Unit 5: Ethical Decision in Software Development and Ethics of IT Organizations

Contents:

- Software Quality and its Importance
- Strategies for Developing Quality Software
- Use of Contingent Workers
- H-1B Workers
- Outsourcing
- Whistle-Blowing
- Green Computing

Software Quality and its Importance

High-quality software systems are systems

- that are easy to learn and use because they perform quickly and efficiently,
- they meet their users' needs, and
- they operate safely and reliably so that system downtime is kept to a minimum.

Such software has long been required to support the fields of air traffic control, nuclear power, automobile safety, health care, military and defense, and space exploration.

Computers and software are integral parts of almost every business, and the demand for high-quality software in a variety of industries is increasing.

End users cannot afford system crashes, lost work, or lower productivity. Nor can they tolerate security holes through which intruders can spread viruses, steal data, or shut down websites. Software manufacturers face economic, ethical, and organizational challenges associated with improving the quality of their software.

A software defect is any error that, if not removed, could cause a software system to fail to meet its users' needs. The impact of these defects can be trivial; for example, a computerized sensor in a refrigerator's ice cube maker might fail to recognize that the tray is full and, therefore, continue to make ice. Other defects could lead to tragedy—the control system for an automobile's antilock brakes could malfunction and send the car into an uncontrollable spin. The defect might be subtle and undetectable, such as a tax preparation package that makes a minor miscalculation; or the defect might be glaringly obvious, such as a payroll program that generates checks with no deductions for Social Security or other taxes.

Here are some recent, notable software bugs:

- The Nest thermostat is a clever device that enables users to monitor and adjust their thermostats using their smartphones. However, during a recent cold spell, a software glitch caused the devices to shut down or go offline for many customers. Temperatures in their homes plunged over night, threatening to freeze pipes and causing potentially serious health issues for the elderly and ill and those with infants.
- In 2016, software problems resulted in thousands of Blue Cross and Blue Shield of North Carolina customers being overbilled, enrolled in the wrong plans, dropped from coverage, or left without proper insurance cards. In addition, nearly 100 health care providers were unable to properly bill the insurance company for their services—and thus went unpaid—for months.

- A faulty software update at the U.S. Customs and Border Protection agency caused a shutdown of the agency's systems that are used to process travelers. The resulting massive lines and delays at airports ruined the end of vacations for many holiday travelers returning to the United States.
- Functional magnetic resonance imaging (fMRI) is used to create images that are intended to show how various areas of our brains react when we are in REM sleep, playing a game of chess, or exercising strenuously, for example. These pictures have served as the basis of tens of thousands of scientific papers and books. However, flaws in the software used to analyze fMRI data were recently uncovered by scientists who studied the results of many different brain studies over the last 15 years. According to the new report, the software flaw frequently caused false positives, suggesting brain activity where there was none. This has raised considerable controversy between critics who have long said fMRI is nothing more than high-tech pseudo medicine and brain-imaging researchers who claim that the software problems are not as serious or widespread as reported.

Software quality is the degree to which a software product meets the needs of its users. Quality management focuses on defining, measuring, and refining the quality of the development process and the products developed during its various stages. These products—including statements of requirements, flowcharts, and user documentation— are known as a deliverable. The objective of quality management is to help developers deliver high-quality systems that meet the needs of their users. Unfortunately, the first release of any software rarely meets all its users' expectations. A software product does not usually work as well as its users would like it to until it has been used for a while, found lacking in some ways, and then corrected or upgraded.

One cause of poor software quality is that many developers do not know how to design quality into software from the very start; others simply do not take the time to do so. To develop high-quality software, developers must define and follow a set of rigorous software engineering principles and be committed to learning from past mistakes. In addition, they must understand the environment in which their systems will operate and design systems that are as immune to human error as possible.

All software designers and programmers make mistakes in defining user requirements and turning them into lines of code.

Coverity (a software testing firm) performed a sophisticated scan of 17.5 million lines of code from the widely used open source packages Linux, Apache HTTP, MySQL, and Perl/PHP/Python, which are used to run millions of web servers around the world. The study found just 0.290 defects per thousand lines of code. A separate analysis found that about 10 to 20 defects per thousand lines of code are identified during in-house testing of Microsoft's applications. By the time those applications are released to the public, that error rate is in the range of 0.5 defects per thousand lines of code. That means the Microsoft Windows 10 operating system (which contains an estimated 50 million lines of code) may have included close to 25,000 defects at the time it was released. Thus, critical software used daily by workers worldwide likely contains tens of thousands of defects.

Another factor that can contribute to poor-quality software is the extreme pressure that software companies feel to reduce the time to market their products. They are driven by the need to beat the competition in delivering new functionality to users, begin generating revenue to recover the cost of development, and show a profit for shareholders. They are also driven by the need to meet quarterly earnings forecasts used by financial analysts to place a value on the stock. The resources and time needed to ensure quality are often cut under the intense pressure to ship a new product. When forced to choose between adding more user features and doing more testing, most software companies decide in favor of more features. They often reason that defects can be patched in the next release, which will give

customers an automatic incentive to upgrade. Additional features make a release more useful and therefore easier to sell to customers.

A major ethical dilemma for software development organizations is: "How much additional cost and effort should we expend to ensure that our products and services meet customers' expectations?" A study published in the International Journal of Software Engineering & Applications concluded that approximately 22 percent of Android apps on the market at the time of the study were of low quality. An app was considered to be low quality if it had one or more of the following characteristics that the developer failed to correct over time: included a poor user interface, did not meet the user requirements, had security issues, failed at certain critical moments, had compatibility and downloading issues, consumed excess battery power, or was overly expensive. Customers are stakeholders who are key to the success of a software application, and they may benefit from new features. However, they also bear the burden of errors that aren't caught or fixed during testing.

As a result of the lack of consistent quality in software, many organizations avoid buying the first release of a major software product or prohibit its use in critical systems; their rationale is that the first release often has many defects that cause problems for users. Because of the defects in the first two popular Microsoft operating systems (DOS and Windows), including their tendency to crash unexpectedly, many believe that Microsoft did not have a reasonably reliable operating system until its third major variation—Windows NT.

Even software products that have been reliable over a long period can falter unexpectedly when they are replaced with a newer version. British Airways implemented a new global check-in system in 2016, which led to five major computer outages between May and September. The result was thousands of flights canceled or delayed, tens of thousands of passengers upset and inconvenienced, and a cumulative stock market loss of 10.54 percent or £92.9 billion (USD $113 billion).

The Importance of Software Quality

A business information system is a set of interrelated components—including hardware, software, databases, networks, people, and procedures—that collects and processes data and disseminates the output. A common type of business system is one that captures and records business transactions. For example, a manufacturer's order-processing system captures order information, processes it to update inventory and accounts receivable, and ensures that the order is filled and shipped on time to the customer. Other examples are an airline's online ticket reservation system and an electronic funds transfer system that moves money among banks. The accurate, thorough, and timely processing of business transactions is a key requirement for such systems. A software defect can be devastating, resulting in lost customers and reduced revenue. How many times would bank customers tolerate having their funds transferred to the wrong account before they stopped doing business with that bank?

Another type of business information system is the decision support system (DSS), which is used to improve decision making in a variety of industries. A DSS can be used to develop accurate forecasts of customer demand, recommend stocks and bonds for an investment portfolio, or schedule shift workers in such a way as to minimize cost while meeting customer service goals. A software defect in a DSS can result in significant negative consequences for an organization and its customers.

Software is also used to control many industrial processes in an effort to reduce costs, eliminate human error, improve quality, and shorten the time it takes to manufacture products. For example, steel manufacturers use process-control software to capture data from sensors about the equipment that rolls steel into bars and about the furnace that heats the steel before it is rolled. Without process-control computers, workers could react to defects only after the fact and would have to guess at the adjustments

needed to correct the process. Process-control computers enable the process to be monitored for variations from operating standards (e.g., a low furnace temperature or incorrect levels of iron ore) and to eliminate product defects before they affect product quality. Any defect in this software can lead to decreased product quality, increased waste and costs, or even unsafe operating conditions for employees.

Software is also used to control the operation of many industrial and consumer products, such as automobiles, medical diagnostic and treatment equipment, televisions, cameras, home security systems, refrigerators, and washers. A software defect could have relatively minor consequences, such as clothes not drying long enough, or it could cause serious damage, such as a patient being overexposed to powerful X-rays.

