SOFTWARE AND MIND

Andrei Sorin

EXTRACT

Chapter 2: *The Mind*

This chapter shows why the mechanistic models promoted by the software elites cannot attain the intelligence, creativity, skills, and intuition of human beings.

The entire book, each chapter separately, and also selected sections, can be viewed and downloaded free at the book's website.

www.softwareandmind.com

SOFTWARE AND MIND

The Mechanistic Myth and Its Consequences

Andrei Sorin

ANDSOR BOOKS

Published by Andsor Books, Toronto, Canada (www.andsorbooks.com)
First edition 2013. Revised 2019.

For disclaimers see pp. vii, xvi.

Designed and typeset by the author with text management software developed by the author and with Adobe FrameMaker 6.0. Printed and bound in the United States of America.

Acknowledgements

Excerpts from the works of Karl Popper: reprinted by permission of the University of Klagenfurt/Karl Popper Library.

Excerpts from *The Origins of Totalitarian Democracy* by J. L. Talmon: published by Secker & Warburg, reprinted by permission of The Random House Group Ltd.

Excerpts from *Nineteen Eighty-Four* by George Orwell: Copyright ©1949 George Orwell, reprinted by permission of Bill Hamilton as the Literary Executor of the Estate of the Late Sonia Brownell Orwell and Secker & Warburg Ltd.; Copyright ©1949 Harcourt, Inc. and renewed 1977 by Sonia Brownell Orwell, reprinted by permission of Houghton Mifflin Harcourt Publishing Company.

Excerpts from *The Collected Essays, Journalism and Letters of George Orwell*: Copyright ©1968 Sonia Brownell Orwell, reprinted by permission of Bill Hamilton as the Literary Executor of the Estate of the Late Sonia Brownell Orwell and Secker & Warburg Ltd.; Copyright ©1968 Sonia Brownell Orwell and renewed 1996 by Mark Hamilton, reprinted by permission of Houghton Mifflin Harcourt Publishing Company.

Excerpts from *Doublespeak* by William Lutz: Copyright ©1989 William Lutz, reprinted by permission of the author in care of the Jean V. Naggar Literary Agency.

Excerpts from *Four Essays on Liberty* by Isaiah Berlin: Copyright ©1969 Isaiah Berlin, reprinted by permission of Curtis Brown Group Ltd., London, on behalf of the Estate of Isaiah Berlin.

Library and Archives Canada Cataloguing in Publication

Sorin, Andrei
Software and mind : the mechanistic myth and its consequences / Andrei Sorin.

Includes index.
ISBN 978-0-9869389-0-0

1. Computers and civilization. 2. Computer software – Social aspects.
3. Computer software – Philosophy. I. Title.

QA76.9.C66S67 2013 303.48'34 C2012-906666-4

Don't you see that the whole aim of Newspeak is to narrow the range of thought?… Has it ever occurred to you … that by the year 2050, at the very latest, not a single human being will be alive who could understand such a conversation as we are having now?

George Orwell, *Nineteen Eighty-Four*

Disclaimer

This book attacks the mechanistic myth, not persons. Myths, however, manifest themselves through the acts of persons, so it is impossible to discuss the mechanistic myth without also referring to the persons affected by it. Thus, all references to individuals, groups of individuals, corporations, institutions, or other organizations are intended solely as examples of mechanistic beliefs, ideas, claims, or practices. To repeat, they do not constitute an attack on those individuals or organizations, but on the mechanistic myth.

Except where supported with citations, the discussions in this book reflect the author's personal views, and the author does not claim or suggest that anyone else holds these views.

The arguments advanced in this book are founded, ultimately, on the principles of demarcation between science and pseudoscience developed by philosopher Karl Popper (as explained in "Popper's Principles of Demarcation" in chapter 3). In particular, the author maintains that theories which attempt to explain non-mechanistic phenomena mechanistically are pseudoscientific. Consequently, terms like "ignorance," "incompetence," "dishonesty," "fraud," "corruption," "charlatanism," and "irresponsibility," in reference to individuals, groups of individuals, corporations, institutions, or other organizations, are used in a precise, technical sense; namely, to indicate beliefs, ideas, claims, or practices that are mechanistic though applied to non-mechanistic phenomena, and hence pseudoscientific according to Popper's principles of demarcation. In other words, these derogatory terms are used solely in order to contrast our world to a hypothetical, ideal world, where the mechanistic myth and the pseudoscientific notions it engenders would not exist. The meaning of these terms, therefore, must not be confused with their informal meaning in general discourse, nor with their formal meaning in various moral, professional, or legal definitions. Moreover, the use of these terms expresses strictly the personal opinion of the author – an opinion based, as already stated, on the principles of demarcation.

This book aims to expose the corruptive effect of the mechanistic myth. This myth, especially as manifested through our software-related pursuits, is the greatest danger we are facing today. Thus, no criticism can be too strong. However, since we are all affected by it, a criticism of the myth may cast a negative light on many individuals and organizations who are practising it unwittingly. To them, the author wishes to apologize in advance.

Contents

Preface xiii

Introduction Belief and Software 1
Modern Myths 2
The Mechanistic Myth 8
The Software Myth 26
Anthropology and Software 42
Software Magic 42
Software Power 57

Chapter 1 Mechanism and Mechanistic Delusions 68
The Mechanistic Philosophy 68
Reductionism and Atomism 73
Simple Structures 90
Complex Structures 96
Abstraction and Reification 111
Scientism 125

Chapter 2 The Mind 140
Mind Mechanism 141
Models of Mind 145

Tacit Knowledge 155
Creativity 170
Replacing Minds with Software 188

Chapter 3 Pseudoscience 200
The Problem of Pseudoscience 201
Popper's Principles of Demarcation 206
The New Pseudosciences 231
The Mechanistic Roots 231
Behaviourism 233
Structuralism 240
Universal Grammar 249
Consequences 271
Academic Corruption 271
The Traditional Theories 275
The Software Theories 284

Chapter 4 Language and Software 296
The Common Fallacies 297
The Search for the Perfect Language 304
Wittgenstein and Software 326
Software Structures 345

Chapter 5 Language as Weapon 366
Mechanistic Communication 366
The Practice of Deceit 369
The Slogan "Technology" 383
Orwell's Newspeak 396

Chapter 6 Software as Weapon 406
A New Form of Domination 407
The Risks of Software Dependence 407
The Prevention of Expertise 411
The Lure of Software Expedients 419
Software Charlatanism 434
The Delusion of High Levels 434
The Delusion of Methodologies 456
The Spread of Software Mechanism 469

Chapter 7 Software Engineering 478
Introduction 478
The Fallacy of Software Engineering 480
Software Engineering as Pseudoscience 494

Structured Programming 501
The Theory 503
The Promise 515
The Contradictions 523
The First Delusion 536
The Second Delusion 538
The Third Delusion 548
The Fourth Delusion 566
The *GOTO* Delusion 586
The Legacy 611
Object-Oriented Programming 614
The Quest for Higher Levels 614
The Promise 616
The Theory 622
The Contradictions 626
The First Delusion 637
The Second Delusion 639
The Third Delusion 641
The Fourth Delusion 643
The Fifth Delusion 648
The Final Degradation 655
The Relational Database Model 662
The Promise 663
The Basic File Operations 672
The Lost Integration 687
The Theory 693
The Contradictions 707
The First Delusion 714
The Second Delusion 728
The Third Delusion 769
The Verdict 801

Chapter 8 From Mechanism to Totalitarianism 804
The End of Responsibility 804
Software Irresponsibility 804
Determinism versus Responsibility 809
Totalitarian Democracy 829
The Totalitarian Elites 829
Talmon's Model of Totalitarianism 834
Orwell's Model of Totalitarianism 844
Software Totalitarianism 852

Index 863

Preface

This revised version (currently available only in digital format) incorporates many small changes made in the six years since the book was published. It is also an opportunity to expand on an issue that was mentioned only briefly in the original preface.

Software and Mind is, in effect, several books in one, and its size reflects this. Most chapters could form the basis of individual volumes. Their topics, however, are closely related and cannot be properly explained if separated. They support each other and contribute together to the book's main argument.

For example, the use of simple and complex structures to model mechanistic and non-mechanistic phenomena is explained in chapter 1; Popper's principles of demarcation between science and pseudoscience are explained in chapter 3; and these notions are used together throughout the book to show how the attempts to represent non-mechanistic phenomena mechanistically end up as worthless, pseudoscientific theories. Similarly, the non-mechanistic capabilities of the mind are explained in chapter 2; the non-mechanistic nature of software is explained in chapter 4; and these notions are used in chapter 7 to show that software engineering is a futile attempt to replace human programming expertise with mechanistic theories.

A second reason for the book's size is the detailed analysis of the various topics. This is necessary because most topics are new: they involve either

entirely new concepts, or the interpretation of concepts in ways that contradict the accepted views. Thorough and rigorous arguments are essential if the reader is to appreciate the significance of these concepts. Moreover, the book addresses a broad audience, people with different backgrounds and interests; so a safe assumption is that each reader needs detailed explanations in at least some areas.

There is some deliberate repetitiveness in the book, which adds only a little to its size but may be objectionable to some readers. For each important concept introduced somewhere in the book, there are summaries later, in various discussions where that concept is applied. This helps to make the individual chapters, and even the individual sections, reasonably independent: while the book is intended to be read from the beginning, a reader can select almost any portion and still follow the discussion. In addition, the summaries are tailored for each occasion, and this further explains that concept, by presenting it from different perspectives.

The book's subtitle, *The Mechanistic Myth and Its Consequences*, captures its essence. This phrase is deliberately ambiguous: if read in conjunction with the title, it can be interpreted in two ways. In one interpretation, the mechanistic myth is the universal mechanistic belief of the last three centuries, and the consequences are today's software fallacies. In the second interpretation, the mechanistic myth is specifically today's mechanistic *software* myth, and the consequences are the fallacies *it* engenders. Thus, the first interpretation says that the past delusions have caused the current software delusions; and the second one says that the current software delusions are causing further delusions. Taken together, the two interpretations say that the mechanistic myth, with its current manifestation in the software myth, is fostering a process of continuous intellectual degradation – despite the great advances it made possible.

The book's epigraph, about Newspeak, will become clear when we discuss the similarity of language and software (see, for example, pp. 409–411).

Throughout the book, the software-related arguments are also supported with ideas from other disciplines – from the philosophies of science, of mind, and of language, in particular. These discussions are important, because they show that our software-related problems are similar, ultimately, to problems that have been studied for a long time in other domains. And the fact that the software theorists are ignoring this accumulated knowledge demonstrates their incompetence.

Chapter 7, on software engineering, is not just for programmers. Many parts

(the first three sections, and some of the subsections in each theory) discuss the software fallacies in general, and should be read by everyone. But even the more detailed discussions require no previous programming knowledge. The whole chapter, in fact, is not so much about programming as about the delusions that pervade our programming practices, and their long history. So this chapter can be seen as a special introduction to software and programming; namely, comparing their true nature with the pseudoscientific notions promoted by the software elite. This study can help both programmers and laymen to understand why the incompetence that characterizes this profession is an inevitable consequence of the mechanistic software ideology.

The book is divided into chapters, the chapters into sections, and some sections into subsections. These parts have titles, so I will refer to them here as *titled* parts. Since not all sections have subsections, the lowest-level titled part in a given place may be either a section or a subsection. This part is, usually, further divided into *numbered* parts. The table of contents shows the titled parts. The running heads show the current titled parts: on the right page the lowest-level part, on the left page the higher-level one (or the same as the right page if there is no higher level). Since there are more than two hundred numbered parts, it was impractical to include them in the table of contents. Also, contriving a short title for each one would have been more misleading than informative. Instead, the first sentence or two in a numbered part serve also as a hint of its subject, and hence as title.

Figures are numbered within chapters, but footnotes are numbered within the lowest-level titled parts. The reference in a footnote is shown in full only the first time it is mentioned within such a part. If mentioned more than once, in the subsequent footnotes it is abbreviated. For these abbreviations, then, the full reference can be found by searching the previous footnotes no further back than the beginning of the current titled part.

The statement "italics added" in a footnote indicates that the emphasis is only in the quotation. Nothing is stated in the footnote when the italics are present in the original text.

In an Internet reference, only the site's main page is shown, even when the quoted text is from a secondary page. When undated, the quotations reflect the content of these pages in 2010 or later.

When referring to certain individuals (software theorists, for instance), the term "expert" is often used mockingly. This term, though, is also used in its normal sense, to denote the possession of true expertise. The context makes it clear which sense is meant.

The term "elite" is used to describe a body of companies, organizations, and individuals (for example, the software elite). The plural, "elites," is used when referring to several entities within such a body.

The issues discussed in this book concern all humanity. Thus, terms like "we" and "our society" (used when discussing such topics as programming incompetence, corruption of the elites, and drift toward totalitarianism) do not refer to a particular nation, but to the whole world.

Some discussions in this book may be interpreted as professional advice on programming and software use. While the ideas advanced in these discussions derive from many years of practice and from extensive research, and represent in the author's view the best way to program and use computers, readers must remember that they assume all responsibility if deciding to follow these ideas. In particular, to apply these ideas they may need the kind of knowledge that, in our mechanistic culture, few programmers and software users possess. Therefore, the author and the publisher disclaim any liability for risks or losses, personal, financial, or other, incurred directly or indirectly in connection with, or as a consequence of, applying the ideas discussed in this book.

The pronouns "he," "his," "him," and "himself," when referring to a gender-neutral word, are used in this book in their universal, gender-neutral sense. (Example: "If an individual restricts himself to mechanistic knowledge, his performance cannot advance past the level of a novice.") This usage, then, aims solely to simplify the language. Since their antecedent is gender-neutral ("everyone," "person," "programmer," "scientist," "manager," etc.), the neutral sense of the pronouns is established grammatically, and there is no need for awkward phrases like "he or she." Such phrases are used in this book only when the neutrality or the universality needs to be emphasized.

It is impossible, in a book discussing many new and perhaps difficult concepts, to anticipate all the problems that readers may face when studying these concepts. So the issues that require further discussion will be addressed online, at *www.softwareandmind.com*. In addition, I plan to publish there material that could not be included in the book, as well as new ideas that may emerge in the future. Finally, in order to complement the arguments about traditional programming found in the book, I have published, in source form, some of the software I developed over the years. The website, then, must be seen as an extension to the book: any idea, claim, or explanation that must be clarified or enhanced will be discussed there.

CHAPTER 2

The Mind

The most pernicious consequence of a mechanistic culture is the disparagement of human capabilities. Because the human environment consists of complex phenomena, the brain has evolved to help us cope with complex structures. We possess naturally, therefore, non-mechanistic capabilities. And we exercise these capabilities every time we do something through personal experience, skills, creativity, and intuition; in other words, simply by knowing how to do it, rather than by following rules and methods.

As we will see in the present chapter, practically all mental acts involved in normal, intelligent behaviour require *non-mechanistic* knowledge. In our mechanistic culture, however, all knowledge that cannot be reduced to simple hierarchical structures is deemed "unscientific," and hence unimportant. We admit that intuition and other personal forms of knowledge are useful in certain fields – in the arts, for instance – but otherwise we prefer knowledge that can be precisely described. Consequently, we are neglecting our superior, non-mechanistic capabilities, and restricting ourselves to the mechanistic ones; that is, to those capabilities we share with our machines. But our most important problems are non-mechanistic. So these problems remain unsolved, and we even cease noticing them. Pleased with our success in solving simple, mechanistic problems, we increasingly ignore the important, complex ones.

Mind Mechanism

We have a special interest in the mechanistic theories of mind, for they are more than just another example of mechanistic delusions. Scientists search for mechanistic explanations of the mind because these explanations would enable them to create models of human intelligence. But in the age of the computer it is more than intellectual curiosity, or the scientistic tradition, that motivates this work. Mechanistic models can be implemented with software, so the search for mechanistic theories of mind is ultimately an attempt to replace minds with software; that is, to find substitutes for human intelligence.

It is relatively easy to study the effects of mind substitutes when software *exceeds* human capabilities – in complicated calculations, for example. But what are the social consequences of mind substitutes when software is *inferior*? We will examine this issue later, in "Replacing Minds with Software." For now, it suffices to remember the connection between the two subjects we are discussing here: the *non-mechanistic* capabilities of human beings and the *mechanistic* theories of mind. Restricted as they are to simple structures, the mechanistic models, and hence the substitutes based on them, can never attain the capabilities of human minds.

❖

The mechanistic philosophy, we recall, is the combination of reductionism and atomism, the belief that every phenomenon can be described as a system of things within things. Our infatuation with mechanism started in the seventeenth century, when philosophers and scientists equated nature with mathematics. All natural phenomena, as well as the phenomena related to human life, were seen as simple structures and hence amenable to mathematical treatment. In particular, knowledge and the working of the mind were represented as simple structures.

