CHAPTER

8 Professional Ethics

We have come through a strange cycle in programming, starting with the creation of programming itself as a human activity. Executives with the tiniest smattering of knowledge assume that anyone can write a program, and only now are programmers beginning to win their battle for recognition as true professionals.

–GERALD WEINBERG, *The Psychology of Computer Programming*, 1971

8.1 Introduction

INFORMALLY, A **PROFESSION** IS A VOCATION THAT REQUIRES A HIGH LEVEL OF EDUCATION and practical experience in the field. Medicine and law are two well-known professions. We pay doctors and lawyers well, trusting that they will correctly ascertain and treat our medical and legal problems, respectively. Professionals have a special obligation to ensure their actions are for the good of those who depend on them because their decisions can have more serious consequences than the choices made by those holding less responsible positions in society.

In this chapter we focus on moral decisions made by people who design, implement, or maintain computer hardware or software systems. We begin by considering the extent to which a computer-related career is a profession along the lines of medicine or law. Next, we present and analyze a code of ethics for an important computer-related discipline: software engineering. Our analysis leads us into a discussion of virtue ethics,

an ethical theory based on the idea that good character is the source of correct moral decisions. Four case studies give us the opportunity to use the software engineering code of ethics as a tool for ethical analysis.

Finally, we discuss whistleblowing: a situation in which a member of an organization breaks ranks to reveal actual or potential harm to the public. Whistleblowing raises important moral questions about loyalty, trust, and responsibility. Two accounts of whistleblowing illuminate these moral questions and demonstrate the personal sacrifices some have made for the greater good of society. We consider the important role management plays in creating an organizational atmosphere that either allows or suppresses internal dissent.

8.2 Are Computer Experts Professionals?

Millions of people have a computer-related job title, such as computer engineer, computer scientist, programmer, software engineer, system administrator, or systems analyst. Is a computer-related career a profession like medicine or law? Let's consider the characteristics of a well-developed profession.

8.2.1 Characteristics of a Profession

A fully developed profession has a well-organized infrastructure for certifying new members and supporting those who already belong to the profession. Ford and Gibbs have identified eight components of a mature professional infrastructure [1]:[1]

- *Initial professional education*—formal course work completed by candidates before they begin practicing the profession
- *Accreditation*—assures that the formal course work meets the standards of the profession
- *Skills development*—activities that provide candidates with the opportunity to gain practical skills needed to practice the profession
- *Certification*—process by which candidates are evaluated to determine their readiness to enter the profession
- *Licensing*—the process giving candidates the legal right to practice the profession
- *Professional development*—formal course work completed by professionals in order to maintain and develop their knowledge and skills
- *Code of ethics*—mechanism by which a profession ensures that its members will use their knowledge and skills for the benefit of society
- *Professional society*—organization promoting the welfare of the profession, typically consisting of most if not all of the members of the profession

1. From Gary Ford and Norman E. Gibbs. "A Mature Profession of Software Engineering." Technical report, Carnegie Mellon University, January 1996.

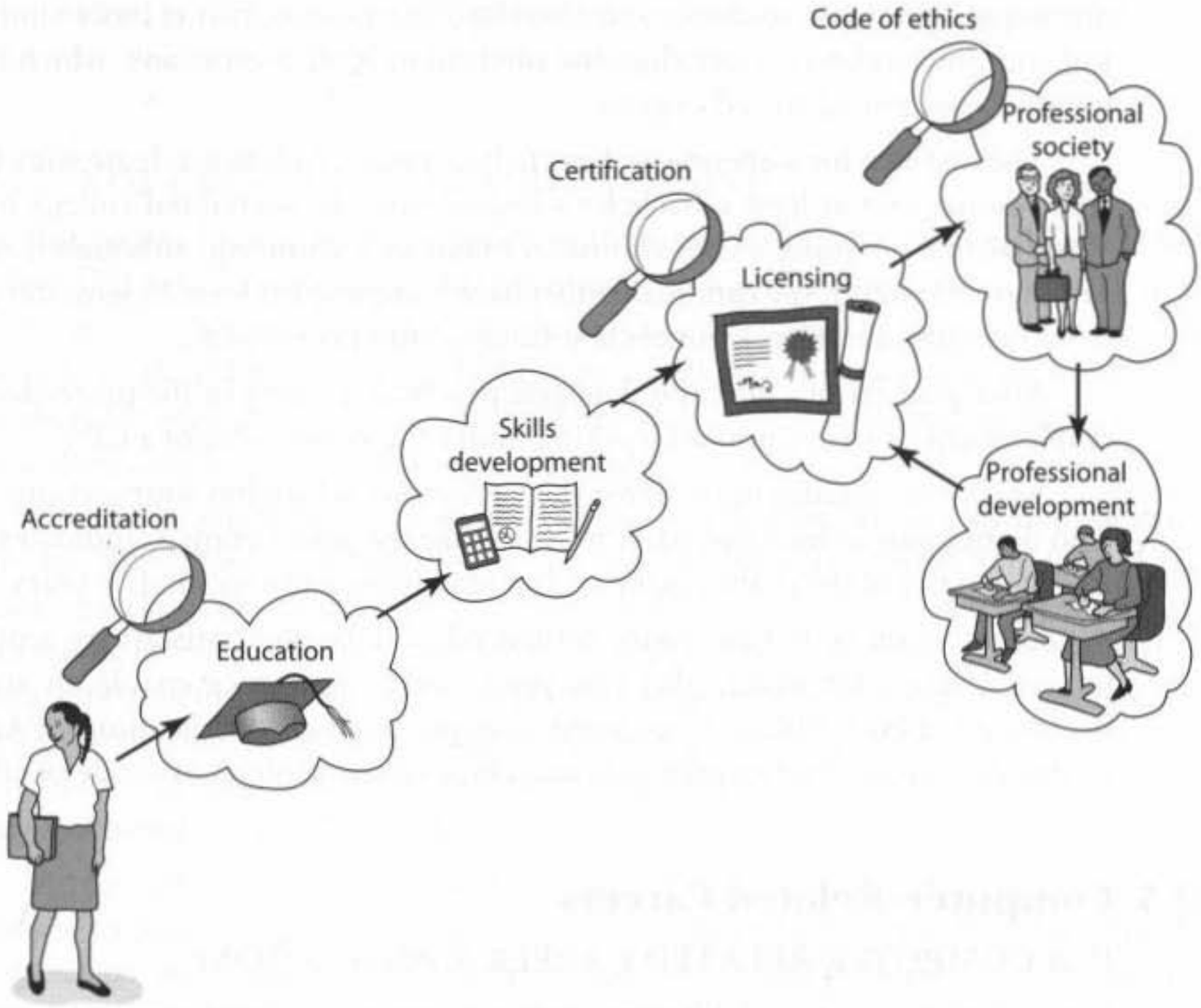

FIGURE 8.1 A mature profession has eight attributes that enable it to certify new members and support existing members [1].

Figure 8.1 illustrates how these components work together to support the profession. A person desiring to join the profession undertakes some initial professional education. A process of accreditation assures that the educational process is sound. After completing their formal education, candidates gain skills through practical experience working in the field. Another check determines if the candidate is ready to be certified. Successful candidates are licensed to practice the profession.

When the public can trust the competence and integrity of the members of a profession, every one of its members benefits. For this reason professionals have a stake in ensuring that fellow members of the profession are capable and act appropriately. For mature professions, professional societies establish codes of ethics and require their members to keep their knowledge current through continuing education and training. Professionals who do not follow the code of ethics or fail to keep up with changes in the field can lose their licenses.

8.2.2 Certified Public Accountants

To illustrate these steps, let's consider how a person becomes a Certified Public Accountant (CPA). We choose accounting because it is a fully developed profession that does

not require graduate study for membership. In this respect it is more similar to a typical computer-related career than the medical or legal professions, which require their members to earn advanced degrees.

The first step for someone wishing to become a CPA is to graduate with 150 semester credit hours and at least a bachelor's degree from an accredited college or university. Many people pursuing a CPA choose to major in accounting, although it is not strictly necessary. However, the candidate must have completed at least 24 semester credit hours in accounting, auditing, business law, finance, and tax subjects.

After graduation, the candidate gets practical training in the profession by finding employment as an accountant working under the supervision of a CPA.

Finally, candidates must sit for the CPA exam, which has four sections. Candidates who do not pass at least two parts must re-take the entire exam. Candidates who pass at least two parts of the exam must pass the remaining parts within five years.

Completion of the necessary formal education, plus satisfactory scores on every section of the CPA exam, plus two years' work experience enable an accountant to become a Certified Public Accountant. In order to retain certification, CPAs must fulfill continuing education requirements and abide by the profession's code of ethics.

8.2.3 Computer-Related Careers

IS A COMPUTER-RELATED CAREER A PROFESSION?

It is easy to find a crucial difference between systems analysts, computer programmers, and system administrators on the one hand and accountants, lawyers, and physicians on the other hand. At the heart of every mature profession is certification and licensing. Certification and licensing allow a profession to determine who will be allowed to practice the profession. For example, a person may not practice law in a state without passing the bar exam and being granted a license. In contrast, people may write computer programs and maintain computer systems, either as consultants, sole proprietors, or members of larger firms, without being certified or having been granted a license.

