SOFTWARE AND MIND

Andrei Sorin

EXTRACT

Chapter 4: *Language and Software*
Sections *The Common Fallacies, The Search for the Perfect Language*

These sections examine the mechanistic philosophy of language, and analyze the mechanistic fallacies common to language theories and software theories.

SOFTWARE AND MIND

The Mechanistic Myth and Its Consequences

Andrei Sorin

ANDSOR BOOKS

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

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.

The Common Fallacies

1

Let us review first our language and software fallacies. Most people, if asked what is the purpose of programming languages, would agree that it is similar to the purpose of natural languages. All languages, it seems, are systems of symbols, definitions, and rules, employed to convey instructions or information. Speaking, along with programming, entails the translation of knowledge from one representation – mental forms, social situations, natural phenomena – into another: words and sentences in the case of speech, operations and statements in the case of programming.

So the resulting sentences or statements, we believe, reflect the original knowledge in a different form and medium. Sentences describing wishes, or feelings, or states of affairs, or scientific facts, or logical arguments, are perceived by most of us to be verbal representations of those things – representations created by selecting appropriate words and arranging them according to certain rules of grammar. And the act of communication takes place, presumably, when the person hearing or reading these sentences translates them, somehow, into a mental representation that resembles the original one. Similarly, programs designed to address specific business requirements in inventory management, production scheduling, or text processing are thought to be software representations of those requirements.

With programming languages as with natural ones, then, we perceive the structures created with language to be *pictures* of the world, replicas of reality.

We believe that the role of language is to generate structures which provide a *one-to-one correspondence* to the knowledge or events we want to represent: each entity in the real world – each object, process, or event – must have its corresponding counterpart in the structures created with language; then, and only then, will utterances correctly express facts or ideas, and programs correctly represent our requirements.

Another term commonly used to describe this relationship is *mirroring*. An even better term is *mapping*: just as a map provides a one-to-one correspondence to those aspects of a territory that we want to represent graphically, our utterances and our programs correspond to, or map, those aspects of the world that we want to express through language. Lastly, the term *isomorphism* is sometimes used to describe these one-to-one relationships: the structures we create with language are isomorphic to the phenomena that occur in the world.

A second quality we believe to be shared by programming languages and natural languages is their hierarchical character. It seems that text can always be represented as a hierarchical structure of linguistic entities – paragraphs, sentences, words; in addition, words can be classified hierarchically on the basis of their meaning. Similarly, it seems that any piece of software can be broken down hierarchically into smaller and smaller software entities. Our mechanistic view of the world tempts us to perceive all phenomena, including language and software, as systems of things within things.

Thus, if we believe that all phenomena can be represented with hierarchical structures, and believe also that languages generate hierarchical structures, it is not surprising that we see languages as mapping systems. We do not doubt for a moment that all aspects of reality can be represented precisely and completely in one language or another. If we assume that the simplest elements of language (words, for instance) correspond to the simplest elements of reality (individual objects or actions, for instance), all we need to do is combine these elements into more and more complex ones, on higher and higher levels. Each level in the structure created by language will then correspond to a more complex aspect of reality. Similarly, if the simplest *software* elements correspond to the simplest parts of our affairs, by combining these elements hierarchically we will generate software applications that represent more and more complex aspects of our affairs.

❖

These, then, are our language and software fallacies. Our naive view of language – the belief that language can provide a one-to-one correspondence to reality, which stems from the belief that both language and reality can be represented with neat hierarchical structures – forms one of the oldest mechanistic delusions in Western history. It has misled philosophers and scientists for centuries, and it is now distorting our perception of software and programming.

The fundamental mistake in this view is, of course, reification. The structures we create with language do indeed represent the world, but not on their own. Languages are human inventions, so they cannot exist independently of their human users. As a result, language structures always interact with various knowledge structures present in human minds, and with the structures formed by human societies. Thus, it is not through language alone but through the totality of these structures that we can hope to understand the world; that is, to attain a close correspondence to reality. The fallacy is the same, whether we expect a one-to-one correspondence between our world and the sentences of a language, or between our affairs and the statements of a software application.

In the domain of logic, the language fallacy has given rise to the belief in the existence of an ideal language. The ideal language is an artificial language, or a modified natural language, which, being logically perfect, would permit us to represent with mathematical precision everything that can exist in the world. The search for the ideal language culminated in the twentieth century with the work of philosophers like Bertrand Russell and Rudolf Carnap. These philosophers held that knowledge can always be represented by means of a language, and that science and philosophy are in fact little more than attempts to represent the world through various types of languages.

These philosophers also believed that the world has a certain logical structure – a structure that can be discovered. But we will not discover it as long as we try to mirror it in our natural languages, because these languages are imperfect, ambiguous and illogical. It is this defect, more than anything else, that prevents us from understanding the world and finding answers to our inquiries. Thus, Russell remarked that "almost all thinking that purports to be philosophical or logical consists in attributing to the world the properties of language."[1] We set out trying to mirror the world in language, and we end up, instead, perceiving the world as similar to our illogical languages.

So, the argument continues, if we represented reality in a logically perfect language, we would find the correct answers simply by expressing our inquiries in that language. This is what we do when we represent the world in the perfect language of mathematics – in astronomy, for instance. It is because we have found a way to represent the world with a perfect language that we are so

[1] Bertrand Russell, quoted in Irving M. Copi, "Artificial Languages," in *Language, Thought, and Culture*, ed. Paul Henle (Ann Arbor: University of Michigan Press, 1965), p. 107.

successful in the exact sciences and in engineering; so we should try to find an equally logical language for the other kinds of knowledge.

But the argument is wrong, because most aspects of the world are too complex to represent with logical languages like mathematics. Our natural languages appear ambiguous and illogical precisely because we use them to represent this complexity. No language can represent with precision the complex aspects of the world.

In the domain of programming, the language fallacy has given rise to our software delusions, to the notion of software engineering, to theories like structured programming, to methodologies, development tools, and programming environments – all stemming from the belief that the problem of programming is the problem of creating exact hierarchical structures of software entities. We believe that the answer to our programming problems lies in inventing programming concepts that are logically perfect, and in the use of application development systems based on these concepts.

Thus, because programming systems can generate hierarchical structures, we ended up attributing to the world the properties of these systems: we ended up believing that those aspects of the world that we wish to represent with software are neat hierarchical structures. But the world is *not* a neat structure, so the neat software structures are seldom adequate. We continue to believe, though, that the problem is the inexactness of the software structures, so the answer must be to improve our programming systems.

The language fallacy, then, has given rise to our preoccupation with programming languages, methodologies, tools, and environments, to the belief that these inventions are the most important aspect of software development, and that the adoption of more and more elaborate versions is the only way to improve our software representation of the world.

This preoccupation, this search for the perfect programming system, is the software counterpart of the age-old search for the perfect language. Instead of overcoming our mechanistic language delusions, we have augmented them with mechanistic *software* delusions.