We attribute this idea to Descartes, because he was the first to describe it in detail. Any problem, he claimed, any inquiry, can be broken down hierarchically into smaller and smaller parts, until we reach parts that are simple enough to understand directly. Thus, all human knowledge can be represented with hierarchical structures. And consequently, everything that is within the capabilities of the mind can be expressed in a form similar to the deductive system of mathematics: "The long chains of simple and easy reasonings by means of which geometers are accustomed to reach the conclusions of their most difficult demonstrations, had led me to imagine that all things, to

the knowledge of which man is competent, are mutually connected in the same way, and that there is nothing so far removed from us as to be beyond our reach, or so hidden that we cannot discover it."[1]

While Descartes's view was becoming the accepted model of mind, there remained a few dissenters, who claimed that much of our problem-solving ability is intuitive rather than mathematical. The best known of these dissenters is Pascal, a contemporary of Descartes, and himself a distinguished mathematician and scientist. Pascal held that we are capable of two types of thinking, the mathematical and the intuitive, and that we need both: "We know the truth not only through our reason but also through our heart.... Principles are felt, propositions proved, and both with certainty though by different means."[2] Those "accustomed to the clearcut, obvious principles of mathematics and to draw no conclusions until they have clearly seen and handled their principles ... become lost in matters requiring intuition, whose principles cannot be handled in this way."[3]

Descartes, too, recognized the need for intuitive thinking, but he believed it to be necessary only in understanding the *terminal* elements of a knowledge structure; that is, those concepts whose truth we accept as self-evident, because there are no antecedent concepts to deduce it from. All other knowledge, he believed, can be represented exactly and completely by reducing it hierarchically to the terminal elements.

Following Descartes, philosophers of all persuasions – idealists and realists, rationalists and empiricists – proposed theories of mind which assumed that the phenomena of knowledge and learning can be explained mechanistically. None of these theories worked, of course. In the twentieth century, despite the continuing popularity of the mechanistic view, some philosophers came to accept the fact that a great part of our knowledge cannot be formally defined or described. Far from being only peripheral, these philosophers say, the informal knowledge that we develop in the mind is essential for intelligent behaviour.

In the last five decades, we have learned a great deal about our mental capabilities from the work done in the field of computer science known as artificial intelligence.[4] Ironically, though, what we have learned is the exact opposite of what research in artificial intelligence set out originally to prove.

[1] René Descartes, *A Discourse on Method* (London: Dent, 1912), p. 16.

[2] Blaise Pascal, *Pensées* (London: Penguin Books, 1966), p. 58. [3] Ibid., p. 211.

[4] The term "artificial intelligence" is used in this book in its traditional sense – emulating human intelligence with software. (Today it is increasingly used as a marketing slogan for software systems unrelated to this research program.) See Hubert Dreyfus's classic study of the fallacies of artificial intelligence: *What Computers* Still *Can't Do: A Critique of Artificial Reason* (Cambridge, MA: MIT Press, 1992). Through philosophical arguments, Dreyfus shows why the very idea that a machine can display human-like understanding is mistaken.

The aim was to create software that emulates human intelligence, and the computer was seen as the device through which we could actually implement – and hence, finally vindicate – our mechanistic theories of mind. What we discovered instead is that these theories, which had been accepted by countless generations of thinkers, fail blatantly when implemented with actual models.

The attempt to emulate human intelligence with software can succeed only for those mental capabilities that are simple enough to be approximated with isolated hierarchical structures; in other words, for mental acts that can be separated from the complex phenomena of the real world. Unsurprisingly, it was this type of acts that researchers chose in their initial projects. They interpreted the immediate successes as evidence that their mechanistic theories were correct, and concluded that machine intelligence of any level can be attained simply by extrapolating those methods; that is, by building larger and larger hierarchical structures. This led to the now famous proclamations made by some of the most respected scientists in the 1960s – namely, that within a few years computers would be intelligent enough to do everything that human minds can do. Some examples: "Extrapolating the recent rapid gains that have been made, we can forecast with some confidence that a decade hence we shall have a reasonably comprehensive theoretical understanding of the processes of human problem solving."[5] "Technologically … machines will be capable, within twenty years, of doing any work that a man can do."[6] "Within a generation, I am convinced, few compartments of intellect will remain outside the machine's realm – the problems of creating 'artificial intelligence' will be substantially solved."[7]

It is not the naivety of artificial intelligence that concerns us here, though, but its mechanistic roots. All theories that attempt to explain mental acts rely, ultimately, on the concept of hierarchical structures. Whether the simplest elements – the knowledge atoms – are believed to be bits of information, or sensations, or innate functions, to account for actual knowledge these elements must be combined hierarchically into more and more complex ones. There is no other way to explain higher levels of knowledge within a deterministic philosophy. So, whether we attempt to explain intelligent acts by breaking them down into successively simpler parts, or whether, conversely, we start with some elementary capabilities and combine them into successively larger parts, it is always the mechanistic concepts of reductionism and atomism that we invoke; specifically, we try to explain intelligence as some atomic entities combined hierarchically through completely specifiable relations.

[5] Herbert A. Simon, *The Shape of Automation: For Men and Management* (New York: Harper and Row, 1965), pp. 89–90. [6] Ibid., p. 96.

[7] Marvin L. Minsky, *Computation: Finite and Infinite Machines*, (Englewood Cliffs, NJ: Prentice Hall, 1967), p. 2.

The rationalist theories of Descartes and Leibniz, the empiricist theories of Locke and Hobbes, the modern theories of behaviourism and linguistics, are all founded ultimately upon the idea of simple hierarchical structures. Before we had computers, philosophers and scientists could only create hierarchies that were small enough to study manually. Noting how powerful the hierarchical concept was, they imagined that there is no limit to the complexity we can reach with it; all we need, they thought, is a way to create hierarchies with a sufficiently large number of elements, levels, and relations. They had no means to verify this assumption, so they could remain confident that the complexity of human knowledge is simply the result of a very large hierarchical structure created somehow in the mind. The only reason they could not prove this, they believed, was that they had no tools to create models of such large hierarchies.

After centuries of speculations, software finally afforded us the opportunity to create large hierarchical structures, and thereby implement the mechanistic theories of mind. But instead of vindicating these theories, as it was hoped, the software models refuted them. Thus, the failure of the software models of mind became an unintentional refutation of *all* mechanistic theories of mind: "Thanks to AI [artificial intelligence] research, Plato's and Kant's speculation that the mind works according to rules has finally found its empirical test in the attempt to use logic machines to produce humanlike understanding. And, after two thousand years of refinement, the traditional view of mind has shown itself to be inadequate."[8]

The more sophisticated our software becomes, the more obvious are the limitations of the mechanistic models, and the greater ought to be our respect for the unique capabilities of our minds. The failure of artificial intelligence has taught us nothing, however. Few people realize that the failure of mechanistic software concepts in the much broader and more important domain of business application development is due to the same delusion: the belief that human minds can be represented with mechanistic models, and hence human knowledge and skills – in this case, programming expertise – can be replaced with formal methods and with software devices.

Innumerable theories, methodologies, languages, and development tools have been invented as substitutes for programming expertise, and all have failed to provide real benefits. The reason is that software and programming

[8] Hubert L. Dreyfus and Stuart E. Dreyfus, *Mind over Machine: The Power of Human Intuition and Expertise in the Era of the Computer* (New York: Free Press, 1988), p. 98.

phenomena consist of interacting structures, and only minds can process these structures together. The mechanistic substitutes must separate them, so even if they manage to represent the individual structures correctly, they cannot deal with their interactions. Still, the failures are described as temporary setbacks, to be resolved by the "next generation" of methods or devices. So we continue to see the same claims and the same failures year after year. (We will study these delusions in the following chapters.)

We are witnessing an amazing spectacle: Our software experts, those individuals to whom we have entrusted our future, are claiming in effect that their vision of the future is a world where human beings – that is, *we* – no longer matter. Their projects can be summed up quite simply as attempts to prove that software devices are better than human minds. These projects repeatedly fail, but instead of recognizing their failure as evidence of the superiority of our minds and taking pride in our capabilities, we continue to be fooled. Instead of admitting that our software experts are in fact impostors and charlatans, we accept their excuses, keep admiring them, and pay them to try again to prove that their devices are better than our minds.

Models of Mind

1

Let us take a closer look at the non-mechanistic capabilities of the human mind. Recall the process of face recognition: while we can easily *recognize* a familiar face, we cannot *describe* that face precisely enough to enable someone unfamiliar with it to recognize it as easily as *we* do (see pp. 108–109). Thus, when we know a face, we possess a type of knowledge that cannot be expressed as methods or rules. We can readily *use* this knowledge, but we cannot explain how we do it. What this means is that the only way to acquire this knowledge is by allowing it to *develop* in our mind; we cannot acquire it by learning some facts. The mind has the capacity to create, as it were, an internal replica of a phenomenon, and it does this simply by being exposed to that phenomenon.

We acquire a similar type of knowledge when we learn to recognize a voice on the telephone. We can recognize, in fact, the voices of dozens of people by hearing just a word or two; yet we cannot describe *how* we do it. As in the case of face recognition, a person already familiar with a certain voice cannot describe that voice to us so that we would recognize it if we heard it; the only way to acquire this knowledge is through our own experience. Nor can we transmit this knowledge to others once *we* have acquired it.

Note how different this knowledge is from the knowledge of isolated facts like birth dates or telephone numbers, which we can precisely convey to others in words or symbols. And it is not the *simplicity* of these facts that allows us to describe them with precision, for we can also describe with precision the make-up of a complicated machine with thousands of parts and connections. What makes these phenomena precisely describable is that they are reducible to simple structures. The process of recognizing faces or voices, on the other hand, is a *complex* structure – a system of interacting structures.

For a phenomenon that is a complex structure, the only exact representation is the phenomenon itself. We saw that some complex phenomena can be *approximated* with simple structures, by representing them with mechanistic models (rules, mathematics, software, etc.). And, even though our mind has the capacity for complex structures, our knowledge – the replicas created by the mind – is still only an approximation of the actual phenomena. But, being complex structures themselves, the approximations created by the mind are closer to the actual phenomena than are the mechanistic approximations. The mind, therefore, can usefully approximate many phenomena that cannot be usefully approximated with mechanistic models.

We note two facts characteristic of non-mechanistic knowledge. First, this knowledge cannot be transferred directly, fully developed, into the mind; the only way it can arise is by developing inside the mind from the isolated bits of knowledge that reach the mind. Second, the only way this knowledge can develop in the mind is through repeated exposure to the phenomena associated with it; that is, through personal experience.

Using the concept of simple and complex structures, we can say that this knowledge does not inhere in the bits of knowledge detected by our senses. The bits of knowledge form simple structures, and the knowledge consists of the complex structures created by the *interaction* of these simple structures. It is the mind's capacity to discover the interactions that permits us to deal intelligently with the complex phenomena surrounding us.

2

Philosopher Gilbert Ryle used the terms *knowing that* and *knowing how* to distinguish between the two kinds of knowledge we are discussing here.[1] *Knowing that* refers to the knowledge of isolated facts and rules – knowledge we can describe precisely and completely. *Knowing how* refers to the knowledge acquired through personal experience, by being exposed to certain situations

[1] Gilbert Ryle, *The Concept of Mind* (Chicago: University of Chicago Press, 1984), p. 28.

and by performing certain acts. This latter knowledge enables us to behave intelligently, but we cannot describe it in terms of facts and rules. *Knowing how*, thus, cannot be reduced to *knowing that*.

When acquiring a skill, we start by learning facts and following rules; but when we attain expertise, our skilled performance is due, not to memorizing many facts and rules, but to our experience. Intelligent behaviour requires both *knowing that* and *knowing how*. The mechanistic doctrine, though, claims that our mind is only capable of *knowing that*. Our apparent *knowing how*, according to this doctrine, is merely the result of combining in the mind various bits of *knowing that*. Mechanism must deny the existence of *knowing how* because it maintains that every phenomenon can be explained precisely and completely.

Ryle attributes modern mind mechanism to the traditional theory of mind, which he calls "with deliberate abusiveness … 'the dogma of the Ghost in the Machine.'"[2] This dogma, inaugurated by Descartes, endows us with two separate entities, bodies and minds. The body is a physical thing that functions just like a machine, so we can analyze its operation with the same methods we employ in the study of mechanical devices. The mind, however, is a rather mysterious thing: it does not exist in space, so we cannot see it, nor describe its operation mechanically; all we can do is observe the external results of its workings, as manifested through the body. The body and the mind must interact continually, but we cannot see the means of interaction either. A human being, therefore, is somewhat like a machine operated by a ghost.

At first, this concept placed the mind safely beyond the reach of mechanistic theories; but the relief was short-lived. The mechanists acknowledge the intelligent capabilities of human beings, and they recognize that the body itself is incapable of intelligence. This is why they postulate a ghostly device and attribute to *it* the mysterious acts known as intelligence. However, while admitting that this invisible device is different from the material body, the mechanists insist that its operation need not be different from the mechanical operation of the body. The ghost and its tools may be invisible, but it is still possible to study them with the familiar principles of reductionism and atomism.

So, in the end, the mind too became the subject of mechanistic theories. The mechanistic theories of mind claim that knowledge, skills, and intelligence can be explained through a reduction to simpler parts, leading ultimately to some elementary bits of knowledge, skills, and intelligence. What the ghostly mechanism of the mind does when we engage in an act of thinking or intelligent behaviour is to combine those invisible bits using some mysterious

[2] Ibid., p. 15.

instructions, and then pass the invisible result of this process to our body, where we notice it as speech or movement of limbs.

But, if this is true, it means that whenever we perform an intelligent act we do two things: first we perform some internal calculations, deliberations, or planning (swiftly and unconsciously), and *then* we perform the actual act. Ryle shows that this view is absurd, that when we behave intelligently or skilfully we perform only one thing: that intelligent or skilful act.[3] If we believe that intelligent or skilful behaviour is the result of other mental acts, we have to explain *those* acts as the result of yet other acts, and so on. But there are no final, atomic, indivisible mental acts: "The regress is infinite, and this reduces to absurdity the theory that for an operation to be intelligent it must be steered by a prior intellectual operation.… 'Intelligent' cannot be defined in terms of 'intellectual,' or 'knowing *how*' in terms of 'knowing *that*'; 'thinking what I am doing' does not connote 'both thinking what to do and doing it.' When I do something intelligently, i.e. thinking what I am doing, I am doing one thing and not two."[4]

3

Ryle's ghost is only one of a whole army of invisible workers that had to be drafted by scientists and philosophers to serve in their mechanistic models of mind. It is impossible, in fact, to invent a theory that "explains" the working of the mind without having to postulate the existence of some invisible agents who perform some invisible work on behalf of the owner of the mind. Mechanistic mind models, therefore, are compelled to commit what is known as the *homunculus* fallacy:[5] the belief that, to explain a mental process, all we have to do is describe it as an internal rather than an external phenomenon.

For example, to explain vision – how we perceive as real objects the images formed on the retina – the mechanists might postulate a mind model that has a visual module for scanning and interpreting these images. But this model is specious. All it does is shift the definition of vision from a capability of the person to a capability of the mind: instead of saying that the mind lets the person see, the model says that the visual module lets the mind see. The phenomenon of vision remains as unexplained as before. The scanning and interpreting operations attributed to the visual module are not an explanation of vision; they are, in fact, the phenomenon that the model was supposed to explain. This mistake is called a homunculus fallacy because the explanation

[3] Ibid., pp. 30–32. [4] Ibid., pp. 31–32.

[5] *Homunculus* means "little man" in Latin; this phrase is explained in the next paragraph.

amounts to a claim that there is a little man in the mind who performs the operations needed to produce the phenomenon in question. What exactly are those operations, and how the little man knows to perform them, is left unanswered. (Thus, because of their circularity, the homunculus theories of mind resemble the inane scholastic explanations of the Middle Ages: fluidity is that quality of a substance which causes it to flow, elasticity is that quality of an object which causes it to bend rather than break, etc.)

Mechanism, we saw in chapter 1, has been exceptionally successful in explaining natural phenomena. So what the mind mechanists are attempting now – and what most research in artificial intelligence and cognitive science amounts to – is to combine the homunculus concept with the mechanistic concept. Since mechanism works in other fields, they say, it *must* work also for minds. They admit that the homunculus concept on its own is erroneous, but they believe that its circularity can be resolved if we combine it with the principles of reductionism and atomism; specifically, if instead of one little man performing one task we assume a hierarchical structure where many little men perform tasks of different complexity at different levels of abstraction.