There are other differences between computer-related careers and mature professions. A person does not have to complete college or serve an apprenticeship under the guidance of an experienced mentor in order to gain employment as a programmer, system administrator, or systems analyst. The vast majority of people who hold computer-related jobs do not belong to either of computing's professional societies. It is up to particular employers to monitor the behavior of their employees and guide their continuing education—no professional organization has the authority to forbid someone from managing computer networks or writing computer programs.

In another important respect computer programmers differ from most professionals, such as dentists and ministers. Typically, professionals work directly with individual clients. A dentist treats one patient at a time. An accountant audits one business at a time. Most computer programmers work inside a company as part of a team that includes many other programmers as well as managers. In this environment the responsibility of

an individual person is more difficult to discern. Low-level technical decisions are made by groups, and final authority rests with management.

STATUS OF CERTIFICATION AND LICENSING

The two largest organizations supporting the computing field are the IEEE Computer Society (IEEE-CS), with about 85,000 members, and the Association for Computing Machinery (ACM), with about 92,000 members. Like organizations supporting mature professions, the IEEE-CS and the ACM strive to advance the discipline and support their members through publications, conferences, local chapters, student chapters, technical committees, and the development of standards.

A **software engineer** is someone engaged in the development or maintenance of software, or someone who teaches in this area. In 1993 the IEEE-CS and ACM set up a joint steering committee to explore the establishment of software engineering as a profession. The joint steering committee created several task forces to address particular issues. One task force conducted a survey of practitioners with the goal of understanding the knowledge and skills required by software engineers. Another task force developed accreditation criteria for undergraduate programs in software engineering. A third task force developed a code of ethics for software engineers.

In May 1999 the ACM Council passed a resolution that stated, in part, "ACM is opposed to the licensing of software engineers at this time because ACM believes that it is premature and would not be effective in addressing the problems of software quality and reliability" [2].

ABILITY TO HARM PUBLIC

The computing "profession" may not be as well developed as the medical or legal professions, but in one key respect—the ability to harm members of the public—those who design, implement, and maintain computer hardware and software systems sometimes hold responsibilities similar to those held by members of mature professions. The Therac-25 killed or gravely injured at least six people, in part because of defective software. While most software engineers do not write code for safety-critical systems such as linear accelerators, society does depend on the quality of their work. People make important business decisions based on the results they get from their spreadsheet programs. Millions rely upon commercial software to help them produce their income tax returns. Errors in programs can result in such harms as lost time, incorrect businesses decisions, and fines. System administrators are responsible for keeping computer systems running reliably without infringing on the privacy of the computer users.

The ability to cause harm to members of the public is a powerful reason why those in computer-related careers should follow a code of ethics, even if they are not professionals in the same sense as physicians, lawyers, and CPAs. As a good example of a code of ethics for those in computer-related disciplines, we present the Software Engineering Code of Ethics and Professional Practice, endorsed by both the ACM and the IEEE-CS.

8.3 Software Engineering Code of Ethics

The Software Engineering Code of Ethics and Professional Practice is a practical framework for moral decision making related to problems that software engineers may encounter.

The Software Engineering Code of Ethics and Professional Practice, reproduced in its entirety below, is copyright © 1999 by the Association for Computing Machinery, Inc. and the Institute for Electrical and Electronics Engineers, Inc.

8.3.1 Preamble

Computers have a central and growing role in commerce, industry, government, medicine, education, entertainment and society at large. Software engineers are those who contribute by direct participation or by teaching, to the analysis, specification, design, development, certification, maintenance and testing of software systems. Because of their roles in developing software systems, software engineers have significant opportunities to do good or cause harm, to enable others to do good or cause harm, or to influence others to do good or cause harm. To ensure, as much as possible, that their efforts will be used for good, software engineers must commit themselves to making software engineering a beneficial and respected profession. In accordance with that commitment, software engineers shall adhere to the following Code of Ethics and Professional Practice.

The Code contains eight Principles related to the behavior of and decisions made by professional software engineers, including practitioners, educators, managers, supervisors and policymakers, as well as trainees and students of the profession. The Principles identify the ethically responsible relationships in which individuals, groups, and organizations participate and the primary obligations within these relationships. The Clauses of each Principle are illustrations of some of the obligations included in these relationships. These obligations are founded in the software engineer's humanity, in special care owed to people affected by the work of software engineers, and the unique elements of the practice of software engineering. The Code prescribes these as obligations of anyone claiming to be or aspiring to be a software engineer.

It is not intended that the individual parts of the Code be used in isolation to justify errors of omission or commission. The list of Principles and Clauses is not exhaustive. The Clauses should not be read as separating the acceptable from the unacceptable in professional conduct in all practical situations. The Code is not a simple ethical algorithm that generates ethical decisions. In some situations standards may be in tension with each other or with standards from other sources. These situations require the software engineer to use ethical judgment to act in a manner which is most consistent with the spirit of the Code of Ethics and Professional Practice, given the circumstances.

Ethical tensions can best be addressed by thoughtful consideration of fundamental principles, rather than blind reliance on detailed regulations. These Principles should influence software engineers to consider broadly who is affected by their work; to examine if they and their colleagues are treating other human beings with due respect; to consider how the public, if reasonably well informed, would view their decisions; to analyze how

the least empowered will be affected by their decisions; and to consider whether their acts would be judged worthy of the ideal professional working as a software engineer. In all these judgments concern for the health, safety and welfare of the public is primary; that is, the "Public Interest" is central to this Code.

The dynamic and demanding context of software engineering requires a code that is adaptable and relevant to new situations as they occur. However, even in this generality, the Code provides support for software engineers and managers of software engineers who need to take positive action in a specific case by documenting the ethical stance of the profession. The Code provides an ethical foundation to which individuals within teams and the team as a whole can appeal. The Code helps to define those actions that are ethically improper to request of a software engineer or teams of software engineers.

The Code is not simply for adjudicating the nature of questionable acts; it also has an important educational function. As this Code expresses the consensus of the profession on ethical issues, it is a means to educate both the public and aspiring professionals about the ethical obligations of all software engineers.

8.3.2 Principles

PRINCIPLE 1: PUBLIC

Software engineers shall act consistently with the public interest. In particular, software engineers shall, as appropriate:

1.01 Accept full responsibility for their own work.

1.02 Moderate the interests of the software engineer, the employer, the client and the users with the public good.

1.03 Approve software only if they have a well-founded belief that it is safe, meets specifications, passes appropriate tests, and does not diminish quality of life, diminish privacy or harm the environment. The ultimate effect of the work should be to the public good.

1.04 Disclose to appropriate persons or authorities any actual or potential danger to the user, the public, or the environment, that they reasonably believe to be associated with software or related documents.

1.05 Cooperate in efforts to address matters of grave public concern caused by software, its installation, maintenance, support or documentation.

1.06 Be fair and avoid deception in all statements, particularly public ones, concerning software or related documents, methods and tools.

1.07 Consider issues of physical disabilities, allocation of resources, economic disadvantage and other factors that can diminish access to the benefits of software.

1.08 Be encouraged to volunteer professional skills to good causes and contribute to public education concerning the discipline.

FIGURE 8.2 Software engineers shall approve software only if they have a well-founded belief that it is safe, meets specifications, passes appropriate tests, and does not diminish quality of life, diminish privacy, or harm the environment. The ultimate effect of the work should be to the public good (Clause 1.03).

PRINCIPLE 2: CLIENT AND EMPLOYER

Software engineers shall act in a manner that is in the best interests of their client and employer, consistent with the public interest. In particular, software engineers shall, as appropriate:

2.01 Provide service in their areas of competence, being honest and forthright about any limitations of their experience and education.

2.02 Not knowingly use software that is obtained or retained either illegally or unethically.

2.03 Use the property of a client or employer only in ways properly authorized, and with the client's or employer's knowledge and consent.

2.04 Ensure that any document upon which they rely has been approved, when required, by someone authorized to approve it.

2.05 Keep private any confidential information gained in their professional work, where such confidentiality is consistent with the public interest and consistent with the law.

2.06 Identify, document, collect evidence and report to the client or the employer promptly if, in their opinion, a project is likely to fail, to prove too expensive, to violate intellectual property law, or otherwise to be problematic.

2.07 Identify, document, and report significant issues of social concern, of which they are aware, in software or related documents, to the employer or the client.

