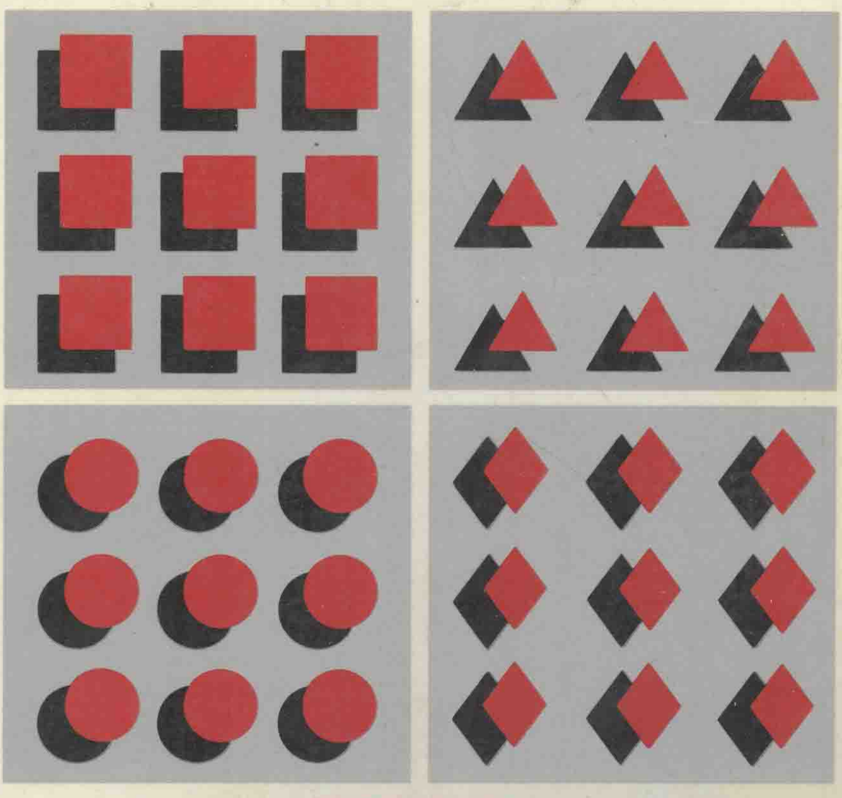
INFORMATION SYSTEMS FOR MANAGEMENT

A book of readings

Hugh J. Watson • Archie B. Carroll • Robert I. Mann



THIRD EDITION

INFORMATION SYSTEMS FOR MANAGEMENT

A Book of Readings

Edited by

Hugh J. Watson
Professor of Management and
Director of MIS Programs

Archie B. Carroll
Professor of Management

Robert I. Mann
Assistant Professor of Management

All of The University of Georgia

1987
Third Edition

BUSINESS PUBLICATIONS, INC.
Plano, Texas 75075

Previous editions of this book were published
under the title *Computers for Business.*

ISBN 0-256-03721-3

Library of Congress Catalog Card No. 86–71807

Printed in the United States of America

1 2 3 4 5 6 7 8 9 0 ML 4 3 2 1 0 9 8 7

INFORMATION SYSTEMS FOR MANAGEMENT

A Book of Readings

PREFACE

Computers are found in organizations of every kind and size. In fact, it is becoming increasingly difficult to find an organization that does not have several. Recognizing this trend, most colleges of business administration now require one or more courses in computers and information systems in their common body of knowledge. Without this training, graduates would be poorly equipped in the skills required to function effectively in the contemporary business world. With this training, graduates are ready to use and help manage this important organizational resource.

What do business school graduates need to know about computers and information systems? From the editors' perspective, the answer to this question is "A little about a lot of things." From our point of view, a student entering the business world needs to know about:

1. Computer hardware and software.
2. Computer-based information systems.
3. Management of information systems.
4. Computer impact on personnel, organizations, and society.

A knowledge of *computer hardware and software* is necessary in order to actively use the computer and communicate with data-processing specialists. In a previous course or the course that you are currently taking, you probably learned about bits, bytes, and other hardware and software concepts; how to use computer software for word processing, spreadsheet, and database applications; and perhaps how to write your own computer programs. In this book of readings we will assume an elementary knowledge of computer hardware and software and build from there.

In organizations that have computers, information about past, present, and future conditions are increasingly being provided by *computer-based information systems* (CBIS). These systems are evolving in sophistication in response to hardware and software advances, the efforts of end users and data-processing personnel, the information requirements of management, and so on. In this book we will explore areas such as the evolving performance capabilities of CBIS, their functioning, and what is required in developing a CBIS.