Philosopher Daniel Dennett, for example, expresses a common view when saying that this idea has helped us, if not to understand how the mind works, at least to confirm some of our mechanistic theories of mind.[6] He attacks the homunculus theories, but is enthusiastic about their potential when fortified with mechanistic principles. He evidently fails to see that this combination merely replaces one delusion with another. He believes that if the tasks performed by the little men become progressively simpler as we move to lower levels, this model constitutes real progress.[7] But the model constitutes progress only *if* the mind is a mechanistic phenomenon; if it is not, the only progress made is to replace the homunculus fallacy with the mechanistic fallacies.

Let us examine these delusions more closely. Since most mechanistic models of mind are now grounded on some combination of homunculus and mechanistic fallacies, by exposing these fallacies we can show the futility of *all* mechanistic theories of intelligence.

To explain or understand a phenomenon within the mechanistic doctrine means to represent it, through a series of reductions, as the result of some elementary processes that are simple enough to understand directly: bits of matter, simple facts, basic mathematical operations, small pieces of software,

[6] Daniel C. Dennett, *Brainstorms: Philosophical Essays on Mind and Psychology* (Cambridge, MA: MIT Press, 1981), pp. 119–123. [7] Ibid., p. 123.

and the like. Thus, we can hope to explain a phenomenon in this sense only if we accept *unquestioningly* that it can be represented as a simple hierarchical structure.

So, to explain the mind we must *assume* that it can be treated as a simple structure; that an intelligent act is the high-level manifestation of some simpler operations, which occur at a lower level; and that those operations are the result of even simpler ones, and so on, down to some operations that are simple enough to be accepted implicitly. Theories differ in the level of reduction claimed to be necessary; some are bold enough to continue the reduction down to the mind's "hardware" (the brain), but most theories in psychology, cognitive science, and artificial intelligence stop at some low level of the mind's "software" (the immaterial mental processes). No matter what level each theory considers low enough to count as explanation, though, to effect the reduction they must all treat the mind as a simple hierarchical structure: the lowest-level elements perform certain operations, the results become the elements that take part in the operations of the next higher level, and so on.

This structure, and all its elements and operations, are imaginary, of course. The mechanists agree that this is only a model, a way to account for the intelligent behaviour we observe from outside. But they insist that the model can be as successful in explaining the working of the mind as similar models are in explaining *physical* phenomena.

Philosophers like Ryle criticize the mechanistic theories of mind by showing that their reductionistic principles are untenable: attempting to explain mental acts hierarchically leads to an infinite regress, so the idea is fallacious. The mind mechanists, though, are not impressed by these objections, and argue in the traditional mechanistic manner: by bringing as evidence the very thing which they are supposed to prove.

Characteristic of all mechanistic thinking is that it starts by *assuming* that there exists a mechanistic model of the phenomenon in question: all phenomena can be explained, and "explanation" means a mechanistic model. The mechanists do not see the need to confirm first that a mechanistic model can exist at all; their task, they believe, is only to *discover* that model. As a result, they see nothing wrong in invoking the principles of reductionism and atomism as a justification of their confidence, when in reality it is precisely these principles that are in doubt. And so, all mechanistic theories end up begging the question. This is a consequence of identifying science, research, and explanation with mechanism, and forgetting that mechanism is only a convention, that it can explain some phenomena but not others.

Let us see how this question-begging logic is used to justify the mechanistic models of mind. The little men and their operations, the mechanists remind us, exist only in our imagination. Their only function is to help us explain the complex mental phenomena by replacing them with combinations of simpler phenomena. The hierarchical levels do not strictly exist: as soon as we explain the first level through the operations performed at the second, lower level, we can dispose of the first level altogether; that level is in fact nothing but the result of the operations of the lower level. Then we dispose of the second level in the same way, once we explain it through the operations of the third level. And we continue this process to lower and lower levels. The only real operations, and hence the only little men that we need to retain, are those at the lowest level, in the terminal elements of the hierarchy. But, because this level is very low, its operations are so simple that we can dispose of those little men too: we can explain the operations, for example, as the result of some trivial physiological functions. This logic – a perfect example of reductionism and atomism – appears impeccable: it seems that we can have our imaginary hierarchical levels and little men after all, and at the same time ground our theory on solid bits of organic matter whose existence no one can doubt.

In this argument, the hierarchical structure of operations in the mind is perceived to be similar to other structures of things within things. In a classification of animals, for example, we invent levels like dogs, species, and genera, but ultimately the only physical entities that really exist are the terminal elements – the animals themselves. The levels of classification exist only in our imagination; no physical entity *dogs* exists in addition to the dogs themselves. Similarly, in a structure like a city we have districts, streets, and buildings, the buildings have walls, and the walls are made up of bricks; but the only physical entities that really exist are the bricks. We can view the entire city as nothing but bricks combined in a hierarchical structure. The buildings and the city do not exist *in addition* to the bricks.

But the structure that best resembles the mind, say the mechanists, is the hierarchical structure of modules and subassemblies that make up a complicated machine – a computer, for instance. We build machines as parts within parts, and we describe their function as operations within operations. In the end, though, the only entities that actually exist are the basic components, the lowest level of the hierarchy. The subassemblies and the machine do not exist *in addition* to these components.

The mechanistic theories assume that the mind can be viewed just like these hierarchical structures. The only physical entities that actually exist are the smallest elements of the brain – perhaps the neurons. The higher levels, up to the mental acts we note as intelligence, are only combinations of states (or values) displayed by these elements. We have no idea how this structure works,

the mechanists admit, but there is no reason to doubt that it exists, that its levels and operations can be discovered. The only reason we haven't succeeded so far is that it is such a complicated structure. We notice, for example, that some intelligent acts – a process of decision making or problem solving, for instance – can be represented as a hierarchical structure of simpler and simpler acts. True, this structure describes only hypothetical mental acts, and we don't know how to continue it down to the level of physical entities; and true, it explains only the simpler decision-making or problem-solving situations. But this is just the beginning; we are evidently making progress, and one day we will understand all levels and operations, and explain all intelligent behaviour.

The mechanists *start* with the assumption that the mind works like a machine, and then necessarily conclude that it can be explained as we explain the working of machines. They propose models based on reductionistic and atomistic principles, so they are committing the fallacy of assuming what in fact needs to be proved. For, the mind may or may not work like a machine; and if it does not, if it is a phenomenon like so many other natural phenomena that cannot be explained mechanistically, then their theories are bound to fail. The search for a mechanistic model of mind is, thus, a mechanistic delusion.

4

The futility of mechanistic mind models becomes obvious if we remember that they could work only if our mental processes were indeed simple structures. But the brain has evolved to help us cope with our natural environment, which consists of interacting phenomena. It has developed, as a result, the capacity for interacting structures, and this permits us to acquire knowledge that demands complex structures. We already know that complex structures cannot be reduced to simple ones, so a mechanistic model of mind can only *approximate* our mental capabilities. Like all mechanistic approximations, this can be done through reification: by extracting *one* of the simple structures that make up the complex knowledge structure. This method may yield useful approximations for our simple skills, when one knowledge structure is dominant and its interactions with the others can be ignored. But it is useless for our advanced capabilities, when the interactions between structures are important.

Thus, the logic we examined previously (whereby some hypothetical levels are assumed in order to explain the high-level processes in terms of low-level ones, and at the same time dismissed in order to account for the fact that only

the low-level processes actually exist) works well for animal classifications, or cities, or machines. It works for these systems because we *purposely* design them as independent hierarchical structures. Specifically, we take into account *some* of the attributes possessed by their elements, and ignore the others; and we do this because we find the resulting approximation useful. In knowledge structures, though, *all* attributes are important, so this logic cannot explain the working of the mind. Mental acts cannot be usefully approximated with a neat structure of operations within operations, as an animal classification can with a structure of dogs within breeds, species, and genera; or as a city can with bricks within walls, buildings, streets, and districts; or as a machine can with components within modules and subassemblies.

Mechanistic mind models must reconcile the evident existence of trivial, low-level physiological processes, with the equally evident existence of complex intelligent acts. And they must accomplish this within the ideal of strict determinism; that is, with precise methods, diagrams, software, and the like. Given these requirements, the mind mechanists are compelled to commit the fallacy of reification, and all mind models end up alike: a hierarchical structure where some simple processes are the starting elements, and the complex intelligent acts are the top element. Some of these models may be very involved, but we must not let their sophistication deceive us. We already know that all deterministic theories – all models based on reductionism and atomism – are logically equivalent to simple hierarchical structures.

If the first fallacy (reification) is to represent the mind with a simple structure, the second fallacy (abstraction) is to use as starting elements relatively high-level entities: the mechanists stop their reductions of little men long before reaching the *basic* mental phenomena. No one knows what is the lowest level in mental phenomena, but even if we agree to take physiological entities like neurons as starting elements, the mechanists do not get anywhere near this level. Their starting elements are such entities as bits of knowledge, small behavioural acts, and simple logical operations; in other words, entities that are not at all basic mental phenomena. They defend their models by invoking the principles of reductionism and atomism, but at the same time they violate these principles and start from relatively high levels.

It is not surprising, therefore, that these models do not work. Mechanistic mind models fail to explain intelligence because their starting elements are not atomic and independent, as starting elements must be. In addition to the one structure that the mechanists recognize, these elements belong to many *other* structures, all formed from the elements that make up the *lower* levels – those levels that the mechanists ignore. The different values of the top element represent the various intelligent acts that the models must explain. And the models fail because they can account for only *some* of these values – those we

would see if the starting elements were indeed atomic and independent. They cannot account for the many alternatives resulting from interactions occurring at the lower levels.

❖

If all we want is to approximate some simple, specific mental capabilities – capabilities that can be isolated from the others – the search for mechanistic models may result in practical applications. But it is futile to search for mechanistic models capable of general intelligence, or for mechanistic theories that explain the working of the mind. This search is equivalent, in either case, to searching for a way to represent with simple structures the complex knowledge structures present in a mind – structures which can be acquired only by being a person, by having a body, by living in a society, and by engaging in the infinite variety of activities that make up a normal life. We cannot discover a model of mind because no such model can exist. The great capabilities of the mind arise largely from the knowledge structures that each one of us develops through personal experiences; that is, through exposure to a complex environment. A model of mind (as might be implemented with software, for example) cannot be exposed to such experiences, and therefore cannot attain the knowledge structures that make intelligence possible. Nor can we copy this kind of knowledge fully developed from a real mind into a model, any more than we can copy it from one mind into another.

When the mind mechanists seek to understand how the mind works, what exactly are they searching for? What do they expect to discover? Only a little reflection reveals that there is nothing to find; in a sense, we *already know* how the mind works. All we have to do is recognize that, when we perform an intelligent or skilful act, this act is not divisible into simpler acts. As Ryle points out again and again: "Overt intelligent performances are not clues to the workings of minds; they are those workings."[8]

We only run into trouble and keep searching for theories if we view the mind as a sort of machine instead of a natural phenomenon, and hence intelligence as a structure of operations within operations. It is this belief that tempts us to search for a model in the form of a simple structure. Consider other natural phenomena: The heavenly bodies simply move – they don't solve equations first. It is we who invented mathematical models to represent their motion. This phenomenon is in reality the complex result of many interacting processes, but we find the mathematical approximation useful. Biological phenomena too are the result of interacting processes, and they too manage to

[8] Ryle, *Concept of Mind*, p. 58.

occur without mathematics. It is we, again, who search for exact theories to explain them; but now we are less successful, because these phenomena are more complex than mechanical ones, and cannot be usefully approximated with simple structures. The human mind is the most complex biological phenomenon. Mental acts are the result of countless interacting processes, and no useful approximation with simple structures is possible.

Tacit Knowledge

1

The best way to observe the development of non-mechanistic knowledge is by studying the process of skill acquisition. Natural skills like face recognition and language processing we practise continuously from the time we are born, and for this reason they may be difficult to assess. Special skills, however, we acquire all the time: using a tool, driving a car, playing a game, programming a computer – we all learn to perform a great variety of tasks. So the process of skill acquisition – that is, approaching a new type of challenge as novices and then improving our performance through personal experience – is a phenomenon we can readily observe in ourselves and in others. Moreover, we can examine this phenomenon for a wide range of skills, from those we master in a few hours to those we improve continually over the years.

Hubert Dreyfus and Stuart Dreyfus, in their comparison of human skills to computer capabilities, distinguish five stages in the process of skill acquisition: novice, advanced beginner, competent, proficient, and expert.[1] The most noticeable change as one progresses from novice to expert is the shift from rule-following, machine-like behaviour to an intuitive and holistic response. The novice responds to a given situation as if it were made up of simpler parts, and as if it could be decomposed into these parts, because his limited experience only permits him to recognize isolated parts. He is incapable, therefore, of responding intelligently to situations that are not a neat structure of simpler situations. The expert, who was exposed to thousands of situations in the past, has developed knowledge that permits him to cope with new situations directly. His mind, it seems, extracted from the previous situations a kind of knowledge that he can apply in new situations, although he cannot express this knowledge in the form of rules or methods.

As novices acquiring a new skill, we learn to identify specific facts relevant

[1] Hubert L. Dreyfus and Stuart E. Dreyfus, *Mind over Machine: The Power of Human Intuition and Expertise in the Era of the Computer* (New York: Free Press, 1988), ch. 1.

to that skill; and we also learn various rules or methods for the appropriate actions. The facts we identify are *context-free* elements, and we learn them as isolated bits of knowledge; namely, knowledge that can be used directly, without reference to the overall context.[2] Thus, as novices, all we can do is break down each situation into recognizable context-free elements, and act according to the rules we learned. At this stage we behave just as a computer does when executing the instructions defined in a program. For example, a novice driver applies standard rules of speed and distance without further consideration of particular traffic conditions; a novice physician attempts to diagnose a disease by matching mechanically the symptoms he observes with the rules he learned; a novice programmer writes small and isolated bits of software by following standard methods, and with no need to consider the broader context of the whole application.

We progress from the novice stage after contending with a large number of real-world situations. This experience increases the number and complexity of rules and context-free elements we know, but it does more than that. Through repeated exposure to similar situations, we learn to identify situations that only *resemble* those that can be identified through rules. For example, we can recognize an object or a face even from an angle from which we never saw it before; as drivers, we can cope with traffic situations that are only similar to previously encountered ones; as programmers, we can use a certain software concept even for a problem that we face for the first time.

This new capability greatly extends the range of situations that we can cope with, by allowing us to recognize *context-dependent*, or *situational* elements – elements that vary from one situation to another.[3] No rules can help us to identify situations that contain such elements; so we learn to recognize them directly, holistically. But we cannot tell *how* we do it. We cannot tell in what ways the new situation is different from previous ones. We do not recognize it by following rules, nor by breaking it down into simpler, familiar elements. We simply recognize it.

Another quality we acquire on the way to expertise is *personal involvement* in our acts. As novices we act in a *detached* manner: because all we do is follow rules provided by others, we do not feel committed, nor responsible for the outcome of our acts. As experienced performers we are deeply involved, because our acts are determined by our own experiences.[4] The new knowledge both affects and is affected by our other knowledge. Having performed those acts many times and under many different conditions, they become associated in our mind with many experiences. Eventually, we no longer think of the new knowledge as a distinct skill; it becomes part of ourselves: "An expert's skill has

[2] Ibid., p. 21. [3] Ibid., p. 23. [4] Ibid., p. 26.

become so much a part of him that he need be no more aware of it than he is of his own body."[5]

Along with personal involvement, it is the ability to act *intuitively* that increasingly distinguishes our performance as we gain expertise. Intuition, in this sense, does not mean wild guessing or supernatural inspiration. It is a practical, valuable know-how: the ability to respond intelligently, without following rules or methods, in complex situations. And we can develop this ability only by being exposed to many similar situations. We need to develop intuitive knowledge because most situations in a complex environment are not identical to some predefined, or previously encountered, situations; nor are they made up of simpler situations. The only way to act intelligently in complex situations, therefore, is by responding intuitively to whole patterns, including patterns we face for the first time. We must recognize these patterns without decomposing them into some constituent elements.[6]

An important point, thus, is that expert performance in any field entails intuitive knowledge. To put it differently, when following rules and methods we do *not* display the highest performance that human beings are capable of.