2.08 Accept no outside work detrimental to the work they perform for their primary employer.

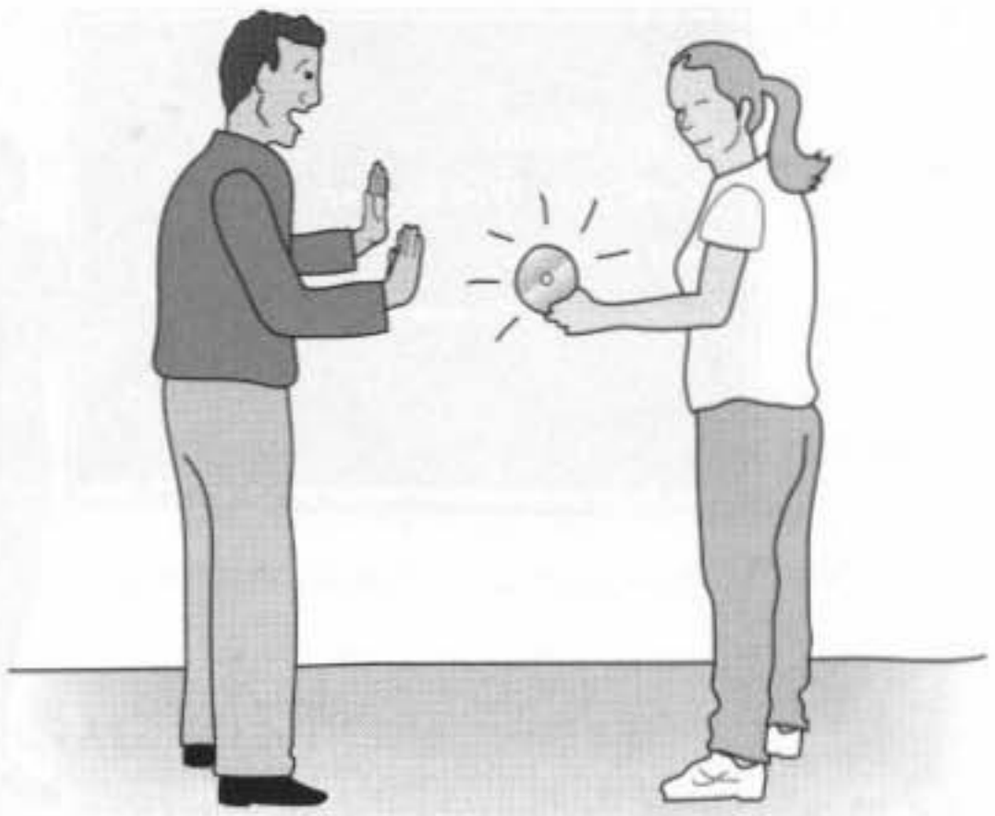

Figure 8.3 Software engineers shall not knowingly use software that is obtained or retained either illegally or unethically (Clause 2.02).

2.09 Promote no interest adverse to their employer or client, unless a higher ethical concern is being compromised; in that case, inform the employer or another appropriate authority of the ethical concern.

PRINCIPLE 3: PRODUCT

Software engineers shall ensure that their products and related modifications meet the highest professional standards possible. In particular, software engineers shall, as appropriate:

3.01 Strive for high quality, acceptable cost and a reasonable schedule, ensuring significant tradeoffs are clear to and accepted by the employer and the client, and are available for consideration by the user and the public.

3.02 Ensure proper and achievable goals and objectives for any project on which they work or propose.

3.03 Identify, define and address ethical, economic, cultural, legal and environmental issues related to work projects.

3.04 Ensure that they are qualified for any project on which they work or propose to work by an appropriate combination of education and training, and experience.

3.05 Ensure an appropriate method is used for any project on which they work or propose to work.

3.06 Work to follow professional standards, when available, that are most appropriate for the task at hand, departing from these only when ethically or technically justified.

3.07 Strive to fully understand the specifications for software on which they work.

FIGURE 8.4 Software engineers shall ensure proper and achievable goals and objectives for any project on which they work or propose (Clause 3.02).

3.08 Ensure that specifications for software on which they work have been well documented, satisfy the users' requirements and have the appropriate approvals.

3.09 Ensure realistic quantitative estimates of cost, scheduling, personnel, quality and outcomes on any project on which they work or propose to work and provide an uncertainty assessment of these estimates.

3.10 Ensure adequate testing, debugging, and review of software and related documents on which they work.

3.11 Ensure adequate documentation, including significant problems discovered and solutions adopted, for any project on which they work.

3.12 Work to develop software and related documents that respect the privacy of those who will be affected by that software.

3.13 Be careful to use only accurate data derived by ethical and lawful means, and use it only in ways properly authorized.

3.14 Maintain the integrity of data, being sensitive to outdated or flawed occurrences.

3.15 Treat all forms of software maintenance with the same professionalism as new development.

PRINCIPLE 4: JUDGMENT

Software engineers shall maintain integrity and independence in their professional judgment. In particular, software engineers shall, as appropriate:

4.01 Temper all technical judgments by the need to support and maintain human values.

4.02 Only endorse documents either prepared under their supervision or within their areas of competence and with which they are in agreement.

4.03 Maintain professional objectivity with respect to any software or related documents they are asked to evaluate.

4.04 Not engage in deceptive financial practices such as bribery, double billing, or other improper financial practices.

4.05 Disclose to all concerned parties those conflicts of interest that cannot reasonably be avoided or escaped.

4.06 Refuse to participate, as members or advisors, in a private, governmental or professional body concerned with software related issues, in which they, their employers or their clients have undisclosed potential conflicts of interest.

PRINCIPLE 5: MANAGEMENT

Software engineering managers and leaders shall subscribe to and promote an ethical approach to the management of software development and maintenance. In particular, those managing or leading software engineers shall, as appropriate:

5.01 Ensure good management for any project on which they work, including effective procedures for promotion of quality and reduction of risk.

5.02 Ensure that software engineers are informed of standards before being held to them.

5.03 Ensure that software engineers know the employer's policies and procedures for protecting passwords, files and information that is confidential to the employer or confidential to others.

5.04 Assign work only after taking into account appropriate contributions of education and experience tempered with a desire to further that education and experience.

5.05 Ensure realistic quantitative estimates of cost, scheduling, personnel, quality and outcomes on any project on which they work or propose to work, and provide an uncertainty assessment of these estimates.

5.06 Attract potential software engineers only by a full and accurate description of the conditions of employment.

5.07 Offer fair and just remuneration.

5.08 Not unjustly prevent someone from taking a position for which that person is suitably qualified.

5.09 Ensure that there is a fair agreement concerning ownership of any software, processes, research, writing, or other intellectual property to which a software engineer has contributed.

5.10 Provide for due process in hearing charges of violation of an employer's policy or of this Code.

5.11 Not ask a software engineer to do anything inconsistent with this Code.

5.12 Not punish anyone for expressing ethical concerns about a project.

FIGURE 8.5 Software engineers shall help develop an organizational environment favorable to acting ethically (Clause 6.01).

PRINCIPLE 6: PROFESSION

Software engineers shall advance the integrity and reputation of the profession consistent with the public interest. In particular, software engineers shall, as appropriate:

6.01 Help develop an organizational environment favorable to acting ethically.

6.02 Promote public knowledge of software engineering.

6.03 Extend software engineering knowledge by appropriate participation in professional organizations, meetings and publications.

6.04 Support, as members of a profession, other software engineers striving to follow this Code.

6.05 Not promote their own interest at the expense of the profession, client or employer.

6.06 Obey all laws governing their work, unless, in exceptional circumstances, such compliance is inconsistent with the public interest.

6.07 Be accurate in stating the characteristics of software on which they work, avoiding not only false claims but also claims that might reasonably be supposed to be speculative, vacuous, deceptive, misleading, or doubtful.

6.08 Take responsibility for detecting, correcting, and reporting errors in software and associated documents on which they work.

6.09 Ensure that clients, employers, and supervisors know of the software engineer's commitment to this Code of ethics, and the subsequent ramifications of such commitment.

6.10 Avoid associations with businesses and organizations which are in conflict with this code.

6.11 Recognize that violations of this Code are inconsistent with being a professional software engineer.

6.12 Express concerns to the people involved when significant violations of this Code are detected unless this is impossible, counter-productive, or dangerous.

6.13 Report significant violations of this Code to appropriate authorities when it is clear that consultation with people involved in these significant violations is impossible, counter-productive or dangerous.

PRINCIPLE 7: COLLEAGUES

Software engineers shall be fair to and supportive of their colleagues. In particular, software engineers shall, as appropriate:

7.01 Encourage colleagues to adhere to this Code.

7.02 Assist colleagues in professional development.

7.03 Credit fully the work of others and refrain from taking undue credit.

7.04 Review the work of others in an objective, candid, and properly documented way.

7.05 Give a fair hearing to the opinions, concerns, or complaints of a colleague.

7.06 Assist colleagues in being fully aware of current standard work practices including policies and procedures for protecting passwords, files and other confidential information, and security measures in general.

7.07 Not unfairly intervene in the career of any colleague; however, concern for the employer, the client or public interest may compel software engineers, in good faith, to question the competence of a colleague.

7.08 In situations outside of their own areas of competence, call upon the opinions of other professionals who have competence in that area.

PRINCIPLE 8: SELF

Software engineers shall participate in lifelong learning regarding the practice of their profession and shall promote an ethical approach to the practice of the profession. In particular, software engineers shall continually endeavor to:

8.01 Further their knowledge of developments in the analysis, specification, design, development, maintenance and testing of software and related documents, together with the management of the development process.

8.02 Improve their ability to create safe, reliable, and useful quality software at reasonable cost and within a reasonable time.

8.03 Improve their ability to produce accurate, informative, and well-written documentation.

8.04 Improve their understanding of the software and related documents on which they work and of the environment in which they will be used.

8.05 Improve their knowledge of relevant standards and the law governing the software and related documents on which they work.