Computers and information systems are an important organizational resource. Companies have committed enormous sums of money to their purchase, development, operation, and maintenance. Many firms could not function or compete without them. Consequently, the *management of information systems* is an important organization concern not only for data-processing managers but general management. Given recent developments such as computer hardware and software advances, end users developing their own applications, and information systems being used for competitive advantage, managing computers and information systems is a challenging task.

Computers and information systems are responsible for a number of *impacts on personnel, organizations,* and *society.* They eliminate, modify, and create positions within the organization. They affect how managerial and nonmanagerial work is done. They create fear and resistance among those unfamiliar with the technology. They pose a challenge for both computer specialists and users as they strive to understand each other. Computers affect the organization's structure, with departments not only being created, modified, or eliminated, but the organizational placement of departments being changed. Finally, computers are affecting society in areas such as privacy, employment, and computer-related crime.

There is much that needs to be learned about computers and information systems in order to be an effective data-processing specialist, user, or manager. The objective of this book is to provide you with some of the knowledge that is required. It should assist you when entering a computerized environment to function effectively both personally and as a contributing organizational member.

This book of readings was prepared for use in courses with a significant information systems focus. Articles were selected on the basis of their readability, level of interest, contributions to understanding, and coverage of current and future topics of importance. Consequently, articles were selected from leading sources such as *Time, Harvard Business Review, Computerworld, MIS Quarterly, Journal of MIS, Sloan Management Review, Communications of the ACM,* and others. The topics covered include:

- How computers have worked in the past, how they work now, and how they will work in the future.
- How software advances are affecting programming productivity.
- Management information systems—concepts and applications.

- Decision support systems—concepts and applications.
- Expert systems—concepts and applications.
- The design of information systems.
- How to meet the information needs of top executives.
- How to manage existing applications.
- Key information systems management issues.
- End-user computing and its management.
- Information systems as a competitive weapon.
- The management of information systems.
- How computers and information systems affect jobs.
- How computers and information systems affect decision making.
- Ethical issues: privacy, accuracy, property, and accessibility.
- Computer-related crime.
- The future of humankind.

We want to acknowledge that the major contributions to this book come from the authors of the articles. While we have written introductions to each of the sections, the articles have made the substantive contributions. We would like to express our appreciation to the authors and the publishers of the journals in which the articles appeared for allowing us to reprint them. Also we would like to thank Marie Dent and Sandi Abicht for assisting in the development of this book.

Hugh J. Watson
Archie B. Carroll
Robert I. Mann

CONTENTS

PART ONE

Computer Hardware and Software

It is through computer hardware and software that computer-based information systems function. The hardware provides the required equipment, while the software provides the required processing instructions. In a very short period of time there have been dramatic advances in hardware and software, and even more impressive developments are certain to occur in the near future. In this section of the book we present five articles that should enhance your knowledge of computer hardware and software technology.

The book begins with a selection from *Time* magazine, "Science: The Numbers Game." This article traces the evolution of computers and the miniaturization of memory and logic circuitry. It also describes in an easy-to-understand manner how computers function. Reading 2, "Superchips: The New Frontier," describes the continuing miniaturization of computer chips and provides insights on the nature of the applications that developing chip technology will make possible. Lewis Branscom's article, "Future Computer" (Reading 3), further explores the future of information technology and the challenges posed by continuing hardware and software changes.

The development of new languages and tools, and the effect that these products are having on the productivity of knowledge workers, is discussed by David H. Freeman in Reading 4, "Programming without Tears." The article also provides insights into the changes that fourth-generation languages are causing in the systems development process. Closing out this section is a selection by Jesse Green, "Productivity in the Fourth Generation: Six Case Studies" (Reading 5). In this article, Green provides further insights into the productivity increase being accomplished by the use of fourth-generation languages, and he illustrates the phenomenon with six case studies from the real world of business.

1

Science: The Numbers Game*

For the young electronics engineer at the newly formed Intel Corp., it was a challenging assignment. Fresh out of Stanford University, where he had been a research associate, M. E. ("Ted") Hoff in 1969 was placed in charge of producing a set of miniature components for programmable desktop calculators that a Japanese firm planned to market. After studying the circuitry proposed by the Japanese designers, the shy, self-effacing Hoff knew that he had a problem. As he recalls: "The calculators required a large number of chips, all of them quite expensive, and it looked, quite frankly, as if it would tax all our design capability."

Pondering the difficulty, Hoff was suddenly struck by a novel idea. Why not place most of the calculator's arithmetic and logic circuitry on one chip of silicon, leaving mainly input/output and programming units on separate chips? It was a daring conceptual move. After wrestling with the design, Hoff and his associates at Intel finally concentrated nearly all the elements of a central processing unit (CPU), the computer's electronic heart and soul, on a single silicon chip.