As a result of the increasing use of computers and software in business, many companies are now in the software business whether they like it or not. The quality of software, its usability, and its timely development are critical to almost everything businesses do. The speed with which an organization develops quality software can put it ahead of or behind its competitors. Mismanaged software can be fatal to a business, causing it to miss product delivery dates, incur increased product development costs, and deliver products that have poor quality.

Business executives frequently face ethical questions of how much money and effort they should invest to ensure the development of high-quality software. A manager who takes a short-term, profit-oriented view may feel that any additional time and money spent on quality assurance will only delay a new product's release, resulting in a delay in sales revenue and profits. However, a different manager may consider it unethical not to fix all known problems before putting a product on the market and charging customers for it.

Other key questions for executives are whether their products could cause damage and what their legal exposure would be if they did. Fortunately, software defects are rarely lethal, and few personal injuries are related to software failures. However, the increasing use of software to control critical functions in vehicles as well as manage the operation of medical devices introduces product liability issues that concern many executives.

Software Product Liability

The liability of manufacturers, sellers, lessors, and others for injuries caused by defective products is commonly referred to as product liability. There is no federal product liability law; instead, product liability in the United States is mainly covered by common law (made by state judges) and Article 2 of the Uniform Commercial Code, which deals with the sale of goods.

If a software defect causes injury or loss to purchasers, lessees, or users of the product, the injured parties may be able to sue as a result. Injury or loss can come in the form of physical mishaps and death, loss of revenue, or an increase in expenses due to a business disruption caused by a software failure. Numerous product liability claims may well be in the future for the self-driving car based upon product liability claims against the vehicle manufacturer or a supplier of a component. This would come about if there was a malfunction in the electronics or software of the vehicle that lead to injuries or property damage.

Software product liability claims are typically based on strict liability, negligence, breach of warranty, or misrepresentation—sometimes in combination with one another. Each of these legal concepts is discussed in the following paragraphs.

Strict liability means that the defendant is held responsible for injuring another person, regardless of negligence or intent. The plaintiff must prove only that the software product is defective or unreasonably

dangerous and that the defect caused the injury. There is no requirement to prove that the manufacturer was careless or negligent, or to prove who caused the defect. All parties in the chain of distribution—the manufacturer, subcontractors, and distributors—are strictly liable for injuries caused by the product and may be sued.

Defendants in a strict liability action may use several legal defenses, including the doctrine of supervening event, the government contractor defense, and an expired statute of limitations. Under the doctrine of supervening event, the original seller is not liable if the software was materially altered after it left the seller's possession and the alteration caused the injury. To establish the government contractor defense, a contractor must prove that the precise software specifications were provided by the government, that the software conformed to the specifications, and that the contractor warned the government of any known defects in the software. Finally, there are statutes of limitations for claims of liability, which means that an injured party must file suit within a certain amount of time after the injury occurs.

Negligence is the failure to do what a reasonable person would do, or doing something that a reasonable person would not do. When sued for negligence, a software supplier is not held responsible for every product defect that causes customer or third-party loss. Instead, responsibility is limited to harmful defects that could have been detected and corrected through "reasonable" software development practices. Contracts written expressly to limit claims of supplier negligence may be disregarded by the courts as unreasonable. Software manufacturers and organizations with software-intensive products are frequently sued for negligence and must be prepared to defend themselves.

The defendant in a negligence case may either answer the charge with a legal justification for the alleged misconduct or demonstrate that the plaintiffs' own actions contributed to their injuries (contributory negligence). If proved, the defense of contributory negligence can reduce or totally eliminate the amount of damages the plaintiffs receive. For example, if a person uses a pair of pruning shears to trim his fingernails and ends up cutting off a fingertip, the defendant could claim contributory negligence.

A **warranty** assures buyers or lessees that a product meets certain standards of quality. A warranty of quality may be either expressly stated or implied by law. Express warranties can be oral, written, or inferred from the seller's conduct. For example, sales contracts contain an implied warranty of merchantability, which requires that the following standards be met:

- The goods must be fit for the ordinary purpose for which they are used.
- The goods must be adequately contained, packaged, and labeled.
- The goods must be of an even kind, quality, and quantity within each unit.
- The goods must conform to any promise or affirmation of fact made on the container or label.
- The quality of the goods must pass without objection in the trade.
- The goods must meet a fair average or middle range of quality.

If the product fails to meet the terms of its warranty, the buyer or lessee can sue for breach of warranty. Of course, most dissatisfied customers will first seek a replacement, a substitute product, or a refund before filing a lawsuit.

Software suppliers frequently write warranties to attempt to limit their liability in the event of nonperformance. Although a certain software application may be warranted to run on a given machine configuration, often no assurance is given as to what that software will do. However, even if a contract specifically excludes the commitment of merchantability and fitness for a specific use, the court may find such a disclaimer clause unreasonable and refuse to enforce it or refuse to enforce the entire contract. In determining whether warranty disclaimers are unreasonable, the court attempts to evaluate if the

contract was made between two “equals” or between an expert and a novice. The relative education, experience, and bargaining power of the parties and whether the sales contract was offered on a take-it-or-leave-it basis are considered in making this determination.

The plaintiff must have a valid contract that was unfulfilled by the supplier in order to win a breach-of-warranty claim. Because the software supplier writes the warranty, this claim can be extremely difficult to prove. For example, the M. A. Mortenson Company—one of the largest construction companies in the United States—installed a new version of bid preparation software for use by its estimators. During the course of preparing one bid, the software allegedly malfunctioned several times, each time displaying the same cryptic error message. Nevertheless, the estimator submitted the bid and Mortenson won the contract. Afterward, Mortenson discovered that the bid was $1.95 million lower than intended, and the company filed a breach-of-warranty suit against Timberline Software, makers of the bid software. Timberline acknowledged the existence of the bug. However, the courts ruled in Timberline’s favor because the license agreement that came with the software explicitly barred recovery of the losses claimed by Mortenson. Even if breach of warranty can be proven, damages are generally limited to the amount of money paid for the product.

Intentional misrepresentation occurs when a seller or lessor either misrepresents the quality of a product or conceals a defect in it. For example, if a cleaning product is advertised as safe to use in confined areas and some users subsequently pass out from the product’s fumes, they could sue the seller for intentional misrepresentation or fraud. Advertising, salespersons’ comments, invoices, and shipping labels are all forms of representation. Most software manufacturers use limited warranties and disclaimers to avoid any claim of misrepresentation.

Strategies for Developing Quality Software

As individuals and organizations have come to increasing rely on software, developers have identified multiple strategies for ensuring the quality of their software. These include the use of tools such as software development methodologies, the Capability Maturity Model Integration (CMMI) process-improvement model, special techniques for safety critical systems, risk management processes, and quality management standards.

Software Development Methodologies

Developing information system software is not a simple process; it requires completing many complex activities, with many dependencies among the various activities. System analysts, programmers, hardware engineers, infrastructure architects, database specialists, project managers, documentation specialists, trainers, and testers are all involved in large software projects. Each of these groups of workers has a role to play, with specific responsibilities and tasks. In addition, each group makes decisions that can affect the software’s quality and the ability of an organization or an individual to use it effectively.

Most software companies have adopted a specific software development methodology— a standard, proven work process that enables systems analysts, programmers, project managers, and others to make controlled and orderly progress in developing high-quality software. A methodology defines activities in the software development process as well as the individual and group responsibilities for accomplishing these activities (see Figure 7-1). Each methodology recommends specific techniques for accomplishing the various activities, such as using a flowchart to document the logic of a computer program. A methodology also offers guidelines for managing the quality of software during the various stages of development. If an organization has developed such a methodology, it is typically applied to any software development that the company undertakes.

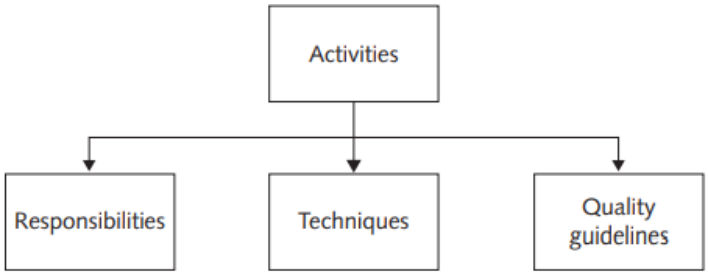

FIGURE 7-1 Components of a software development methodology

The waterfall system development model is a sequential, multistage system development process in which development of the next stage of the system cannot begin until the results of the current stage are approved or modified as necessary. This approach is referred to as a waterfall process because progress is seen as flowing steadily downward (like a waterfall) through the various stages of the development.