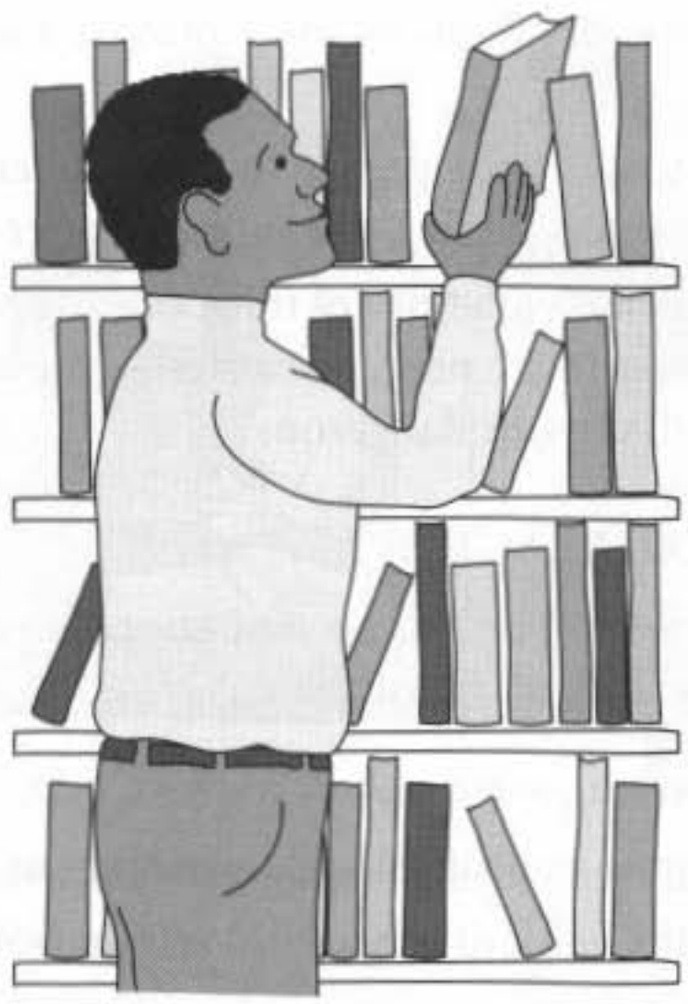

FIGURE 8.6 Software engineers shall continually endeavor to improve their ability to create safe, reliable, and useful quality software at reasonable cost and within a reasonable time (Clause 8.02).

8.06 Improve their knowledge of this Code, its interpretation, and its application to their work.

8.07 Not give unfair treatment to anyone because of any irrelevant prejudices.

8.08 Not influence others to undertake any action that involves a breach of this Code.

8.09 Recognize that personal violations of this Code are inconsistent with being a professional software engineer.

8.4 Analysis of the Code

In this section we analyze the Code and derive an alternate set of underlying principles upon which it rests.

8.4.1 Preamble

The preamble to the Code points out that there is no mechanical process for determining the correct actions to take when faced with a moral problem. Our experience evaluating moral problems related to the introduction and use of information technology confirms this statement. Even two people with similar philosophies may reach different conclusions when confronted with a moral problem. Two Kantians may agree on the basic facts of a moral problem, but disagree on how to characterize the will of the moral agent. Two utilitarians may agree on the benefits and harms resulting from a proposed action, but assign different weights to the outcomes, causing them to reach opposite conclusions.

The preamble also warns against taking an overly legalistic view of the Code. Simply because an action is not expressly forbidden by the Code does not mean it is morally acceptable. Instead, judgment is needed to detect when a moral problem has arisen and to determine the right thing to do in a particular situation.

While the Code is expressed as a collection of rules, these rules are based on principles grounded in different ethical theories. This is not surprising, considering that the Code was drafted by a committee. When we encounter a situation where two rules conflict, the preamble urges us to ask questions that will help us consider the principles underlying the rules. These questions demonstrate the multifaceted grounding of the Code:

1. *Who is affected?*

   Utilitarians focus on determining how an action benefits or harms other people.

2. *Am I treating other human beings with respect?*

   Kant's Categorical Imperative tells us to treat others as ends in themselves, rather than simply as a means to an end.

3. *Would my decision hold up to public scrutiny?*

   A cultural relativist is concerned about whether an action conforms with the mores of society.

4. *How will those who are least empowered be affected?*

   Rawls's second principle of justice requires us to consider whether inequalities are to the greatest benefit of the least-advantaged members of society.

5. *Are my acts worthy of the ideal professional?*

   The ethics of virtue is based on imitation of morally superior role models. Since we did not discuss virtue ethics in Chapter 2, let's examine it now.

8.4.2 Virtue Ethics

ORIGIN OF VIRTUE ETHICS

In *The Nicomachean Ethics*, Aristotle expresses the opinion that happiness results from living a life of virtue [3]. He distinguishes between *intellectual virtue*, which is developed through education, and *moral virtue*, which comes about through repetition of the appropriate acts (Figure 8.7). You can acquire the virtue of honesty, for example, by habitually telling the truth. According to Aristotle, deriving pleasure from a virtuous act is a sign that you have acquired that virtue.

There is a wealth of virtues, of course. Here is a brief list of two dozen virtues given by James Rachels: benevolence, civility, compassion, conscientiousness, cooperativeness, courage, courteousness, dependability, fairness, friendliness, generosity, honesty, industriousness, justice, loyalty, moderation, patience, prudence, reasonableness, self-discipline, self-reliance, tactfulness, thoughtfulness, and tolerance [4].

A person who possesses many moral virtues has a strong moral character. According to Aristotle, when people with strong character face a moral problem, they know the

FIGURE 8.7 According to Aristotle, happiness derives from living a life of virtue. You acquire moral virtues by repeating the appropriate acts.

right thing to do, because the action will be consistent with their character. As Justin Oakley and Dean Cocking put it, "An action is right if and only if it is what an agent with a virtuous character would do in the circumstances" [5].

STRENGTHS OF VIRTUE ETHICS

Virtue ethics has two advantages over the ethical theories we considered in Chapter 2. First, it provides a motivation for good behavior. The calculus of utility and the Categorical Imperative say nothing about motivation. Virtue ethics, on the other hand, stresses the importance of loyalty, thoughtfulness, courteousness, dependability, and other characteristics of healthy social interactions.

A second advantage of virtue ethics is that it provides a solution to the problem of impartiality. Recall that utilitarianism, Kantianism, and social contract theory require us to be completely impartial and treat all human beings as equals. This assumption leads to moral evaluations that are hard for most people to accept. For example, when a couple is faced with the choice between using $4,000 to take their children to Disneyland for a week or feeding 1,000 starving Africans for a month, the calculus of utility would conclude saving 1,000 lives was the right thing to do. However, most of us expect that good parents will show more kindness to their children than to people living on the other side of the world.

Virtue ethics avoids the pitfall of impartiality by rejecting the notion that every action must be designed to produce the maximum benefit for people overall [5]. Instead, some virtues are partial toward certain people, while others are impartial and treat everyone equally. Love, friendship, and loyalty are examples of virtues that allow a person to be partial toward friends and family members. Honesty, civility, and courteousness are examples of virtues that a person would extend equally to all human beings.

WEAKNESS OF VIRTUE ETHICS

However, virtue ethics has a significant liability. Using virtue ethics alone, it is often difficult to determine what to do in a particular situation. Suppose you are in charge of dispatching crews to fight brush fires in southern California. Three fires erupt at the same time: one near a mountain resort frequented by the wealthy; another near a town of 10,000 people with a high rate of poverty; and a third close to a middle-class suburb. You only have the resources to fight two of the fires. Which fires do you attack? Looking back on the list of virtues, which ones come into play? Compassion? Fairness? Justice? Prudence?

Suppose we decide the most relevant virtue is prudence. What is the prudent thing to do? Perhaps prudence dictates that you allocate fire crews to minimize property damage, but making your decision based on the total value of the property that each fire is threatening is an example of the utilitarian approach to moral problem solving.

Perhaps the most relevant virtue is justice. Suppose only two of the fires are inside the fire-control district that funds the fire-fighting brigade. Using justice as your virtue, you may decide to abide by the fire district policies. Following written policies looks suspiciously like a Kantian approach to decision making.

You may argue that the desire to be prudent came before the decision to take a utilitarian approach, or the will to be just was an antecedent to a Kantian analysis. Even if this were so, a fundamental problem remains. If the desire to exercise different virtues compels you toward different actions, which action should you take? Put another way, what is the methodology for answering the question "What would a person with strong moral character do in these circumstances?"

VIRTUE ETHICS COMPLEMENTS OTHER THEORIES

Rather than treating virtue ethics as a stand-alone theory, some ethicists believe it makes more sense to see virtue ethics as a complement to one of the other theories, such as utilitarianism. Adding virtue ethics allows ethical decision makers to consider their rationale for taking the action as well as the beneficial or harmful effects of the action.

Remember the problem of moral luck, one of the major criticisms of act utilitarianism? Since an action is judged right or wrong based solely on its consequences, an unlucky, unintended consequence can result in an action being considered wrong. Suppose your mother-in-law is in the hospital and you send her an expensive and beautiful bouquet of flowers. Unfortunately, she gets an allergic reaction to one of the flowers in the bouquet. As a result, she must spend an additional four days in the hospital. From a

purely act utilitarian point of view, you did the wrong thing when you sent your mother-in-law the flowers. In a mixed act utilitarian/virtue ethics theory, we would also take into account that you were acting out of thoughtfulness, a virtue. If nothing else, introducing the virtue ethics component makes it easier for us to think about some of the other consequences of the action. Despite the allergic reaction, your mother-in-law appreciated your kind gesture, a benefit. In addition, you strengthened your habit of thoughtfulness by practicing it on your mother-in-law, another benefit.

8.4.3 Alternative List of Fundamental Principles

The start of each section of the Code begins with the statement of a fundamental principle. For example, the first section begins with the fundamental principle, "Software engineers shall act consistently with the public interest." All of these statements of fundamental principles are expressed from the point of view of what software engineers ought to do.

