Getting the most out of reading

Reading assignments

Suppose your lecturer says to you, "Before your next tutorial I expect you to have read Chapter 3 of Tiddley-Push's book". How do you go about this assignment? If you think "it all depends", what does it depend on?

Your answer may, "What a strange question; I just read the chapter, of course!" or you may have several suggestions on how to go about reading different sorts of material for different reasons. If you said, "I just read the chapter, of course", you probably have a rather passive approach to reading.

To make a success of **writing** ... a textbook, a magazine article, a handout, an essay ... all sorts of questions need to be asked and answered. For example:

- Why am I going to write this? Who am I writing it for?
- What points do I want to make?
- Which are the most important points?
- My readers are going to find this idea difficult: how can I best explain it? Should I include an example? What's the best example?
- Should I include some exercises?   ... and so on ...

When you are **reading** you need to ask and answer corresponding questions:

- Why am I going to read this?
- What points was the author trying to make?
- Which are the most important points? ... to the author?... to me?
- What did my lecturer expect me to get out of this reading?
- Can I follow this example? What point is it supposed to illustrate?
- Do I agree with the author's opinion at this point? ... has he justified it?
- Can I do this exercise? (Have I really understood?)   ...and so on...

The main ingredient of **active reading** is this process of asking and finding answers to questions. Some of the questions need asking and answering before you start a reading assignment, some need tackling as you read.

Why am I going to read this?

The first thing an active reader tries to do is find a preliminary answer to the question, "Why am I going to read this?" The answer, "Because I was told to," doesn't really help much! Even if you set yourself the reading, make sure you know why you are going to do it.

Here are some possible answers to the "Why am I going to read this?" question:

- to get a general overview without getting bogged down in detail
- so that I understand and can explain each concept in detail, and can apply this understanding to solve problems
- to gather information on one or two particular topics
- to compare what this author has to say on this subject with what I have read in other books or with the views of my lecturer
- just for fun!

There are, of course, other possible answers, and you may care to add to the list.

If lecturers set reading, make sure you take note of any information they give you on why you are to do the reading. For example, your lecturer may say,

- "Read this chapter to get a general idea of what it's about: don't worry if you find some of it difficult; I'll explain the tricky bits in my lecture next week."

or ...

- "Read Chapter 3; I'm not going to lecture on this material, but you need to understand it thoroughly. I'll give you a self-test in the tutorial next week."

Even if you are already quite clear about why you are going to read a text, it is very sensible to start off by skimming through it.

Skim-reading

A "skim read" is a "look through", but it is a "look through" with a purpose. The purpose is to get an overview, a general idea of what the piece is about, the author's approach and main concerns.

If you have set yourself the reading (as part of an information gathering process) your skim-read may be enough to tell you that this particular text is not what you are looking for. If it **is** what you are looking for, the skim-read will help by providing an overall map as you do your later, thorough reading of the relevant parts.

Why skimming is useful

Here are some reasons why an initial skim-read is useful:

- You will start to get a feel for "what it's all about". Your brain can begin to round up any existing knowledge you have that will help you as you read.
- You will begin to get an idea of what you need to get out of the text, what is important for you: there may be a difference between the author's priorities and your own.
- You will get a feel for how difficult the reading is going to be for you. Different people find different things difficult.

How to do it

Obviously, the way you go about skimming through a single chapter, or a couple of pages, will be rather different from the way you tackle skimming a 200 page book. The hints given here assume that what you are going to read is between about five and thirty pages in length.

Hints:

- Keep moving! Don't get bogged down. If you read a sentence you don't understand, leave it and carry on.
- As you skim, look out for headings and subheadings: these will give you an idea of the author's train of thought, and how the text is organised, as well as what it's about. (Headings and subheadings act as signposts).
- If there is a short section headed "Introduction", or something similar, it's a good idea to read this in full. An introduction may not have a special heading, so make sure you spot it.
- If there is a section headed "Conclusions", or "Summary" stop and read this. If there is no such heading, read the last paragraph of the text anyway.
- Look out for phrases like "in this section we ...", and "to sum up ...", and "the most important point to remember is ...", and words and phrases underlined, in italics, or bold font. Such phrases provide more signposts: if you spot one, it's usually worth stopping to read more carefully the sentence containing it.
- On your skim through, don't stop to read anything that starts "for example", "to illustrate", etc. Keep on track ... You are trying to spot the main points.
- If you aren't used to skimming, don't try to go too fast, but don't lapse into word-by-word reading. As you practise skim-reading, try to improve your speed.