Under the **agile development methodology**, a system is developed in iterations (often called sprints) lasting from one to four weeks, as illustrated in Figure 7-3. Unlike the waterfall system development model, agile development accepts the fact that system requirements are evolving and cannot be fully understood or defined at the start of the project. Agile development concentrates instead on maximizing the team's ability to deliver quickly and respond to emerging requirements—hence the name agile. In an agile development project, the team evaluates the system every one to four weeks, giving it ample opportunity to identify and implement new requirements. The Manifesto for Agile Software Development (https://www.agilealliance.org/agile101/the-agile-manifesto) was developed by a group of software practitioners. The manifesto is built around a set of 4 values and 12 principles that help agile project teams make ethical decisions in the development of quality software.

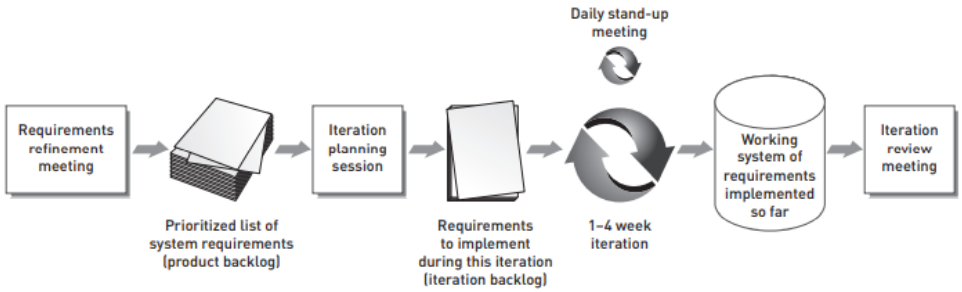

FIGURE 7-3 Agile system development methodology

TABLE 7-1 Pros and cons of waterfall and agile

Waterfall		Agile	
Pros	**Cons**	**Pros**	**Cons**
Formal review at end of each stage allows maximum management control.	Often, users' needs go unstated or are miscommunicated or misunderstood. Users may end up with a system that meets those needs as understood by the developers; however, this might not be what the users really needed.	For appropriate projects, this approach puts an application into production sooner.	It is an intense process that takes considerable time and effort on the part of project members and can result in burnout for system developers and other project participants.
Structured processes produce many intermediate products that can be used to measure progress toward developing the system.	Users can't easily review intermediate products and evaluate whether a particular product will lead to a system that meets their business requirements.	Forces teamwork and lots of interaction between users and project stakeholders so that users are more likely to get a system that meets their needs.	Requires stakeholders and users to spend more time working together on the project.

As with most things, it is usually easier and cheaper to avoid software problems at the beginning than to attempt to fix the damages after the fact. Studies have shown that the cost to identify and remove a defect in an early stage of software development can be up to 100 times less than removing a defect in a piece of software that has been distributed to customers (see Figure 7-4).13,14 (Although these studies were conducted several years ago, their results still hold true today.)

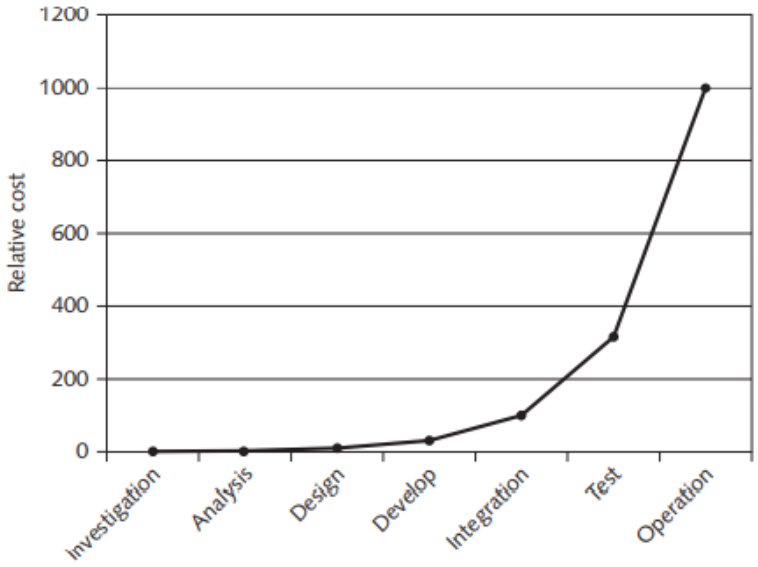

FIGURE 7-4 The cost of removing defects

Source: Used with permission from LKP Consulting Group.

If a defect is uncovered during a later stage of development, some rework of the deliverables produced in preceding stages will be necessary. The later the error is detected, the greater the number of people who will be affected by the error; thus, the greater the costs will be to communicate and fix the error. Consider the cost to communicate the details of a defect, distribute and apply software fixes, and possibly

retrain end users for a software product that has been sold to hundreds or thousands of customers. Thus, most software developers try to identify and remove errors early in the development process not only as a cost-saving measure but also as the most efficient way to improve software quality.

A product containing inherent defects that harm the user may be the subject of a product liability suit. The use of an effective methodology can protect software manufacturers from legal liability in two ways. First, an effective methodology reduces the number of software errors that might occur. Second, if an organization follows widely accepted development methods, negligence on its part is harder to prove. However, even a successful defense against a product liability case can cost hundreds of thousands of dollars in legal fees. Thus, failure to develop software carefully and consistently can have serious consequences in terms of liability exposure.

Quality assurance (QA) refers to methods within the development process that are designed to guarantee reliable operation of a product. Ideally, these methods are applied at each stage of the development cycle. However, some software manufacturing organizations without a formal, standard approach to QA consider testing to be their only QA method. Instead of checking for errors throughout the development process, such companies rely primarily on testing just before the product is shipped to ensure some degree of quality.

Several types of tests are used in software development, as discussed in the following section.

Software Testing Software is developed in units called subroutines or programs. These units, in turn, are combined to form large systems. One approach to QA is to test the code for a completed unit of software by actually entering test data and comparing the results to the expected results in a process called dynamic testing. There are two forms of dynamic testing:

- **Black-box testing** involves viewing the software unit as a device that has expected input and output behaviors but whose internal workings are unknown (a black box). If the unit demonstrates the expected behaviors for all the input data in the test suite, it passes the test. Black-box testing takes place without the tester having any knowledge of the structure or nature of the actual code. For this reason, it is often done by someone other than the person who wrote the code.
- **White-box testing** treats the software unit as a device that has expected input and output behaviors but whose internal workings, unlike the unit in blackbox testing, are known. White-box testing involves testing all possible logic paths through the software unit with thorough knowledge of its logic. The test data must be carefully constructed so that each program statement executes at least once. For example, if a developer creates a program to calculate an employee's gross pay, the tester would develop data to test cases in which the employee worked less than 40 hours, exactly 40 hours, and more than 40 hours (to check the calculation of overtime pay).

Other forms of software testing include the following:

- **Static testing**—This is a software-testing technique in which software is tested without actually executing the code. It consists of two steps—review and static analysis. During the review step, analysts and/or programmers review pertinent documentation to find and eliminate any errors in system requirements or design specifications. They also read the code that has been written. There are several types of review—informal, walk-through, peer review, and inspection—in increasing order of effort and thoroughness. During the static analysis step, special software programs called static analyzers are run against the code. Rather than reviewing input and output, the static analyzer looks for suspicious patterns in programs that might indicate a defect. Static

analyzers can identify the following types of errors: a variable with an undefined value, variables that are declared but never used, unreachable code that can never be executed, programming standards violations, and potential system security vulnerabilities. Static testing can be performed while the code is being written, prior to any other type of testing, which gives static testing an important advantage in that it can detect and eliminate defects early in the software development process when they are easier and less costly to fix.

- **Unit testing**—This involves testing individual components of code (subroutines, modules, and programs) to verify that each unit performs as intended. Unit testing is accomplished by developing test data that ideally force the code to execute all of its various functions and user features. As testers find problems, they modify the code to work correctly.
- **Integration testing**—After successful unit testing, the software units are combined into an integrated subsystem that undergoes rigorous testing to ensure that the linkages among the various subsystems work successfully.
- **System testing**—After successful integration testing, the various subsystems are combined to test the entire system as a complete entity.
- **User acceptance testing**—Trained end users conduct independent user acceptance testing to ensure that the system operates as they expect.