The process of skill acquisition can be summarized by saying that it is "the progression *from* the analytic behavior of a detached subject, consciously decomposing his environment into recognizable elements, and following abstract rules, *to* involved skilled behavior based on holistic pairing of new situations with associated responses produced by successful experiences in similar situations."[7]

A novice pilot uses the controls to guide the airplane; an expert pilot simply experiences flying: the controls, and even the airplane, become an extension to his body. A novice air traffic controller watching the blips on a radar screen interprets them analytically; an expert controller sees, in effect, the airplanes themselves. A novice chess player moves the pieces according to learned rules and strategies; a master player identifies with the game positions: he no longer sees himself as player, but as participant in a world of opportunities, threats, strengths, and weaknesses.[8]

Interestingly, when we attain expertise in a professional skill we behave just as we do when exercising *natural* skills – skills like walking, talking, or recognizing objects. We are all experts in natural skills: when we walk, we don't consciously control our legs following laws of balance; when we talk, we don't put together words following rules of grammar; we simply *perform* these activities. We have the opportunity to observe beginner levels in natural skills when studying the performance of people who suffer from mental disorders. Individuals suffering from agnosia, for example, have difficulty

[5] Ibid., p. 30. [6] Ibid., pp. 28–29. [7] Ibid., p. 35. [8] Ibid., pp. 30–31.

recognizing familiar objects, and it is significant that their behaviour resembles the behaviour displayed by all of us when acquiring a new skill: they have to mentally break down each object into parts and features before recognizing it. These individuals fail to function adequately in everyday activities because they are restricted to analytical and logical behaviour. Without the capacity for intuitive and holistic thinking, they are forever novices in ordinary human skills.[9] Other interesting disorders are those of autistic and schizophrenic individuals (we will examine these cases in the next section).

Consider also the skill of visual perception. We learn to perceive the world around us while growing up, just as we learn to talk. But we cannot explain, for example, how we recognize objects – their shape, size, and distance – from the two-dimensional images they form on the retina; we do it effortlessly, intuitively. The only occasion we have to observe novice performance in this skill is when studying individuals who were born blind and begin to see late in life. These individuals must *learn* to distinguish objects visually, and they do it by analyzing them and memorizing their features. They behave just as we all do when acquiring a new skill.

The most important lesson from the study of skill acquisition is that a computer program cannot emulate the involved and intuitive performance of human beings. Since it is this kind of performance that we display when attaining expertise, we must conclude that computers, which are detached and analytic, cannot reach expert skill levels. Thus, in activities where a skill level lower than expertise is insufficient, human minds cannot be replaced with software. Programming, for instance, is such an activity; and this is why development systems – which are, in effect, attempts to replace programming expertise with software – fail to provide any measurable benefits.

2

A similar view was expressed by scientist and philosopher Michael Polanyi,[10] who called the kind of knowledge that can exist only in a mind – the knowledge that we can possess but cannot describe – *tacit* knowledge. (Tacit knowledge, then, is what I called complex, or non-mechanistic, knowledge.) Polanyi created a philosophy of knowledge that recognizes the participation of the subject in the process of knowing, as opposed to the traditional view that knowledge is always objective and impersonal. All knowledge, Polanyi says,

[9] Ibid., p. 64.

[10] Michael Polanyi, *Personal Knowledge: Towards a Post-Critical Philosophy*, corr. ed. (Chicago: University of Chicago Press, 1962).

has tacit aspects. Most importantly, tacit knowledge plays a part in our most rational pursuits, even in science. No science learning or practice, and no scientific discovery, would be possible if we were to rely exclusively on *specifiable* facts and methods.

Tacit knowledge manifests itself in that "*we can know more than we can tell.*"[11] We have the capacity to develop and use complex knowledge structures, but we cannot explain how we do it: "We know a person's face and can recognize him among a thousand, indeed among a million. Yet we usually cannot tell how we recognize a face we know. There are many other instances of the recognition of a characteristic appearance – some commonplace, others more technical – which have the same structure as the identification of a person. University students are taught in practical classes to identify cases of diseases and specimens of rocks, plants and animals. This is the training of perception that underlies the descriptive sciences. The knowledge which such training transmits cannot be put into words, nor even conveyed by pictures; it must rely on the pupil's capacity to recognize the characteristic features of a physiognomy and their configuration in the physiognomy.… What the pupil must discover by an effort of his own is something we could not tell him. And he knows it then in his turn but cannot tell it.… [This] exemplifies not only that the subsidiary elements of perception may be unspecifiable, but shows also that such tacit knowledge can be *discovered*, without our being able to identify what it is that we have come to know. This holds equally for the learning of skills: we learn to ride a bicycle without being able to tell in the end how we do it."[12]

Scientism, we saw in chapter 1, is the attempt to apply in the human sciences the methods of the exact sciences. Modern scientists believe that phenomena related to living things can be precisely explained, just like mechanical phenomena, through reductionism and atomism. Accordingly, they reject the need for tacit knowledge in disciplines like psychology, physiology, and biology. But many of the phenomena related to living things are unspecifiable: we cannot find a mathematical theory that explains the whole of a plant or animal. These phenomena consist of interacting processes, which must be observed simultaneously; so the only way to study them is through direct observation of the phenomena themselves.

Our study of living things, therefore, must depend on our capacity for tacit learning and knowing, and ultimately on personal experience. Exact

[11] Michael Polanyi, *The Tacit Dimension* (Gloucester, MA: Peter Smith, 1983), p. 4.

[12] Michael Polanyi, "The Logic of Tacit Inference," in *Knowing and Being: Essays by Michael Polanyi*, ed. Marjorie Grene (Chicago: University of Chicago Press, 1969), p. 142 – paper originally published in *Philosophy* 41 (1966): 1–18.

theories cannot replace the tacit knowledge developed in the mind of each individual scientist: "Morphology, physiology, animal psychology – they all deal with comprehensive entities. None of these entities can be mathematically defined, and the only way to know them is by comprehending the coherence of their parts.… Our knowledge of biotic phenomena contains a vast range of unspecifiable elements, and biology remains, in consequence, a descriptive science heavily relying on trained perception. It is immeasurably rich in things we know and cannot tell.… An attempt to de-personalize our knowledge of living beings would result, if strictly pursued, in an alienation that would render all observations on living things meaningless. Taken to its theoretical limits, it would dissolve the very conception of life and make it impossible to identify living beings."[13]

What is true of biology is also true of those disciplines concerned with human behaviour and human societies. The complex human phenomena studied by these disciplines are the result of interacting processes that take place in the minds of individual subjects, and between different minds; and these interactions, we saw, cannot be precisely specified. Consequently, a scientist may get to understand some of these phenomena after observing them for a long time, while being unable to describe his knowledge to others. It is the tacit aspects of his knowledge that he cannot describe – those aspects that reflect the interactions.

The theories that attempt to reduce mental processes to exact models – in behaviourism and cognitive science, for example – claim in effect that it is possible to explain precisely and completely all human knowledge. They are bound to fail, though, because they ignore the tacit, unspecifiable aspects of knowledge.

The scientists who advance these theories believe that, if they can *recognize* certain processes which occur in other minds, they should be able to represent these processes with exact models. But *they* can recognize mental processes containing tacit aspects because *they themselves* have a mind, which in its turn is capable of developing tacit knowledge; so they can create similar processes in their own minds.[14] When attempting to express their knowledge about the other minds in the form of mathematical or software models, they must leave behind the tacit aspects, which can only exist in their minds – and of which, moreover, they are not even aware. The models, then, will represent only the *mechanistic* aspects of their knowledge, and will provide a poor approximation of it: "The claim of cybernetics to generate thought

[13] Ibid., pp. 150–152.

[14] It is, in fact, for this reason that we can communicate at all with one another and get to share knowledge, values, and traditions – complex phenomena rich in tacit aspects – while the actual communication is limited to simple structures like symbols and sounds.

and feelings rests … on the assumption that mental processes consist in explicitly identifiable performances which, as such, would be reproducible by a computer. This assumption fails, because mental processes are recognized to a major extent tacitly, by dwelling in many particulars of behaviour that we cannot tell."[15]

Most significantly, tacit knowledge is essential even in the exact sciences. There are "unformalizable mental skills which we meet even in that citadel of exact science, that show-piece of strict objectivity, the classical theory of mechanics."[16] We can understand a mathematical theory only if we understand also the concepts underlying the theory, when and how to apply it, and so on. Being largely informal, this kind of knowledge cannot be represented with precise rules or methods. We cannot teach or learn it as we teach or learn the theory itself. We cannot reduce it to simpler elements, as we can the theory itself, leading ultimately to some basic atoms of knowledge. And yet, we must possess this kind of knowledge if we are to practise science at all. Scientists, evidently, acquire it and use it, although they cannot describe with precision what they acquire or how they do it. They do it simply by growing up in a particular society, by being part of a scientific community, by learning and practising within a certain cultural environment. On the whole, "the ideal of a strictly explicit knowledge is indeed self-contradictory; deprived of their tacit coefficients, all spoken words, all formulae, all maps and graphs, are strictly meaningless. An exact mathematical theory means nothing unless we recognize an inexact non-mathematical knowledge on which it bears and a person whose judgment upholds this bearing."[17]

Even though mathematics forms a perfect hierarchical structure, its fundamental principles – the terminal elements – are not atoms of absolute truth, but informal axioms and conventions. Concepts like *number* and *infinity*, for instance, can only be understood intuitively. The attempts to revolutionize mathematics by reducing it to logic, which gave rise to a large number of complicated theories in the first decades of the twentieth century, generally failed. Such attempts necessarily result in contradictions and circular arguments. For example, to avoid depending on our intuitive notion of what the number *one* means, the symbol "1" would be defined with a lengthy chain of expressions employing a formal symbolic language, but which depend in the end on an informal interpretation of words like "one." This is well understood today, but few scientists saw the futility of these projects at the time.

[15] Polanyi, "Tacit Inference," p. 152.

[16] Michael Polanyi, "The Unaccountable Element in Science," in *Knowing and Being*, ed. Grene, p. 106 – paper originally published in *Philosophy* 37 (1962): 1–14.

[17] Michael Polanyi, "Sense-Giving and Sense-Reading," in *Knowing and Being*, ed. Grene, p. 195 – paper originally published in *Philosophy* 42 (1967): 301–325.

In conclusion, "a mathematical theory can be constructed only by relying on *prior* tacit knowing and can function as a theory only *within* an act of tacit knowing, which consists in our attending *from* it to the previously established experience on which it bears."[18] And if mathematics depends on personal, informal knowledge, so must all science. Ultimately, all types of knowledge contain unspecifiable aspects: "*All knowledge falls into one of these two classes: it is either tacit or rooted in tacit knowledge.*"[19]

3

Let us see how the two kinds of skills we have been examining can be explained if we interpret the phenomenon of mind as the processing of simple and complex structures. Skills described as *knowing that*, and requiring only isolated, context-free bits of knowledge, arise from processing *simple* structures in the mind; so they reflect *mechanistic* capabilities. Skills described as *knowing how*, and requiring intuitive, tacit knowledge, arise from processing *complex* structures; so they reflect *non-mechanistic* capabilities. (See pp. 146–147, 155–156). The phenomena we are exposed to may be simple or complex. But the mind has the capacity to create replicas of both kinds of phenomena, so we can develop both kinds of knowledge.

The mind treats all phenomena the same way. Its function is to discover the structures that best depict the phenomena it is exposed to; in particular, the relations between the elements of these structures. For simple phenomena, it needs to discover only the relations *within* structures; but for complex phenomena, it discovers also those *between* structures. The mind, of course, doesn't "know" that some structures are simple and others complex; it discovers all relations the same way. But when the relations are between elements belonging to different structures, the result is non-mechanistic knowledge.

The relations cannot be acquired directly through the senses, but must be developed by the mind; and the mind can only develop them through repeated exposure to a particular phenomenon. The mind always starts by discovering the relations within the individual structures (because stronger or more common, perhaps); but after further exposure it also discovers the relations between elements of different structures.

This is indeed what we observe in the process of learning, when we acquire new knowledge and skills. At first, as novices, we develop the ability to deal with *isolated aspects* of the new knowledge – the simple structures. After gaining further experience, we can deal with larger parts – the complex

[18] Polanyi, *Tacit Dimension*, p. 21.

[19] Polanyi, "Sense-Giving," p. 195.

structures. To cope with situations where the interactions between structures are weak, we need to deal only with individual structures, so mechanistic knowledge suffices. But when the interactions are important, we need non-mechanistic knowledge.

❖

I have stated that the mind can process both simple and complex structures, but in reality no knowledge structure can exist in the mind on its own, isolated from the others. Recall our conclusion in chapter 1: no phenomenon is totally isolated, so all phenomena surrounding us are complex structures. Since the brain has evolved to help us cope with our environment, the knowledge we develop in the mind must mirror the actual phenomena; so our knowledge too comprises only complex structures.

Even a basic concept like *one plus one is two*, which can be easily represented as a simple structure, cannot exist in the mind on its own – the way it might exist in a computer, for instance. It is precisely the fact that this concept can be related to others that allows the mind to combine it with existing knowledge structures. Thus, the only way to acquire and use this new concept is by relating it to concepts like *one*, *number*, and *addition*, by relating *these* concepts to the objects we add and to the effects of addition, and by relating then *these* concepts to others yet.

But, just as some phenomena in the world are only weakly related to others and can be usefully approximated with simple structures, so the knowledge structures in the mind form mutual links that are stronger or weaker, depending on the phenomena they mirror, and can sometimes be usefully approximated with simple structures. This is perhaps what leads the mind mechanists to believe that *all* knowledge can be represented with simple structures.

It is the interactions between structures that transform a group of isolated simple structures into a complex structure. This may be hard to understand, because it is precisely the interactions that cannot be completely described, as can the structures themselves. It is these interactions that define the complex phenomena, and hence also the knowledge structures which mirror these phenomena. It is not too much to say that in complex knowledge structures the interactions *are* the knowledge.

Consider, for example, the word "beauty," the abstract concept of beauty, and the way we recognize beauty. As children growing up in a particular culture we learn all these things, but we don't know how this happens. Clearly, these notions taken separately are meaningless. Can we learn how to recognize beautiful things without knowing that beauty can exist? Can we learn that

beauty exists without knowing that we can refer to concepts like beauty with words? Can we learn the meaning of the word "beauty" without knowing already how to recognize beautiful things? The circle is endless. The way we learn about beauty as children is by learning all these notions at the same time. We usually have no difficulty learning about beauty, but if we had to write instructions for aliens, explaining how to recognize beautiful things, we wouldn't know how to do it. Nor can we program a computer to appreciate beauty as *we* do, although we can store the word "beauty" in its memory, and we can implement with software any facts or rules related to beauty. The reason for these difficulties is that most of our knowledge of beauty is contained, not in facts or rules, not in the isolated knowledge structures related to beauty, but in the *interactions* between these structures – and also in the interactions with *other* knowledge structures, for the concept of beauty is related to other concepts.

Human knowledge inheres mostly in the interactions between concepts, rather than in isolated facts and rules; and, unlike facts and rules, these interactions can seldom be specified completely and precisely. Complex knowledge, thus, can exist only in the phenomena themselves and in the mind. The only way to learn about beauty is by being exposed to the structures related to beauty and letting the mind discover the interactions. And we would reach the same conclusion if we analyzed any other concept. It is only when we forget how much of our knowledge is based on interacting structures that we try to imitate intelligence with deterministic models – which, no matter how sophisticated, can only represent simple structures.

The best way to appreciate the importance of these interactions is by studying the process of learning – the acquisition of knowledge. It is on such occasions that we notice how often the knowledge that develops in the mind is, in fact, several kinds of knowledge that are acquired together. The best-known case is that of language. All studies of linguistic skills conclude that language is inseparable from thinking and knowledge. As children, we learn to speak at the same time we learn everything else, and it is generally agreed that intelligence and verbal ability grow together. We must have the capacity to learn the word for beauty and how to recognize beauty at the same time, otherwise we could learn neither.

As already noted, acquiring natural skills early in life is similar to acquiring professional and general skills as adults, but this latter process we can more readily observe and study. Thus, a process similar to the acquisition of linguistic knowledge together with other knowledge takes place whenever we

have to communicate in order to develop a complex skill. Polanyi takes the interpretation of chest X-rays as example.[20] A medical student learns to diagnose pulmonary diseases from X-ray pictures by observing diverse cases while listening to an expert radiologist comment on the various features in technical language. Since both the pictures and the verbal descriptions are unintelligible to a novice, they must be learned together.

At first, the student is completely puzzled, for all he can see is light and dark patches; he cannot even distinguish the lungs, let alone the disease. He listens to the expert's explanations, but he can see in the pictures nothing of what the expert is describing. After several weeks, however, after studying in this fashion many different cases, the X-ray pictures slowly turn into a rich panorama of significant details: physiological variations and signs of disease. And then the student also begins to understand the expert's explanations. Although he still sees only a fraction of what an expert can see, both the pictures and the comments are beginning to make sense: "Thus, at the very moment when he has learned the language of pulmonary radiology, the student will also have learned to understand pulmonary radiograms. The two can only happen together. Both halves of the problem set to us by an unintelligible text, referring to an unintelligible subject, jointly guide our efforts to solve them, and they are solved eventually together by *discovering* a conception which comprises a *joint understanding* of both the words and the things."[21]

The process whereby an expert interprets X-ray pictures cannot be separated into two distinct structures – the technical description of X-ray pictures, and the identification of lung physiology and disease. Nor is the description a structure *within* the identification, or the identification a structure *within* the description. The process of interpretation and diagnosis is the *interaction* of these structures (and probably other structures too), and this complex knowledge structure can only exist in a mind. The experienced radiologist cannot convey his expertise to the student directly, because he can only communicate with the student through the simple structures of images and descriptions. The knowledge, therefore, must develop in the mind of each student through personal experience; it consists in the complex structure that is the *combination* of those structures.

