became leaders in fine chemicals. They and others also turned to the specialties market.

This story of the development of chemistry and of its application is the basis of Fred Aftalion's *Histoire de la chimie*, now in an English edition, appropriately entitled *A History of the International Chemical Industry*. Benfey's translation brings over the full drama of the rise of the industry, despite the enormous number of facts and the company mini-histories. The occasional bias toward events in France is actually quite useful, since it provides readers in the English language with a good overview of the understudied French chemical industry. The analysis of how the chemical industries of individual countries, including Japan and the Middle East, have been shaped by sociopolitical factors is particularly illuminating. The translation, unlike the original, includes extensive name and company indexes. This is a useful guide, especially of developments over the past half century.

ANTHONY S. TRAVIS
*Sidney M. Edelstein Center for the History and Philosophy of Science, Technology and Medicine, Hebrew University, Jerusalem, 91 904, Israel*

## Programming Programming

**Japan's Software Factories**. A Challenge to U.S. Management. MICHAEL A. CUSUMANO. Oxford University Press, New York, 1991. xii, 513 pp., illus. $35.

The business historian Michael Cusumano has followed his successful 1985 study *The Japanese Automobile Industry: Technology and Management at Nissan and Toyota* with a study of Japanese management practices in the software industry. As the subtitle of his new book suggests, Cusumano's implicit hypothesis is that superior Japanese management practices, represented by a commitment to treat software development as just another manufacturing process, threaten the perceived U.S. dominance of the software industry. In contrast with the case of automobiles, where the Japanese success is now obvious, the outcome of Japanese–U.S. competition in software is by no means clear. It is therefore more difficult to tell a convincing story, and the book, despite its length, falls short of proving its controversial hypothesis.

It does, however, clearly demonstrate that Japanese managers' emphasis on continuously improving the software development process by applying techniques commonly used in manufacturing has enabled them to write software programs more efficiently. The book therefore reinforces the important lesson from automobiles that viewing manufacturing as a static and technologically determined function ultimately leads to competitive disadvantage.

After an introductory chapter that lays out the book's conceptual framework, and so contains its conclusions, Cusumano describes the essential challenge for software development—how individuals whose expertise is often tacit can adapt a process that is much like an art form to efficiently write programs of varying language, length, and content. One approach, which Cusumano implies is American, is to manage programmers as sensitive craftsmen working in small teams and to treat each program as a new work of art. The Japanese, in contrast, apply the "factory approach." Cusumano identifies nine elements that typify this approach, and the bulk of the book examines its application inside four Japanese firms (Toshiba, Hitachi, NEC, and Fujitsu) and an early failed attempt inside a U.S. corporation, System Development Corporation.

As one reads the chapters on each Japanese company, a strong sense of the common approach they have adopted does indeed build. First, it is clear that, notwithstanding a historical tendency to centralize programming in large units of several thousand employees, the physical analogy to a factory is misplaced. Technology allows programmers to be geographically dispersed although working together, and NEC, in particular, is moving toward a network of coordinated production sites that exploits this capability. Nor does the factory approach imply assembly-line production using standardized and interchangeable components. Rather it involves a strategic commitment by corporate management to treat software development like any manufacturing process and to focus on building an organizational capability that favors efficiency over the development of unique or breakthrough products.