Capability Maturity Model Integration

Capability Maturity Model Integration (CMMI) models are collections of best practices that help organizations improve their processes. A best practice is a method or technique that has consistently shown results superior to those achieved with other means, and that is used as a benchmark within a particular industry. CMMI models are developed by product teams with members from industry, government, and the Carnegie Mellon Software Engineering Institute (SEI). The models are general enough to be used to evaluate and improve almost any process, and a specific application of CMMI—CMMI-Development (CMMI-DEV)—is frequently used to assess and improve software development practices. There are additional CMMI applications for the acquisition and delivery of products and services. CMMI defines five levels of software development maturity (see Table 7-2) and identifies the issues that are most critical to software quality and process improvement. A maturity level consists of practices for a set of process areas that improve an organization's overall performance. Identifying an organization's current maturity level enables it to specify necessary actions to improve the organization's future performance. The model also enables an organization to track, evaluate, and demonstrate its progress over the years. From 2007 to June 2016, more than 13,700 CMMI appraisals of organizations have been performed. Of these, only 11 percent were determined to be high-maturity (level 4 or 5) organizations.

TABLE 7-2 Definition of CMMI maturity levels

Maturity level	Description
Initial	Process is ad hoc and chaotic; organization tends to overcommit and processes are often abandoned during times of crisis.
Managed	Projects employ processes and skilled people; status of work products is visible to management at defined points.
Defined	Processes are well defined and understood and are described in standards, procedures, tools, and methods; processes are consistent across the organization.
Quantitatively managed	Quantitative objectives for quality and process performance are established and are used as criteria in managing projects; specific measures of process performance are collected and statistically analyzed.
Optimizing	Organization continually improves its processes; changes are based on a quantitative understanding of its business objectives and performance needs.

CMMI-DEV is a set of guidelines for 22 process areas related specifically to systems development. The premise of the model is that those organizations that do these 22 things well will have an outstanding software development process. After an organization decides to adopt CMMI-DEV, it must conduct an assessment of its software development practices (using trained, outside assessors to ensure objectivity) to determine where the organization fits in the capability model. The assessment identifies areas for improvement and establishes action plans needed to upgrade the development process. Over the course of a few years, the organization can improve its maturity level by executing the action plan.

CMMI-DEV can also be used as a benchmark for comparing organizations. In the awarding of software contracts—particularly by the federal government—organizations that bid on a contract may be required to have adopted CMMI and to be performing at a certain level.

Achieving Maturity Level 5—the highest possible rating—is a significant accomplishment for any organization, and it can lead to substantial business benefits. It means that the organization is able to statistically evaluate the performance of its software development processes. This in turn leads to better control and continual improvement in the processes, making it possible to deliver software products of high quality on time and on budget.

Honeywell is a Fortune 100 company that provides aerospace products and services, control technologies for homes and businesses, turbochargers, and performance materials to customers around the world. Quality software development is a critical part of Honeywell's commitment to produce high-quality, software-enabled products. The company has achieved CMMI Maturity Level 5 in 100 percent of its global software divisions, enabling its software teams to develop better products, faster and at a lower cost—providing Honeywell with an important competitive advantage. According to Honeywell chairman and CEO Dave Cote, "Like total quality management, a best practice widely adopted in Western manufacturing companies, CMMI is a similar best practice in software engineering."

Developing Safety-Critical Systems

Although defects in any system can cause serious problems, the consequences of software defects in certain systems can be deadly. In these kinds of systems, the stakes involved in creating quality software are raised to the highest possible level. The ethical decisions involving a trade-off—if one must be considered—between quality and factors such as cost, ease of use, and time to market require extremely serious examination.

A safety-critical system is one whose failure may cause human injury or death. The safe operation of many safety-critical systems relies on the flawless performance of software. Such systems control an ever-increasing array of products and applications, including antilock brakes, adaptive cruise control functionality, and a myriad of other safety related features found in newer automobiles; nuclear power plant reactors; airplane navigation; elevators; and a wide range of medical devices. The process of building software for such systems requires highly trained professionals, formal and rigorous methods, and state-of-the-art tools. Failure to take strong measures to identify and remove software errors from safety-critical systems "is at best unprofessional and at worst leads to disastrous consequences." However, even with these types of precautions, the software associated with safety-critical systems is still vulnerable to errors that can lead to injury or death. The following are some examples of safety-critical system failures:

- Problems with uncontrollable acceleration and a faulty antilock braking system resulted in lost lives and required Toyota to issue three separate recalls costing it nearly $3 billion.

- Neonatal ventilators manufactured by Covidien were recalled because a software problem caused the amount of air being delivered to the patient to be less that the amount specified by the physician or nurse. The problem could lead to serious injury or death.
- As many as 4.3 million General Motors cars and trucks were recalled because they had potentially defective airbags that may fail to deploy in an accident due to flawed embedded software in the vehicles.

When developing safety-critical systems, a key assumption must be that safety will not automatically result from following an organization's standard development methodology. Safety-critical software must go through a much more rigorous and time-consuming development process than other kinds of software. All tasks—including requirement definition, systems analysis, design, coding, fault analysis, testing, implementation, and change control—require additional steps, more thorough documentation, and vigilant checking and rechecking. As a result, safety-critical software takes much longer to complete and is much more expensive to develop.

Software developers working on a safety-critical system must also recognize that the software is only one component of the system; other components typically include system users or operators, hardware, and other equipment. Software developers need to work closely with safety and systems engineers to ensure that the entire system, not just the software, operates in a safe manner.

The key to ensuring that these additional tasks are completed is to appoint a system safety engineer, who has explicit responsibility for the system's safety. The safety engineer uses a logging and monitoring system to track hazards from a project's start to its finish. This hazard log is used at each stage of the software development process to assess how it has accounted for detected hazards. Safety reviews are held throughout the development process, and a robust configuration management system tracks all safety-related matters. However, the safety engineer must keep in mind that his or her role is not simply to produce a hazard log but rather to influence the design of the system to ensure that it operates safely when put into use.

The increased time and expense of completing safety-critical software can draw developers into ethical dilemmas. For example, the use of hardware mechanisms to back up or verify critical software functions can help ensure safe operation and make the consequences of software defects less critical. However, such hardware may make the final product more expensive to manufacture or harder for the user to operate—potentially making the product less attractive than a competitor's. Companies must carefully weigh these issues to develop the safest possible product that also appeals to customers.

Another key issue is deciding when the QA staff has performed sufficient testing. How much testing is enough when you are building a product whose failure could cause loss of human life? At some point, software developers must determine that they have completed sufficient QA activities and then sign off

When designing, building, and operating a safety-critical system, a great deal of effort must be put into considering what can go wrong, the likelihood and consequences of such occurrences, and how risks can be averted, mitigated, or detected so the users can be warned. One approach to answering these questions is to conduct a formal risk analysis.

Risk Management

Risk is the potential of gaining or losing something of value.

Risk can be quantified by three elements:

- a risk event,

- the probability of the event happening, and
- the impact (positive or negative) on the business outcome if the risk does actually occur.

The annualized rate of occurrence (ARO) is an estimate of the probability that this event will occur over the course of a year.

The single loss expectancy (SLE) is the estimated loss that would be incurred if the event happens.

The annualized loss expectancy (ALE) is the estimated loss from this risk over the course of a year.

The following equation is used to calculate the annual loss expectancy: ARO × SLE =ALE

For example, if an undesirable event has a one percent probability of occurring over the course of a year (ARO) and the consequences of that event occurring would cost $1,000,000 (SLE), then the annualized loss expectancy (ALE) can be calculated as: 0.01× $1,000,000 = $10,000

The risk for this event would be considered greater than that of an event that has a 10 percent probability of occurring, at a cost of $50,000 per occurrence.