2

We are interested in the modern theories of a perfect language – the theories that have emerged since the seventeenth century – for it is from these language delusions that our software delusions were ultimately born. Umberto Eco,[2]

[2] Umberto Eco, *The Search for the Perfect Language* (Oxford: Blackwell, 1997). The title of the next section reflects this book's title.

however, notes that the idea of a language that mirrors the world perfectly was preoccupying philosophers long before the Scientific Revolution. An attempt to explain how language corresponds to reality can be found even in Plato's dialogues. On the whole, "the story of the search for the perfect language is the story of a dream and of a series of failures.… [It is] the tale of the obstinate pursuit of an impossible dream."[3]

If the search for a logically perfect language has been based since Descartes on mechanistic notions, earlier theories were based on mystical or religious notions. One theory, for instance, was inspired by the biblical story of Babel: In the Garden of Eden there was only one language – the language God used to speak to Adam. This was a perfect language, but it was lost at Babel, when Man started to build a mighty tower in an attempt to reach the heavens. This arrogant project incurred the wrath of God, who decided to stop it by confounding the language used by the workers. Construction was disrupted, the tower was never finished, and we have suffered ever since the confusion of a multitude of illogical languages. Thus, the belief in a perfect language can be seen as an attempt to restore the ideal state of the Beginning: a language that mirrors reality perfectly would enable Man to again understand his world, attain omniscience and happiness, and perhaps communicate with God.

An early example of this belief, Eco notes, is the linguistic project of Dante Alighieri, started in 1303. Dante, who is best known as poet and philosopher, attempted to create a poetic language that would serve the needs of an ideal society – a language suited for expressing truth and wisdom, and capable of accurately reflecting reality. "Opposing this language to all other languages of the confusion, Dante proclaimed it as the one which had restored that primordial affinity between words and objects which had been the hallmark of the language of Adam."[4]

Interpreters have concluded that Dante, influenced by even earlier scholars, believed that the divine gift received by Adam was not so much a language as the *capacity* to understand and create languages – a view similar to Chomsky's idea of an innate language faculty (see "Universal Grammar" in chapter 3): "What God gave Adam … was neither just the faculty of language nor yet a natural language; what he gave was, in fact, a set of principles for a universal grammar."[5] And what Dante believed to be the most important characteristic of these principles was the capability to provide a one-to-one correspondence to the actual world. This capability is what he saw as the distinguishing quality of a perfect language – that quality which our natural languages have lost, and which he strove to restore through his poetic language: "It seems most likely that Dante believed that, at Babel, there had disappeared the perfect *forma*

[3] Ibid., p. 19. [4] Ibid., p. 35. [5] Ibid., p. 44.

locutionis [i.e., linguistic model] whose principles permitted the creation of languages capable of reflecting the true essence of things; languages, in other words, in which the *modi essendi* of things [i.e., their essence] were identical with the *modi significandi* [i.e., their representation]."[6]

George Steiner points out that every civilization had its version of Babel in its mythology.[7] Thus, the belief in a language that mirrors reality stems perhaps from a common human need to understand the world. And the one-to-one correspondence provided by such a language derives from its divine nature: "The vulgate of Eden contained, though perhaps in a muted key, a divine syntax – powers of statement and designation analogous to God's own diction, in which the mere naming of a thing was the necessary and sufficient cause of its leap into reality.... Being of direct divine etymology, moreover, [it] had a congruence with reality such as no tongue has had after Babel.... Words and objects dovetailed perfectly. As the modern epistemologist might put it, there was a complete, point-to-point mapping of language onto the true substance and shape of things."[8]

❖

Throughout history we have been searching for an ideal language, motivated by the belief that language can provide an exact correspondence to reality – the same belief we recognize in modern linguistics. We should not be surprised, therefore, that this mechanistic view of language has prompted twentieth-century thinkers like Russell, Carnap, and Chomsky to propose linguistic theories that are so similar to the mystical notions held by medieval scholars. Nor should we be surprised that our *software* theories, which arose out of the same mechanistic culture and are grounded on the same beliefs, make the same mistake: they regard the relationship between *programming* languages and reality just as the linguistic theories regard the relationship between *natural* languages and reality.

Ancient or modern, founded on mysticism or science, all attempts to find an ideal language, or a logically perfect language, have been sheer fantasies. It is important, however, to recognize their common fallacy. Philosophers recognize the potency of language, its power to describe and explain the world, to express knowledge and ideas; and they also recognize the power and simplicity of hierarchical structures, their ability to represent apparently complex phenomena with neat systems. These philosophers believe, then, that

[6] Ibid., p. 45.

[7] George Steiner, *After Babel: Aspects of Language and Translation*, 2nd ed. (Oxford: Oxford University Press, 1992), p. 59. [8] Ibid., pp. 60–61.

it is possible to combine the potency of language with the neatness of a hierarchical structure. The terminal elements of the hierarchy (the words, typically) would correspond directly to the basic objects, processes, and events that make up the world; and the rules that define the relations between elements in language would correspond to the natural laws that govern the relations between the real things. Since such a system would provide an exact correspondence between words and reality, it would constitute a perfect language.

What these philosophers fail to see is that the potency of language derives, not from a capability to mirror the world through one-to-one correspondence, but from its capability to generate interacting structures. The structures formed by language elements interact with one another, and also with the knowledge structures present in human minds, and with the structures formed by human societies. It is the complex structures generated by these interactions that are the source of richness and potency in language.

It is futile, therefore, to try to explain the world by inventing a language that is better, or more logical, than our natural languages. For, if we cannot explain a complex phenomenon, the problem is not the imperfection of our languages but the complexity of the world, and also the lack of certain knowledge structures in our minds. If we are at all capable of understanding a certain phenomenon, we will understand it through any language, for the language structures themselves play only a small part in this process. It is their interaction with various knowledge structures present in our minds that gives rise to the intelligence we need to understand a complex phenomenon.

And so it is with our *software* structures. If our applications fail to answer our needs (fail, that is, to provide an exact correspondence to the world), the problem is not the imperfection of our programming languages or development tools, but the complexity of the world, and the lack of adequate knowledge structures (that is, programming skills) in our minds. Consequently, it is futile to seek a solution by improving the programming languages or the development tools. As is the case with natural languages, the richness and potency of software derives, not from a capability to mirror the world perfectly (something no programming language or tool can do), but from its capability to generate interacting structures; namely, structures that interact with one another, with the knowledge structures present in our minds, and with the structures formed by our social and business affairs. It is the complex structures emerging from these interactions that we observe as software benefits.

Therefore, when the benefits are not forthcoming, improving the *language* structures will not solve the problem; it is the *knowledge* structures that we must improve. For, the programming languages themselves play only a small part in this process, and, in any case, they are already adequate. It is not a new language or a new development system that we need, but greater programming knowledge.

The Search for the Perfect Language

1

Let us start with the philosophies of the seventeenth century. The three great rationalists, Descartes, Spinoza, and Leibniz, believed that everything can be represented in the form of a deductive system; namely, as a hierarchical structure similar to the system of mathematics. Their philosophies varied in detail, but were all based on the vague notion of *substances* (the hypothetical things that make up the world).

Substances are taken to be independent of one another. In particular, the material world is made of one kind of substance, and mental processes are made of a different kind. There are objects that occupy physical space, attributes that can be perceived, and events that can be observed to occur in time; then, there are their mental counterparts – the notions we hold in the mind when we are aware of these objects, attributes, and events. The real things are said to have an *extension*, while thoughts and feelings do not. There were difficulties in explaining how the material world interacts with the mental one (as when a thought causes the movement of a limb), but these difficulties could usually be resolved by postulating divine intervention in one form or another.

In their effort to explain the world mathematically – that is, with simple structures – the rationalists had to assume that it is made up of several independent aspects, each one consisting of a different substance. They could not explain the *real* world – the complex structure – mathematically, so they extracted various aspects, thinking that by explaining each one separately they would eventually explain the world. And two aspects they always had to separate were the material and the mental.