After the initial skim ....

When you've completed your first skim through, ask yourself:

- What is this text about? What is the author trying to do?
  (Can you write down a few sentences that seem to capture the answer?)
- Do I need to revise my view of why I am going to read it?
  (If it is reading you have set yourself, you may of course decide at this point that you're not going to read it at all.)
- Roughly how long will it take me to get what I need out of it?
  (With practice, you'll get better at making this kind of estimate.)

The next step: reading it "properly"

You are now ready to study the text properly. What "properly" means depends partly on why you are reading a particular text. If you are gathering information about one particular topic from a book or paper that is about many different topics, then you will skip or just re-skim chunks of the text that are not relevant to your current task, but will still (one hopes) be reading the relevant parts "properly". Reading properly means reading with understanding. Never say to yourself, "This is hard, I can't understand it, so I'll just have to remember it".

It is very difficult to remember what you do not understand. If you do understand, remembering will usually take care of itself. Look at the two word sequences in capital letters below: which one are you most likely to remember in three days' time? Both contain the same number of symbols!

"WIGOG AGGLE STRUMSIK DRI POLMI OPLESTOPLIKENS"

"YOU ARE A VERY TALENTED AND CREATIVE STUDENT"

There are of course other problems with learning like a parrot: you won't be able to use your knowledge to solve problems, because you won't know what knowledge you've got; you won't be able to pick relevant ideas from it to put together an essay, for example, because you won't know which bits are relevant; you certainly won't be able to make any intelligent critical comments on it.

Active reading

The process of **active reading**, is be similar to the process of **active listening** described earlier: it requires the same questioning approach. To demonstrate this, we repeat the list of the kinds of questions you should be asking yourself during a lecture: you should be able to see that they are equally relevant to reading.

- What is the point he's making by using this example?
- He just said "Therefore ..." (or "so", or "it follows that").
  Can I see why this is a consequence of what he said earlier?
- Can I think of an example of what he just said?
- He's obviously building up to some conclusion. Can I guess what it will be?
- He just said "It's important to remember ...". Can I see why it's important?
- "Transient"? What does that mean?

While reading, you are working at your own pace, you can spend more time trying to answer question as you go along: in a lecture, you have to go at the lecturer's pace. Active readers "do things" as they read. Unless you have a quite remarkable short term memory, you will need to "do things with pencil and paper"

It is best to make a distinction between

- notes you need to make as part of the process of reading and understanding a piece of text: we shall call these **working notes**.
- the kind of notes you want to have in your folder for the longer term, to use for reference and revision: we shall call these **reference notes**.

Making working notes

What your working notes look like depends partly on the kind of material you are reading, partly on your purpose in reading, and partly on you: the important thing

is to think of them as "throw-away" notes, and not worry about how chaotic they look as long as they are helping you understand. You may make your jottings in sequence down a page, or prefer to construct the sort of mind-map we illustrated in Part 1 as a way of taking notes in lectures.

As you read, write down questions as they occur to you, and look out for answers to to them as you continue to read.

- If you have written down a question like "**why** does he think the terms 'modernism' and 'post-modernism' are unhelpful?" or "**why** does he say 'hardware development is the bottle-neck holding up the progress of computing'?" somewhere, later on, he will (or should) give his reasons.
  - Stop: summarise his reasoning
  - Does it satisfy you? Are you convinced? Why? Why not?
  - Try to note down further evidence of your own, for and against the author's reasoning. Think about other related reading you've done, lectures, your own experience ...
  - Has doing all this raised more questions you need to note down and look for answers to?
- If you hit a word you don't understand then stop and look it up right away. (What **does** "transient" mean?)
- If the reading includes exercises, tackle these as a standard part of "doing the reading": your attempts at working out the answers will form part of your working notes.

Jotting down questions, working through exercises, making links with your existing knowledge or personal experience, trying to express difficult bits of the text in your own words, and so on, all make a major contribution to active reading.

You may need to make a second reading. One reading may not be enough. Some of the questions you jotted down on your first reading may still be unanswered when you get to the end. Sometimes you will have to look elsewhere for the answers, but often they will be in the text itself: another reading may be required to answer them.