0.10 × $50,000 = $5,000

Risk management is the process of identifying, monitoring, and limiting risks to a level that an organization is willing to accept. The level of risk that remains after managing risk is called residual risk. Ultimately, senior management must choose a level of acceptable residual risk based on the organization's goals and the resources (people, dollars, and time) the organization is willing to dedicate to mitigate the risk. Strategies for addressing a particular risk include the following:

- Acceptance—When the cost of avoiding a risk outweighs the potential loss of a risk, an organization will likely accept the risk. For example, spending $1 million to avoid a risk that might cost the organization $1,000 per year clearly doesn't make sense. A decision to accept a risk can be extremely difficult and controversial when dealing with safety-critical systems because making that determination involves forming personal judgments about the value of human life, assessing potential liability in case of an accident, evaluating the potential impact on the surrounding natural environment, and estimating the system's costs and benefits.
- Avoidance—An organization may choose to eliminate the vulnerability that gives rise to a particular risk in order to avoid the risk altogether. This is the most effective solution, but often not possible due to organizational requirements and factors beyond an organization's control.
- Mitigation—Risk mitigation involves the reduction in either the likelihood or the impact of the occurrence of a risk. N-version programming is an approach to minimizing the impact of software errors by independently implementing the same set of user requirements N times (where N could be 2, 3, 4 or more); N versions of software are run in parallel; and, if a difference is found, a "voting algorithm" is executed to determine which result to use. For example, if two software versions calculated the answer to a particular question to be 2.4 and the third version calculated 4.1, the algorithm might choose 2.4 as the correct answer. In N-version programming, each software version is built by different teams of people using different approaches to write programming instructions designed to meet the user's requirements. In some cases, instructions are written by teams of programmers from different companies and run on different hardware devices. The rationale behind N-version programming is that multiple software versions are highly unlikely to fail at the same time under the same conditions. Thus, one or more of versions should yield a correct result. Triple-version programming is common in airplane and spacecraft control systems.

- Redundancy is the provision of multiple interchangeable components to perform a single function in order to cope with failures and errors. An example of a simple redundant system would be an automobile with a spare tire or a parachute with a backup chute attached. A more complex redundant system is a redundant array of independent disks (RAID), which is commonly used in high-volume data storage for file servers. RAID systems use many small capacity disk drives to store large amounts of data to provide increased reliability and redundancy. Should one of the drives fail, it can be removed from service and a new one substituted in its place.
- Transference—A common way to accomplish risk transference is for an individual or an organization to purchase insurance, such as auto or business liability insurance. Another way to transfer risk is to outsource the risk by contracting with a third party to manage the risk.

Manufacturers of safety-critical systems must sometimes decide whether to recall a product when data indicate a problem. For example, automobile manufacturers have been known to weigh the cost of potential lawsuits against that of a recall. Drivers and passengers in affected automobiles (and, in many cases, the courts) have not found this approach to be ethically sound. Manufacturers of medical equipment and airplanes have sometimes made similar decisions. Making such a decision is often extremely complicated, especially if the available data cannot pinpoint the cause of a particular problem. For example, there was great controversy over the use of Firestone tires on Ford Explorers after numerous tire blowouts and Explorer rollovers caused multiple injuries and deaths. However, it was difficult to determine if the rollovers were caused by poor automobile design, faulty tires, or improperly inflated tires. Consumers' confidence in both manufacturers and their products was nevertheless shaken.

Reliability is a measure of the rate of failure in a system that would render it unusable over its expected lifetime. For example, if a component has a reliability of 99.9 percent, it has a one in one thousand chance of failing over its lifetime. Although this chance of failure may seem low, remember that most systems are made up of many components. As you add more components, the system becomes more complex, and the chance of failure increases. For example, assume that you are building a complex system made up of seven components, each with 99 percent reliability. If none of the components has redundancy built in, the system has a 93.8 percent (.997) probability of operating successfully with no component malfunctions over its lifetime. If you build the same type of system using 10 components, each with 99 percent reliability, the overall probability of operating without an individual component failure falls to 90 percent. Thus, building redundancy into systems that are both complex and safety critical is imperative. System engineers sometimes refer to a system with "five nines" (99.999 percent) reliability. This translates to a system that would have only 5.4 minutes of total downtime in a year.

Reliability and safety are two different system characteristics. Reliability has to do with the capability of the system to continue to perform; safety has to do with the ability of the system to perform in a safe manner. Thus, a system could be reliable but not safe. For example, an antiaircraft missile control system may continue to operate under a wide range of operating conditions so that it is considerably reliable. If, however, the control system directs the missile to change direction and to fly back into its launching device, it is certainly unsafe.

One of the most important and challenging areas of safety-critical system design is the system–human interface. Human behavior is not nearly as predictable as the performance of hardware and software components in a complex system. The system designer must consider what human operators might do to make a system work less safely or effectively. The challenge is to design a system that works as it should and leaves little room for erroneous judgment on the part of the operator. For instance, a self-medicating pain-relief system must allow a patient to press a button to receive more pain reliever, but must also regulate itself to prevent an overdose. Additional risk can be introduced if a designer does not anticipate

the information an operator needs and how the operator will react under the daily pressures of actual operation, especially in a crisis. Some people keep their wits about them and perform admirably in an emergency, but others may panic and make a bad situation worse.

Poor design of a system interface can greatly increase risk, sometimes with tragic consequences. For example, in July 1988, the guided missile cruiser USS Vincennes mistook an Iranian Air commercial flight for an enemy F-14 jet fighter and shot the airliner down over international waters in the Persian Gulf. All 290 people on board were killed. Some investigators blamed the tragedy on a lack of training and experience on the part of the operators and the confusing interface of the $500-million Aegis radar and weapons control system. The Aegis radar on the Vincennes locked onto an Airbus 300, but it was misidentified as a much smaller F-14 by its human operators. The Aegis operators also misinterpreted the system signals and thought that the target was descending, even though the airbus was actually climbing. A third human error was made in determining the target altitude—it was off by 4,000 feet. As a result of this combination of human errors, the Vincennes crew thought the ship was under attack and shot down the plane.

Quality Management Standards

The International Organization for Standardization (ISO), founded in 1947, is a worldwide federation of national standards bodies from 161 countries. The ISO issued its 9000 series of business management standards in 1988. These standards require organizations to develop formal quality-management systems that focus on identifying and meeting the needs, desires, and expectations of their customers.

The ISO 9001 family of standards serves as a guide to quality products, services, and management. ISO 9001 provides a set of standardized requirements for a quality management system. In 2015, more than 1.5 million ISO 9001 certificates were issued to organizations around the world. ISO standards are updated every five years to keep them current and relevant. Although companies can use the ISO standards as a management guide for their own purposes in achieving effective control, the priority for many companies is having a qualified external agency certify that they have achieved ISO 9001 certification. Many businesses and government agencies both in the United States and abroad insist that a potential vendor or business partner have a certified quality management system in place as a condition of doing business. Becoming ISO 9001 certified provides proof of an organization's commitment to quality management and continuous improvement.

To obtain this coveted certificate, an organization must submit to an examination by an external assessor and must fulfill the following requirements:

- Have written procedures for all processes
- Follow those procedures
- Prove to an auditor that it has fulfilled the first two requirements; this proof can require observation of actual work practices and interviews with customers, suppliers, and employees.

Many software development organizations are applying for ISO 9001 to meet the special needs and requirements associated with the purchase, development, operation, maintenance, and supply of computer software.

Failure Mode and Effects Analysis

Failure mode and effects analysis (FMEA) is an important technique used to develop ISO 9000–compliant quality systems by both evaluating reliability and determining the effects of system and equipment failures. Failures are classified according to their impact on a project's success, personnel safety,

equipment safety, customer satisfaction, and customer safety. The goal of FMEA is to identify potential design and process failures early in a project, when they are relatively easy and inexpensive to correct.

A failure mode describes how a product or process could fail to perform the desired functions described by the customer. An effect is an adverse consequence that the customer might experience. Unfortunately, most systems are so complex that there is seldom a one-to-one relationship between cause and effect. Instead, a single cause may have multiple effects, and a combination of causes may lead to one effect or multiple effects. It is not uncommon for a FMEA of a system to identify 50 to 200 potential failure modes.

The use of FMEA helps to prioritize those actions necessary to reduce potential failures with the highest relative risks. The following steps are used to identify the highest priority actions to be taken:

- Determine the severity rating: The potential effects of a failure are scored on a scale of 1 to 10 (or 1 to 5) with 10 assigned to the most serious consequence (9 or 10 are assigned to safety- or regulatory-related effects).
- Determine the occurrence rating: The potential causes of that failure occurring are also scored on a scale of 1 to 10, with 10 assigned to the cause with the greatest probability of occurring.
- Determine the criticality: Criticality is the product of severity times occurrence.
- Determine the detection rating: The ability to detect the failure in advance of it occurring due to the specific cause under consideration is also scored on a scale of 1 to 10, with 10 assigned to the failure with the least likely chance of advance detection. For software, the detection rating would represent the ability of planned tests and inspections to remove the cause of a failure.
- Calculate the risk priority rating: The severity rating is multiplied by the occurrence rating and by the detection rating to arrive at the risk priority rating. Raytheon is a technology company that designs and manufactures aerospace and defense systems that incorporate cutting-edge electronic components. The company employs 63,000 people worldwide and generated $24 billion in recent sales.