This process emphasis appears to have been applied in three stages in Japan. Although different companies pursued each stage at different and often overlapping times, a normative prescription for how to implement this, which Cusumano does not explicitly articulate, does emerge. Stage 1 involves implementing the *work methods* of manufacturing. First, development is divided among organizational units so that each "factory" specializes in the type of software that suits it, from generalized business applications, which can be the most standardized, to the most demanding artificial intelligence applications. Second, the software development process is divided into standardized tasks—requirements planning, system design, coding, testing, and so on—in order to facilitate project management. Next, measurement systems are installed to predict and control these individual tasks. Time to write a line of code, number of defects, pages of documentation, and other performance measures are recorded to facilitate forward planning, monitor a project's progress, and identify unfavorable variances that require management attention. Fourth, workers are hired without specific skills (many Japanese programmers are only high school graduates) and are trained and provided a career path inside the firm. In some cases company-specific programming methods necessitate this, but in all cases it serves to minimize skill differences among programmers. The ensuing interchangeability of personnel, Cusumano suggests, allows the Japanese, unlike U.S. software companies, to assign additional programmers to a project and actually reduce the time to project completion. Finally, the well-known Japanese techniques of quality control are applied at all stages. Indeed, standardizing the process, if not the product, facilitates the use of many existing generic techniques and tools for project management, quality control, and cost accounting.

Stage 2 in the application of the "factory approach" introduces *automated tools* to software development after the manual process has been controlled. At first these tools might be simple compilers. Later they become design-support tools that automate coding from standardized design languages or automatically check errors and generate documents. Later still they might be artificial-intelligence tools that translate system requirements directly into code. At each stage these tools are tied to the company's project management system to maintain the emphasis on process efficiency.

Stage 3 involves *standardized components*, or the reuse of modules of code from a library of standard packages. Although this might be the goal of the factory approach, it is clear that it is not yet common—reuse rates, even between similar applications programs, are often less than 20%. Indeed, the initial expense of writing and documenting a flexible module and the fact that starting from scratch is usually cheaper if a substantial portion (around 40%) of a module has to be amended make reuse impractical except for a few functions in general business applications. The Japanese seem to view automated programming and reusability as by-products of their stress on controlling the software development process, not as ends in themselves. In contrast, the few U.S. firms that have tried an approach other than the craft job shop have focused on these as goals and have given up the factory ap-

proach when they were found to be unattainable.

Cusumano recognizes that the Japanese approach to software development was facilitated by the structure of the domestic computer industry. Fewer competitors and a focus on mainframes rather than minicomputers and PCs (at least historically) meant that software was developed primarily for large computers with relatively stable architectures and applications. For those needs an emphasis on process refinement was appropriate. The U.S. market, with its rapid technological change and a preference for standardized software packages for small computers, is less suited to such a degree of control and conformity (which is probably also anathema to the average U.S. programmer).

Unfortunately, the lessons of Cusumano's detailed research are obscured by a tendency to core-dump all he has learned about a company into the book. As a consequence the reader must wrestle with succeeding sets of acronyms and charts of process flows. Moreover, Cusumano ultimately falls between two stools by including too much detail to make for exciting reading by the layperson, and too little detail for real learning by software engineers.

However, the more important drawback to the book is its failure to prove the superiority of the Japanese approach to software development. The detailed evidence presented on performance refers only to the Japanese marketplace. If the Japanese are so good at software development why do they not sell software outside Japan? Lacking a convincing argument on this point, the book becomes mainly descriptive. It does not, for example, compare the introduction of the factory approach to software development with its introduction in another industry (management of contracting, for example, seems to be analogous to software and is another area in which the Japanese claim to be superior to the United States), and so cannot elucidate broadly applicable lessons. Nor can the reader compare and contrast the two countries' management styles, for the book's description of the U.S. style of software development is limited to four companies and 25 pages. I believe there is basically a trade-off between the two approaches to software development—between a combination of productivity and program reliability on the one hand and product sophistication on the other. The Japanese approach may indeed be efficient at generating customized application programs in vertical markets, such as automated banking machines, traffic control, or reservations systems. It has been less effective in developing new standard programs, whether for operating systems or general business application. This has been the strength of U.S. software development. That more tightly managed development processes sacrifice innovation for efficiency is no surprise; whether the sacrifice is economically justified has yet to be determined.