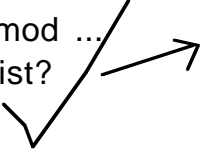## When you've finished ... make "reference notes"

Reading usually has an immediate purpose (for example, preparing for your next class) **and** a long term purpose. A lecturer is unlikely to want you to understand something for next Wednesday but be happy for you to forget all about it after that! Any reading that was worth more than an initial skim contains something worth remembering (otherwise you should have abandoned it after the skim).

Some day, in a month or perhaps a year's time, you may want to use information, opinions, arguments from this reading as part of an essay or presentation. If all you have are the working notes you made as you tackled the reading you will probably find them almost useless. We've all had this unhappy experience.

Working notes reflect the way you were thinking at the time of reading, and this will have been changing as you went along. You may have lots of jottings about one small section that puzzled you at the time, but which, once understood, turned out not to be particularly important; you may have only very brief reference to the most important points of all, if these did not cause you any difficulty as you read.

Working notes are tied very closely to the (changing) state of your knowledge and the (changing) way you were thinking as you did the reading. They contain cryptic jottings like ...

```
mod + post-mod ...          explained p7
not useful dist?            see **
WHY???
```

"Seeing **" a couple of months later may not shed much light, and the book's probably back in the library. When you return to your notes for some future purpose, you don't usually want to have to go back and read the original text again: you want to have **reference** notes that stand alone.

Whenever you are given, or set yourself, a reading assignment, you need to think about whether you need to keep reference notes on it, and if so of what kind. Reference notes need to be a little more polished, and should ideally be understandable without having to refer back to the original text.

Key points about reference notes

The best time to make your reference notes is when you feel you have understood the particular text as well as you are going to, and before you have a chance to start forgetting it.

Always include:

- A full "bibliography entry" for the text:

  (It is maddening if you want to include a reference to this text in an essay or report and you can't remember what it was called or who wrote it. You will be expected to acknowledge sources, and will need to give full details. Also, if you want to look at the original text again yourself you need to be able to **find** it!)

- A brief "review" in your own words:

  (If you were going to write one to two hundred words about this text to serve as a "book review", what would you say? What is it about? Who or what is it useful for? What does it do well? What does it do less well or badly?)

- Key learning points:

  (There's little point in writing notes for reference that say things you knew inside out already before you did this reading: for example "The original DOS operating system was built to address only 64K of RAM." Note down the **new things** you learnt from this reading, and why they were worth learning.)

- Useful quotations:

  (If you want to refer to this text from your own writing at some future point, short quotations will be useful. Pick them carefully: don't fall into the trap of writing out half the text! With each quotation, make a note of **why** you have chosen it, how the author justifies the statement, how you would justify it … or argue against it: quotations you can disagree with are useful too.)

It isn't possible to give a comprehensive general list of what you should include: particular texts, and your purpose in reading them, may suggest additions to the list above.

Write your reference notes so that they will half a long shelf-life (like a tin of soup rather than an apple in a fruit bowl). Write in complete sentences, rather than "note form". Write neatly … well, neatly enough for **you** to be able to read your hand-writing a year later, even if no one else can!

Why not write reference notes as you do the reading?

This is a reasonable question, and if you are reading something very straightforward it may be feasible. If the reading is difficult, it isn't a good idea to try to do this.

Doing difficult reading is a bit like playing a complicated adventure game, like Myst for example.

Supposing you wanted to provide one of those "crib sheets" for completing a game like this. The crib sheet you finally produced would **not** reflect all the exploring, questioning, missed clues and back-tracking you did in order to complete the quest yourself: if it did it wouldn't be very useful to anyone else.

As you play the game yourself, and try to progress through it to completion, your head is full of "working notes" - if you are really dedicated, you may actually write working notes as you play: "can't get in!!! need key … back courtyard".

As you play, you don't know what's going to be really important and what isn't, so you can't start writing the crib sheet. When you complete the quest, **then** you can go back and start to write the crib sheet.

As you tackle a difficult piece of reading, you are on a quest for understanding, you don't know what's going to be really important and what isn't, so you can't write your "crib sheet" as you go along.

A passage for reading, analysis and evaluation

Read <u>all</u> of <u>this</u> page before turning to the next!

