrow slit that has to be adjusted perfectly for the customer to see the screen. This added feature, probably thought to be a clever security enhancement, drives me crazy. Some models set a maximum allowed withdrawal amount but don't tell you what it is! If you ask for more than the maximum, it rudely refuses, and you have to start the whole procedure over: insert card, type password, and make another blind guess. Why doesn't the line on the screen that offers the option of a withdrawal also state the maximum? Probably because nobody bothered to watch customers use the system before the design was finalized.

Reservation Systems

Airline reservation systems generate roughly a billion dollars in revenues annually. There are hundreds of airlines, thousands of routes, tens of thousands of airplanes, hundreds of thousands of travelers each day. Managing schedules, making reservations and connections, changing them, and keeping track of fares and payments is an extraordinarily complicated job that is virtually impossible to do by hand. Had computerized systems not existed, perhaps a hand ledger system operating by telephone could have evolved. A rather large airline in India operated this way. However, once the records and schedules were computerized in America, there was no turning back. Any airline that wanted to stay in business had to get itself listed in a reservation system, and to serve their function, all the systems had to talk to each other.

The usefulness of reservation systems is obvious. Business efficiency requires keeping track of what seats are occupied, what cars are where, what rooms are reserved, occupied, and cleaned. Customers like having reservations confirmed anywhere instantly. I lived in England for a while before the introduction of networked airline reservation services. To book, I had a travel agent call the airline and ask whether a seat was available on a particular flight. After some delay, a message came back saying no seat was available. The agent looked up another flight, requested a seat, and repeated until an opening was found—sometimes days later.

But now look at usability. Reservation systems are generally a disgrace. They don't appear too bad from a traveler's perspective. One calls the travel agent and requests a point-to-point connection in a certain time slot at a certain service level; the agent does the rest and usually comes up fairly soon with a reasonable solution. True, with the proliferation of flights, fares, and restrictions, one can be puzzled by the different answers that different agents give on different occasions. And, of course, the system may be down or unavailable, requiring a call back and another Muzak hold. However, these are minor nuisances compared to what can

happen in an exceptional case. The systems are baroquely complicated and horridly hard to learn. Few agents know how to do much beyond the most routine transactions.

My ticket on an international flight got locked in a secretary's drawer over the weekend I was to leave. When I called, the airline said I could easily get a new ticket and apply for a credit on the old, but I had to get to the counter an hour ahead of time. That seemed like a lot of time, but I complied. As it turned out, no one at the counter knew how to enter the transaction into the computer. The first agent, a trainee, made a few attempts, then called for help. The next agent tried several keyboard incantations, consulted an older colleague at the far end of the counter, tried some more, and gave up. One by one, three other agents appeared, palavered, tried experiment after experiment. The desk supervisor had no better luck; manuals and online help were to no avail. It was over an hour before I had a ticket and a receipt whose legitimacy the supervisor doubted.

The direct users of reservation systems are clerks and agents. Their job is fraught with difficulties. Destination airports, airlines, and service categories are entered by arcane codes. Hundreds of these must be memorized and entered flawlessly. It is an intellectually demanding job, too often filled by people with little experience because of rapid turnover (partly due to the nastiness of the job). The computer systems do a miserable job of helping the agents but are nevertheless successful because they are useful: they do something that both the customer and the supplier want done and do it in a much more efficient and effective way than might otherwise be possible. They are successful despite severe problems in usability. This is all too common. Designers assume that the system only needs to perform its job well; Difficulties in operating it are the employees' problem. Employees should be selected, trained, and "motivated" to cope with the wonderful system.

The Macintosh User Interface

The dramatic market success of the Apple Computer Co. is often attributed to the ease of use of its Macintosh interface. This interface—the way

in which information is presented, the machine is controlled by its user, the screen is laid out—is hailed as the epitome of good usability design, the great breakthrough that brought computing to the masses.

The Macintosh interface is based on the use of windows (different portions of the screen devoted to different functions in a flexible and independent manner), the use of a "mouse," icons, and menus (lists of options to be selected). The design of the screen is quite attractive. The Macintosh employs a consistent interaction style: "pull-down" menus are listed at the top and activated by clicking a mouse, then selected by a further mouse click, and so on. This consistent style is generally followed by people who write application programs to be used on the Macintosh as well as Apple's own designers and has become an article of passionate faith on the part of devoted users and developers. Most of these techniques originated in experiments on interaction techniques conducted in the 1960s at the Stanford Research Institute, further developed at Xerox Palo Alto Research Center, and then commercially exploited by Apple.