Nevertheless, it is easy to criticize a book for what it is not. Cusumano has written a fine, detailed historical account of how the Japanese manage software development. It is possible that, as the ratio of hardware costs to software costs decreases, a relatively inelegant program, cheap to write but requiring more processing time and memory than a carefully crafted program, becomes economically desirable. If it does, then the Japanese approach to the management of software development may indeed pose, as Cusumano implies, an effective challenge to the current U.S. dominance of the software industry.

DAVID COLLIS
*Harvard Business School,*
*Boston, MA 02163*

## Acoustic Strategies

**Acoustic Behavior of Insects**. An Evolutionary Perspective. WINSTON J. BAILEY. Chapman and Hall (Routledge, Chapman and Hall), New York, 1991. xvi, 225 pp., illus. $87.50.

Insects are the most successful of terrestrial beings, and in their lives we can find counterparts to our own. The ways in which they have adapted to life on earth can be viewed as models for the "Johnny-come-latelys" of evolution, including human beings. One is reminded that some renowned commentators on human social behavior—Alfred Kinsey, R. D. Alexander, and E. O. Wilson—were entomologists first (even the young Freud had a brief obsession with the brains of insects and crayfish before moving on). Behavioral biologists are rightly fascinated by the signals contained within the songs of birds, the croaking of frogs, and the howlings of monkeys, but it should not surprise us that crickets were chirping long before these animals and for the same "reasons." For all these animals, the "songs" are signals for survival.

Bailey's little book is about the role that acoustic signals play in the lives of insects. It is unusual in that it attempts to bring together two major themes of behavioral biology—evolution and mechanism—in the analysis of acoustic communication. Among recent books on the subject, the evolutionary theme is represented by Thornhill and Alcock's *The Evolution of Insect Mating Systems* (Harvard University Press, 1983) and the mechanist point of view by Ewing's *Arthropod Bioacoustics* (Cornell University Press, 1990) and Huber *et al.*'s *Cricket Behavior and Neurobiology* (Cornell University Press, 1988). None of these, nor a half-dozen other books on insect communication behavior, really provide a synthesis of the two disparate viewpoints of sociobiology and neuroethology. It is to Bailey's credit that he makes this attempt, and that is the value of this slim volume.

To do all this in one book is very ambitious given the extensive literature in insect bioacoustics and neuroethology. Nonetheless, Bailey provides a brief and readable introduction to the diverse mechanisms by which insects produce sounds, the multitude of kinds of "ears" that have evolved to hear them, and a glimpse at the way acoustic signals are detected, localized, and "recognized" by the nervous system. Like nearly all animals that hear, insects "use" acoustic signals to mediate behaviors relating to mating and territorial rights among members of the same species. Insects must also detect the presence of predators, some of which give themselves away by the sounds that they make. For a look at reproductive behavior, Bailey draws his examples from the orthopteroid insects (grasshoppers, crickets, katydids) as well as from his own work on cicadas. As an example of how insects detect acoustically active predators, Bailey recounts the story of insectivorous bats—which use ultrasonic biosonar signals to detect and localize flying insects—and the moths that evade the bats by hearing them and flying away.

A certain amount of detail is missing from Bailey's brief chapters that would deepen the reader's understanding of the bioacoustics and neurobiology that underlie insect bioacoustics, but a mechanistic account is not his major objective. The subtitle of his book is "An Evolutionary Perspective," and, taking a strict selectionist perspective, he tackles some big questions of behavioral ecology, such as the advantages of singing in a chorus, a behavior observed in cicadas, katydids, and tree crickets, among insects—but also seen in frogs and birds. He discusses the role of natural and sexual selection in the evolution of mating calls in insects and the "conflicts of interest" that confront male and female in sending and receiving these acoustic signals.

Bailey ends his book with a thought-provoking chapter on the evolution of social signals. Because this subject is as controversial as it is important in sociobiology, his musings will not find universal agreement. But I welcome this book because it represents an attempt at a synthesis that is sorely needed in behavioral biology, where evolutionary biologists tend to ignore questions