Raytheon employs FMEA throughout its product development lifecycle. Starting early in the design cycle, the company invites suppliers to review its designs to identify potential failure modes, assess the ability to detect the modes, and estimate the severity of the effects. Raytheon then uses this input to prioritize the product design issues that need to be eliminate or mitigated to create superior products.

Table 7-3 shows a sample FMEA risk priority table.

TABLE 7-3 Sample FMEA risk priority table

Issue	Severity	Occurrence	Criticality	Detection	Risk priority
#1	3	4	12	9	108
#2	9	4	36	2	72
#3	4	5	20	4	80

Many organizations consider those issues with the highest criticality rating (severity occurrence) as the highest priority issues to address. They may then go on to address those issues with the highest risk priority (severity × occurrence × detection). So although Issue #2 shown in Table 7-3 has the lowest risk priority, it may be assigned the highest priority because of its high criticality rating. Table 7-4 provides a manager's checklist for upgrading the quality of the software an organization produces. The preferred answer to each question is yes.

What is meant by software quality, why is it so important, and what potential ethical issues do software manufacturers face when making decisions that involve trade-offs between project schedules, project costs, and software quality?

- High-quality software systems are easy to learn and use. They perform quickly and efficiently to meet their users' needs, operate safely and reliably, and have a high degree of availability that keeps unexpected downtime to a minimum.
- High-quality software has long been required to support the fields of air traffic control, nuclear power, automobile safety, health care, military and defense, and space exploration.
- Computers and software are integral parts of almost every business, and the demand for high-quality software is increasing. End users cannot afford system crashes, lost work, or lower productivity. Nor can they tolerate security holes through which intruders can spread viruses, steal data, or shut down websites.
- A software defect is any error that, if not removed, could cause a software system to fail to meet its users' needs.
- Software quality is the degree to which a software product meets the needs of its users. Quality management focuses on defining, measuring, and refining the quality of the development process and the products developed during its various stages.
- Software developers are under extreme pressure to reduce the time to market of their products. They are driven by the need to beat the competition in delivering new functionality to users, to begin generating revenue to recover the cost of development, and to show a profit for shareholders.
- The resources and time needed to ensure quality are often cut under the intense pressure to ship a new software product. When forced to choose between adding more user features and doing more testing, many software companies decide in favor of more features.
- A business information system is a set of interrelated components—including hardware, software, databases, networks, people, and procedures—that collects and processes data and disseminates the output.
- Software product liability claims are typically based on strict liability, negligence, breach of warranty, or misrepresentation—sometimes in combination. Strict liability means that the defendant is held responsible for injuring another person regardless of negligence or intent.
- A warranty assures buyers or lessees that a product meets certain standards of quality and may be either expressly stated or implied by law. If the product fails to meet the terms of its warranty, the buyer or lessee can sue for breach of warranty.

What are some effective strategies for developing quality systems?

- A software development methodology is a standard, proven work process that enables systems analysts, programmers, project managers, and others to make controlled and orderly progress in developing high-quality software. Software methodologies define the activities in the software development process as well as the individual and group responsibilities for accomplishing objectives, recommend specific techniques for accomplishing the objectives, and offer guidelines for managing the quality of the products during the various stages of the development cycle. The waterfall system development model is a sequential, multistage system development process in which development of the next stage of the system cannot begin until the results of the current stage are approved or modified as necessary.
- Under the agile development methodology, a system is developed in iterations (often called sprints), lasting from one to four weeks. Agile development, which accepts the fact that system

requirements are evolving and cannot be fully understood or defined at the start of the project, concentrates on maximizing the team's ability to deliver quickly and respond to emerging requirements.

- Using an effective development methodology enables a manufacturer to produce high-quality software, forecast project-completion milestones, and reduce the overall cost to develop and support software. An effective development methodology can also help protect software manufacturers from legal liability for defective software in two ways: by reducing the number of software errors that could cause damage and by making negligence more difficult to prove.
- The cost to identify and remove a defect in the early stages of software development can be up to 100 times less than removing a defect in a piece of software that has been distributed to customers.
- Quality assurance (QA) refers to methods within the development process that are designed to guarantee reliable operation of a product. Ideally, these methods are applied at each stage of the development cycle.
- There are several tests employed in software development including black-box and whitebox dynamic testing, static testing, unit testing, integration testing, system testing, and user acceptance testing.
- Capability Maturity Model Integration (CMMI) models are collections of best practices that help organizations improve their processes. A best practice is a method or technique that has consistently shown results superior to those achieved with other means, and that is used as a benchmark within a particular industry. CMMI-Development (CMMI-DEV)—is frequently used to assess and improve software development practices.
- CMMI defines five levels of software development maturity: initial, managed, defined, quantitatively managed, and optimizing. CMMI identifies the issues that are most critical to software quality and process improvement. Its use can improve an organization's ability to predict and control quality, schedule, costs, and productivity when acquiring, building, or enhancing software systems. CMMI also helps software engineers analyze, predict, and control selected properties of software systems.
- A safety-critical system is one whose failure may cause human injury or death. In the development of safety-critical systems, a key assumption is that safety will not automatically result from following an organization's standard software development methodology.
- Safety-critical software must go through a much more rigorous and time-consuming development and testing process than other kinds of software; the appointment of a project safety engineer and the use of a hazard log and risk analysis are common in the development of safety-critical software.
- Risk is the potential of gaining or losing something of value. Risk can be quantified by three elements: a risk event, the probability of the event happening, and the impact (positive or negative) on the business outcome if the risk does actually occur.
- The annualized rate of occurrence (ARO) is an estimate of the probability that an event will occur over the course of a year. The single loss expectancy (SLE) is the estimated loss that would be incurred if the event happens. The annualized loss expectancy (ALE) is the estimated loss from this risk over the course of a year.
- The following equation is used to calculate the annual loss expectancy: ARO × SLE =ALE.
- Risk management is the process of identifying, monitoring, and limiting risks to a level that an organization is willing to accept.
- Reliability is a measure of the rate of failure in a system that would render it unusable over its expected lifetime.

- The International Organization for Standardization (ISO) issued its 9000 series of business management standards in 1988. These standards require organizations to develop formal quality management systems that focus on identifying and meeting the needs, desires, and expectations of their customers.
- The ISO 9001 family of standards serves as a guide to quality products, services, and management; it provides a set of standardized requirements for a quality management system. Many businesses and government agencies specify that a vendor must be ISO 9001 certified to win a contract from them.
- Failure mode and effects analysis (FMEA) is an important technique used to develop ISO 9001–compliant quality systems. FMEA is used to evaluate reliability and determine the effects of system and equipment failures.

Use of Contingent Workers

The Bureau of Labor Statistics defines contingent work as a job situation in which an individual does not have an explicit or implicit contract for long-term employment. A firm is likely to use contingent IT workers if it experiences pronounced fluctuations in its technical staffing needs. For example, contingent workers may be hired as consultants on an organizational restructuring project, as technical experts on a product development team, and as supplemental staff for many other short-term projects, such as the design and installation of a new information system. Typically, these workers join a blended team of full-time employees and other contingent workers for the life of the project and then move on to their next assignment. Whether they work, when they work, and how much they work depends on the company's need for them. They have neither an explicit nor an implicit contract for continuing employment. Organizations can obtain contingent workers through temporary staffing firms, employee leasing organizations, and professional employer organizations (PEOs).

Temporary staffing firms recruit, train, and test job seekers in a wide range of job categories and skill levels, and then assign them to clients as needed. Temporary employees are often used to fill in during staff vacations and illnesses, handle seasonal workloads, and help staff special projects. However, they are not employees of the client company, so they are not eligible for company benefits such as vacation, sick pay, and medical insurance. Temporary working arrangements may appeal to people who want maximum flexibility in their work schedules, as well as a variety of work experiences. Other workers take temporary work assignments because they are unable to find more permanent work.