Another way to devise a list of fundamental principles is to consider those virtues we would like to instill among all the members of any profession. We end up with a set of general, discipline-independent rules that cut across the eight categories of the Code. Here is an alternative list of fundamental principles derived using that approach:

1. *Be impartial.*

   The good of the general public is equally important to the good of your organization or company. The good of your profession and your company are equally important to your personal good. It is wrong to promote your agenda at the expense of your firm, and it is wrong to promote the interests of your firm at the expense of society. (Supports Clauses 1.02, 1.03, 1.05, 1.07, 3.03, 3.12, 4.01, and 6.05.)

2. *Disclose information that others ought to know.*

   Do not let others come to harm by concealing information from them. Do not make misleading or deceptive statements. Disclose potential conflicts of interest. (Supports Clauses 1.04, 1.06, 2.06, 2.07, 3.01, 4.05, 4.06, 5.05, 5.06, 6.07, 6.08, 6.09, 6.12, and 6.13.)

3. *Respect the rights of others.*

   Do not infringe on the privacy rights, property rights, or intellectual property rights of others. (Supports Clauses 2.02, 2.03, 2.05, and 3.13.)

4. *Treat others justly.*

   Everyone deserves fair wages and appropriate credit for work performed. Do not discriminate against others for attributes unrelated to the job they must do. Do not penalize others for following the Code. (Supports Clauses 5.06, 5.07, 5.08, 5.09, 5.10, 5.11, 5.12, 7.03, 7.04, 7.05, 7.07, and 8.07.)

5. *Take responsibility for your actions and inactions.*

   As a moral agent, you are responsible for the things you do, both good and bad. You may also be responsible for bad things that you allow to happen through your

inaction. (Supports Clauses 1.01, 3.04, 3.05, 3.06, 3.07, 3.08, 3.10, 3.11, 3.14, 3.15, 4.02, and 7.08.)

6. *Take responsibility for the actions of those you supervise.*

   Managers are responsible for setting up work assignments and training opportunities to promote quality and reduce risk. They should create effective communication channels with subordinates so that they can monitor the work being done and be aware of any quality or risk issues that arise. (Supports Clauses 5.01, 5.02, 5.03, and 5.04.)

7. *Maintain your integrity.*

   Deliver on your commitments and be loyal to your employer, while obeying the law. Do not ask someone else to do something you would not be willing to do yourself. (Supports Clauses 2.01, 2.04, 2.08, 2.09, 3.01, 3.02, 3.09, 4.03, 4.04, 6.06, 6.10, 6.11, 8.08, and 8.09.)

8. *Continually improve your abilities.*

   Take advantage of opportunities to improve your software engineering skills and your ability to put the Code to use. (Supports Clauses 8.01, 8.02, 8.03, 8.04, 8.05, and 8.06.)

9. *Share your knowledge, expertise, and values.*

   Volunteer your time and skills to worthy causes. Help bring others to your level of knowledge about software engineering and professional ethics. (Supports Clauses 1.08, 6.01, 6.02, 6.03, 6.04, 7.01, 7.02, and 7.06.)

In the following section we will use these fundamental, discipline-independent principles to facilitate our analysis in four case studies related to computing.

9.4 Globalization

Globalization refers to the process of creating a worldwide network of businesses and markets. Globalization results in a greater mobility of goods, services, and capital around the world. Investments are made across national boundaries. Products manufactured in

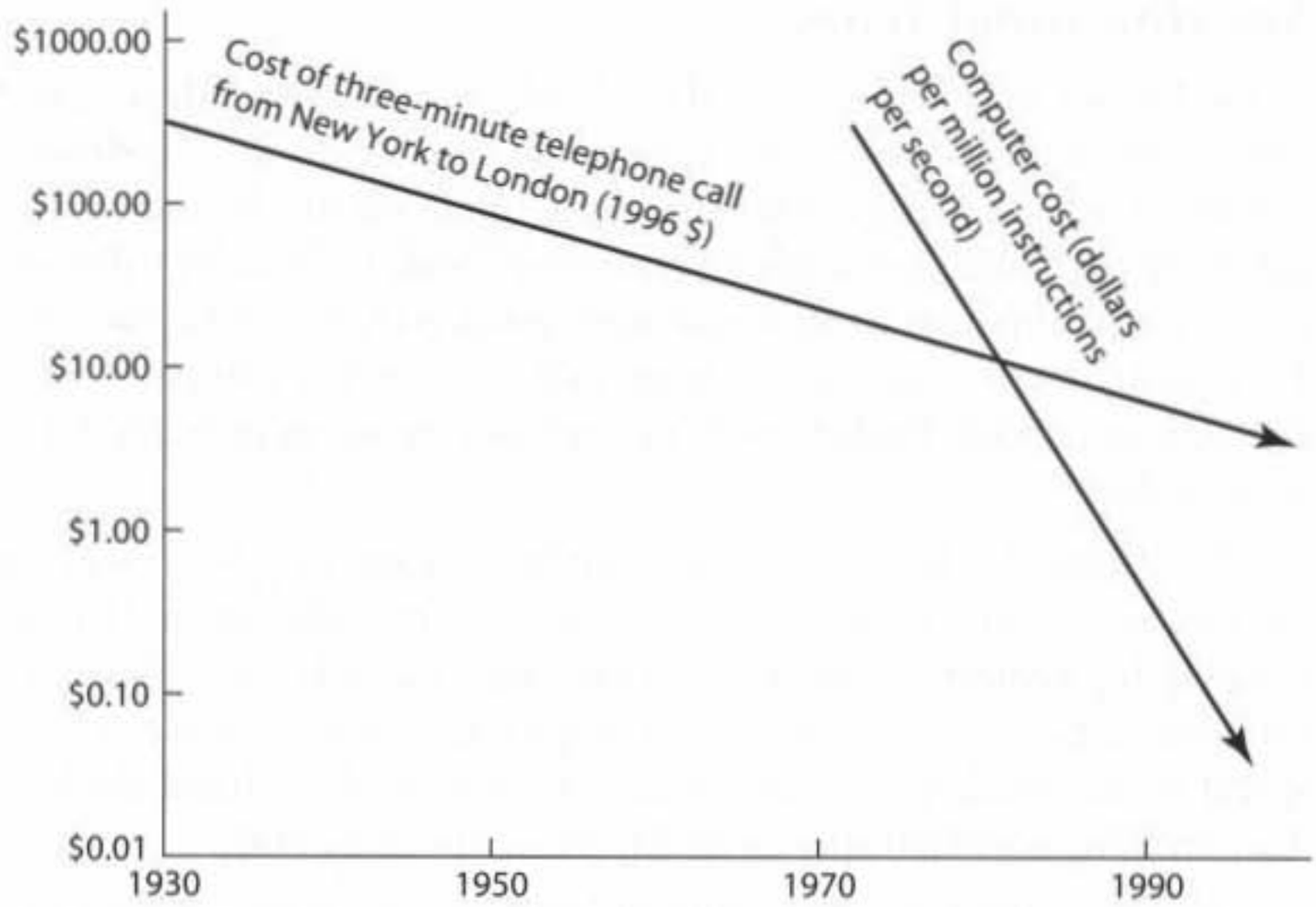

FIGURE 9.7 The dramatic declines in the cost of computing and communications have made global enterprises feasible.

one country are sold in another. Consumers calling a telephone help center get connected with support technicians located on the other side of the world.

The rapidly decreasing cost of information technology has made globalization possible (Figure 9.7). The cost of computing dropped by 99.99 percent between 1975 and 1995. The cost of an international telephone call from New York to London dropped by 99 percent between 1930 and 1996 [47]. Companies have made extensive use of low-cost information technology to coordinate operations distributed around the planet.

9.4.1 Arguments for Globalization

Those who favor globalization seek the removal of trade barriers between nations. The North American Free Trade Agreement (NAFTA) between Canada, the United States, and Mexico is a step toward globalization. The World Trade Organization (WTO) is an international body that devises rules for international trade. It promotes the goal of free trade among nations.

The WTO and other proponents of globalization support free trade with these arguments:

1. *Globalization increases competition among multiple possible providers of the same product.*

   Competition ensures that higher-quality products are sold at the best possible prices. Consumers get better prices when each area produces the goods or services it does best: corn in Kansas, automobiles in Ontario, semiconductors in Singapore,

and so on. When prices are lower, the real purchasing power of consumers is higher. Hence globalization increases everyone's standard of living.

2. *People in poorer countries deserve jobs, too.*

   When they gain employment, their prosperity increases.

3. *Every example in the past century of a poor country becoming more prosperous has been the result of that country producing goods for the world market rather than trying for self-sufficiency* [48].

4. *Creating jobs around the world reduces unrest and leads to more stability.*

   Countries with interdependent economies are less likely to go to war with each other.

9.4.2 Arguments against Globalization

Ralph Nader, American trade unions, the European farm lobby, and organizations such as Friends of the Earth, Greenpeace, and Oxfam oppose globalization. They give these reasons why globalization is a bad trend:

1. *The United States and other governments should not be subordinate to the WTO.*

   The WTO makes the rules for globalization, but nobody elected it. It makes its decisions behind closed doors. Every member country, from the United States to the tiniest dictatorship, has one vote in the WTO.