Thus, Descartes's system "presents two parallel but independent worlds, that of mind and that of matter, each of which can be studied without reference to the other."[1] Spinoza held that the world is only one substance, one system,

[1] Bertrand Russell, *A History of Western Philosophy* (New York: Simon and Schuster, 1972), p. 567.

but had to separate the physical from the mental anyway. He interpreted them as two attributes, each one providing "complete and adequate knowledge of the essence of a single substance. Thought and extension each represents reality as it essentially is, and each attribute gives a *complete* account of that reality."[2] Leibniz preferred a system called *monadology*, according to which the world is a hierarchical structure of *monads*: atomic entities that form independent substances while also being related through the mystical quality known as "pre-established harmony." Leibniz was trying to explain how physical and mental structures, which are different substances and do not interact, can nevertheless influence each other: some of the properties of monads give rise to physical structures, and others to mental structures, but the pre-established harmony keeps the two kinds of structures synchronized at all times.

The dichotomy of mind and matter has remained a major problem of philosophy. This problem can be described as the problem of accounting for mental phenomena by means of mechanistic theories: explaining how it is possible for such phenomena as knowledge, consciousness, intelligence, and emotions, which have no material existence, to arise in a world made up of material entities. It ought to be obvious that mental phenomena are complex structures, and thus irreducible to deterministic models; but such an explanation is inadmissible in a mechanistic culture like ours. So, for more than three hundred years, one philosophy after another has been advanced in an attempt to bridge the gap between the mental and the material worlds; that is, to reduce mental phenomena to physical processes. Today's theories of artificial intelligence, for example, are merely the modern equivalent of the mechanistic fantasies of the seventeenth century: just another attempt to account for consciousness and intelligence by means of deterministic models.

Returning to the rationalist philosophers, we can understand why they liked the two-world conception of reality – the separation of reality into a material world and a mental world. By accepting this notion, they abandoned in effect the goal of understanding the real world, and replaced it with the lesser challenge of understanding the material and the mental worlds independently. Now they were able to apply the mechanistic principles, reductionism and atomism, to two separate worlds, which individually are simpler than the real one.

In the material world, this has resulted in the spectacular advances in science and technology of the last three centuries. In the mental world, the benefits of the mechanistic methods have been much more modest. The reason is that mechanistic theories and models, as embodied in mathematical systems and other simple hierarchical structures, can provide only *approximations* of

[2] Roger Scruton, *Spinoza* (Oxford: Oxford University Press, 1986), pp. 56–57.

reality. In the material world, many phenomena are sufficiently independent – sufficiently isolated from other phenomena – for these approximations to be useful. But the phenomena created by human minds cannot be studied in isolation, because they are closely linked. Moreover, they are linked to the phenomena of the material world. A mental world separated from the material world can exist only in our imagination. In practice we cannot observe a person's actual thoughts or feelings, but only the phenomena that arise when those thoughts and feelings interact with the material world.

The conclusion must be that processes occurring in a mind cannot be studied as we do those involving physical objects. Searching for a theory that explains mechanistically mental phenomena is an absurd pursuit, so we must not be surprised that so little progress has been made in disciplines like psychology and sociology. If there exists a way to explain mental phenomena, it must involve the study of the complex structures of the real world – structures created by the interaction of the phenomena of one mind with those of other minds, and with those occurring in the material world. This may well be an impossible challenge.

❖

When the rationalist philosophers addressed themselves to the problem of language, they naturally tried to solve it as they had tried to solve the problem of mind: by separating language from the complex reality. Just as they had separated the real world into mental and material worlds, they treated language as yet another world – a third kind of substance, capable of independent existence. They believed that if such matters as grammar and word meaning were fairly well understood on their own, there was no reason why they could not discover a complete theory of language, a model that explained with precision every aspect of it. And, needless to say, "they held that a truly systematic conception and formation of language could be obtained only through the application of the method and standards of mathematics."[3] Language, they believed, is a mechanistic phenomenon, a system of things within things, and must be investigated in the same way as the phenomena of the material world.

Language, however, is a *complex* structure, the result of many interacting structures; and it involves not only linguistic structures like syntax and semantics, but also various knowledge structures present in the mind, and the structures that make up the context in which it is used. Since they could not

[3] Ernst Cassirer, *The Philosophy of Symbolic Forms*, vol. 1, *Language* (New Haven: Yale University Press, 1955), p. 127.

explain the whole phenomenon of language, the rationalists attempted to create exact linguistic models by reducing language to one of its component structures; typically, a structure based on grammar, or one based on word meaning. But this reification can yield only a crude approximation of the whole phenomenon.

The rationalist philosophies of the seventeenth century ended up depicting reality, thus, as three independent structures: the material, the mental, and the linguistic. These three structures were seen as three parallel worlds, each one reflecting reality, and each one being a reflection of the others.

But the notion of three parallel and independent worlds did not originate with the rationalists. It went back, in fact, at least to the thirteenth century, when a group of scholars called Modistae "asserted a relation of *specular* correspondence between language, thought and the nature of things. For them, it was a given that the *modi intelligendi* [i.e., mental forms] and, consequently, the *modi significandi* [i.e., symbolic forms] reflected the *modi essendi* [i.e., actual forms] of things themselves."[4] They believed, in other words, that the linguistic symbols we use to express knowledge correspond to both the knowledge held in the mind and the actual things depicted by that knowledge.

It is this fallacy – the idea that language structures can map mental structures and material structures perfectly, coupled with the idea that all three are neat hierarchical structures and hence amenable to mechanistic treatment – that we can recognize in all the theories we are examining here, down to the theories of our own time. The fallacy can manifest itself in one of two ways: the philosophers believe that *natural* languages form simple structures, and therefore any language, if carefully employed, can reflect perfectly the mental world or the material world (also taken to form simple structures); or they believe that an *artificial* language based on a simple structure can be invented – a language which would reflect perfectly the mental or the material world. Let us start by examining how these two beliefs influenced the philosophies of the seventeenth century; then, we will study their influence in the twentieth century, and how they gave rise to the current *software* delusions.

2

The most naive manifestation of the language fallacy was the belief that it is possible to use ordinary language to build systems of logical concepts, simply by arranging these concepts into neat hierarchical structures that emulate the deductive method of mathematics.

[4] Umberto Eco, *The Search for the Perfect Language* (Oxford: Blackwell, 1997), p. 44.

In a mathematical system we start with a small number of axioms, and derive simple theorems by showing that their validity can be logically deduced, or demonstrated, from these axioms. Then, we derive more complex theorems by deducing them from the simple ones; and we continue this process with more and more complex theorems, on higher and higher levels. Ultimately, by creating a hierarchical structure of theorems, we can confidently determine the validity of very difficult ones.

The rationalist philosophers believed, as we saw, that language can provide an exact, one-to-one correspondence to the material and the mental worlds, and that all three can be represented with simple hierarchical structures. So, they concluded, if they expressed their arguments in ordinary language very carefully, the resulting sentences would function just like the axioms and theorems of mathematics: they would start with simple assertions about the world – assertions whose truth no one doubts, and which could therefore act as axioms; then, they would formulate more and more ambitious statements by moving up, one level at a time, and expressing each statement as a logical combination of statements from the previous level.