In **employee leasing**, a business (called the subscribing firm) transfers all or part of its workforce to another firm (called the leasing firm), which handles all human resource-related activities and costs, such as payroll, training, and the administration of employee benefits. The subscribing firm leases these workers, but they remain employees of the leasing firm. Employee leasing creates a coemployment relationship, in which two employers have actual or potential legal rights and duties with respect to the same employee or group of employees. Employee leasing firms are subject to special regulations regarding workers' compensation and unemployment insurance. Because the workers are technically employees of the leasing firm, they may be eligible for some company benefits through the leasing firm. Once the relationship between the employee leasing firm and its client ends, the leased employees remain with the leasing firm and move on to other projects with new clients.

A professional employer organization (PEO) is a business entity that coemploys the employees of its clients and typically assumes responsibility for all human resource management functions. The PEO typically remits wages and withholdings of the worksite employees and reports, collects, and deposits employment taxes with local, state, and federal authorities. The PEO also issues the Form W-2 for the compensation

paid by it under its EIN (Employer Identification Number). The client company remains responsible for directing and controlling the daily activities of the employees, tracking actual hours worked and reporting them to the PEO for payment processing, and ensuring that payroll funds are paid to the PEO. The exact terms of the arrangement are specified in a client service agreement. The client maintains a long-term investment and commitment to the employees, but uses the PEO as a means to outsource the human resource activities. If the agreement between the client company and the PEO is terminated, the workers continue to be employed by the client. By assigning the nonrevenue-producing administrative tasks to the PEO, the client company can focus on managing and growing its business while letting administrative tasks be handled by human resource experts, ideally at a lower total cost.

The Gig Economy

The term gig, which originated in the entertainment business, refers to a short-term job. The Bureau of Labor Statistics refers to a gig as a "a single project or task for which a worker is hired—often through a digital marketplace—to work on demand." The gig economy refers to a work environment in which temporary positions are common and organizations contract with independent workers for short-term engagements. In the gig economy, instead of earning a regular hourly wage, workers get paid for specific gigs, such as complete testing of a unit of software code, writing end user documentation, or delivering a training class via webinar. Employers are desperate for highly qualified workers with specialized skills. And thanks to the power of the Internet and the nature of much IT work, those individuals may be located anywhere in the world. Many of those workers are willing to work a gig for highly competitive rates. Online staffing websites such as 99 Designs, Freelancer, Guru, Toptal, Witmart, and over a dozen others can help organizations find these workers. Some of these websites draw from a database of over one million professionals, enabling organizations to find the talent they need with just a few clicks. Organizations can post their project and have contractors bid on it, or they can search for workers and contact those they are interested in directly. Some sites vet the workers to some degree, or at least let potential employers view feedback and ratings of prior clients. The sites typically handle all payment services.

Independent Contractors

An independent contractor is an individual who provides services to another individual or organization according to terms defined in a written contract or within a verbal agreement. A study by Intuit predicted that by 2020, more than 40 percent of American workers would be independent contractors. There are many factors driving this trend toward the use of independent contractors, as shown in Table 10-1.

TABLE 10-1 Factors behind the trend toward independent contractors

From the employee's perspective	From the employer's perspective
Freedom to select from among temporary jobs and projects around the world	Ability to choose the best individuals for a specific project from a larger pool of candidates than that available in a given geographic area
Opportunity to change "jobs" frequently	Financial pressure to reduce staff and associated costs, such as payroll and benefits as well as costs associated with office space and training
Greater flexibility in terms of work hours and location where work is performed	Enables organization to focus on its core functions and on building its business

Organizations that use contingent workers must be extremely careful how they pay and treat those workers, or they run the risk of getting dragged into a class action lawsuit over misclassification of workers.

Table 10-2 details some of the factors that come into play in classifying a worker as either an employee or an independent contractor.

TABLE 10-2 Factors that are considered in classifying a worker as either an employee or an independent contractor

Employee	Independent contractor
Receives net salary after employer has withheld income, Social Security, and Medicare taxes	No income, Social Security, or Medicare taxes are withheld from paycheck; worker must make arrangements to pay his or her own taxes
Receives a W-2 form from the employer for the purposes of filing federal, state, and local taxes	Receives a form 1099-MISC for amounts exceeding $599 of nonemployee income for the purposes of filing federal, state, and local income taxes
Eligible for employment benefits, such as health and disability insurance, vacation and holiday pay, and contributions to an employee-sponsored retirement account	Receives no employment benefits from the employer
Eligible to receive unemployment compensation after layoff or termination	Not eligible for unemployment compensation benefits
Covered by federal and state wage and hour laws, such as those related to minimum wage and overtime	Paid according to the terms of the contract, typically does not receive overtime pay
Can receive worker's compensation benefits for any workplace injury	Not eligible for worker's compensation benefits
Protected by workplace safety and employment antidiscrimination laws	Not protected by employment antidiscrimination and workplace safety laws
Unless employment is "at will," can be terminated by the employer only for just cause and with prior notice	Unless the consulting contract is for a specified term, can be let go by the employer for any reason, at any time
Employer has right to control or direct not only the result of the work but also how it will be done and when	Client has the right to control or direct only the result of the work; other than that, worker operates independently and decides what will be done and how the work will be accomplished
Works hours set by the employer	Sets own work hours
All equipment, materials, and tools are provided to perform the work	Provides his or her own equipment, materials, and tools

Advantages of Using Contingent Workers

When a firm employs a contingent worker,

- It does not usually have to provide benefits such as insurance, paid time off, and contributions to a retirement plan.
- A company can easily adjust the number of contingent workers it uses to meet its business needs and can release contingent workers when they are no longer needed. An organization cannot

usually do the same with full-time employees without creating a great deal of ill will and negatively impacting employee morale.

- Moreover, because many contingent workers are already specialists in a particular task, a firm does not customarily incur training costs for contingent workers. Therefore, the use of contingent workers can enable a firm to meet its staffing needs more efficiently, lower its labor costs, and respond more quickly to changing market conditions.

Disadvantages of Using Contingent Workers

- One downside to using contingent workers is that those workers may not feel a strong connection to the company for which they are working. This can result in a low commitment to the company and its projects, along with a high turnover rate.
- Although contingent workers may already have the necessary technical training for a temporary job, many contingent workers gain additional skills and knowledge while working for a particular company; those assets are lost to the company when a contingent worker departs at a project's completion.

Deciding When to Use Contingent Workers

- When an organization decides to use contingent workers for a project, it should recognize the trade-off it is making between completing a single project quickly and cheaply versus developing people within its own organization.
- If the project requires unique skills that are probably not necessary for future projects, there may be little reason to invest the additional time and costs required to develop those skills in full-time employees. Or, if a particular project requires only temporary help that will not be needed for future projects, the use of contingent workers is a good approach. In such a situation, using contingent workers avoids the need to hire new employees and then fire them when staffing needs decrease.
- However, organizations should carefully consider whether or not to use contingent workers when those workers are likely to learn corporate processes and strategies that are key to the company's success. It is next to impossible to prevent contingent workers from passing on such information to subsequent employers. This can be damaging if the worker's next employer is a major competitor.
- Although using contingent workers is often the most flexible and cost-effective way to get a job done, their use can raise ethical and legal issues about the relationships among the staffing firm, its employees, and its customers—including the potential liability of a staffing firm's clients for withholding payroll taxes, payment of employee retirement benefits and health insurance premiums, and administration of workers' compensation to the staffing firm's employees.

H-1B Workers

An H-1B visa is a temporary work visa granted by the U.S. Citizenship and Immigration Services (USCIS) for people who work in specialty occupations—jobs that require at least a four-year bachelor's degree in a specific field, or equivalent experience.

People working in the United States on H-1B visa are referred to as "nonimmigrant workers" because they are in the country on a temporary work visa; they are not being granted a visa to immigrate to the United States. Many companies turn to H-1B workers to meet critical business needs or to obtain essential technical skills and knowledge that they claim cannot be readily found in the United States. An individual can work for a U.S. employer as an H-1B employee for a maximum continuous period of six years. The top

countries of birth for H-1B workers in 2015 were India with 70.9 percent of all approved H-1B petitions and China with 9.7 percent.