2. *American workers should not be forced to compete with foreign workers who do not receive decent pay and working conditions.*

   The WTO does not require member countries to protect the rights of their workers. It has not banned child labor. Dictatorships such as the People's Republic of China are allowed to participate in the WTO even though they do not let their workers organize into labor unions.

3. *Globalization has accelerated the loss of both manufacturing jobs and white-collar jobs overseas.*

4. *The removal of trade barriers hurts workers in foreign countries, too.*

   For example, NAFTA removed tariffs between Canada, Mexico, and the United States. Because they receive agricultural subsidies from the U.S. government, large American agribusinesses grow corn and wheat for less than its true cost of production and sell the grain in Mexico. Mexican farmers who cannot compete with these prices are driven out of business. Most of them cannot find jobs in Mexico and end up immigrating to the United States [49].

Even if globalization is a good idea, there are reasons why a company may not choose to move its facilities to the place where labor is the least expensive. Interestingly, these arguments are more relevant to "blue collar" jobs such as manufacturing than they are to "white collar" jobs such as computer programming. With automation, the cost of labor becomes a smaller percentage of the total cost of a product. Once the labor cost is reduced to a small enough fraction, it makes little difference whether the factory is

located in China or the United States. Meanwhile, there are definite additional costs associated with foreign factories. If you include products in transit, foreign factories carry more inventory than identical factories in the United States. There are also more worries about security when the product is being made in a foreign country. For these reasons, moving a factory to a less-developed country is not always in the best interest of a company [4].

9.4.3 Dot-Com Bust Increases IT Sector Unemployment

In the 1990s Intel's stock rose 3,900 percent, Microsoft's stock increased in value 7,500 percent, and Cisco System's stock soared an incredible 66,000 percent. That means $1,000 of Cisco stock purchased in 1990 was worth $661,000 at the end of 1999. Investors looking for new opportunities for high returns focused on **dot-coms,** Internet-related start-up companies. Speculators pushed up the values of many companies that had never earned a profit. Early in 2000 the total valuation of 370 Internet start-ups was $1.5 trillion, even though they had only $40 billion in sales (that's *sales*, not profits) [50].

In early 2000 the speculative bubble burst, and the prices of dot-com stocks fell rapidly. The resulting "dot-com bust" resulting in 862 high-tech start-ups going out of business between January 2000 and June 2002. Across the United States, the high-tech industry shed half a million jobs [51]. In San Francisco and Silicon Valley, the dot-com bust resulted in the loss of 13 percent of nonagricultural jobs, the worst downturn since the Great Depression [52].

9.4.4 Foreign Workers in the American IT Industry

Even while hundreds of thousands of information technology workers were losing their jobs, U.S. companies hired tens of thousands of foreigners to work in the United States. The U.S. government grants these workers visas allowing them to work in America. The two most common visas are called the H-1B and the L-1.

An H-1B visa allows a foreigner to work in the United States for up to six years. In order for a company to get an H-1B visa for a foreign employee, the company must demonstrate that there are no Americans qualified to do the job. The company must also pay the foreign worker the prevailing wage for the job. Information technology companies have made extensive use of H-1B visas to bring in skilled foreign workers and to hire foreign students graduating from U.S. universities.

In the midst of the high-tech downturn, the U.S. government continued to issue tens of thousands of H-1B visas: 163,600 in 2000–2001 and 79,100 in 2001–2002. Meanwhile, the unemployment rate among American computer science professionals was about 5.1 percent. Many of the 100,000 unemployed computer scientists complained to Congress about the large number of H-1B visas being issued. Some professional organizations argued against giving out any H-1B visas at all [53]. Congress decided to drop the H-1B quota to 65,000 for the fiscal year beginning October 1, 2003, and it initially set a quota of 65,000 for the following fiscal year. However, the 65,000 H-1B visas approved for 2004–2005 were filled in a single day; representatives of universities and technology companies said the quota was set too low [54]. Bill Gates said, "Anyone who's got

the education and the experience, they're not out there unemployed" [55]. Congress responded in May 2005 by allowing an exemption for an additional 20,000 foreigners with advanced degrees (master's or higher).

The annual quota of 65,000 H-1B visas and the exemption for 20,000 foreigners with advanced degrees remain in effect. During the deep economic recession of 2008–2009, the unemployment rate rose sharply, and the U.S. Citizenship and Immigration Service had a difficult time filling the quota mandated by Congress. With about a month to go in the 2008–2009 fiscal year, the USCIS had received only 45,000 petitions for the regular H-1B vis and about 20,000 petitions for the advanced degree exemption [56].

The other important work visa is called the L-1. American companies use L-1 visas to move workers from overseas facilities to the United States for up to seven years. For example, Intel employees in Bangalore, India, could be transferred to Hillsboro, Oregon, if they held an L-1 visa. Employees brought in to the United States under an L-1 visa do not need to be paid the prevailing wage. That saves employers money.

Critics of L-1 visas claim lower-paid foreign workers are replacing higher-paid American workers within the walls of high-tech facilities located in the United States. The U.S. Congress has put no limit on the number of L-1 visas that may be issued in any given year, but the number of foreigners working in the United States under L-1 visas is much smaller than the number holding H-1B visas. In 2006 about 50,000 foreigners were employed in the United States under this visa program [57].

9.4.5 Foreign Competition

The debate over the number of visas to grant foreign workers seeking employment in the United States should not mask another trend: the increasing capabilities of IT companies within developing nations, particularly China and India.

In 2004 IBM agreed to sell its PC division to Chinese computer manufacturer Lenovo for $1.75 billion, making Lenovo the number three manufacturer of PCs in the world [58]. A few months later, Chinese Premier Wen Jiabao visited India to encourage new collaborations between Chinese hardware companies and Indian software companies [59]. Today, China is the world's number one producer of computer hardware (Figure 9.8).

India's IT outsourcing industry is growing rapidly; Indian companies now employ more than a million people and have annual sales exceeding $17 billion. About 70 percent of these sales are in software engineering work, such as designing, programming, and maintaining computer programs. The other 30 percent of these sales are in IT-related services, such as call centers, medical transcription, and x-ray interpretation [60].

The number of college students in China is increasing rapidly, from 11 million in 2000 to 16 million in 2005 [61]. Some Chinese universities are becoming recognized for their research expertise. For example, Intel's new Pentium Extreme Edition chip makes use of a compiler developed at China's Tsinghua University [62].

More evidence of global competition comes from the annual Association for Computing Machinery Collegiate Programming Contest. When the contest began 29 years

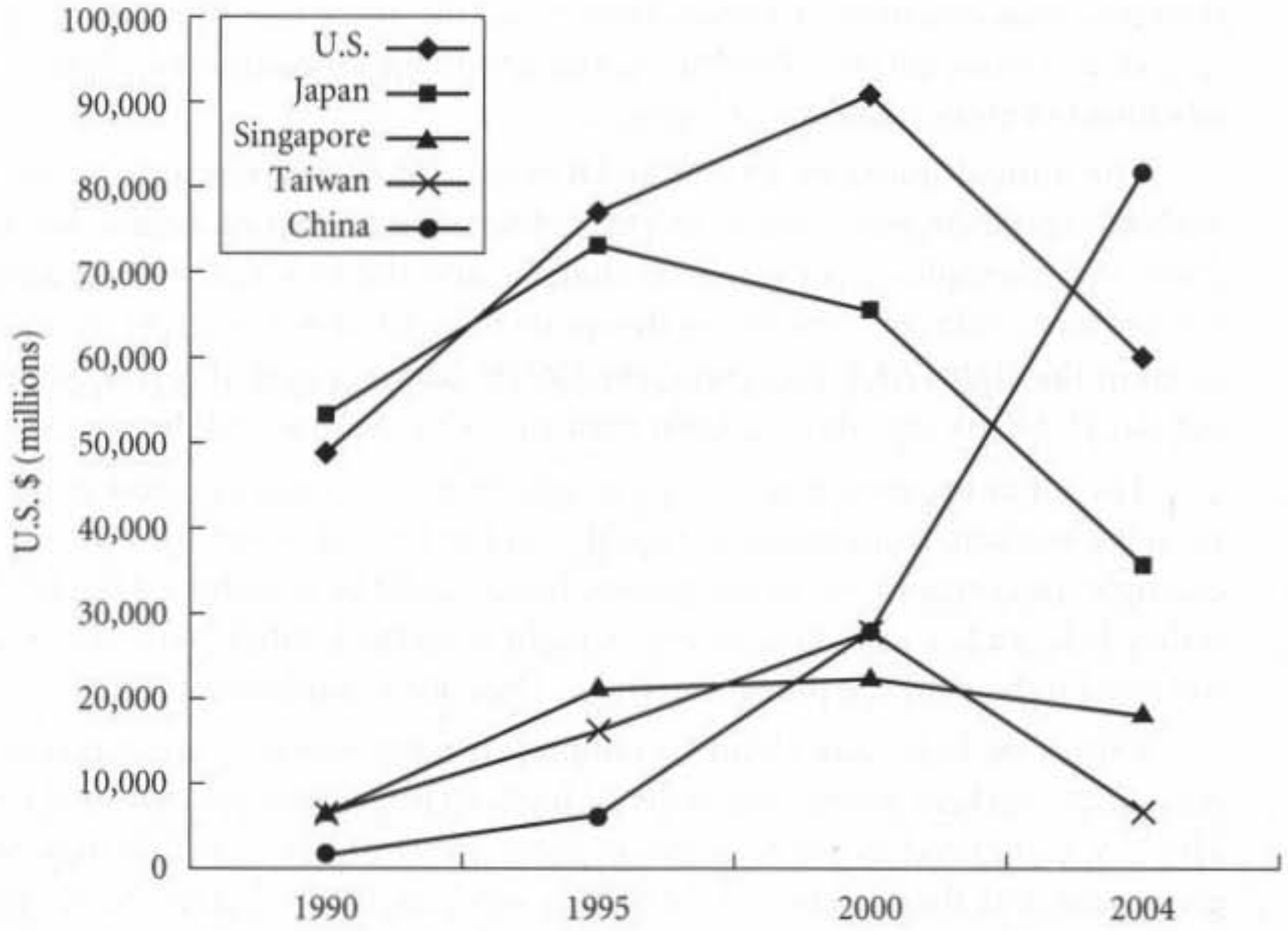

FIGURE 9.8 In 1990 China's computer-hardware industry was virtually nonexistent. By 2004 China had become the world's leading computer-hardware-producing nation.