This method, they believed, is identical to the deductive method employed in mathematics, so it should allow us to determine the truth of the most difficult statements. But language mirrors reality, so these demonstrations will function at the same time as demonstrations of certain states of affairs that exist in the world. Simply by manipulating sentences, then, we will be able to determine the validity of any concept, including concepts that cannot be verified directly.

Thus, with this method Descartes needed only a few pages of text to "establish the existence of God, and the distinction between the mind and body of man."[5] The logic system through which he "establishes" these facts includes: ten definitions (for example, definition I is "By the term *thought* … I comprehend all that is in us, so that we are immediately conscious of it…."[6]); seven postulates, which preface and explain ten axioms (for example, axiom I is "Nothing exists of which it cannot be inquired what is the cause of its existing…."[7]); and four theorems, or propositions, with their demonstrations.

Similarly, in the *Ethics*, Spinoza presented his entire philosophy in this fashion. The book consists of definitions, postulates, axioms, propositions, demonstrations, and corollaries, arranged in five hierarchical structures, just like mathematical systems. Spinoza addresses such topics as the existence of God, knowledge and emotions, the relation between matter and mind, the power of reason over passion, and human freedom. And Samuel Clarke,

[5] René Descartes, *A Discourse on Method* (London: Dent, 1912), p. 229. [6] Ibid.
[7] Ibid., p. 232.

another philosopher of that period, presented in the form of a mathematical system his ideas on religion and morals.

We are not concerned here with the content or merit of these theories, but only with their mathematical pretences. These philosophers were convinced that, by mimicking in ordinary language the deductive methods of mathematics, they were actually proving their "theorems." This delusion is easy to understand if we remember their belief that the knowledge embodied in a set of sentences can form a self-contained system, an independent structure. They fail to see that the reason we can comprehend these sentences at all – the reason they can communicate their ideas to us through language – is the common knowledge structures that both they and we *already hold* in our minds. Thus, it is the *combination* of this previous knowledge and the knowledge found in the new sentences that forms, in reality, their philosophical systems.

The sentences themselves may or may not form a perfect hierarchical structure, but in either case the new knowledge is the result of several interacting structures. What we derive from reading the axioms, propositions, and demonstrations is not just a new and independent structure of neatly related concepts, but the complex structure formed by the interaction of these concepts and many *other* concepts, which already exist in our minds. For, if this were not the case, if all the concepts required to understand a philosophical system were contained in the structure of sentences alone, we could implement them as a *software* system – by storing the definitions, axioms, and propositions in the elements of a hierarchical data structure. Then, simply by interpreting the high levels of this structure, the computer would understand the idea of good and evil, or the meaning of reason and passion, or the value of freedom, just as *we* do – a preposterous notion.

The delusion becomes obvious, thus, when we represent these verbal theorems as simple and complex structures. Entities can function as the starting elements of a simple structure only if atomic and independent, otherwise they give rise to a *complex* structure. The rationalist philosophers invoke the mechanistic principles of reductionism and atomism, but do not, in fact, follow them rigorously: they employ as starting elements – as definitions, postulates, and axioms – entities that are neither atomic nor independent. To use as definition, postulate, or axiom a sentence like the ones previously quoted, we must know the meaning of its words, understand the facts asserted, and place these facts in the current context. So we must appreciate its significance, and we do this by analyzing and interpreting it on the basis of *previous* knowledge – the same knowledge that helps us to appreciate the significance of the *other* definitions, postulates, and axioms. Thus, since the starting elements are interrelated, what these philosophers are building is not an isolated hierarchical structure but a system of interacting structures.

These philosophers also fail to see that if they could use the neat, deductive methods of mathematics with these topics, they wouldn't need all those sentences in the first place. They could substitute symbols like *x* and *y* for the starting elements, and then restrict themselves to manipulating these symbols with equations, as we do in mathematics. It is precisely because they need to express concepts that *cannot* be represented mathematically – specifically, because they want us to link the new structures with some *other* knowledge structures – that the philosophers use *verbal* theorems and demonstrations. They need the words because it is only through the multiple meanings of words that we can link these structures. But then, how can they continue to believe that those verbal hierarchies function as isolated structures, as mathematical systems? Since their method is unsound, their conclusions are unwarranted, and this casts doubt on their entire philosophy.

This is what George Boole showed nearly two centuries later.[8] We know Boole as the mathematician who established modern symbolic logic, and in particular, what is known today as Boolean logic – the system that provides the mathematical foundation for (among other things) digital electronics, and hence computers. In Boolean logic, entities are reduced to the values *0* and *1*, or *False* and *True*, and are manipulated with logical operators like AND, OR, and NOT; large and intricate hierarchical structures can be built starting with these simple elements and operations.

Like the rationalist philosophers, Boole held that a logic system is not limited to mathematical problems, but can also be used with general propositions. Unlike them, however, he recognized that it is adequate only in situations that can be reduced to a symbolic form. Thus, he criticized Spinoza and Clarke, showing that the ideas they attempted to prove with deductive methods do not lend themselves to this treatment. As a result, their philosophical systems – which continue to be seen even today as impeccable – are not as sound as they appear: "In what are regarded as the most rigorous examples of reasoning applied to metaphysical questions, it will occasionally be found that different trains of thought are blended together; that particular but essential parts of the demonstration are given parenthetically, or out of the main course of the argument; that the meaning of a premiss may be in some degree ambiguous; and, not unfrequently, that arguments, viewed by the strict laws of formal reasoning, are incorrect or inconclusive."[9]

Boole checked some of Spinoza's and Clarke's demonstrations by substituting symbols for their verbal propositions, and then reproducing their verbal arguments by means of logical formulas. Thus, he notes that some of the

[8] George Boole, *The Laws of Thought* (New York: Dover, 1958), ch. XIII.
[9] Ibid., p. 186.

original definitions and axioms turn out to be vague or ambiguous when we try to represent them with precise symbols. And, even when they can be accurately reduced to logical formulas, he notes that some of the conclusions cannot, in fact, be logically derived from the premises.

3

So far we have examined the belief that natural languages can be used like mathematical systems. The most common manifestation of the language fallacy, however, is the belief that our natural languages are hopelessly inadequate for expressing rational thought; that only a formal system of symbols and rules can accurately reflect reality; that we must start from scratch, therefore, and *invent* a perfect language.

Descartes, who held that the totality of human knowledge can be represented, as it were, with one giant structure of things within things, envisaged the possibility of a universal language that would express this knowledge: all we have to do is create a hierarchical linguistic structure that matches, element for element, the whole structure of knowledge. He believed that "just as there is a very definite order among the ideas of mathematics, e.g., among numbers, so the whole of human consciousness, with all the contents that can ever enter into it, constitutes a strictly ordered totality. And similarly, just as the whole system of arithmetic can be constructed out of relatively few numerical signs, it must be possible to designate the sum and structure of all intellectual contents by a limited number of linguistic signs, provided only that they are combined in accordance with definite, universal rules."[10] Unfortunately, Descartes admitted, the only way to design such a language is by first determining all the elements in the hierarchy of knowledge – a task that would require "the analysis of all the contents of consciousness into their ultimate elements, into simple, constitutive 'ideas.'"[11]

Descartes never attempted this project, but his immediate successors did: "In rapid sequence they produced the most diverse systems of artificial universal language, which, though very different in execution, were in agreement in their fundamental idea and the principle of their structure. They all started from the notion that [the totality of knowledge is ultimately based on] a limited number of concepts, that each of these concepts stands to the others in a very definite factual relation of coordination, superordination or subordination, and that a truly perfect language must strive to express this natural hierarchy of concepts adequately in a system of signs."[12]

[10] Cassirer, *Symbolic Forms*, p. 128. [11] Ibid. [12] Ibid.