The extract on pages 17 to 21 is from Sherry Turkle's book,*Life on the Screen: Identity in the Age of the Internet* , published by Weidenfeld and Nicholson, 1996. The extract doesn't start at the beginning of a chapter. For full understanding, you need to know this chapter begins with a description of how the author wrote "French compositions" as a student, by writing little bits of the composition on scraps of paper, then pushing them around on the floor until they fitted together to make the essay she wanted ... She had to submit a "plan", but wrote this **last**!

[We've taken a few liberties with the footnotes, removing the additional commentary from most of them and leaving only the references to sources.]

It is important that you read and understand this passage, as reference is made to it at several points in later Parts of the booklet.

Exercises on the passage

We hope you'll enjoy the passage for its own sake, but the following exercises are aimed at helping you get as much out of it as possible, and practice some of the ideas and skills discussed in this booklet so far.

Exercise 1:

> (First look back at Part 2 and review the hints on how to skim read.)
> Make an initial skim read of the passage (this should take three or four minutes).
> Write a few sentences to record your first impressions of what the extract is about.

Exercise 2:

> (First look back at Part 2 and review the material on making working notes.)
> Now give the passage a thorough reading. Make working notes as you read.

Exercise 3:

> (First look back at Part 2 and review the material on making reference notes.)
> Make some reference notes on the passage.

Exercise 4:

> What audience do you think Turkle is writing for? Who is going to be interested? Who is going to be able to understand? How would you describe her writing style? Is it appropriate to the subject matter and her audience?

Exercise 5:

> Are you, by nature, or from time to time, a "bricoleur"? If so, you may at some stage in your student career need to justify this way of working. What points could you use from this passage to support your case?

Extract from Chapter 2 of

Life on the Screen: Identity in the Age of the Internet

PLANNING AND TINKERING

The instructor in my 1978 programming class at Harvard - the one who called the computer a giant calculator - described programming methods in universal terms, which he said were justified by the computer's essential nature. But from the very beginning of my inquiries into the computer culture, it became clear that different people approach programming in very different ways. Where my professor saw the necessary hegemony of a single correct style, I found a range of effective yet diverse styles among both novices and experts.[1]

The "universal" method recommended by my Harvard instructor is known as structured programming. A model of the modernist style, it is rule-driven and relies on top-down planning. First you sketch out a master plan in which you make very explicit what your program must do. Then you break the task into manageable subprograms or subprocedures, which you work on separately. After you create each piece, you name it according to its function and close it off, a procedure known as black boxing. You need not bother with its details again. By the 1970s, this structured, planner's method was widely accepted as the canonical style in computing. Indeed, many engineers and computer scientists still see it as the definitive procedure, as simply the way things must be done. They have a powerful, practical rationale for this method. In real organisations, many people have to be able to understand and use any particular piece of software. This means it has to be understandable and fixable (debuggable) long after its programmer has left the research team or business setting.

Others, however, had a style of programming that bore a family resemblance to my associative style of writing, a "soft" style as opposed to a "hard" one. It was bottom-up rather than top-down. It was built up by playing with the elements of a program, the bits of code, much as I played with the elements of my essay, the bits of paper strewn across my room. It is best captured by a word, bricolage, that Claude Levi-Strauss has used to contrast the analytic methodology of Western science with an associative science of the concrete practiced in many non-Western societies.[2] The tribal herbalist, for example, does not proceed from top-down design but by

---

[1]   Sherry Turkle, *The Second Self : Computers and the Human Spirit* (New York: Simon and Shuster, 1984); Sherry Turkle and Seymoor Papert, "Epistemological Pluralism: Styles and Voices within the Computer Culture," *Signs* 16 (1990): 128-57; and Sherry Turkle and Seymoor Papert, "Epistemological Pluralism and the Revaluation of the Concrete," *Journal of Mathematical Behaviour* 11 (1992): 3-33.

[2]   Claude Levi-Strauss, *The Savage Mind* (Chicago: University of Chicago Press, 1968) pp. 16-33.

arranging and rearranging a set of well-known materials can be said to be practicing bricolage. They tend to try one thing, step back, reconsider, and try another. For planners, mistakes are steps in the wrong direction; bricoleurs navigate through midcourse corrections. Bricoleurs approach problem-solving by entering into a relationship with their work materials that has more the flavour of a conversation than a monologue. In the context of programming, the bricoleur's work is marked by a desire to play with lines of computer code, to move them around almost as though they were material things - notes on a score, elements of a collage, words on a page.