The way these elements are combined and used in the basic Macintosh operations (that is, ignoring special applications like income tax programs) created a giant step in ease of use. Tests of users doing basic text editing and spreadsheet operations show 40 to 70 percent improvements in work efficiency over other interfaces.

Often the ease of the use of the Macintosh is attributed simply to the presence of the components listed above, to its GUI (graphical user interface, pronounced "gooey"). This explanation is simplistic and false. It is possible, as many applications program for the Macintosh have made clear, to combine all of these elements in ways that make it impossible to work with reasonable efficiency and accuracy. A recent study compared the use of a variety of functions on the Mac with doing the same things on another widely marketed GUI system. The Mac advantage was almost as big over its competitor as it is over typical non-GUI interfaces. Moreover, there are applications on other systems for which excellent usability is obtained without using any GUI features. The effectiveness of the basic Macintosh's functionality derives from the way in which the elements have been combined, which ones are used when for what purposes, what icons and menu options are used for what, and, especially, the well-chosen set of interlocking functions that have been provided. The way in

which the sequence of user actions interacts with how the screen responds has been engineered into a smoothly working whole that optimizes the user's performance of common tasks.

Macintosh's secret of success is the opposite of that of the reservation systems: it is easy to use. Its usefulness is not nearly so dramatic, though. Macs are being used for word processing, graphics, spreadsheets, and databases, but as the superiority of the personal computer for these tasks over their paper and ink predecessors is rather tenuous. It is hard to show that these systems make it possible to do important operations that couldn't be done before or to do old things in a much easier, more effective, or more economical way. Most of their apparent usefulness seems to lie in letting people do themselves what they previously had to have done for them by specialists, though it may not be especially economical or effective to do so.

In the summer of 1991, the National Science Foundation sponsored a conference in Boulder, Colorado, in which researchers were invited to present case histories of systems that had been explicitly designed to help people do mental tasks better and that were proven successes. The participants came up almost empty-handed. They presented mostly designs that were either still in the laboratory or in only limited use. For a real-world example, the participants kept coming back to Macintosh and its sales. We are not swimming in a sea of computer inventions of easily demonstrated value.

Some Not So Successful Examples

Two attempts to build large systems for the U.S. government bombed in instructive ways. One was a system for the patent office, the other for the Internal Revenue Service. A central feature of these two notorious cases is that they are phase two applications, aimed at improving the efficiency and effectiveness of a job done by humans.[5]

Keeping track of the ever increasing flood of patent applications, checking them against the growing mountain of past patents and claims, and processing the new documents is a daunting job. There are about 1,500 patent examiners, over 100,000 applications per year. The main job is searching information files and handling documents, tasks for

which computers seem perfectly suited. The project, begun in 1982, was targeted at greatly increasing the efficiency of the examiner's job, a clear case of intellectual augmentation. A decade later, the system was not quite in operation. Online it will have cost most of $1 billion and increased operating expenses by an amount sufficient to add another 1,000 patent examiners. Strassmann (1990) quotes the assistant commissioner of the U.S. Patent Office as saying, "We really get no short-term increases in productivity. The real gains . . . are in improvement in the quality of work." The lack of evidence of quality improvements from other applications of automated information retrieval is cause for doubt as to whether even this claim will be realized.

The IRS awarded a contract for 18,000 portable computers for field tax auditors, with the goal of increasing their productivity and the accuracy with which they could find errors. The delivered system required an agent to load as many as eighteen different diskettes to do an audit. Three-quarters of the agents said they didn't like the new tool, and two-thirds of them refused to use it.

Another report of the value of computers for government comes from France (Fontaine 1992). Information technology applications in sixteen ministries were studied, with uses ranging from automobile registration to statistical analyses of the labor force and inventory control for state-managed pharmacies. Most of the applications involved either automation of record keeping or databases for management information, with a few intended to help in planning and decision making. There were observations, interviews, and analyses of before-and-after output and staffing. Of the sixteen, only three could be counted as clear wins. Seven clearly cost more than they were worth in labor saved. Poor economic results and poor effects on service quality went hand in hand.

We know about these examples only because they were done in public for the public good. There are undoubtedly scores of equally ignominious failures, although perhaps on smaller scales, in the private sphere. But private businesses do not often tell us about foolish investments that they have made.