Two language systems designed as a hierarchical classification of concepts were those of George Dalgarno and John Wilkins.[13] These men attempted to classify all known objects, attributes, qualities, relations, actions, etc., into one giant hierarchical structure. Each one of these entities was believed to occupy a precise place in the structure of knowledge, and to form therefore one of the terminal elements in the structure of words that mirrors the structure of knowledge. But familiar words can be misleading, so Dalgarno and Wilkins invented elaborate systems of symbols to represent these elements, as well as rules of grammar to combine them into sentences. It was widely believed that an artificial language of this kind, designed logically from scratch, would enable scientists and philosophers to express ideas more effectively than it is possible with our natural languages. And few doubted that such languages can, indeed, be invented.

Among the seventeenth-century language fantasies, it is Leibniz's work that has probably received the most attention. Although Leibniz did not actually attempt to create a language system, he was preoccupied with the relation between language and knowledge throughout his career. Leibniz's work was guided by two beliefs. First, anything that is complex – mental as much as material – is necessarily made up of simpler things; and these things, if still complex, are made up of even simpler things, and so on, ending eventually with some elements that are no longer divisible into simpler ones. These elements function, therefore, as an "alphabet": they are the building blocks from which everything is made up.

This, of course, is a belief in reductionism and atomism. Applying these principles to mental entities, Leibniz held that there must exist an "alphabet of human thought": a set of simple, elementary concepts that constitute the building blocks of all knowledge, of all science and philosophy, of all truths known or yet to be discovered. It is the use of imprecise natural languages to express ideas that leads us into error and slows down intellectual progress. With an alphabet of thought we could formulate inquiries logically, solve problems rationally, and quickly expand our knowledge. All we would have to do is combine the basic elements of thought into more and more complex concepts, one level at a time, following precise rules. We could then deal safely with concepts of any complexity, because their validity would be guaranteed by the method's formality.

Leibniz's second belief, thus, was that a logical language is not limited to helping us express what we already know, but can also help us attain *new* knowledge. The language of mathematics, for example, allows us to represent very large numbers, or very complex relations, by starting with a small set of

[13] See, for example, Eco, *Perfect Language*, chs. 11–12.

symbols and rules. By combining and manipulating these symbols logically on paper, we have made great discoveries about the real world; that is, about the actual, physical entities represented by the symbols. These discoveries, clearly, could not have been made by observing or manipulating the physical entities themselves.

Similarly, Leibniz believed, a language based on an alphabet of thought would enable us to solve problems and to make progress in any domain that involves rational thinking. He called the symbols used to represent the elementary concepts *characters*, and the language *universal characteristic*. This language, he believed, would function as a sort of mathematical system: "It is obvious that if we could find characters or signs suited for expressing all our thoughts as clearly and exactly as arithmetic expresses numbers or geometrical analysis expresses lines, we could do in all matters *insofar as they are subject to reasoning* all that we can do in arithmetic and geometry. For all investigations which depend on reasoning would be carried out by the transposition of these characters and by a species of calculus."[14] Thus, Leibniz was convinced that it is possible to invent a language "in which the order and relations of signs would so mirror the order and relations of ideas that all valid reasoning could be reduced to an infallible, mechanical procedure involving only the formal substitution of characters."[15]

❖

The fallacy of these language systems lies in the belief that human knowledge can be represented with *one* hierarchical structure. We recognize this as the fallacy of reification. When a hierarchical classification of concepts is used as the basis of a universal language, what is wrong is not the classification, which in itself may be accurate and useful. The reason the system fails is that there are *additional ways* to classify the same concepts, on the basis of other attributes, or characteristics, or criteria (see pp. 98–102).

Only elements that are atomic and independent can function as starting elements in a simple hierarchical structure; and it is impossible to find such elements in the phenomenon of knowledge. The concept of a flower, for example, is an element in many knowledge structures: one that depicts the botanical aspects of flowers, one that depicts the business of flowers, one that depicts social customs involving flowers, and so forth. These are not distinct subsets of a hierarchical structure of knowledge, but different structures that

[14] Gottfried Leibniz, quoted in Donald Rutherford, "Philosophy and Language in Leibniz," in *The Cambridge Companion to Leibniz*, ed. Nicholas Jolley (New York: Cambridge University Press, 1995), p. 234. [15] Rutherford, "Leibniz," p. 231.

share some of their elements. The concept of a flower, therefore, is not a starting element in the structure of knowledge envisaged by mechanists. And breaking it down into even simpler concepts would not help; for, were the lower-level elements independent of other knowledge structures, the concept of a flower itself would be independent.

The same is true of any other concept. When Leibniz attempts to discover the alphabet of human thought by repeatedly breaking down complex concepts into simpler ones, what he fails to see is that, at each level, he is taking into account *only one* of the attributes that characterize the concepts of that level. There are always other ways that a concept can be expressed as a combination of simpler ones. We can stop, in fact, at any level we like and call its elements the alphabet of thought; but these elements always have other attributes besides those we took into account, so they are part of other structures too. These elements, therefore, are not independent, so they cannot function as the starting elements of a simple hierarchical structure.

When we acquire knowledge, our mind develops structures matching *all* these classifications, and their interactions. This is why we can *acquire* and *use* knowledge, while being unable to represent the same knowledge with rules or diagrams. The knowledge is embodied largely in the *interactions* between structures, and it is these interactions that can exist in a mind but not in a mechanistic model.

It is not difficult, then, to find some building blocks of human thought. But, while the elements at any level could be called the alphabet of thought, they would not serve this function as Leibniz hoped (in the way the building blocks of mathematics are the foundation of mathematical systems). We could indeed create the entire human knowledge from these elements, but only by taking into account *all* their attributes: we would have to combine the elements through many structures simultaneously, so what we would be building is a *complex* structure.

For example, if we decide that certain *words* should function as the alphabet of thought, it is not enough to see them as the elements of *one* structure. When words are used to express ideas, their meanings give rise to many structures: just as the things themselves link, through their attributes, the structures that *constitute* reality, the words depicting these things link in our mind the knowledge structures that *mirror* reality. Knowledge, therefore, like reality itself, is a complex structure, and cannot be represented with one hierarchy, as Leibniz hoped. (We will return to this problem in the next section.)

But perhaps Leibniz *did* recognize the limitations of simple structures. In the passage previously quoted, for instance, he emphasizes that his system would be useful "in all matters *insofar as they are subject to reasoning*." His error, therefore, is perhaps not so much in believing that *all* knowledge can be represented with simple structures, as in failing to see how little *can* in fact be so represented; that is, how little is "subject to reasoning," if we define reasoning in the narrow sense of mathematical reasoning. For, if we did that, we would have to declare almost all knowledge as *not* being subject to reasoning. His system could indeed work, but only if we restricted our mental acts to whatever can be accomplished with neat mathematical methods; in other words, if we restricted human thought to the capabilities of machines.