Unveiled in 1971, the one-chip CPU—or microprocessor—contained 2,250 transistors in an area barely a sixth of an inch long and an eighth of an inch wide. In computational power, the microprocessor almost matched the monstrous ENIAC—the first fully electronic computer, completed in 1946—and performed as well as an early 1960s IBM machine that cost $30,000 and re-

*Source: *Time,* February 20, 1978, pp. 54–58.

quired a CPU that alone was the size of a large desk. On his office wall, Hoff still displays Intel's original advertisement: "Announcing a new era of integrated electronics. . . a microprogrammable computer on a chip."

Intel's little chip had repercussions far beyond the pocket-calculator and minicomputer field. It was so small and cheap that it could be easily incorporated into almost any device that might benefit from some "thinking" power: electric typewriters with a memory, cameras, elevator controls, a shopkeeper's scales, vending machines, and a huge variety of household appliances. The new chip also represented another kind of breakthrough: because its program was on a different chip, the microprocessor could be "taught" to do any number of chores. All that had to be done was to substitute a tiny program chip with fresh instructions. In a memorable display of this versatility, the Pro-Log Corp. of Monterey, California, built what was basically a digital clock. But by switching memory chips and hitching it to a loudspeaker, it became first a "phonograph," playing the theme from *The Sting,* then an electric piano.

The Intel chip and one developed at about the same time at Texas Instruments Incorporated—the question of priority is still widely debated in the industry—were the natural culmination of a revolution in electronics that began in 1948 with Bell Telephone Laboratories' announcement of the transistor. Small, extremely reliable, and capable of operating with only a fraction of the electricity needed by the vacuum tube, the "solid-state" device proved ideal for making not only inexpensive portable radios and tape recorders but computers as well. Indeed, without the transistor, the computer might never have advanced much beyond the bulky and fickle ENIAC, which was burdened with thousands of large vacuum tubes that consumed great amounts of power, generated tremendous quantities of heat, and frequently burned out. In an industry striving for miniaturization, the transistors, too, soon began to shrink. By 1960, engineers had devised photolithographic and other processes that enabled them to crowd many transistors as well as other electronic components onto a tiny silicon square.

The advent of such integrated circuits (ICs) drastically reduced the size, cost, and electrical drain of any equipment in which they were used. One immediate byproduct: a new generation of small, desk-size minicomputers as well as larger, high-speed machines. Their speed resided in the rate at which electric current races through wire: about one foot per billionth of a second, close to the velocity of light. Even so, an electrical pulse required a significant fraction of a second to move through the miles of wiring in the early, large computers. Now even circuitous routes through IC chips could be measured in inches—and traversed by signals in an electronic blink. Computers with ICs not only were faster but were in a sense much smarter. Crammed with more memory and logic circuitry, they could take on far more difficult workloads.

Like the tracks in a railroad yard, ICs were really complex switching systems, shuttling electrical pulses hither and yon at the computer's bidding. Still, ICs could not function by themselves; other electronic parts had to keep the switches opening and closing in proper order. Then came the next quantum leap in miniaturization: the development in the late 1960s of large-scale integration (LSI). Unlike their single-circuit predecessors, which were designed to do only one specific job, LSIs integrated a number of circuits with separate functions on individual chips. These in turn were soldered together on circuit boards. Out of such modules, entire computers could be assembled like Erector sets.

But the new LSIs had an innate drawback. Because they were made in rigid patterns and served only particular purposes—or were, as engineers say, "hard-wired"—they lacked flexibility. That limitation was ingeniously solved by the work of Hoff and others on microprogramming—storing control instructions on a memory-like chip. For the first time, computer designers could produce circuitry usable for any number of purposes. In theory, the same basic chip could do everything from guiding a missile to switching on a roast.

Such computational prowess seems dazzlingly unreal and reinforces the popular image of computers as electronic brains with infinite intelligence. Yet most scientists regard computers, including those on chips, as dumb brutes. "They do only what they are told," insists Louis Robinson, director of scientific computing at IBM's data-processing division, "and not an iota more." What all computers, large and small, do extremely well is "number crunching"; they can perform prodigious feats of arithmetic, handling millions of numbers a second. Equally important, they can store, compare, and arrange data at blinding speed. That combination lets the computer handle a broad range of problems—from designing a complex new telescopic lens to sending TV images across the solar system.

Humans have been calculating since the dawn of history—and before. Stone age man, making scratches on animal bones, tried to keep track of the phases of the moon. Other prehistoric people reckoned with pebbles. Indeed, the Latin world *calculus* means a stone used for counting. Perhaps the most enduring calculating device is the abacus, which was used in China as early as the sixth century B.C. But the first really serious efforts to make mechanical calculators, in which some of the tallying was done automatically, did not come until the 17th century.