ago, only schools from North America and Europe competed. In 2005, 4,109 teams from 71 countries entered the contest. The top 78 teams did well enough in regional competitions to attend the world finals, held in Shanghai, China. The winning team was from Shanghai Jiao Tong University, and the second and third place teams were from Russian universities. The top American team came from the University of Illinois, which tied for 17th place [63, 64]. This result was no fluke; over the next two years every team placing in the top three was from either Russia, Poland, or China, and only four American universities finished in the top 20 [65].

During the deep recession of 2008 and 2009, American corporations like Microsoft, General Electric, JP Morgan Chase, and Best Buy continued "offshoring" white-collar jobs to India and other countries in order reduce their cost of doing business [66].

9.5 The Digital Divide

The **digital divide** refers to the situation where some people have access to modern information technology while others do not. The underlying assumption motivating the term is that people who use telephones, computers, and the Internet have opportunities denied to people without access to these devices. The idea of a digital divide became popular in the mid-1990s with the rapid growth in popularity of the World Wide Web.

According to Pippa Norris, the digital divide has two fundamentally different dimensions. The **global divide** refers to the disparity in Internet access between more industrialized and less industrialized nations. The **social divide** refers to the difference in access between the rich and poor within a particular country [67].

9.5.1 Evidence of the Digital Divide

GLOBAL DIVIDE

There is plenty of evidence of what Norris calls the global divide. One piece of evidence is the percentage of people with Internet access (Figure 9.9). In 2006 about 1.1 billion people, representing about 17 percent of the world's population, had access to the Internet. Access to the Internet in Oceania, the Americas, and Europe was significantly above this average, while access in Asia and Africa was below this average. Only about 5 percent of the population—1 out of every 20 persons—had Internet access in Africa in 2006 [68].

What is hampering Internet development in less technologically developed countries?

1. *Often there is little wealth.*

   In many of these countries there is not enough money to provide everyone in the country with the necessities of life, much less pay for Internet connections.

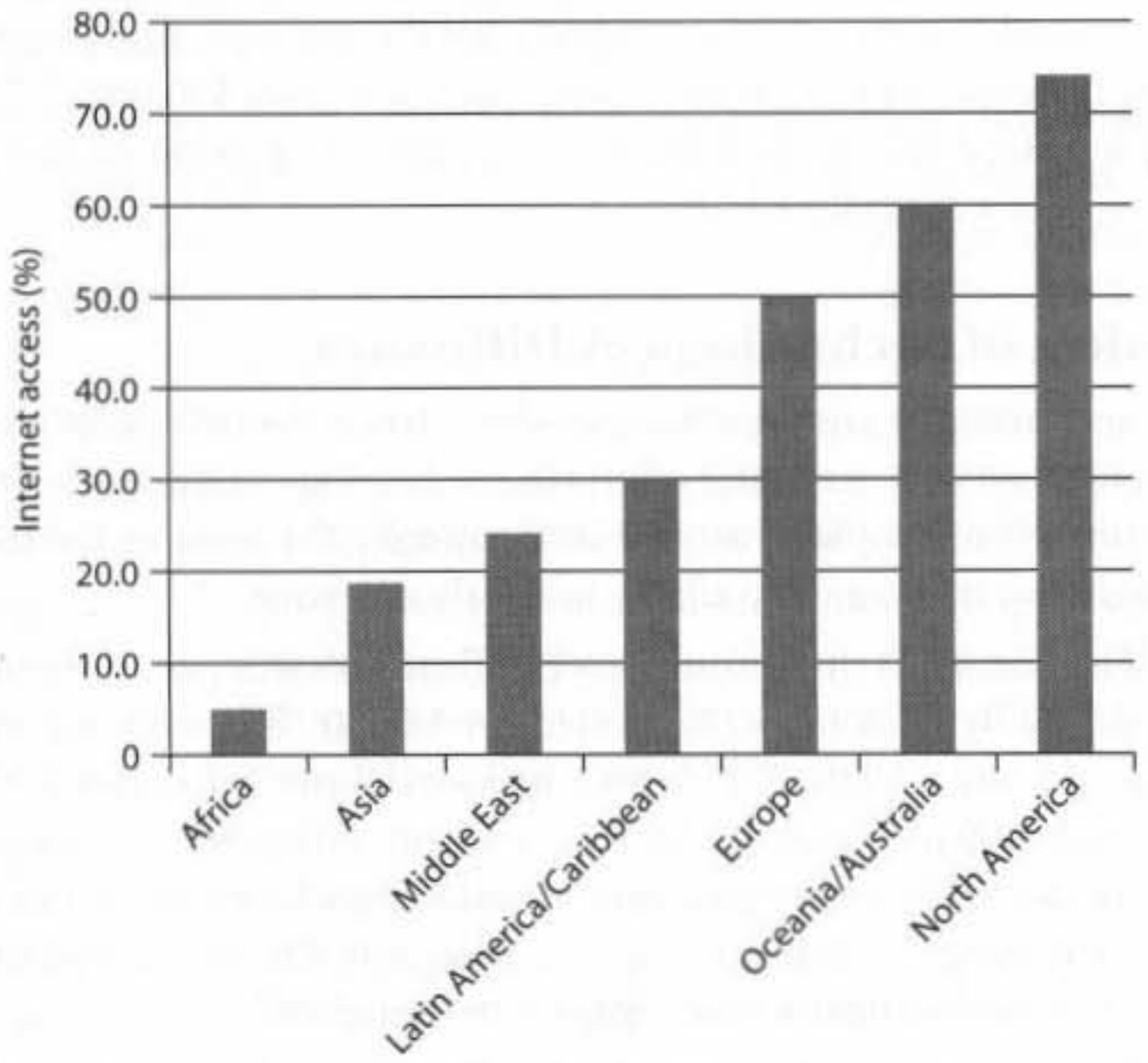

FIGURE 9.9 Percentage of people with Internet access, by world region.

2. *Many of these countries have an inadequate telecommunications infrastructure.*

   For example, less than 5 percent of the people in the African nations of Burundi, the Central African Republic, Chad, Comoros, Eritria, Ethiopia, Guinea, Malawi, Niger, Rwanda, and Sierra Leone subscribe to a telephone service [69]. Many poor people have no access to newspapers, radio, or television [67].

3. *The primary language is not English.*

   English is the dominant language for business and scientific development, giving English-speaking countries a comparative advantage with respect to competing in the global marketplace.

4. *Literacy is low, and education is inadequate.*

   Half of the population in poorer countries has no opportunity to attend secondary schools. There is a strong correlation between literacy and wealth, both for individuals and for societies [31].

5. *The country's culture may not make participating in the Information Age a priority* [70].

SOCIAL DIVIDE

Even within wealthy countries such as the United States, the extent to which people use the Internet varies widely according to age, wealth, and educational achievement. Pew Surveys polled Americans to find out how many made use of the Internet in the year 2008. Online access varied from 93 percent of 12–17 year olds to 27 percent for those 76 and over [71]. A 2006 study revealed that fully 91 percent of adults living in households with annual incomes at least $75,000 used the Internet, compared to 53 percent of adults living in households with annual incomes less than $30,000. While 91 percent of those with a college degree used the Internet, only 40 percent of those who dropped out of high school went online [72].

9.5.2 Models of Technological Diffusion

New technologies are usually expensive. Hence the first people to adopt new technologies are those who are better off. As the technology matures, its price drops dramatically, enabling more people to acquire it. Eventually the price of the technology becomes low enough that it becomes available to nearly everyone.

The history of the consumer VCR illustrates this phenomenon. The first VHS VCR, introduced by RCA in 1977, retailed for $1,000. That's $3,562 in 2009 dollars. In 2009 you could buy a VHS VCR from a mass-marketer for under $30. That means between 1977 and 2009 the price of a VCR in constant dollars fell by more than 99 percent! As the price declined, more people could afford to purchase a VCR and sales increased rapidly. The VCR progressed from a luxury that only the rich could afford into a consumer product found in nearly every American household.