4

We cannot leave this early period without recalling the satire of Jonathan Swift. Through imaginative writing, Swift ridiculed the society of his time, and especially the ignorance, hypocrisy, and corruption of the elites. From politics to religion, from education to morals, from arts to science, he fearlessly questioned all accepted values. In his best-known work, *Gulliver's Travels*, the hero finds himself in some strange lands, giving Swift the opportunity to expose the preoccupations of the contemporary British society by projecting them in modified and exaggerated forms onto the fictional societies of those lands.

Now, one of the things that Swift detested was the excesses of the mechanistic philosophy, which by the end of the seventeenth century had become for most scientists practically a new religion: the belief that mechanistic theories can explain any phenomenon; the obsession with finding a neat model that would describe the whole world; and, of course, the search for an artificial language that would mirror reality perfectly.

In his voyage to Laputa, Gulliver has the occasion to visit the Grand Academy of Lagado, where hundreds of "projectors" are engaged in fantastic research projects: ideas that promise extraordinary benefits to society, although so far none of them work. Here Swift is ridiculing the Royal Society and other institutions, which, in their infatuation with mechanism, were studying quite seriously all sorts of utopian schemes. And some of these schemes involved the use of artificial languages. (In 1668, for example, the Royal Society appointed a commission of distinguished scientists to study the possible applications of the language invented by John Wilkins, mentioned earlier; see p. 312.[16])

Swift recognized the absurdity of the mechanistic language theories; specifically, the belief that the totality of knowledge can be represented with exact, mathematical systems, which allows us to treat knowledge mechanically. To satirize these theories, he describes the language machine invented by one of

[16] Eco, *Perfect Language*, p. 229.

the professors at the Academy: a mechanical contraption that generates random combinations of words. Then, simply by selecting those combinations that form meaningful sentences, any person can create any text in a given domain: "Every one knows how laborious the usual method is of attaining to arts and sciences; whereas by his contrivance, the most ignorant person at a reasonable charge, and with a little bodily labour, may write books in philosophy, poetry, politics, law, mathematics and theology, without the least assistance from genius or study."[17] (We will discuss the delusion of language machines in greater detail in chapter 6; see pp. 447–450.)

Then, to ridicule the idea that the elements of language mirror the world through one-to-one correspondence, Swift describes another research project: abolishing language altogether and communicating instead directly through physical objects (which would obviate both the need to produce sounds and the need to translate words from one language into another): "Since words are only names for *things*, it would be more convenient for all men to carry about them such *things* as were necessary to express the particular business they are to discourse on."[18]

Our challenge today is to recognize that our preoccupation with programming languages and systems stems from delusions that are the software counterpart of the mechanistic language delusions of previous ages. When a programming theory claims that our affairs constitute a neat hierarchical structure of concepts, and therefore applications should be built as neat hierarchical structures of modules, we are witnessing the same fallacy as in Leibniz's idea of human knowledge and the mathematical language that would represent it.

Structured programming methodologies, object-oriented systems, fourth-generation languages, not to mention pursuits like Chomskyan linguistics and software models of mind – these are the projects that Gulliver would have found the professors of the Grand Academy engaged in, had Swift lived in our time. And the language machine that permits ignorant people to write books on any subject has its counterpart today in the application development environments, programming tools, database systems, and other software devices that promise ignorant programmers and users the power to generate applications "without writing a single line of code." (We will study these delusions in chapters 6 and 7.)

[17] Jonathan Swift, *Gulliver's Travels and Other Writings* (New York: Bantam Books, 1981), p. 181. [18] Ibid., p. 183.

5

Let us pass over the dozens of other language proposals and go straight to the last years of the nineteenth century, when the modern theories of language and meaning were born. The first thing we notice is that there is little difference between these theories and the earlier ones, as the passage of two centuries did not alter the two fundamental beliefs: that there exists a one-to-one correspondence between language and reality, and that both can be represented as simple hierarchical structures.

A major concern of twentieth-century philosophy has been the formal analysis of language structures – not in the linguistic sense, but as regards their *meaning*; specifically, the study of the relationship between statements and reality. One of the aspects of this study has been the attempt to derive and interpret the meaning of ordinary sentences using the methods of formal logic; that is, to determine from the grammatical and logical structure of a sentence whether it describes something that actually exists in the world. This is how Irving Copi puts it: "The linguistic program for metaphysical inquiry may be described as follows. Every fact has a certain ontological form or structure. For a given sentence to assert a particular fact, the sentence must have a grammatical structure which has something in common with the ontological structure of the fact. Hence, on the reasonable expectation that sentences are easier to investigate than the facts they assert, the examination of sentences will reveal metaphysical truths about the world."[19]

But, philosophers say, while the world has presumably a neat and logical structure, our natural languages do not, so we will never be able to understand the world through ordinary language. We must design, therefore, a special language: "The relevance of constructing an artificial symbolic language which shall be 'ideal' or 'logically perfect' to the program for investigating metaphysics by way of grammar is clear. If we have a 'logically perfect' language, then its structure will have something in common with the structure of the world, and by examining the one we shall come to understand the other."[20] Copi points out, however, that "even if an 'ideal' or 'logically perfect' language could be devised, the proposed program for investigating the logical or ontological structure of reality by investigating the syntactical structure of an 'ideal' language is impossible of fulfillment."[21] The reason is simple: if a

[19] Irving M. Copi, "Artificial Languages," in *Language, Thought, and Culture*, ed. Paul Henle (Ann Arbor: University of Michigan Press, 1965), p. 108. [20] Ibid., p. 109.
[21] Ibid., p. 110.

language is to mirror reality perfectly, we must understand reality before we design that language, thus contradicting the original goal of designing the language in order to understand reality.

These linguistic theories demonstrate once again the circularity that characterizes mechanistic thinking. The philosophers attempt to describe the world with languages based on simple structures because they *assume* that the world has a neat hierarchical structure. They start by assuming the truth of what, in fact, needs to be proved. For, the structure of the world is what they do not know, what those languages were meant to uncover. In other words, they use their mechanistic fantasy about the world to justify their mechanistic fantasy about language. Actually, the world is not a simple structure but the interaction of many structures, so it is not surprising that these languages do not work. It is *because* they are logically perfect that they cannot mirror the world. Thus, the mechanistic language dream – a complete analysis of language and knowledge through mathematical logic – was never attained.

The first philosopher to investigate this possibility was Gottlob Frege, who rejected the view that a precise language like the symbolic language of mathematics can represent only the formal aspects of knowledge. He held that *all* human thought can be reduced to a precise language – a language that can "be taken care of by a machine or replaced by a purely mechanical activity."[22] Frege, however, "recognized from the very beginning that for most sentences of natural languages 'the connection of words corresponds only partially to the structure of the concepts.' But instead of drawing Kant's defeatist conclusion, Frege attempted to identify what others would call a 'perfect language,' a fragment of German that expressed perspicuously the content of what we say."[23]

The language fragment Frege was seeking had to fulfil two conditions: "(a) Every German sentence has a translation into this fragment, and (b) the grammatical form of every sentence in this fragment mirrors isomorphically the constituents of the content it expresses, as well as their arrangement in that content.... In effect, the idea was to produce a language in which, even though inference was based on meaning, one need no longer think about meanings ... since one could now restrict oneself to the signs 'present to the senses' and their symbolic correlations."[24] What Frege was seeking, thus, was a symbolic system that would fulfil for all the knowledge we can express in a natural language like German, the same function that the symbolic language of mathematics fulfils for that portion of knowledge we can express mathematically.