By then numbers had become especially important because of great advances in astronomy, navigation, and other scientific disciplines. More than ever before, it was necessary to rely on long tables of such elementary mathematical functions as logarithms, sines, and cosines. Yet compiling these essential tools often required years of slavish toil.

Still, mathematical illiteracy continued to plague Europe. In the early 19th century, Charles Babbage, an idiosyncratic mathematician and inventor of the railroad cowcatcher and the first tachometer, was becoming increasingly incensed by the errors he found in insurance records, logarithm tables, and other data. His fetish for accuracy was so great, in fact, that after reading Lord Tennyson's noted line "Every moment dies a man/ Every moment one is born," he wrote the poet: "It must be manifest that if this were true, the population of the world would be at a standstill." Babbage's recommended change: "Every moment dies a man/ Every moment $1\frac{1}{16}$ is born."

In 1822, Babbage began work on a machine, called the difference engine, that could help solve polynomial equations to six places. The Chancellor of the Exchequer was so impressed by the machine's potential for compiling accurate navigational and artillery tables that he subsidized construction of a still larger difference engine that could compute to 20 places. Unfortunately, the metalworkers of Babbage's day were not up to making the precision parts required, and the machine was never completed. But Babbage had a bolder dream: he wanted to build a machine, which he dubbed the analytical engine, that could perform any arithmetical and logical operations asked of it. In effect, it would have been programmable—that is, a true computer instead of a mere calculator.

To "instruct" the machine, Babbage borrowed an idea that had just revolutionized the weaving industry. Using a string of cards with strategically placed holes in them, like those in a piano roll, the Frenchman Joseph Marie Jacquard automatically controlled which threads of the warp would be passed over or under with each pass of the shuttle. Babbage planned to use the same technique to program his machine; instead of the positions of threads, the holes in his cards would represent the mathematical commands to the machine. Wrote Babbage's mathematically knowledgeable friend Lady Lovelace, daughter of the poet Lord Byron: "We may say most aptly that the Analytical Engine weaves algebraical patterns just as the Jacquard-loom weaves flowers and leaves."

Babbage's loom, alas, never wove anything. By the time the eccentric genius died in 1871, he had managed to put together just a few small parts; only his elaborate drawings provide a clue to his visionary machine. Indeed, when Harvard and IBM scientists rediscovered Babbage's work in the 1940s while they were building a pioneering electromechanical digital computer called Mark I, they were astonished by his foresight. Said the team leader, Howard Aiken: "If Babbage had lived 75 years later, I would have been out of a job."

The Harvard machine occupied a large room and sounded, in the words of physicist-author Jeremy Bernstein, "like a roomful of ladies knitting." The noise came from the rapid opening and closing of thousands of little switches, and it represented an enormous information flow and extremely long calculations for the time. In less than five seconds, Mark I could multiply two 23-digit numbers, a record that lasted until ENIAC's debut two years later. But how? In part, the answer lies in a beguilingly simple form of arithmetic: the binary system. Instead of the 10 digits (0 through 9) of the familiar decimal system, the computer uses just the binary's two symbols (1 and 0). And with enough 1s and 0s any quantity can be represented.

In the decimal system, each digit of a number read from right to left is understood to be multiplied by a progressively higher power of 10. Thus the number 4,932 consists of 2 multiplied by 1, plus 3 multiplied by 10, plus 9 multiplied by 10 × 10, plus 4 multiplied by 10 × 10 × 10. In the binary system, each digit of a number, again read from right to left, is multiplied by a progressively higher power of 2. Thus the binary number 11010 equals 0 times 1, plus 1 times 2, plus 0 times 2 × 2, plus 1 times 2 × 2 × 2, plus 1 times 2 × 2 × 2 × 2—for a total of 26 (see Figure 1).

Working with long strings of 1s and 0s would be cumbersome for humans—but it is a snap for a digital computer. Composed mostly of parts that are essentially on-off switches, the machines are perfectly suited for binary computation. When a switch is open, it corresponds to the binary digit 0; when it is closed, it stands for the digit 1. Indeed, the first modern digital computer completed by Bell Labs scientists in 1939 employed electromechanical switches called relays, which opened and closed like an old-fashioned Morse telegraph key. Vacuum tubes and transistors can also be used as switching devices and can be turned off and on at a much faster pace.

But how does the computer make sense out of the binary numbers represented by its open and closed switches? At the heart of the answer is the work of two other gifted Englishmen. One of