I want to conclude this discussion with another example of tacit knowledge: the unusual skill of chick sexing – determining the sex of newly born chicks.

[20] Polanyi, *Personal Knowledge*, p. 101.

[21] Ibid. (italics added).

The reason it is important to sort chicks as early as possible is that egg producers need only pullets, so it is uneconomical to grow both pullets and cockerels. The differences at birth are very subtle, though, so the sorting accuracy of an unskilled person is not much better than 50 percent – the equivalent of guessing. Some recognition rules *have* been developed, based on anatomical details; but because each detail individually is unreliable, the best accuracy that a worker can achieve by following these rules is about 70 percent. Speed in this work is as important as accuracy, and sorters who merely follow rules cannot improve their speed either.

The Japanese, however, discovered in the 1920s that experienced sorters could achieve much better results by relying entirely on intuition. After sorting more than a million chicks, these workers had learned to distinguish chicks almost instantly and very accurately. But they were doing it through a combination of visual and tactile clues of which they themselves were unaware; that is, they could not explain what it was that they had learned to recognize. Their expertise, therefore, could not benefit other workers. So the way a novice could acquire this skill remained unchanged: sorting a large number of chicks and developing the same intuition. With intuitive sorting, an expert can attain rates exceeding one thousand chicks per hour, with an accuracy of 98 percent.

Following its success in Japan, the method of intuitive sorting was adopted all over the world. It is still being practised, but it is becoming less and less important, as chick hatchers now prefer breeds in which the chicks can be distinguished by their feathers (so they can be easily sorted even by unskilled workers).

Much studied by psychologists, this skill is an excellent example of tacit knowledge. Although the recognition process remains a mystery, it is probably similar in nature to other skills that involve intuitive pattern recognition – recognizing faces, for instance (see pp. 108–109). Since an expert sorter needs only about three seconds (which includes the handling operations), he is presumably distinguishing the chick immediately, the way we recognize a familiar face. So this *must* be an intuitive act; the sorter is not merely applying some recognition rules quickly.

Also, the learning process itself must be intuitive. There isn't much that an experienced worker can teach an apprentice, apart from confirming whether he was right or wrong about a particular chick. And this, indeed, is how workers are trained. Each apprentice, therefore, must discover on his own, unconsciously, the various patterns; and he does it simply by examining a large number of chicks. So the learning process, too, is similar to face recognition: we learn to recognize faces simply by seeing them many times.

Let us say that there are hundreds of basic variations in chick anatomy, and

perhaps even some variations within the basic ones. Even so, a worker who examines millions of chicks will come across each variation many times, and will eventually develop in his mind a set of familiar patterns. He will learn to distinguish these variations, just as we all learn to recognize hundreds of faces after seeing them repeatedly. And he cannot tell what it is that he learned to recognize, any more than we can tell how we recognize those faces. Perhaps the set of familiar patterns in his mind includes only the basic variations, and he recognizes the other patterns as variations of the basic ones – the way we recognize a familiar face from different angles.

The process of face recognition, we saw, is a complex structure: it entails several interacting recognition processes (simple structures). Similarly, we can think of several structures that make up the chick anatomy, so the process of distinguishing a chick must include several structures; for example, recognizing relations between the sizes, shapes, or proportions of some chick parts, or relations between these relations, and so on. These structures interact, because they exist at the same time and share their elements – the chick parts. This makes the whole recognition process a complex structure.

To follow recognition rules, a worker needs only mechanistic knowledge. This permits him to recognize *separately* the individual structures, and his performance is restricted to those patterns obvious enough to identify in this manner; that is, those for which the links between structures can be ignored. When attaining expertise, the worker's knowledge includes the *interactions* between structures. His performance improves because this complex knowledge permits him to recognize additional patterns – those for which the links between structures are important.

4

The study of unusual skills like chick sexing demonstrates that seemingly mysterious capabilities can be accounted for through the concept of tacit knowledge. These skills are striking because they are unusual – we would find similar capabilities if we studied *normal* activities. Normal skills are more difficult to observe, though, because we practise them all the time and take them for granted. We acquire them naturally and effortlessly while growing up, and this is why we are liable to forget that they are in fact important and impressive skills.

It is when artificial intelligence researchers fail to simulate with software what are for us simple, everyday tasks, that we are reminded how important is our *unspecifiable* knowledge. Thus, an unusual skill that can be acquired under controlled conditions stands out and provides a striking example of tacit

knowledge, but we must bear in mind that it merely brings to attention what are in reality *normal* human capabilities.

The fact that we can acquire unusual skills also casts doubt on the theories that postulate *specialized* higher mental functions. Although we cannot doubt that all mental acts are based ultimately on the physiological characteristics of the brain, and that there exist specialized functions at the *lower* cognitive levels, we have no reason to assume that we need a great variety of mental processes to produce intelligent behaviour. Since we can acquire so many skills (skills as diverse as distinguishing chicks, interpreting X-rays, and programming computers), it is unlikely that we use specialized mental functions for any one of them, and more likely that we possess some *generic* mental capabilities, which take part in *all* mental acts. If we use generic capabilities to acquire one skill, then why not others? If we use them for chick recognition, why not also for face recognition, which seems so similar? And why not also for linguistic skills, and for all other intelligent acts?

No one would seriously suggest that we possess an innate chick sexing faculty, because then we would have to conclude that we possess a separate faculty for every other skill that we can acquire. Yet this is precisely what many theories of mind suggest. Chomskyan linguistics, for example, confidently assumes that we possess an innate *language* faculty. But if we can perform remarkable mental feats in diverse areas without specialized mental functions, why do we need specialized functions for language? Our linguistic capabilities are so closely linked to other mental capabilities that it is for language, particularly, that we do *not* need to postulate specialized functions. We certainly have a faculty for language; but this is the same faculty that takes part in visual perception, in understanding music, in driving cars, in programming computers, and in every other intelligent act. (See pp. 268–271.)

The mind mechanists cannot deny the role played by tacit knowledge in intelligent behaviour, but they refuse to accept it as an irreducible mental process. All their attempts to reduce tacit knowledge to simpler knowledge, however, have failed. Our non-mechanistic model of mind is less ambitious: if there is so much evidence both for the existence of tacit knowledge and for the existence of mental capabilities common to all forms of intelligence, we are content with a model that incorporates these two facts. The mechanistic models fail because they attempt to explain mental acts by treating them as isolated processes: simple hierarchical structures of operations, relations, and bits of knowledge. Our model represents the working of the mind as one complex structure – countless interacting structures. In practice, we can often approximate it with *separate* complex structures; we can treat one skill, for example, as several interacting knowledge structures, but without relating it to all the structures that make up the mind. And sometimes we can even

approximate mental acts with isolated *simple* structures; in this reduced version, the model works as a mechanistic one.

Tacit knowledge is the knowledge that can develop only in a mind, and is therefore unspecifiable; it is knowledge that manifests itself in that "we can know more than we can tell." In our model, this knowledge is embodied in the *interactions* between structures. We can acquire and express knowledge only if it is specifiable, only as simple structures (in the form of symbols or sounds, for instance). Our communication with the world is limited, therefore, to mechanistic processes. The interactions that develop in the mind are a reflection of the interactions that exist in the complex phenomena we are exposed to. When we learn something, the mind creates an approximate replica of the phenomenon by discovering the interactions; later, when we perform an intelligent act, the mind uses the interactions found in that replica. The interactions can exist only in the phenomenon itself and in the mind; they cannot be transmitted directly between the mind and the environment through our senses or bodies.

So what we notice as intuition is the external manifestation of tacit knowledge, an expression of the interactions. It is impossible to explain intuition by taking into consideration only the simple structures we detect in our environment. This is why we cannot explain, for example, how we recognize faces, or distinguish chicks, or communicate with language. But if we describe intuitive knowledge as a complex structure, the interactions provide the critical pieces needed to account for the intelligent acts. The model, however, cannot explain the interactions themselves: complex structures cannot be reduced to a precise and complete description of their constituent simple structures, plus a precise and complete description of their interactions.

This is an informal model. But we don't need a more precise one, because, unlike the mechanistic models, ours is not intended to *explain* intelligence. Its purpose, on the contrary, is to show that it is *impossible* to explain mental acts with exact theories, as we explain the working of machines. Its purpose, in other words, is to show that the principles of reductionism and atomism alone cannot account for intelligence. What this model claims, in effect, is that there can be no deterministic model of mind – no model that can be implemented with tools like mathematics or software. It claims that the concept of complex structures is the only way to account for the phenomenon of mind, and complex structures can exist only as natural phenomena; we can describe them, but we cannot implement them with mechanistic means.

Creativity

1

Unusual mental capabilities like technical talent, poetic ingenuity, and appreciating art or humour do not appear to have much in common. They are closely related, however, insofar as they can all be described with a model of mind that takes into account the interaction of knowledge structures. Moreover, when studying these unusual capabilities we recognize that they are caused, in fact, by the same mental processes that give rise to *normal* performance. What we will learn from their study, therefore, is this: if we cannot explain or emulate *extreme* capabilities, we cannot hope to explain or emulate *normal* intelligence either. When observed from the lower levels of mental processes, the performance of a genius is not very different from that of an average person. The mind always works the same way, and the difference between the extremes of mental capabilities is smaller than it appears.

The mind mechanists keep inventing models that emulate trivial mental acts, in the belief that these models will be improved later to handle complex tasks. Our study, however, will challenge this principle. A mechanistic model may succeed in emulating an isolated mental act, but without emulating the *capabilities* of the mind. (A calculator, for instance, can add numbers correctly without understanding the meaning of addition as *we* do.) Thus, there is no reason why a model that emulates simple mental acts, but working differently from a human mind, should reach even an average level of intelligence later, when improved.

❖

In our discussion of intelligence and creativity we encounter frequently the term *context*. We find, for example, that intelligent behaviour must take the context into account, or that the correct interpretation of an item depends on the context. While the notion of context – in the sense of situation, or conditions, or circumstances – is well understood, I want to take a moment to show how it fits within the model of complex structures. Then, we will be able to refer to contexts in the study of intelligence and creativity without having to interpret them as structures each time.

Recall the discussion of complex structures in chapter 1 (pp. 96–98). The entities that function as elements in structures have a large number of attributes; and each attribute relates them in a particular way to other entities, thereby forming one simple structure. A physical object, for example, has a shape, a colour, a weight, a date of creation, an owner, and so forth; it belongs,

therefore, to several structures of objects at the same time – a different structure for each attribute. And these structures interact, because they share their elements – those objects.

A context can be seen as a set of attributes, not unlike the regular attributes of an entity, but which exist *in addition* to the regular ones. The context includes attributes related to time and space (day or night, summer or winter, indoor or outdoor, a specific room or building), to a social setting (a specific town, a public place, the company of friends), to a personal situation (being in a good mood, or sick, or frightened), and so on. Thus, the context attributes are generally *more variable* than the regular ones. It is obvious that, as is the case with the regular attributes, the number of attributes that determine a context can be very large. It is also obvious that an entity *must* have some attributes that determine a context, in addition to its regular attributes, which remain the same from one context to the next. Just as its regular attributes place the entity in various structures and relate it to other entities that have those attributes, the context attributes place it in other structures yet.

This model explains why the context plays such an important part in the interpretation of a given situation. We already know that any entity is an element of a complex structure. But now we see that this structure includes, in addition to the structures formed by its regular attributes, which are constant, the structures formed by the *context* attributes, which are changeable. This makes the entity a somewhat different entity in different contexts, just as a change in its regular attributes (its shape or colour, for instance) would make it a different entity. To interpret correctly an object, or an event, or a process, its current context is as important as are its physical attributes, or its uses, or its effects. Because of the changing nature of the context attributes, we need greater knowledge to cope with the complex structure when it includes a context, than we do when the context is unimportant, or when something always occurs in the same context. The reason, obviously, is that we have to deal with more structures.

Recall also the observation we made earlier on the acquisition of skills. As novices, we start by recognizing only the *context-free* aspects of a new domain – the structures reflecting attributes that remain the same in all contexts. As we gain expertise, however, we learn to recognize also the *context-dependent* aspects of that domain – those structures that include the context attributes. (See p. 156.) The process of skill acquisition entails, therefore, an increase in our ability to recognize complex structures *and* their variations in different contexts; in other words, the ability to cope with structures that are more and more complex.

2

When we think of creativity, we usually think of outstanding accomplishments in science, art, music, poetry, or literature. Creativity in these fields, however, is only one aspect of a wide range of intelligent performance – a range that includes normal, everyday activities. We may not realize it, but we have to be creative just to behave normally. When the mechanistic models of mind fail to display what is called common-sense knowledge and understanding, it is human creativity that they cannot reproduce.

Much has been written on creativity, and a frequent observation is that a creative act produces something totally new, yet relatively simple. What is striking is the novelty, not the complexity, of the result. It is as if the mind searched through a large number of combinations of pieces of knowledge and retrieved a combination that did not exist before. The combination is not random, but especially interesting, or useful, or attractive.

Here is how Jerome Bruner expresses this view: "An act that produces *effective surprise* – this I shall take as the hallmark of a creative enterprise.... Surprise is not easily defined. It is the unexpected that strikes one with wonder or astonishment."[1] Whether the creation consists in a scientific discovery, a work of art, a poem, or a solution to a mundane problem, when we try to explain the new concept we find that it can always be described as a combination of *existing* concepts. On analysis, there are no new facts in the new concept. If we want to describe it as new knowledge, the knowledge is in the combination, not in the individual facts: "I would propose that all of the forms of effective surprise grow out of combinatorial activity – a placing of things in new perspectives."[2] In literature, a great story may be nothing more than novel combinations of human qualities and situations. A poem may be largely new metaphors, words and meanings combined in original and imaginative ways. A technological discovery may be simply a novel combination of processes and applications.

Henri Poincaré,[3] who studied the process of discovery in mathematics, observes that the discovery of new laws is like searching through many combinations of known facts and principles until the right combination is found. The search and selection are performed by the unconscious mind, although they must be preceded and followed by conscious work. But the mind

[1] Jerome S. Bruner, *On Knowing: Essays for the Left Hand*, rev. ed. (Cambridge, MA: Harvard University Press, 1979), p. 18. [2] Ibid., p. 20.

[3] Henri Poincaré, *Science and Method* (New York: Dover, 1952).

does not work like a machine: "It is not merely a question of applying certain rules, of manufacturing as many combinations as possible according to certain fixed laws. The combinations so obtained would be extremely numerous, useless, and encumbering. The real work of the discoverer consists in choosing between these combinations with a view to eliminating those that are useless, or rather not giving himself the trouble of making them at all. The rules which must guide this choice are extremely subtle and delicate, and it is practically impossible to state them in precise language; they must be felt rather than formulated."[4]

All we can tell is that the combinations generated in the mind "are those which, directly or indirectly, most deeply affect our sensibility."[5] Thus, they reflect the knowledge already present in the mind: a mathematician is sensible to combinations that may result in a new mathematical law, while a poet is sensible to combinations that may result in a new metaphor. Discovery is an act of selection – the selection of facts "which reveal unsuspected relations between other facts, long since known, but wrongly believed to be unrelated to each other.... Among the combinations we choose, the most fruitful are often those which are formed of elements borrowed from widely separated domains."[6]

We can summarize these remarks by saying that discovery is the creation of links between knowledge structures that existed previously in the mind but were unrelated. The mind does not *generate* combinations of concepts, of course, and does not *select* combinations, although this may be a useful way of picturing the process of discovery. The combinations are the interactions between structures, which always exist. As we saw earlier, all knowledge structures in the mind are connected as part of one complex structure, but most links are very weak. So the mind can be viewed as a combination of many *separate* complex structures, formed where the links are stronger. The selection that is the act of discovery occurs when the links *between* some structures strengthen; and this typically follows a period of intensive mental activity involving those structures. At that point, a new complex structure is formed.