[22] Gottlob Frege, quoted in J. Alberto Coffa, *The Semantic Tradition from Kant to Carnap* (New York: Cambridge University Press, 1993), p. 65.

[23] Coffa, *Semantic Tradition*, p. 64. [24] Ibid., p. 66.

And, once this perfect language is discovered, the interpretation of the meaning of sentences will be automatic. By translating all discourse into this language, we will be able to determine whether a given statement is meaningful or not "by a purely mechanical activity": all we will have to do is manipulate symbols, just as we do in mathematics.

❖

In the first two decades of the twentieth century, it was Bertrand Russell and Ludwig Wittgenstein who made the greatest contribution to the philosophy of language. And, since the two men collaborated during this period, their theories have much in common. Wittgenstein's most important work, however, was done later. For this reason, and because Wittgenstein's philosophy is especially important to us, I will discuss it separately in the next section.

Russell based his philosophy of language on a theory he called *logical atomism* – the analysis of language using the principles of atomism and mathematical logic. He developed and modified this philosophy over a period of more than forty years, yet his goal remained the same: to find a formal, mathematical language that can express with preciseness all knowable facts, and hence all mental processes. His goal, in other words, was to represent human knowledge and thought with a neat structure of concepts within concepts. And although the theory never worked, Russell continued to believe in the possibility of such a language to the end of his life: "There is, I think, a discoverable relation between the structure of sentences and the structure of the occurrences to which the sentences refer. I do not think the structure of non-verbal facts is wholly unknowable, and I believe that, with sufficient caution, the properties of language may help us to understand the structure of the world."[25]

Thus, Russell's linguistic project is an excellent example of the futile struggle to reduce complex phenomena to simple structures. It is also an example of the corruptive effect of our mechanistic culture – the effect I described in chapter 3 in connection with Chomsky's work (see p. 280): Russell was a professional logician and philosopher, but his mechanistic beliefs compelled him to pursue absurd linguistic theories, not unlike those of crank intellectuals. He was also a humanist, but at the same time he was convinced that the phenomena of knowledge and intelligence can be explained with deterministic theories. He failed to see that what he was trying to prove was, in effect, that human minds are no different from machines. While as humanist he was

[25] Bertrand Russell, *An Inquiry into Meaning and Truth*, rev. ed. (London: Routledge, 1995), p. 341.

concerned with freedom, justice, and peace, as scientist he promoted theories that, although invalid, undermine our respect for human beings.

❖

Russell stresses the mechanistic nature of his philosophy of language. The principles of reductionism and atomism figure prominently in his theory, and it is obvious that the formal language he is seeking would form a simple hierarchical structure: "My own logic is atomic, and it is this aspect upon which I should wish to lay stress."[26] Thus, Russell maintains that there are two kinds of entities, "simples" and "complexes."[27] Simples are the atoms of thought, what we find at the limit of analysis, and are represented in language with symbols (or names). Complexes are those entities that can still be divided into simpler ones; they are not represented with symbols, since they are merely combinations and relations of simples: "I confess it seems obvious to me (as it did to Leibniz) that what is complex must be composed of simples, though the number of constituents may be infinite."[28]

We have yet to discover this symbolic language, Russell admits, but we can assume that it will have several levels of abstraction, and that the levels will reflect the actual facts: the complexity of the elements at each level will match the complexity of the facts described by these elements: "I shall therefore … assume that there is an objective complexity in the world, and that it is mirrored by the complexity of propositions."[29] Basic language elements will represent simple facts directly, and combinations of elements will represent complex facts: "In a logically perfect language the words in a proposition would correspond one by one with the components of the corresponding fact, with the exception of such words as 'or,' 'not,' 'if,' 'then,' which have a different function. In a logically perfect language, there will be one word and no more for every simple object, and everything that is not simple will be expressed by a combination of words."[30]

The simplest facts are those that are not deduced from other facts; that is, facts of which we are aware through direct knowledge or perception. (Russell's term for this awareness is *acquaintance*.) An example of a simple fact is "the possession of a quality by some particular thing."[31] More complex facts occur when two or more facts are combined with relations; for example, "*A* gives *B* to *C*."[32] Russell calls all these facts *atomic facts*, and the language elements that express them *atomic sentences*. He then defines certain operations –

[26] Bertrand Russell, *The Philosophy of Logical Atomism* (Peru, IL: Open Court, 1985), p. 157. [27] Ibid., p. 173. [28] Ibid. [29] Ibid., p. 58.
[30] Ibid. [31] Ibid., p. 59. [32] Ibid.

substitution, combination, generalization – by means of which increasingly complex propositions can be built.[33] For example, the operation of combination connects atomic sentences with words like *or*, *and*, *not*, and *if-then*, and yields *molecular sentences*; thus, "the truth or falsehood of a molecular sentence depends only upon that of its 'atoms.'"[34]

Russell calls "the assemblage of sentences obtained from atomic judgments of perception by the three operations of substitution, combination, and generalization, the *atomistic* hierarchy of sentences."[35] The principle of atomicity "asserts that all propositions are either atomic, or molecular, or generalizations of molecular propositions; or at least, that [if this is not true of ordinary languages] a language of which this is true, and into which any statement is translatable, can be constructed."[36]

Russell's mistake, like Leibniz's, is the mistake we note in all mechanistic delusions; that is, whenever scientists attempt to represent complex phenomena with simple structures (see pp. 313–315). They praise reductionism and atomism, but the starting elements in their structures are *not* atomic and independent, as starting elements must be. Russell calls his theory atomic, but his "atomic facts" are not atomic at all: they are relatively high-level elements. His logical atomism could perhaps work, but only if the reduction ended with some truly atomic and independent entities. Russell cannot perform such a reduction, so he starts his structure from certain "facts." But even those facts that he assumes to be perceived directly (like the possession of a quality by an object) are not really "simple": we appreciate their significance by relying on previous experiences and on the current context; that is, on the same knowledge we use to understand *other* facts. These facts are, therefore, interrelated. They derive from elements and interactions occurring at *lower* levels, so they form multiple, interacting structures. They are not the starting elements of a simple structure, as Russell assumes.

6

Let us turn next to the philosophical school called *logical positivism* (also known as logical empiricism), which flourished between the 1920s and the 1950s. Its best-known members were Moritz Schlick, Friedrich Waismann, Rudolf Carnap, Otto Neurath, and A. J. Ayer. Logical positivism was concerned with *verifiability*; namely, ways to determine from the logical structure of a sentence whether the facts it describes can actually occur.

[33] Russell, *Meaning and Truth*, pp. 194–197. [34] Ibid., p. 195. [35] Ibid., p. 197.
[36] Ibid., p. 266.

The logical positivists held that a sentence is meaningful only if what it says can be logically broken down into a combination of some basic statements – statements simple enough to verify through direct observation. A sentence is true if the basic statements are found to be true, and false otherwise; but in either case it is meaningful. And sentences that cannot be reduced to such basic statements must be considered, not just false, but meaningless. This view of meaningfulness (which is similar to that found in other mechanistic theories of knowledge) is useless as criterion of demarcation, however.

The logical positivists were attempting to establish a revolutionary ideology: a scientific philosophy, grounded entirely on verifiable propositions. As part of this project, they were trying to reduce all knowledge to a system of propositions related through the precise rules of symbolic logic. And they believed that a strict criterion of demarcation is essential, in order to ensure that the system includes all the scientific propositions and none of the metaphysical or meaningless ones. Their criterion, however, was so strict that it ended up labeling as meaningless practically all sentences, including the theories of empirical science. The reason is that only trivial statements and theories can be reduced to facts that are verifiable through direct observation. Almost all knowledge is based, ultimately, on various *hypotheses* about the world; that is, on assertions which cannot be verified.