Technological diffusion refers to the rate at which a new technology is assimilated into a society. Two different theories predict how a new technology is acquired by people

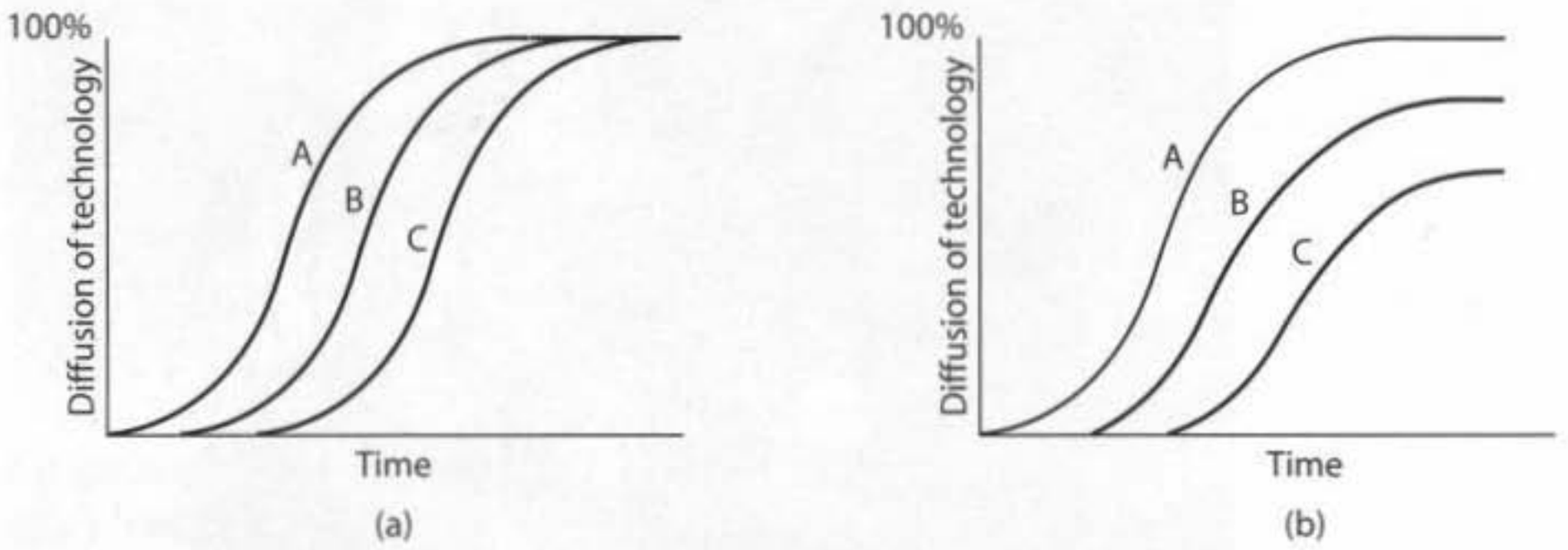

FIGURE 9.10 Two models for technological diffusion. In both models the most advantaged group A is the first to adopt a new technology, while the least advantaged group C is the last to adopt it. (a) In the normalization model, the technology is eventually embraced by nearly everyone in all groups. (b) In the stratification model, the eventual adoption rate of the technology is lower for less-advantaged groups.

in a society, based on their socioeconomic status (Figure 9.10). We divide society into three groups. People with the highest socioeconomic status are in group A, people with the lowest socioeconomic status are in group C, and group B consists of those people in the middle.

In the **normalization model** (Figure 9.10a), group A begins to adopt the technology first, followed by group B, and finally group C. However, at some point nearly everyone in all three groups is using the new technology.

In the **stratification model** (Figure 9.10b), the order of adoption is the same. However, in this model the eventual number of people in group C who adopt the technology is lower than the number of adoptees in group A. The percentage of people in group B who adopt the technology is somewhere between the levels of the other two groups.

Technological optimists believe the global adoption of information technology will follow the normalization model. Information technology will make the world a better place by reducing poverty in developing countries. Creating opportunities elsewhere will reduce the number of people trying to immigrate into the United States.

Technological pessimists believe information technology adoption will follow the stratification model, leading to a permanent condition of "haves" and "have nots." Information technology will only exacerbate existing inequalities between rich and poor nations and between rich and poor people within each nation [67].

Technological pessimists point out that the gap between the richest 20 countries and the poorest 20 countries grew between 1960 and 1995. In 1960 the average gross domestic product (GDP) of the richest countries was 18 times larger than the average GDP of the poorest countries. By 1995 the gap had grown to become 37 times greater. Some of the poorest countries grew even poorer during the last third of the twentieth century [31].

FIGURE 9.11 Unemployed workers in Ennis, Ireland, resisted using the Internet to receive their benefits, preferring to report in person to the social welfare office, where they could visit with other people. (Richard Cummins / Corbis)

9.5.3 Critiques of the Digital Divide

Mark Warschauer has suggested three reasons why the term "digital divide" is not helpful. First, it tends to promote the idea that the difference between the "haves" and the "have nots" is simply a question of access. Some politicians have jumped to the conclusion that providing technology will close the divide. Warschauer says this approach will not work. To back his claim, he gives as an example the story of a small town in Ireland.

While many factories in Ireland produce IT products, there is not a lot of use of IT among Irish citizens. Ireland's telecommunications company held a contest in 1997 to select and fund an "Information Age Town." The winner was Ennis, a town of 15,000 in western Ireland (Figure 9.11). The $22 million of prize money represented $1,200 per resident, a large sum for a poor community. Every business was equipped with an Integrated Services Digital Network (ISDN) line, a Web site, and a smart-card reader. Every family received a smart card and a personal computer.

Three years later, there was little evidence of people using the new technology. Devices had been introduced without adequately explaining to the people why they might want to use them. The benefits were not obvious. Sometimes the technology competed with social systems that were working just fine. For example, before the introduction of the new technology, unemployed workers visited the social welfare office three times a

week to sign in and get an unemployment payment. These visits served an important social function for the unemployed people. It gave them an opportunity to visit with other people and keep their spirits up. Once the PCs were introduced, the workers were supposed to "sign in" and receive their payments over the Internet. Many of the workers did not like the new system. It appears that many of the PCs were sold on the black market. The unemployed workers simply went back to reporting in person to the social welfare office.

For IT to make a difference, social systems must change as well. The introduction of information technology must take into account local culture, which includes language, literacy, and community values.

Warschauer's second criticism of the term "digital divide" is that it implies everyone is on one side or another of a huge canyon. Everybody is put into one of two categories: "haves" and "have nots." In reality access is a continuum, and each individual occupies a particular place on it. For example, how do you categorize someone who has a 56K modem connecting his PC to the Internet? Certainly that person has online access, but he is not able to retrieve the same wealth of material as someone with a broadband connection.

Thirdly, Warschauer says that the term "digital divide" implies that a lack of access will lead to a less advantaged position in society. Is that the proper causality? Models of technological diffusion show that those with a less advantaged position in society tend to adopt new technologies at a later time, which is an argument that the causality goes the other way. In reality, there is no simple causality. Each factor affects the other [31].

Rob Kling has put it this way:

> [The] big problem with "the digital divide" framing is that it tends to connote "digital solutions," i.e., computers and telecommunications, without engaging the important set of complementary resources and complex interventions to support social inclusion, of which informational technology applications may be enabling elements, but are certainly insufficient when simply added to the status quo mix of resources and relationships" [31].

Finally, Warschauer points out that the Internet does not represent the pinnacle of information technology. In the next few decades dramatic new technologies will be created. We will see these new technologies being adopted at different speeds, too.

9.5.4 Net Neutrality

The corporations that operate the long-distance Internet backbone connections in the United States have suggested that they may begin **tiered service**—charging more for higher-priority routing of Internet packets. These companies have said that tiered service will be needed in the future to guarantee a satisfactory level of service to companies that require it, such as Voice-over-IP (VoIP) providers [73].

Content providers, such as Google and Yahoo!, have combined with the American Library Association and consumer groups to oppose any notion of tiered service. These groups have asked the U.S. Congress to enact "net neutrality" legislation that

would require Internet service providers to treat all packets the same. Consumer groups suggest that if tiered service is enacted, only large corporations would be able to pay for the highest level of service. Small start-up companies wouldn't be able to compete with established corporate giants. Hence tiered service would discourage innovation and competition. Another argument against tiered service is based on the concern that companies controlling the Internet might block or degrade access to non-favored content or applications [73]. For example, a customer with an AT&T/Yahoo! DSL connection might find that high definition video content from AT&T channels performs better than high definition video from other providers [74]. Net neutrality advocates say this is unfair and must be prevented, pointing out that 95 percent of consumers have only two choices for broadband access: the local cable company or the local telephone company [75].

Opponents of "net neutrality" legislation suggest that allowing people to pay more to get a higher quality of service can sometimes be to the benefit of consumers. For example, rapid delivery of data packets is more valuable to a person using the Internet for videoconferencing than a person who simply sends email messages. Internet backbone providers argue that even though there is currently enough bandwidth, the rapidly increasing popularity of YouTube and other online video sites will soon fill the Internet's data pipes. A significant amount of money is needed to upgrade the Internet infrastructure to support the higher-bandwidth applications of the future. This money ought to come from the companies that are selling access to the data-intensive content [73].

In a 2007 report, the U.S. Federal Trade Commission concluded the market was becoming more, not less, competitive, and suggested that Congress "proceed with caution" before passing any legislation [73]. Even though it appears unlikely that Congress will pass any net neutrality legislation in the near future, Internet backbone providers are unlikely to move toward tiered service without getting some kind of approval from the FTC [76].