Bruner notes a similar process in artistic creativity. Works of art like paintings and sculptures achieve their effect by connecting two disparate mental structures: themes or contexts that normally evoke entirely different, even contradictory, feelings. A great work of art manages to accomplish this with economy, using a "compact image or symbol that, by its genius, travels great distances to connect ostensible disparities."[7] One theme, for instance, may be rooted in pragmatic, common-sense knowledge, while the other may

[4] Ibid., p. 57. [5] Ibid., p. 58. [6] Ibid., p. 51.
[7] Bruner, *On Knowing*, p. 65.

be a fantasy; the work of art creates then in our mind a new experience, which is the fusion of the two existing ones: "An image is created connecting things that were previously separate in experience, an image that bridges rationality and impulse."[8]

But it is not enough that the *artist* link those structures; we, the beholders, must participate, through individual effort. The artist can communicate with us only through separate images (i.e., simple structures). We must *discover* the links between these images, and the act of discovery constitutes the artistic experience. The difference between great and mediocre art is that great art makes it easy for us to discover the links: "Where art achieves its genius is in providing an image or a symbol whereby the fusion can be comprehended and bound."[9] Comprehending art is a creative act: we must *relive*, as best we can, the original creative act of the artist. And the only way to do this is through a process similar to the process of learning we examined in the previous section: we expose ourselves to the phenomenon and let our mind discover the links between structures, thus creating a new knowledge structure, similar to the one that existed in the artist's mind.

We cannot discuss creativity without mentioning the process of metaphoric thinking. Metaphors are phrases employed in a figurative rather than a literal sense. Metaphors, thus, attempt to connect two knowledge structures on the basis of a relation that is normally very weak, even illogical. Some examples: "time flies" (can time pass faster than normal?), "warm colour" (can colours have a temperature?), "flood of tears" (can a few drops cause an inundation?).

Clearly, then, metaphoric thinking – at work both when inventing and when comprehending metaphors – depends on our capacity to connect knowledge structures that have little in common. But, as we have already seen, this is the mental capability we display when acting creatively. Comprehending metaphors, as much as inventing them, is an act of creativity, of discovery. The metaphor does not work by *connecting* the structures in our mind, but by inviting us to *discover* the connection. Poetry, for instance, is largely metaphoric language, and what makes it enjoyable is the opportunity it affords us to discover new connections. Were these connections *explained*, rather than expressed through metaphors, reading poems would constitute an entirely different experience.

Bruner suggests that the metaphoric mode of thinking is a creative act common to art and science, so scientific discovery and artistic creation have a lot in common.[10] Now we see, though, that it is more accurate to say that scientific discovery, artistic creation, and metaphoric expression are all rooted in a fundamental mental process: combining knowledge structures.

[8] Ibid., p. 62. [9] Ibid., p. 72. [10] Ibid., pp. 65–66.

❖

Arthur Koestler[11] is another thinker who found that all forms of creativity derive from the connection of previously independent mental structures. He coined the term *bisociation* for this process, distinguishing it from what takes place within *individual* structures, where only previously established facts, rules, and relations are involved: "The term 'bisociation' is meant to point to the independent, autonomous character of the matrices which are brought into contact in the creative act, whereas associative thought operates among members of a single pre-existing matrix."[12] Koestler uses various terms to convey the idea of mental structures: frames of reference, associative contexts, types of logic, codes of behaviour, universes of discourse, and matrices of thought.[13] He shows how such diverse acts as scientific discovery, aesthetic experience, poetic expression, and understanding humour can be explained as different external manifestations of the same basic mental process. There is a similarity, for instance, "between the scientist seeing an analogy where nobody saw one before, and the poet's discovery of an original metaphor or simile.… In the scientist's [discovery] process two previously unconnected frames of reference are made to intersect, but the same description may be applied to the poet's … discovery of a felicitous poetic comparison."[14]

Koestler includes humour in the range of creative acts, since understanding comical situations – jokes, satire, irony, wit, puns – is in effect an act of discovery.[15] Like the other types of creativity, humour is due to new connections formed between existing mental structures. In scientific discovery, we saw, two previously independent structures are combined and become one. The same thing happens in art and poetry, but here the relation between the two structures remains tenuous: they represent opposing themes, like reality and fantasy, so we can only *imagine* the new combined structure. In the case of humour, the two structures are totally incompatible; they represent different contexts, so they cannot be logically combined at all.

The pattern underlying all types of humour is this: We are exposed first to one structure, which creates a certain context and certain expectations; then, we are exposed to the second structure, which creates an entirely different context. Our mind attempts to combine the two, but this is impossible. Whereas mentally we can quickly shift from one context to the other, emotionally we cannot, and the tension created by this conflict is released through the reflex action of laughter or smiling. The structures are perfectly logical on their own, but they represent incompatible contexts. Individually, they evoke

[11] Arthur Koestler, *The Act of Creation* (New York: Macmillan, 1964).
[12] Ibid., p. 656. [13] Ibid., p. 38. [14] Ibid., p. 320. [15] Ibid., chs. I–IV.

simple, logical, well-known situations; it is their unrelatedness that causes the tension.

Almost any pair of unrelated contexts can be used to produce humour. The simplest kind of humour is created by using words or phrases with double meaning (for example, one literal and the other figurative, or one in common usage and the other in specialized usage). Comic situations can be created with coincidences, cases of mistaken identity, or confusion of time and occasion – all causing the collision of incompatible contexts. The opposition between man and machine, the human and the automaton, has a long tradition in humour (for example, Charlie Chaplin's classic scenes in *Modern Times*). Mind and matter form another successful opposition (for example, the absent-minded genius bumping into a lamppost – the great mind defeated by a simple object).

In jokes, the opposition between two contexts may be more subtle, but it is still the source of their humour. Here is an example: In a store, the check-out line is long and slow, and a man at the end of the line loses his patience. "I am going to complain to the manager," he tells his neighbour, and leaves. A moment later he returns and takes his place back in line. "What happened?" inquires his neighbour. "The line of people waiting to complain to the manager," he explains, "is even longer." Why do we find this amusing? We are asked to combine two incompatible contexts: one is the situation of a long waiting line and the expectation of a resolution – familiar and logical; the other is the situation of a person choosing the shorter of two waiting lines – also familiar and logical, but the two contexts clash.

Here is another example: A man in a restaurant tells the waiter, "Please bring me a coffee without cream"; and the waiter replies, "Sorry, sir, we have no cream. May I bring you instead a coffee without milk?" One context here is the familiar situation of a person preferring black coffee, and the other is the equally familiar situation of a waiter suggesting a substitute for an item. The two situations are logical individually, but their combination is senseless.

3

We have identified the combination of knowledge structures as the mental process that leads to creativity. And, while easy to observe in the work of exceptional minds – in science, art, or poetry – the same process occurs in lesser mental acts: in the acts performed by any one of us when appreciating art or humour, when using words in a figurative sense, and, in the end, when engaged in any intelligent activity. Another way to confirm the importance of interacting knowledge structures is by noting what happens when there is a deficiency in processing these structures, in individuals suffering from mental

disorders. We saw that researchers who study creativity reach a mind model based on interacting structures; it should not surprise us, therefore, that researchers who study mental disorders also reach such a model.

The term *schizophrenia* (literally, split mind), introduced in 1908 by Eugen Bleuler, expresses his observation that patients behave as if their thought processes were separated and could not be combined properly. Bleuler suggested that the mental processes of schizophrenics are fundamentally similar to those of normal persons; these processes differ only under certain conditions, and then the difference is nothing more than a failure to respond appropriately to a particular situation. The problem is not that the schizophrenic is incapable of certain thought patterns, but that he is unable to match his thoughts to the current context. If the mind works by combining structures, then schizophrenia can be described as a disorder that prevents the mind from connecting structures as a normal mind does: it fails to connect a particular structure to others, or connects it to the wrong ones. This results in behaviour that appears strange when judged by accepted standards. In extreme cases, the schizophrenic cannot function adequately in society.

The failure of schizophrenics to connect mental structures is described by psychiatrists with such terms as "disconnectedness," "thought blocking," "concreteness," and "overinclusive thinking."[16] The term "reification" is sometimes used – the term we adopted for the fallacy of ignoring the interactions in a complex structure. The following examples show that seemingly unrelated disorders can be traced to the same deficiency if we use a mind model based on interacting structures.[17]

The world of the schizophrenic is fragmented. Patients report that they are flooded with thoughts and sensations, and cannot put the pieces together into a coherent pattern. They notice the colour or the shape of an object, rather than simply *seeing* the object. They notice in turn the eyes or nose or hair of a person, rather than the face as a whole. They notice isolated words when reading a book or in a conversation, and fail to grasp the meaning of the whole sentence. They are distracted by small details – details that normal individuals integrate tacitly into whole patterns; they are disturbed by irrelevant background sounds, for instance, or they become acutely aware of mild sensations or parts of their own body. Many schizophrenics find it impossible to watch television, because they cannot connect the image and sound into a coherent whole – they have to concentrate on one or the other; also, they are confused by images or sounds that are too complex or change too fast, since they can only

[16] E. Fuller Torrey, *Surviving Schizophrenia: A Family Manual* (New York: Harper and Row, 1985), p. 17; Theodore Lidz, *The Origin and Treatment of Schizophrenic Disorders* (Madison, CT: International Universities Press, 1990), p. 54.

[17] The examples are from Torrey, *Surviving Schizophrenia*, pp. 8–22 passim.

assimilate them a bit at a time. They fail to put together all the sensory data in a typical social situation: they notice each movement and word separately, so the people's behaviour appears strange to them. An action as simple as getting a drink of water can become a problem, as the schizophrenic must concentrate on each step and detail – hold cup, turn tap, fill cup, drink water. Delusions and hallucinations, the best-known symptoms of schizophrenia, can be explained as the failure to integrate mental structures and sensations correctly; they are "a direct outgrowth of overacuteness of the senses and the brain's inability to synthesize and respond appropriately to stimuli."[18]

At each moment, our mind processes many external stimuli and internal thought patterns, but normally it has no difficulty combining these structures into the most appropriate complex structure. It is this capacity to connect new bits of knowledge, and also to connect them to previous knowledge, that is impaired in schizophrenic minds. The disturbance may affect vision, or hearing, or touch, or thoughts, or emotions, or actions, but the underlying deficiency is the same: the schizophrenic mind can only process one structure at a time. Within each structure, it can be as good as a normal mind. But we cannot function adequately in everyday situations without the capacity to connect structures, because most situations consist of interacting structures.

Even more interesting is the tendency of schizophrenics to connect mental structures *incorrectly* (rather than not at all), a disorder described as derailment of associations, or loose associations.[19] This disorder is interesting because, in addition to affecting their thought patterns, it is seen in their language; it serves thus to support the thesis that the phenomenon of language involves many knowledge structures present in the mind, not just language structures.

The language of schizophrenics often includes jumbled words, as they fail to match words and thoughts. While being highly significant to the patient, sentences may be incomprehensible to others. Schizophrenics also tend to interpret all language literally, so they have difficulty with sentences where words are used figuratively. They cannot appreciate the meaning of proverbs, for example, as they attempt to interpret them literally.[20] Metaphoric language, we saw, relies on the mind's capacity to connect knowledge structures that are only weakly related; and this capacity is limited in schizophrenics. Alternatively, their own language may appear metaphoric, when they create meaningless connections.

Autism is another disorder that can be described as a failure of the mind to combine structures correctly. Autistic individuals "cannot easily understand

[18] Ibid., p. 23. [19] Ibid., p. 17; Lidz, *Schizophrenic Disorders*, p. 53.

[20] J. S. Kasanin, ed., *Language and Thought in Schizophrenia* (New York: W. W. Norton, 1964), pp. 72–88; Torrey, *Surviving Schizophrenia*, p. 19.

language that is flippant or witty, and … instead they are excessively literal.… Sometimes their comments are perceived as inappropriate by others, as rude or as funny, or else over-polite."[21] This is because they have difficulty matching their response to the current context.

Whereas the normal mind can form coherent wholes by correctly combining mental structures, this capacity is diminished in autistic individuals. Their behaviour, as a result, is characterized by detachment.[22] In isolated situations they may perform as well as, or even better than, normal individuals; but when the correct response depends on recognizing the broader context, their behaviour appears strange: "In everyday life we cannot afford too many errors of literal interpretations; we seek interpretations that are coherent within a wider context that takes in social and cultural experiences."[23]

4

What conclusions can we draw from this analysis? In the previous section, in our discussion of learning and skill acquisition, we noted that we can explain all forms of intelligent behaviour through the phenomenon of tacit knowledge, and that this phenomenon can be represented as a system of interacting structures. Thus, the mind's capacity for complex structures is all we need in order to account for our capability to acquire diverse skills. Now a different study of mental performance takes us to the same mind model: we explained diverse forms of creativity as the capacity to process complex structures; and we saw that even a slight deficiency in this capacity is immediately noticeable as mental disorder, and prevents a person from behaving normally. We must conclude, therefore, that we benefit from this capacity not only when exceptionally creative, but also in ordinary activities. The knowledge that allows us to cope with everyday situations, as much as the knowledge responsible for great feats of creativity, is grounded on the mind's capacity for complex structures.

Only a model based on interacting structures can account for what we know about the mind. By adopting this model, we affirm that no *mechanistic* models of mind are possible. In our model, mental acts are complex structures, and complex structures cannot be reduced to simple ones. This model claims, therefore, that mental acts cannot be reduced to simpler operations.

The mind mechanists claim the opposite: the mind works like a machine, they say, and can be emulated with models based on algorithms, rules, and

[21] Uta Frith, *Autism: Explaining the Enigma* (Oxford: Blackwell, 1989), p. 134.
[22] Ibid., p. 100. [23] Ibid., p. 178.

databases of facts; it is true that today's models display only limited intelligence, but there is no reason why they cannot be improved. By putting together what we have just learned about the mind, however, we can demonstrate the absurdity of this idea.

If we think of the mind as a sort of machine, we will inevitably conclude that the capabilities we observe from outside are a reflection of the complexity of the internal processes: as in a machine, the more sophisticated the mind processes, the more intelligent the resulting acts will be. This is why the mind mechanists attempt to improve the performance of their models by refining the algorithms, increasing the number of stored facts and rules, and so on. This, after all, is how we improve our machines, or our software systems. If today the models can only emulate minds of very low intelligence, the mechanists say, we will find ways to improve them, just as we improve everything else we build. The models will progress, one step at a time, through the entire range of capabilities – from low to normal to superior intelligence.

This is their expectation, but when we analyze the mind's performance we realize that this expectation is unwarranted. For, the mind's performance cannot be rated using a scale of capabilities. We do not recognize any processes in the mind that change as the mind's capabilities increase. On the contrary, we find the same processes in a deficient mind as we do in a normal or superior mind. The capabilities displayed by these minds overlap. The only quality that differentiates them is an ability to create the right connections between knowledge structures; and what determines the right connections depends largely on phenomena external to the mind, and which, moreover, cannot be defined with precision.

Thus, it is impossible to improve the performance of a mind model by enhancing its capabilities, because the chief determinant of performance is not its capabilities but its interaction with the environment to which it is exposed. We note that a superior mind is more likely to connect the right knowledge structures, but we cannot define this capacity in terms of some internal mental processes. What we find, in other words, is that the mind's tremendous information processing capability, impressive as it is, does not explain intelligence. What gives rise to intelligent performance is the fact that the mind belongs to a human being who has a body, grows up in a society, and is exposed to the countless phenomena that make up a normal life. It is its interaction with the environment that lets the mind accumulate the complex knowledge structures which determine its future performance, and there is no other way to create these structures.

The mistake of the mind mechanists, therefore, is not just to underestimate the capacity of the mind, but to see intelligence as the result of computational processes. So they conclude that they can enhance the performance of their

models by improving the algorithms or the hardware. No matter how much the models are improved, however, they can never attain human intelligence, simply because they cannot emulate the *existence* of human beings; that is, their interaction with a complex environment.

❖

We saw that creativity is the result of connecting knowledge structures which were previously unrelated. If we think of the mind as a sort of machine, it is tempting to conclude that we can become more creative simply by increasing the number of connections between structures. And that, similarly, to enhance the creativity of a mind model, all we have to do is increase the number of connections between *its* structures. But we can easily show that this is not the case.

We have to be creative, for example, to recognize a pen. Pens come in an endless variety of sizes, shapes, colours, and mechanisms, as pen manufacturers never stop conceiving new designs. Despite this challenge, though, we usually have no difficulty recognizing (visually) as pen an object we have never seen before. We do this by considering some of its attributes and ignoring others, and by taking into account many clues, including the context in which we encounter it. Using this information and our previous experiences, we determine that the object fits best in the category of pens.

Imagine now a software model of mind that has difficulty recognizing pens. How could we enhance its capabilities? We know that *we* recognize a new pen by generating new links between knowledge structures. With an unusual pen, the links between the structures formed in our mind when we first see it, and the pen-related structures that already exist in our mind, are very weak; and our creativity consists in discovering these links. But we cannot enhance the model's creativity simply by instructing it to connect as many structures as possible.

Consider how our mind works. If our creativity were limited to the strongest links, we would recognize as pens only those objects that are practically identical to previously encountered pens; we would always be right, but we would fail to recognize most *new* pens. And if we did not limit our creativity, and connected even structures with very weak links, then we would indeed recognize all new pens; but at the same time we would mistake for pens most elongated objects we encounter. In both cases we would act abnormally, and would fail to function properly in everyday situations. When we act normally, it seems that our creativity is neither too low, nor too high, but just right. What is "right," though? What range of pen-recognition creativity represents normal, intelligent behaviour? Although we can identify this behaviour in people and

notice any deviation from it, we cannot define it with precision. And if we cannot define it, how could we emulate it in a mind model?