Consequently, much of the subsequent work of the logical positivists consisted in searching for a way to resolve this difficulty, and to formulate a practical criterion of demarcation. But, as Popper showed, it is impossible to determine with absolute certainty the truth or falsehood of empirical propositions. Popper criticized the logical positivist project, and held that a criterion of demarcation must be based on falsifiability, not verifiability (see "Popper's Principles of Demarcation" in chapter 3).

Logical positivism was committed to the mechanistic principles of reductionism and atomism. Its chief contribution to the mechanistic culture was the *linguistic* interpretation of science: the attempt to reduce all scientific knowledge to a logical analysis of sentences. Thus, the mechanistic principles were to be applied, not directly to scientific knowledge, but to the *sentences* in which this knowledge is expressed (on the familiar assumption that linguistic structures mirror through one-to-one correspondence the reality they describe).

The logical positivists believed that there is no need to take into account such imprecise information as the context in which a sentence is used. They held that the *logical structure* of a sentence, if properly analyzed, contains all the information we need to determine whether what it expresses is meaningful or not. And if this is not entirely true of natural languages, they argued, we can undoubtedly invent a precise language into which and from which we can translate our sentences. Then, by expressing knowledge in *this* language, we will automatically restrict ourselves to meaningful propositions.

Many attempts were made over the years, especially by Carnap and Neurath, to design that precise language upon which all knowledge could be based. Some theories, for example, involved "protocol sentences": isolated statements that describe such simple and verifiable facts as the position of an object, a particular attribute, a movement, or the time of day. Since reality is ultimately made up of such simple facts, it was argued, the sentences describing these facts can act as the basic elements of discourse. We should be able, then, to express any proposition as a combination of these sentences. Other theories claimed that the language of *physics* must be considered the basic language of knowledge. The basic elements would then be sentences that describe simple processes in space and time. Since everything in the world is ultimately based on elementary physical processes, we should be able to reduce all propositions to linguistic structures built from sentences that describe basic physical processes.

None of these theories worked, but this did not stop the logical positivists from promoting an ambitious project – called *the unity of science* – which, they claimed, would be one of the benefits of a precise language. The unity of science is the culmination of the scientistic dream: a reduction of all knowledge, of all the theories from all sciences, to a common, universal representation. Carnap believed that this is the only way for science to advance, and that only the language of physics can provide a universal representation. We may well develop other universal languages, he says, but such languages would always be reducible to the language of physics: "Every systematic language of this kind can be translated into the physical language.... Because the physical language is thus the basic language of Science *the whole of Science becomes Physics*."[37]

Needless to say, the phenomena studied by sciences like biology, psychology, and sociology must also be reduced to the language of physics. The reason we have not been as successful in these disciplines as we have in physics is that their languages are specialized, and hence limited, unlike the language of physics, which is universal. Their reduction to the language of physics is, therefore, the only way to make progress in these disciplines.[38]

Recall the discussion of formal reductionism in chapter 1 (pp. 76–78): mechanists claim that everything in the world – from material entities to biological phenomena, mental acts, and social life – can ultimately be reduced to physics; and physics can be reduced to mechanics, to the motion of bits of

[37] Rudolf Carnap, *The Unity of Science* (Bristol: Thoemmes, 1995), p. 97.
[38] Ibid., p. 100.

matter. Viewed in this light, logical positivism, along with the concept of the unity of science, is just another manifestation of the reductionistic project – but in a linguistic dress. Since these philosophers believe that language structures can mirror the world through one-to-one correspondence, they inevitably invent theories that postulate, instead of the traditional reduction of biology, psychology, and sociology to the motion of bits of matter, the reduction of the *sentences* employed in these disciplines to *sentences* describing bits of matter.

7

The language delusions of the first half of the twentieth century are reflected in the software delusions of the second half. Our software delusions stem from the same fallacy: the belief that a language – a formal system of rules and symbols – can generate hierarchical structures that mirror reality perfectly. This was the belief of Russell and Carnap, but, while the language delusions are limited to theories, we are actually *implementing* their software counterpart.

The software counterpart of the search for the perfect language is the search for the perfect programming language, or the perfect development system, or the perfect database model, or the perfect application. We recognize it in the endless succession of programming theories, methodologies, environments, languages, and tools, and the perpetual changes, versions, and "generations."

The belief in a perfect language, like the mechanistic doctrine of which it is part, has undoubtedly influenced our conception of language and knowledge, of mind and society. But this is where the harm ended. Its software counterpart – the belief in a perfect programming system – is just as fallacious, yet the mechanists are now asking us to alter our lives, and to lower our expectations, in order to conform to software theories based on this fallacy. Despite the continued belief in a logically perfect language, we never downgraded our conception of human capabilities to what can be represented with simple structures – the only structures possible in such a language. But this is precisely what we do with software when we agree to depend on mechanistic concepts (theories, methodologies, programming aids, ready-made applications), whose express purpose is to restrict us to simple knowledge structures.

It bears repeating: The potency of language and software derives from their ability to mirror reality. They do not mirror reality, however, through structures that provide a one-to-one correspondence to the world. The world consists of complex structures, whereas the entities that make up language and software give rise to simple structures. What mirrors reality is the *interactions* between the structures of language or software, and between these and various knowledge structures.

The greatest thinkers of the twentieth century fell victim to the language fallacy and could not see that their ideas were practically identical to the language fantasies of earlier times. So we should not be surprised that so many people today fall victim to the software and programming fallacies. Russell and Carnap built elaborate logic systems, which may even be faultless, but which cannot represent reality – because the premises of one-to-one correspondence between language and reality, and of a simple hierarchical structure that can represent all knowledge, are both invalid. Similarly, mechanistic software theories may be faultless as logic systems, but are useless in practice, because they start from the same invalid premises. They cannot represent reality any better than can the language theories.

Russell, even after forty years of futile search for a language that would represent logically all knowledge, still did not admit that human minds hold types of knowledge which cannot be reduced to simple structures of symbols.[39] But, as we saw in chapter 2, practically all types of knowledge consist, in fact, of *complex* structures. No one has managed to represent the act of recognizing contexts, for example, as a precise structure of elementary mental acts; yet recognizing contexts is something we all do, continually and effortlessly. Thus, the knowledge involved in this act cannot be mapped perfectly in a language like the one proposed by Russell. It cannot be mapped in *any* language, because it consists of *interacting* structures, and a neat system of symbols can only map *individual* structures.

Similarly, the concepts of software engineering – the relational database model, object-oriented systems, structured programming, and the like – claim that the reality we want to represent with software can be mapped perfectly into hierarchical structures of symbols. But these concepts cannot work, because reality consists of *interacting* structures. For software as for language, it is the interactions that are lost when we attempt to map reality with precise systems of symbols. And when these interactions are important, the resulting systems can provide only a poor approximation of reality.

The language and software theories, thus, are part of the same project: the attempt to reduce to neat hierarchical structures the complex phenomena that make up the world. For software as for language, it is the same world that we try to map with simple structures of symbols, so there is no reason to expect the software theories to succeed where the language theories have failed.

[39] Russell, *Meaning and Truth*, pp. 327–330.