Through the mid-1980s, soft-style programmers, programming's bricoleurs, received their own discouraging "French lessons" from a mainstream computer culture deeply committed to structured programming. People who did not program according to the canon were usually told that their way was wrong. They were forced to comply with the officially sanctioned method of doing things. Today, however, there has been a significant change. As the computer culture's center of gravity has shifted from programming to dealing with screen simulations, the intellectual values of bricolage have become far more important. In the 1970s and 1980s, computing served as an initiation into the formal values of hard mastery. Now, playing with simulation encourages people to develop the skills of the more informal soft mastery because it is so easy to run "What if?" scenarios and tinker with the outcome.

The revaluation of bricolage in the culture of simulation includes a new emphasis on visualization and the development of intuition through the manipulation of virtual objects. Instead of having to follow a set of rules laid down in advance, computer users are encouraged to tinker in simulated microworlds. There, they learn about how things work by interacting with them. One can see evidence of this change in the way businesses do their financial planning, architects design buildings, and teenagers play with simulation games.

There is something ironic about the computer presence playing a role in nurturing such "informalist" ways of knowing, since for so long, the computer was seen as the ultimate embodiment of the abstract and formal. But the computer's intellectual personality has always had another side. Computational objects - whether lines of code or icons on a screen - are like abstract and mathematical objects, defined by the most formal of rules. But at the same time, they are like physical objects - like dabs of paint or cardboard cutouts. You can see them and move them, and in some cases you can place one on top of another. Computational objects have always offered an almost-physical access to the world of formal systems.[3] There have

---

[3]   Seymoor Papert, *Mindstorms: Children, Computers, and Powerful Ideas,* 2nd rev. ed. (New York: Basic Books, 1993 [1980])

always been people whose way of interacting with them had more in common with the style of the painter than with that of a logician.

Consider Lisa, an eighteen-year-old freshman in my Harvard programming course. Lisa's first experiences in the course were very positive. She wrote poetry and found that she was able to approach programming with ways of thinking that she had previously found useful in working with words. But as the term progressed, she came under increasing pressure from her instructors to think in ways that were not her own. Her alienation did not stem from an inability to cope with programming but rather from her preference to do it in a way that came into conflict with the structured and rule-driven style of the computer culture she had entered.
     .... [*about a page omitted here*] ...

Lisa's classmate, Robin, was a pianist with a similar learning style. She wanted to play with computational elements, to manipulate the bits of code, as though they were musical notes or phrases. She, too, was told her way was wrong. Her instructor told her it was "a waste of time."

Lisa and Robin came to the programming course with anxieties about not belonging because they did not see themselves as "computer people." Although both could master the class material intellectually, the course exacerbated their anxieties about not belonging because it insisted on a style of work so different from their own. Both received top grades, but each had to deny who she was in order to succeed. Lisa said that she turned herself "into a different kind of person," and Robin described what she had to do as "faking it."

In the 1970s and 1980s, soft mastery was computing's "different voice."[4] Different and in no way equal. The authorities (teachers and other experts) actively discouraged it, deeming it incorrect or improper. But I found many Lisas and many Robins in schools, universities, and local computer clubs. These were boys and girls, men and women, novices and experts, who reported that they had changed their styles to suit the fashion when they had started to interact with the official computer world. "I got my wrists slapped enough times and I changed my ways," says a college student for whom soft style programming was a passion until he entered MIT and was instructed in the canonical programming style. The cost of such wrist slapping was high. On an individual level, talent was wasted, self-image eroded. On the social level, the computer culture was narrowed.

DISCOVERS AND DENIGRATIONS OF THE CONCRETE

The elite status of abstract thinking in Western thought can be traced back at least to

---

[4]   Carol Gillighan, *In a Different Voice:: Psychological Theory and Women's Development* (Cambridge, Mass.: Harvard University Press, 1982).

Plato. Western scientific culture has traditionally drawn a firm line between the abstract and the concrete. The tools of abstraction are propositions, the tools of concrete thinking are objects, and there has always been a right and wrong side of the tracks. The terms "pure science" and "pure mathematics" made clear the superiority of selecting for the pristine propositions and filtering out the messy objects. In the twentieth century, the role of things-in-thinking has had powerful intellectual champions. But, even among these champions there has been resistance to the importance of the bottom-up style of thought preferred by Lisa and Robin. For example, Levi-Strauss and the noted Swiss psychologist Jean Piaget both discovered ways of reasoning that began with objects and moved to theory, but then they found ways to marginalise them.