Normal behaviour combines the rational and the irrational, the logical and the illogical. We must be sufficiently illogical (and thus sufficiently creative) to be prepared at any time to recognize as pens objects that are inconsistent with our established conception of pens, and at the same time sufficiently logical (and thus sufficiently uncreative) to avoid seeing pens where there are none. The term "arational" describes, perhaps, this type of behaviour.[24]

It is not only with pens, of course, that we need this unspecifiable blend of rationality and irrationality in order to act normally. Every object has its own narrow range of creativity that we identify as normal behaviour, and so has every feeling, sound, idea, phrase, or situation. We are never exposed to a thing in isolation from all other things, so we associate with certain contexts everything we know: we connect every knowledge structure to other structures. We cannot specify what is normal behaviour, or what is the right degree of creativity, because these qualities do not have measurable values. They are not processes that can be measured or improved as we measure or improve the performance of machines. Depending on the context, new connections between mental structures may appear either as an act of creativity or as mental disorder: the same act may be intelligent one moment and absurd the next.

The only way to assess whether a certain act is correct, or intelligent, or normal, is by noting the context in which it occurred; for the chief criterion that determines whether an act is right or wrong is how appropriate it is in the current context. So, if we wanted to bring the behaviour of a mind model closer to that of humans, we would have to improve its ability to recognize the current context, just as *we* are (intuitively) aware at all times of the context we are in. But then, all we would have done is shift the problem from having to specify what is a correct response, to having to specify how to distinguish one context from another. We recognize contexts tacitly, by combining many pieces of knowledge. Contexts, just like mental acts, are complex structures, and hence irreducible to precise rules and definitions. We cannot improve the mind model by defining contexts, therefore, any more than we could by defining what is an intelligent response.

To recognize a pen, we are helped by noticing that it lies on a flat surface, rather than hanging in a tree; we are also helped by knowing that the surface is a desk, and to recognize the desk we are helped by perceiving it as a piece of furniture in a room, which we do because we know what it means to be in a

[24] Hubert L. Dreyfus and Stuart E. Dreyfus, *Mind over Machine: The Power of Human Intuition and Expertise in the Era of the Computer* (New York: Free Press, 1988), p. 36.

room, and so forth. Clearly, if we want to define a context we run into an infinite regress, as we attempt to define each piece of knowledge in terms of other pieces of knowledge, and then define *those* in terms of others yet. Nor are these recognition processes neat structures of things within things. The structures share elements – objects in the immediate environment, for instance – and hence interact. Like the context itself, the process whereby we recognize the current context is a complex structure.

At any moment, we are influenced by such facts as where we are, what happened moments before, specific objects or people, the sensations received by our senses, cultural background, expectations about the future, and all the related thoughts. Only by tacitly combining these facts can we arrive at the correct interpretation of the current context; and the slightest deficiency in this process leads to abnormal behaviour. The strange responses of schizophrenic or autistic persons can be attributed to their failure to assess correctly the current context; that is, their failure to combine all the clues into one complex structure.

It is notable that the failure of artificial intelligence programs to attain significant performance levels has been attributed to the impossibility of conveying to them all the details they need in order to identify the current context. Contexts are called *frames* in artificial intelligence, and this problem is known as the frame problem. Just like schizophrenic and autistic persons, artificial intelligence programs tend to interpret a given situation too literally, too logically. And we cannot improve their performance by improving the software, because the knowledge structures they lack can only be created by a human mind through exposure to a human environment.

The following passage describes the importance of context in explaining the abnormal behaviour of autistic individuals, but it could just as easily explain the failure of the software models of mind: "Context is at once the most essential ingredient in full intentional communication, and the one feature that distinguishes it from bare message transmission. The hallmark of the latter is the piecemeal handling of information. In principle, there is nothing wrong with this. On the contrary, this mode of information processing guarantees stability: the same code always means the same thing, as in a computer. In everyday human communication this guarantee does not apply. Here, there is an *obligation* to use context. This means often having to say 'it depends.' The meaning of any utterance in word or gesture can only be properly understood by *not* treating it piecemeal, but placing it in context."[25]

No matter how elaborate the software models are, they remain mechanistic; so they can only deal with the complex structures of a context by separating

[25] Frith, *Autism*, p. 180.

them into simple structures. Thus, even if they managed somehow to account for most structures, they would still lack the essential information contained in the *interactions* between structures. This is why the performance of mechanistic models of mind cannot advance beyond that of individuals who suffer from mental disorders.

❖

We note the same problem in understanding humour. We saw that humour is the result of trying to connect two mental structures which cannot be logically combined. But if we wanted to design a model of mind that understands humour, we would find it an impossible task: we wouldn't know how to instruct it to recognize pairs of structures that cause humour.

Consider how our mind recognizes a funny situation: we must be rational enough to understand the logic of each structure on its own, and at the same time irrational enough to attempt to combine them; then, we must be rational enough again to recognize the absurdity of this attempt. Only thus can we appreciate the humour inherent in the situation. If our mind fails to grasp the logical parts of the situation, then we cannot understand it at all and we are accused of being stupid. And if we understand the situation so well that we instantly see its absurdity and do not even try to combine the two incompatible structures, then we can see nothing funny and we are accused of lacking a sense of humour. We have to be both logical and illogical, and there is only a narrow range that is "right," that is deemed normal behaviour. But we cannot define this range with precision; and if we cannot define it, how could we design a software model of mind that understands humour?

Recall the joke about the waiter suggesting milk instead of cream. A software mind model processing that story would not recognize it as a joke. The model would notice the absurdity of the dialogue, and would immediately optimize the text – by replacing the dialogue with one statement, simply requesting black coffee. Computers have no sense of humour. They are completely logical, because we design them that way. But imagine that we wanted to improve the model so that, in addition to being logical, it would appreciate our jokes. How could we do it?

We all know, of course, that there are no definitions or specifications for understanding humour. The range of behaviour that is "right" is not a value that can be measured or improved. If humour springs from the conflict between two knowledge structures, a model of mind must recognize this conflict. But it can only recognize the conflict if it understands the two structures as *we* do; that is, if these structures are connected to *other* knowledge structures. To appreciate that joke, for instance, the model would have to be

familiar with our coffee drinking habits, with our restaurant customs, with the functions of cream and milk, with the concept of choice, and so on.

But possessing many knowledge structures is not enough. In the case of humour, too, the model must be able to recognize contexts – combinations of structures – and we already saw that this is an unspecifiable process. We notice how delicate this process is when a remark meant to be funny becomes absurd or insulting as the context changes (on a different occasion or with different listeners), or when people with a different cultural background fail to understand a certain type of humour. In both cases, the problem is that some of the knowledge structures in the minds of those exposed to that humour are different from the structures needed to complete the humorous situation. But it is impossible to specify with precision which structures are wrong and how to correct them.

It is, again, by studying mentally impaired individuals that we notice how difficult it is to specify what is normal behaviour. Mental disorders and humour both manifest themselves as shifts between unrelated mental structures, and only the context can tell us which one we are witnessing: "Such shifts from one realm into another are very frequent with schizophrenics, and therefore the patients' statements sometimes strike one as witticisms."[26]

Whether we observe creativity, or humour, or any other act of discovery, the mental process is the same – generating new connections between knowledge structures – and to judge the result we must know the context: "When two independent matrices of perception or reasoning interact with each other the result … is either a *collision* ending in laughter, or their *fusion* in a new intellectual synthesis, or their *confrontation* in an aesthetic experience. The bisociative patterns found in any domain of creative activity are tri-valent: that is to say, the same pair of matrices can produce comic, tragic, or intellectually challenging effects."[27]

Now, one may argue that it is not important that models of mind be so advanced that they display creativity, or appreciate art or humour. But this, of course, is not the issue. If the same mental processes that take part in creative acts, aesthetic experiences, and humour also take part in normal behaviour, then if we cannot emulate these unusual mental acts we cannot hope to emulate normal human intelligence either. In other words, functioning normally in everyday situations already requires the *full* capacity of the mind. Emulating highly creative acts, therefore, is *not* more difficult than emulating normal behaviour – not if "more difficult" means faster computers, or more advanced algorithms, or more facts and rules.

[26] Kasanin, *Schizophrenia*, p. 121.

[27] Koestler, *Creation*, p. 45.

5

Mind mechanism, then, is the belief that human skills and intelligence can be replaced with software devices. In the next section we will examine the social consequences of this fallacy, but let us first analyze its origin; namely, the assumption that it is possible to emulate the mental capabilities of human beings by separating intelligence into independent knowledge structures.

Normal behaviour depends on generating the right connections between knowledge structures; and what is "right" on each occasion depends on the existing structures and connections, which themselves were generated on earlier occasions through the "right" connections, and so on. What we recognize as knowledge and intelligence is embodied in the structures and connections developed by the mind when we were exposed to certain phenomena. Although we must not underrate the data processing capabilities of the brain, it is ultimately our experiences that determine our knowledge. So, even if we join the mind mechanists and hope to have one day a device with the same capabilities as the brain, the only way to emulate human knowledge would be to expose the device to a human environment.

But how could a computer be designed to live the life of a human being? No one expects to have mind models of this kind, of course. Thus, because they deny the need to emulate *all* of a person's existence, the mind mechanists reject, in effect, the concept of learning. Specifically, they reject the fact that the intelligence required to cope with a certain environment can only be acquired through exposure to that environment. If the device must have knowledge that does not come from interacting with its environment, the only alternative is to program somehow this knowledge directly into the device – through algorithms, rules, and databases of facts. This explains the attempts to reduce human intelligence and skills to a mechanistic representation, as only mechanistic concepts can be translated into software concepts. And this explains also the attempts to emulate only one act at a time: implementing only one knowledge structure, the mechanists believe, is easier than trying to incorporate in a model *all* the knowledge of a human being. But these ideas are fallacious.

To behave intelligently, we must recognize at any moment the context we find ourselves in. This context is a complex structure, and we recognize it because we are already familiar with the interacting structures that make it up. If we try to isolate some of these structures, we end up losing the interactions; besides, we discover that they too are made up of interacting structures, which are made up of others yet, and so on. The only way to recognize a context,

therefore, is intuitively: we must recognize it as a whole, by processing all its structures simultaneously. We *can* recognize contexts, but we cannot explain how we do it. Because so many knowledge structures are involved in this process, we must conclude that we use a great part of our knowledge every time we recognize a context.

Returning to the previous examples, there are no specific mental processes for recognizing pens, or for understanding coffee jokes; consequently, there is no specific pen recognition skill, or coffee jokes skill, that one can acquire. The knowledge we need in order to behave intelligently in these situations is about the same as the knowledge we need in a thousand other situations. We can find no knowledge structure that, when isolated from our other knowledge, embodies the intelligence required to perform a particular act. The idea that models of mind can be programmed to perform specific intelligent acts is, therefore, nonsensical. This can succeed for *context-free* acts, which can indeed be isolated from the others and approximated with simple structures (data processing tasks, for instance), but not for the general acts we perform in everyday situations. To put it differently, the only way to program a mind model to emulate *one* of our intelligent acts is by programming it to emulate *most* of our acts. The mind mechanists admit that the latter task is too ambitious, so they address the simpler one. They don't see that the simpler one is only an illusion.

It is because no knowledge structure in the mind can be isolated from the others that normal behaviour is so hard to define. If we stay within one structure, we are *too logical* – and our behaviour is considered abnormal. We *must* connect knowledge structures, but then we find that, from the infinity of possible links, very few are right: only those that reflect the current context. If we allow more links, we become *too illogical* – and our behaviour is again considered abnormal. Normal behaviour, thus, is far from being banal behaviour. When we behave normally we display great expertise – expertise in recognizing contexts.

The same mental processes are involved in everything we do: when behaving normally as much as when displaying expertise, creativity, or humour. No one knowledge structure can be isolated from the others, because no mental act can be represented mechanistically. This is why it is no easier to program a mind model to display intelligence in one specific act than to program it to display the general intelligence of a person, and no easier to program it to display normal, common-sense behaviour than to program it to display expertise, or creativity, or humour.

Replacing Minds with Software

1

The greatest challenge facing us today is to recognize how the mechanistic delusions are shaping the future of our society. And it is the mechanistic *mind* delusions that are the most dangerous, because how we think of our mental capabilities affects our decisions in all fields where knowledge and skills play a part. Thus, "what we do now will determine what sort of society and what sort of human beings we are to become. We can make such a decision wisely only if we have some understanding of what sort of human beings we already are. If we think of ourselves only as repositories of factual knowledge and of information processing procedures, then we understand ourselves as someday to be surpassed by bigger and faster machines running bigger and more sophisticated programs. Those who embrace that limited conception of intelligence welcome the change with enthusiasm."[1]

The danger we face when perceiving human minds as a kind of machine is getting to depend on charlatans who promise us devices that are better than our minds. For, once we replace knowledge and skills with devices and stop using our non-mechanistic capabilities, we will indeed become inferior to devices. We will then *have* to depend on devices, and our minds will be further degraded, in a process that feeds on itself.

I have devoted so much space to mind mechanism because it is more than just another mechanistic delusion. The belief that we can build models of mind leads in practice to the belief that there exist substitutes for knowledge and skills. The delusion that it is possible to embed human intelligence in a device has now gone beyond the speculations of artificial intelligence: it has escaped the academic confines, and is increasingly influencing our business and social decisions. Our belief in mind mechanism has given rise to a phenomenon that may be called *the culture of knowledge substitutes*.

Thus, instead of developing certain knowledge, we expect a software device to function as a substitute for that knowledge. And it is this fantastic idea that has allowed the software companies to rise to their glamorous and dominating position. These companies cannot provide knowledge substitutes, of course – no one can. But the bytes they sell us in those colourful boxes produce illusions that act very much like the religious or political illusions of the past, tempting us with promises of salvation, of easy solutions to difficult problems.

[1] Hubert L. Dreyfus and Stuart E. Dreyfus, *Mind over Machine: The Power of Human Intuition and Expertise in the Era of the Computer* (New York: Free Press, 1988), p. 206.

❖

Mind mechanism is the most widespread of our mechanistic delusions, and is a factor in many other delusions, including our language and software theories. Mind mechanism is typical of what I have called *the new pseudosciences* – the subject of the next chapter.

Scientists who engage in these worthless pursuits can create entire academic disciplines out of nothing more substantial than a mechanistic fantasy. They start by making an extravagant claim and presenting it in the form of a theory: an explanation for a complex phenomenon, or a solution to a complex problem. This generates a great deal of excitement, even though everyone can see that the theory is only a speculation, a wish. Its foundation on mechanistic principles – which principles are accepted unquestioningly – is what makes the theory credible, imparts to it a "scientific" image, and persuades everyone that it will soon be useful.

The theory appears to work in a few cases (as mechanistic approximations always do), and this gives the scientists additional confidence. Since they are now convinced that it is an important contribution, a "research program" is initiated: a long series of modifications, which make the theory increasingly complicated and thereby mask its shaky foundation. These modifications often result in mechanistic solutions to isolated problems – problems discovered when trying to apply the theory, and perceived now to be part of the research. Encouraged by these successes, the scientists are convinced that they are making progress, that these solutions will be combined one day into an answer to the *original* problem. Non-mechanistic phenomena, however, cannot have mechanistic explanations. So the theory never works, and is eventually abandoned. But we learn nothing from these delusions: new mechanistic theories, equally fallacious, always emerge to replace those that failed.

2

Our chief concern in this book is a specific pseudoscience: the belief that knowledge and skills can be replaced with software. In particular, we are concerned with the belief that *programming* knowledge and skills can be replaced with software – the belief that there exist substitutes for programming expertise.

We hear about innovations like application development environments, object-oriented languages, relational databases, or computer-aided software engineering, and we see the many software devices that embody these ideas. Every year, there are hundreds of new versions of these devices, thousands of

books, periodicals, and courses promoting them, millions of practitioners struggling to assimilate the endless novelties, and billions of dollars being spent by society to support it all. And even a casual study would reveal that it is these innovations and the resulting complexity, rather than the software applications themselves, that constitute the chief preoccupation in the world of programming.

Accordingly, a person unacquainted with programming must think that programming substitutes are indispensable for creating applications. This impression would be reinforced by the fact that, for more than forty years, the notion of programming substitutes has been upheld by professors and gurus, taught in famous universities, endorsed by professional associations, approved by renowned experts, and embraced by corporations large and small in their effort to improve programming practices.

Faced with this reality, the lay person must conclude that the need for programming substitutes, and for their perpetual changes and ever increasing complexity, is based on solid theories, or on irrefutable evidence. That person would be surprised, therefore, when he found out that no such theories or evidence exist. He would be surprised to hear that these innovations are totally unnecessary for developing or maintaining software applications; that their sole purpose is *to eliminate the need for programming expertise*; and that the enormous social and economic structure we have created around them is merely a system of belief – a system founded on the *assumption* that we can invent substitutes for programming expertise.

In chapter 7 we will see that the programming substitutes are mechanistic delusions: despite their variety, they are all based ultimately on the principles of reductionism and atomism – on the belief that programming tasks can be neatly broken down into simpler and simpler tasks. But we can already see the part played in software delusions by the delusions of *mind* mechanism. For, the very notion of programming substitutes must assume that the mental processes involved in programming can be precisely specified, represented with rules and methods, and then implemented with software. And this assumption is identical to the assumption of mind mechanism; namely, that human knowledge and skills can be emulated with mechanistic models. Thus, the programming substitutes are in effect models of mind, substitutes for human intelligence.

❖

The mechanists notice the various processes that make up a software application – the business practices embodied in it, the subroutines, the database, the flow of execution, the data entry or display operations – and conclude that each

process can be extracted from the whole act of programming and dealt with separately. Once this idea is accepted, any number of methodologies, theories, or development tools can be invented for each isolated process, all claiming great improvements in programming productivity. Typically, these expedients attempt to simplify programming by providing higher-level starting elements.

In these concepts, we recognize the two fallacies of mechanistic thinking: reification (the belief that we can separate the simple structures that make up a complex phenomenon) and abstraction (the belief that we can start from higher levels without losing any alternatives at the top level). By reifying complex programming phenomena into simple structures, the software mechanists hope to replace the difficult task of programming a whole application with the easier task of programming separately each one of its processes. This reification also opens the way to abstraction: less work and lower programming skills are required to complete the application if we start from higher levels within each process; in other words, if our starting elements are modules or operations that already incorporate many others. With this method, it is believed, applications of any size and complexity can be created even by inexperienced programmers.

Although useful in domains like manufacturing and construction, this method is inadequate for software-related phenomena; it is inadequate because these phenomena consist, not of processes *within* processes, but of *interacting* processes. In an application, the various software processes interact with one another, and also with the personal, social, and business processes affected by that application. A software process is not a structure *within* other software processes, or *within* a business process; nor is a business process a structure *within* a software process. Thus, software-related phenomena can only be represented with *complex* structures. Thanks to its non-mechanistic capabilities, the human mind can successfully deal with these phenomena. But the only way to create in our mind the complex knowledge structures that reflect the complex software phenomena is through personal experience: by being exposed to these phenomena for many years. We *can* acquire programming expertise; but, like other difficult skills, this takes a long time.

3

When the mechanists claim that programming methods and devices can reduce or eliminate the need for programming – and can therefore act as substitutes for the programming skills of humans – how are we to understand the claim? Since models of mind can emulate only those mental functions that can be represented mechanistically, the claim is in effect that programming is

such an easy challenge that it requires only mechanistic knowledge. But is this true?

Programming methods and devices are based on the assumption that the various activities in software development can be represented with *simple* structures; namely, independent mental acts and independent pieces of software related to one another only through the precise definition of a hierarchy. We saw, however, that practically everything we do requires our mind's capacity for *complex* structures. Is programming, then, an exception? Is it simpler than recognizing faces or voices, or driving a car, or interpreting X-rays, or distinguishing chicks? Is it simpler than coping with everyday situations, or understanding jokes or metaphors? Can we program without being able to recognize contexts? Since it is absurd to think that programming is simpler than everything else we do, the burden of proof rests on those who claim it is: those who claim that there can exist mechanistic substitutes for programming knowledge.

The software mechanists, of course, do not claim that programming is easy. On the contrary, they say, we need the substitutes precisely because programming is difficult and there is no other way to simplify it. But note the contradiction: their attempt to simplify programming through mechanistic theories is equivalent to claiming that programming *is* a simple activity – so simple that it can be reduced to exact, fully specifiable structures of things within things. This contradiction stems from the same fallacy that mechanists have being committing for centuries (we will study this fallacy in chapter 4): They see the richness of a complex phenomenon (minds, language, software, etc.), and the simplicity of the hierarchical concept, and they wish to have both: they wish to use simple hierarchical structures to represent the complex phenomenon. They fail to understand that it is precisely the fact that the phenomenon is complex, and hence irreducible to simpler ones, that gives it its richness and potency.

This explains why programming theories, methodologies, tools, and aids, no matter how sophisticated, have so little effect on programming productivity. Programming is one of the most difficult tasks that we have ever had to perform; and it is so difficult precisely because there is no way to reduce it to a mechanistic representation, and we must depend therefore on personal experience. It is similar, in this respect, to our linguistic or visual performance. Like these other acts, programming can only be performed by human minds, because it depends on our capacity for complex knowledge. It involves discovering the right connections between many knowledge structures: the various concepts that make up the application, technical details, business matters, and even personal and social concerns. And, like all intelligent acts, it requires the full capacity of a mind.

So, in the end, the issue is this: Are programming skills a mechanistic phenomenon? Or are they a complex phenomenon, like our linguistic capability and the many other skills that cannot be reduced to simpler mental acts? Our software pursuits are based on the assumption that our software-related problems are mechanistic, so programming can be restricted to mechanistic acts. But, as we have seen here, all the evidence indicates that this assumption is mistaken: software involves complex phenomena, so programming requires complex knowledge. Thus, we must accept the conclusion that our current software pursuits are largely mechanistic delusions.

Unlike other mechanistic delusions, however, which did not survive for long in the real world, outside academia, the software delusions keep growing and are now spreading throughout society. Thus, the ideology of software mechanism has become the greatest mass delusion in history: a myth more powerful and widespread than all the religious and political systems of belief we had in the past.

But how did this happen? How was it possible for the software elites, in just a few years, to reach the point where they practically control society through this myth? By persuading us to shift our preoccupations from *real* problems to *software-related* ones. The shift occurred first for programmers; then, the incompetence it caused in application development forced the *users* of software to undergo a similar transformation. (We will study this evolution in chapter 6.) Whether the victims are programmers or other workers, it is the same belief – software mechanism – that leads to the change in preoccupations.

Typically, the software mechanists start by promising us solutions to *real* business, social, or personal concerns. Because of our mechanistic tradition, we fail to see that only *mechanistic* problems can have software solutions, so we believe all their claims. For the simple, mechanistic problems, their solutions work well, and these successes enhance their credibility. These successes, however, do not guarantee that software will also solve our complex, non-mechanistic problems – those requiring human expertise. But, even though software fails in those tasks, just by depending on software we create a whole category of problems we did not have before: software-related problems. Since we do not doubt the mechanistic ideology, we continue to believe that the original, real problems can be solved with software. So we seek solutions to the new, software-related problems, hoping that by solving *them* we will solve the original ones. Since most software-related problems are mechanistic, we can usually solve them with mechanistic means – which entail even more

software. But these software solutions are only solutions to the *software-related* problems, which we created ourselves, not to the *original* problems.

Every time we adopt a piece of software, we get to depend on *it*, instead of depending on some other expedient or on our own skills. And even if that software does not solve our real problems, we continue to depend on it and must deal with the problems it generates. So the software elites control our life simply by persuading us to accept software solutions. We need more and more software solutions because we have more and more software-related problems. Our real problems may remain unsolved, but as long as we believe that they can be solved with software we continue to adopt new types of software, and thus create new types of software-related problems, for which we need new software solutions, and so on.

The more difficult it is to solve our real problems with software, the more software solutions we will try, and the more software-related problems we will have. Attempting to find a substitute for a complex phenomenon – a difficult human skill like programming, for instance – is guaranteed to generate a never-ending series of software problems and solutions, of software innovations and devices, precisely because it is an impossible quest. Eventually, a great part of our life is taken by software preoccupations that have little to do with our real problems. But in our infatuation with mechanism we fail to notice this. On the contrary, we *like* our new, software-related problems. We like them because they seem to have easy solutions: we always seem to be *solving* problems. Besides, the software propaganda makes us feel modern and successful simply by *having* this type of problems.

It is this shift in preoccupations – from the real problems to the spurious, software-related ones – that has allowed the software elites to gain so much power. As with any tool, there is nothing wrong in depending on software devices when they are indeed the best answer. It is not from *solving* problems that the elites derive their power, however, but from the exact opposite: *preventing* us from solving problems. For, if the problems are so complex that only a human mind can solve them, then by depending on software devices we forgo the opportunity to develop the skills that *could* solve them. The devices will provide only limited answers, or no answers at all; but we will continue to depend on them, and on the elites behind them, because we will no longer have our own expertise as measure.

Since the only thing the elites can offer us is software devices, they must ensure our dependence on these devices regardless of their degree of usefulness. And the simplest way to achieve this is by degrading the concept of expertise: from the difficult skills needed to solve important problems, to the easy skills needed to operate software devices. This – the prevention of true expertise – is the final goal of the software elites. For, only in a world

where we can no longer use the full capacity of our minds do the software devices outperform us. The reason it appears to us that the software elites are indispensable is that we are becoming increasingly dependent on them for software solutions to the software-related problems we created when we adopted some other software solutions. To *these* problems they can indeed provide answers; and this keeps increasing their prestige and influence, even as they are preventing us from developing our minds and from solving our *real* problems.

4

The programming profession is the only field in which the destruction of knowledge and skills caused by software mechanism is now complete. The incompetence of programmers, therefore, provides the best illustration of the delusion of knowledge substitutes; and the programming theories are the best medium for studying this delusion. (This is the subject of chapter 7.)

It is not hard to see why the programming profession was so easy to destroy: unlike other skills, programming lacks a tradition that could act as standard of expertise. We have centuries of tradition in writing, for example. We already know what good prose is, so if the mechanists claimed that certain language devices can replace the talent and expertise of humans, we would simply compare the results, and recognize the absurdity of their claims. In programming, however, we had no time to attain a similar wisdom before this activity was taken over by incompetents and charlatans.

We have had many opportunities to observe the work of talented programmers. But, while in other professions it was always the performance of the *best* workers, in programming it was the performance of the *mediocre* ones, that was taken as the highest level we should expect. From the start, the official doctrine was to avoid relying on exceptional individuals (because they are hard to find) and to degrade programming to an activity that can be performed even by the least experienced practitioners. The software theorists, we were promised, will show us how to replace programming expertise with methods, tools, and aids. Thus, while in other professions education and training took years, and expertise was attained only after decades, programmers were trained in weeks or months, and were not expected to ever attain expertise.

I have stated that the software elites can control our life by inducing us to shift our preoccupations from real problems to spurious, software-related ones. In the case of programming, the real problem was how to create and maintain software applications efficiently and reliably. This is a difficult task,

demanding much knowledge and experience; but there are enough men and women who can acquire the necessary skills, if given proper education and training, and the opportunity to practise. Programming is no different in this respect from other difficult professions – medicine, for instance. As in other professions, we may use methods, tools, or aids, but it is our own expertise that is the critical factor. No methods, tools, or aids can replace this expertise, because it can exist only in a mind. The expertise is embodied in the complex knowledge structures formed in the mind through lengthy exposure to the complex phenomena of programming.

Thus, if the problem is programming expertise, the answer is a great deal of programming practice – which should not surprise us. But it is a fundamental assumption in our software culture that programming is different from other professions; that creating software applications is akin to the routine work performed by assembly workers when putting together appliances from prefabricated parts; and that the software elites have invented methods and systems which allow us to create applications in this fashion. So, rather than practising for many years to attain programming expertise, it is believed that programmers can accomplish the same tasks merely by using these methods and systems.

And so it is how programming expertise was redefined to mean expertise in the use of substitutes for expertise. The preoccupation of programmers shifted from developing their skills and creating useful applications, to learning how to use programming substitutes. The problem shifted from improving their knowledge of programming to improving their knowledge of ways to *avoid* programming.

It was evident from the start, though, that the substitutes do not work: applications took too long to develop, or were never completed, or were inadequate. Still, no one wanted to give up the idea of programming substitutes, so the conclusion was that we needed better substitutes, not better programmers. And when the new substitutes also failed to replace expertise, the conclusion was, once again, that we needed different substitutes. Thus, changes in programming theories, methodologies, environments, languages, and tools became the chief preoccupation of all software practitioners.

This is still true today. After more than forty years of failures, the doctrine of programming substitutes continues to define this profession. The substitutes, in fact, are now even more complicated and are changing even more frequently, so programmers are wasting even more time with them. Every software deficiency is blamed on the current programming environment, tools, or methods, and is used to justify the next round of changes. Programming incompetence is never suspected. As in the past, the level of a novice is considered the highest level of knowledge that we can expect from a program-

mer. No matter how many years of experience they have, programmers are rarely required to perform a task they could not have performed after the first year or two of practice.

A programmer's expertise is measured, not by his ability to develop and maintain software applications, not by his skills in solving important problems, but by how many methodologies, programming languages, or development environments he is acquainted with, or how many computer periodicals he reads, or how many courses he attended. He must be aware of the latest releases and versions of software products, of announcements and rumours. These are the kind of qualifications that companies are looking for when hiring a programmer; that is, how well he conforms to the mechanistic software ideology. Often, just a few months of experience with a particular system is the only qualification needed.

By degrading the programming profession, by lowering the definition of programming expertise to mean expertise in the use of programming substitutes, the original problem – how to create and maintain applications efficiently and reliably – has been replaced with a multitude of *software* problems. The work of programmers has been degraded to the point where they are only expected to know a number of facts, rather than possess a body of knowledge and skills recognizable as a profession. We have all but forgotten that programming practice ought to mean simply the solution of business or social problems through software. Only the smallest part of a programmer's activities is directly related to this objective.

To understand why this inefficiency is seldom evident, we must remember how common it is for programmers to perform activities that, while perceived as important and urgent, are in reality spurious: dealing with the problems generated by programming substitutes. And, even though what I have said applies mainly to *application* programmers, we must not forget that the work of *system* programmers consists almost entirely in creating the substitutes needed by the application programmers. Thus, no matter how successful or impressive is the creation of the programming tools, development environments, or database systems themselves, this is a spurious activity.

And so are the other activities engendered by the programming substitutes. Were the application programmers experienced professionals, there would be no need for the programming substitutes, or the organizations that sell them, or the schools that teach how to use them, or the conventions that promote them, or the publications that explain, advertise, and review them. It is only because we have forgotten the real purpose that computers and software ought to have in society that we are not outraged by the stupidity of these preoccupations.

❖

As our dependence on computers is growing, the cost of programming incompetence is becoming a social issue that affects us all, because the businesses and governments that depend on incompetent programmers simply pass the cost to the rest of society. The cost to society caused by the destruction of programming skills exceeds probably one trillion dollars a year, globally – money that ultimately ends up in the pockets of the software elites and the software bureaucrats. We need only recall the so-called Y2K date problem (the need to modify the existing applications to handle correctly dates beyond 1999, which cost society hundreds of billions of dollars), to recognize the immense power that software practitioners derive from providing solutions to problems they create themselves. Nor is this problem as unique as it appears. It stands out because it was due to the same deficiency in millions of programs, but it is otherwise similar to all the other software-related problems.

The present consequences of programming incompetence, however, while important, are small compared to what the future holds. Programming is only the first type of knowledge to be destroyed by the software elites. Others will follow, as our dependence on software is spreading beyond the dependence on inadequate business applications. We can already recognize similar delusions, for example, in the kind of software known as office productivity systems, which attempts in effect to replace a variety of skills – skills that take many years to acquire – with acts that anyone can learn to perform in a few days. Similarly to what happened in programming, the preoccupation of managers and office workers is shifting: from improving their skills to learning how to use substitutes for these skills, and from providing professional services to searching for solutions to the endless problems generated by the substitutes. The expertise of office workers is increasingly measured, not by their ability to perform important tasks, but by how well they are acquainted with software devices. Valuable knowledge that can only develop in a mind is being destroyed and replaced with inferior substitutes.

Each time we replace human expertise with substitutes that are in fact only illusions, we lose our capacity to solve the *real* problems, because we forgo the opportunity to develop, through practice, the complex knowledge needed to solve those problems. What is worse, we add to our software-related problems, and thus increase our dependence on software charlatans.

This makes our study of software mechanism in the domain of programming even more important: not only does it help us understand why knowledge substitutes cannot work, and why we must encourage expertise and responsibility among programmers; not only can it help us derive greater benefits from our computers and at the same time save society trillions of

dollars, which can then be directed toward better causes than the support of a software bureaucracy; but it can help us anticipate the consequences of replacing human expertise with mechanistic substitutes in *other* fields of knowledge. If we understand how the mechanistic fallacies have contributed to our software delusions in the domain of programming, we can perhaps learn to recognize and avoid the formation of software delusions from mechanistic fallacies in other domains.