In the 1920s and 1930s, Piaget first noticed concrete modes of reasoning among children.[5] Children thought that when you spread three marbles apart there were more marbles than when you moved three marbles close together. Through such observations, Piaget was able to see what others had not. Concrete mapping and manipulation of objects enable children to develop the concept of number, a concept that only gradually becomes a formal sense of quantity. The construction of number, in other words, is born through bricolage.

Piaget fought for the recognition of this kind of concrete thinking, but at the same time he saw it as something to be outgrown. The adult was "beyond" the concrete. For Piaget there was a progression in modes of reasoning that culminates in a final, formal stage when propositional logic liberates intelligence from the need to think with things. So Piaget both discovered the power of the concrete in the construction of the fundamental categories of number, space, time, and causality, and denigrated what he had found by relegating concrete ways of knowing to an early childhood stage of development.

Piaget's discoveries about the processes of children's thinking challenged a kind of cultural amnesia. Adults forget the way they reasoned as children. And we forget very quickly. While Freud discovered the forgetting of infantile sexuality, Piaget identified a second amnesia, the forgetting of concrete styles of thinking. In both, our stake in forgetting is highly charged. In our culture, the divide between abstract and concrete is not simply a boundary between propositions and objects but a way of separating the clean from the messy, virtue from taboo.

Levi-Strauss, too, both discovered and denied the concrete. He described bricoleur scientists who do not move abstractly and hierarchically from axiom to theorem to

---

[5] Jean Piaget, *The Child's Conception of Number,* trans. C. Gattegno and F. M. Hodgson (London: Routledge and Kegan Paul, 1952); *The Child's Conception of Space,* trans. F. J. Langdon and J. L. Lunzer (London: Routledge and Kegan Paul, 1956); *The Child's Conception of Physical Causality,* trans. Marjorie Gabain (London: Routledge and Kegan Paul, 1930).

corollary but construct theories by arranging and rearranging a set of well-known materials. But the bricoleur scientists he described all operated in non-Western societies. As Piaget had relegated the concrete to childhood, Levi-Strauss relegated it to the so-called "primitive" and to modern Western humanists. What Levi-Strauss had a hard time seeing were the significant elements of bricolage in the practice of Western science.[6]

.... [*about a page omitted here*] ...

THE REVALUATION OF THE CONCRETE

Soft mastery is not a stage, it is a style. Bricolage is a way to organize work. It is not a stage in a progression to a superior form. Richard Greenblatt is a renowned first-generation MIT hacker. a computer culture legend whose virtuoso style of work incorporates a strong dose of bricolage. He has made significant contributions to the development of chess programs as well as systems programming. In the spirit of the painter who steps back to look at the canvas before proceeding to the next step, Greenblatt developed software that put him in a conversation, a negotiation with his materials. He used bricolage at a high level of artistry.[7]

Yet even internationally recognized bricoleur virtuosos such as Richard Greenblatt lived within a dominant computer culture that was scornful of their approach. One of that culture's heroes was the mathematician Richard Dykstra. Dykstra, the leading theorist of hard, structured programming, emphasized analytical methods and scientific rigor in the development of programs. In Dykstra's view, rigorous planning coupled with mathematical analysis should produce a computer program with mathematically guaranteed success. In this model, there is no room for bricolage. When Dykstra gave a lecture at MIT in the later 1970s, he demonstrated his points by taking his audience step by step through the development of a short program. Richard Greenblatt was in the audience, and the two men had an exchange that has entered into computer culture mythology. It was a classic confrontation between two opposing aesthetics. Greenblatt asked Dykstra how he could apply his mathematical methods to something as complicated as a chess program. "I wouldn't write a chess program," Dykstra replied, dismissing the issue.

[... the chapter doesn't end here; if you are interested, why not get the book ...]

---

[6]   Levi-Strauss, *The Savage Mind*   [See footnote 2]
[7]   Of course, hackers like Greenblatt were technically able to write programs with well-delineated subprocedures, but their way of working had little in common with the techniques of the hard master. They did not write a program in sections that could be assembled into a product. They wrote simple working programs and shaped them gradually by successive modifications. If a particular small change did not work, they undid it with another small change. Hackers saw themselves as artists. They were proud of sculpting their programs.