Each year the U.S. Congress sets an annual cap on the number of H-1B visas to be granted—although the number of visas actually issued often varies greatly from this cap. Since 2004, the cap has been set at 65,000, with an additional 20,000 visas available for foreign graduates of U.S. universities with advanced degrees. The cap only applies to certain IT professionals, such as programmers and engineers at private technology companies. A petition to extend an H-1B nonimmigrant's period of stay, change the conditions of the H-1B nonimmigrant's current employment, or request new H-1B employment for an H-1B worker already in the United States does not count against the H-1B fiscal year cap. In addition, a large number of foreign workers are exempt from the cap, including scientists hired to teach at American universities, work in government research labs, or work for nonprofit organizations.

Foreign professionals can convert their H-1B visas into permanent green cards, which grants them authorization to live and work in the United States on a permanent basis, provided their employers file the necessary paperwork. This process can take a few years to complete, but the federal government allows H-1B professionals to obtain additional one year H-1B visas to remain in their jobs until they get approval for their green card. This process also results in an increase of the total number of H-1B visas above the 85,000 cap.

In 2015, USGIS approved a total of 275,317 H-1B petitions submitted on behalf of alien workers. Of these, 70,902 initial employment H-1B petitions plus an additional 112,174 petitions for continuing employment were approved for workers in computer-related occupations.

Using H-1B Workers Instead of U.S. Workers

In order to compete in the global economy, U.S. firms must be able to attract the best and brightest workers from all over the world. Most H-1B workers are brought to the United States to fill a legitimate gap that cannot be filled from the existing pool of workers. However, there are some managers who reason that as long as skilled foreign workers can be found to fill critical positions, why invest thousands of dollars and months of training to develop the available pool of U.S. workers? Although such logic may appear sound for short term hiring decisions, it does nothing to develop the strong core of permanent IT workers that the United States will need in the future. Heavy reliance on the use of H-1B workers can lessen the incentive for U.S. companies to educate and develop their own workforces.

Companies using H-1B workers, as well as the workers themselves, must also consider what will happen at the end of the six-year H-1B visa term. The stopgap nature of the visa program can be challenging for both sponsoring companies and applicants. If a worker is not granted a green card, the firm can lose a worker without having developed a permanent employee. Many of these foreign workers, finding that they are suddenly unemployed, are forced to uproot their families and return home.

Gaming the H-1B Visa Program

Even though companies applying for H-1B visas must offer a wage that is at least 95 percent of the average salary for the occupation, some companies use H-1B visas as a way to lower salaries. Because wages in the IT field vary substantially, unethical companies can get around the average salary requirement. Determining an appropriate wage is an imprecise science at best. For example, an H-1B worker may be classified as an entry-level IT employee and yet fill a position of an experienced worker who would make $10,000 to $30,000 more per year.

Companies that employ H-1B workers are required to declare that they will not displace American workers. But companies are exempt from that requirement if 15 percent or more of their workers are on H-1B visas and the H-1B workers are paid at least $60,000 a year. Thus, H-1B workers at outsourcing firms often receive wages slightly above $60,000—below what similarly skilled American technology professionals earn, allowing those firms to offer services to U.S. companies at a lower cost, undercutting U.S. workers. The majority of people hired with H-1B visas in 2015 were hired primarily as computer programmers and systems analysts, and were paid a median salary of between $61,131 and $71,150.16 However, the median salary for all U.S. computer programmers was $79,840, and for systems analysts, the median salary was $87,220.

The Need for H-1B Workers

The heads of many U.S. companies complain that they have trouble finding enough qualified IT employees and have urged the USGIS to loosen restrictions on visas for qualified workers. Meanwhile, unemployed and displaced IT workers in the United States, as well as other critics, challenge whether the United States needs to continue importing tens of thousands of H-1B workers every year.

IT firms and many other organizations cite concerns about a shortfall in the number of skilled U.S. workers as justification for hiring H-1B workers, and many tech companies have advocated for an increase in the H-1B hiring cap. However, based on the data from the National Center for Education Statistics, there are some 130,000 new computer and information science graduates each year to fill what the BLS projects as a need of roughly 53,000 new U.S. tech job openings per year. The supply of graduates is substantially larger than the demand. Indeed, in surveys of recent computer science graduates, 32 percent of those graduates not entering the IT workforce say it is because IT jobs are unavailable, while 53 percent say they found better job opportunities outside of IT occupations.

Outsourcing

Outsourcing is another approach for meeting staffing needs. Outsourcing is a long-term business arrangement in which a company contracts for services with an outside organization that has expertise in providing a specific function. A company may contract with an organization to provide services such as operating a data center, supporting a telecommunications network, developing software, or staffing a computer help desk.

Stonewood Financial Tools, a privately held financial services company with headquarters in Louisville, Kentucky, provides investment tools that enable people to make well-informed financial decisions. Recently, Stonewood decided it wanted to develop a custom application that could recommend the best retirement income products for its clients based on a broad set of criteria. With fewer than 50 employees, Stonewood did not have sufficient software development expertise in-house, so it outsourced the work to GlowTouch, an outsourcing firm with 1,200 employees that provides small and mid-sized companies with fast, scalable, responsive IT services. Stonewood and GlowTouch worked together to define and implement an application that meets the needs and provides real-time retirement estimates to Stonewood's clients.

Coemployment legal problems with outsourcing are minimal because the company that contracts for the services does not generally supervise or control the contractor's employees. The primary rationale for outsourcing is to lower costs, but companies also use it to obtain strategic flexibility and to keep their staff focused on the company's core competencies.

Offshore Outsourcing

Offshore outsourcing is a form of outsourcing in which services are provided by an organization whose employees are in a foreign country. Any work done at a relatively high cost in the United States may become a candidate for offshore outsourcing—not just IT work. However, IT professionals in particular can do much of their work anywhere—on a company's premises or thousands of miles away in a foreign country. In addition, companies can reap large financial benefits by reducing labor costs through offshore outsourcing. As a result, and because a large supply of experienced IT professionals is readily available in certain foreign countries, the use of offshore outsourcing in the IT field is common.

Outsourcing is not a recent phenomenon. Accenture, Cap Gemini, Hewlett Packard, and IBM have sent thousands of jobs offshore since the mid-1990s. However, the use of offshore outsourcing has been declining in recent years. While 20 percent of leading tech firms employed offshore outsourcing in 2015, this was down from 27 percent in 2014 and 37 percent in 2013.

Many U.S. software firms set up development centers in low-cost foreign countries where they have access to a large pool of well-trained candidates. Intuit—maker of the Quicken tax preparation software—currently has software development facilities in California and India. Accenture, IBM, and Microsoft all maintain large development centers in India. Cognizant Technology Solutions is headquartered in Teaneck, New Jersey, but operates primarily from technology centers in India.

Because of the high salaries earned by application developers in the United States and the ease with which customers and suppliers can communicate, it is now quite common to use offshore outsourcing for major programming projects. India, with its rich talent pool (a high percentage of whom speak English) and low labor costs, is considered one of the best sources of programming skills outside Europe and North America. Other sources of skilled contract programmers are China, Russia, Poland, Switzerland, and Hungary.

Pros and Cons of Offshore Outsourcing

Wages that an American worker might consider low represent an excellent salary in many other parts of the world, and some companies feel they would be foolish not to exploit such an opportunity. Why pay a U.S. IT worker a six-figure salary, they reason, when they can use offshore outsourcing to hire three India-based workers for the same cost? However, this attitude might represent a short-term point of view—offshore demand is driving up salaries in India by roughly 15 percent per year. Because of this, Indian offshore suppliers have begun to charge more for their services. The cost advantage for offshore outsourcing to India used to be 6:1 or more—you could hire six Indian IT workers for the cost of one U.S. IT worker. The cost advantage is shrinking, and once it reaches about 1.5:1, the cost savings will no longer be much of an incentive for U.S. offshore outsourcing to India.

Another benefit of offshore outsourcing is its potential to dramatically speed up software development efforts. For example, FirstBest Systems, a leading provider of insurance software solutions and services, contracted the integration and implementation of an underwriting management system to Syntel, one of the first U.S. firms to successfully launch a global delivery model that enables workers to work on a project around the clock. With technical teams working from networked facilities in different time zones, Syntel executes a virtual "24-hour workday" that saves its customers money, speeds projects to completion, and provides continuous support for key software applications.

While offshore outsourcing can save a company in terms of labor costs, it will also result in other expenses. In determining how much money and time a company will save with offshore outsourcing, the firm must take into account the additional time that will be required to select an offshore vendor as well as the

additional costs that will be incurred for travel and communications. In addition, organizations often find it takes years of ongoing effort and a large up-front investment to develop a good working relationship with an offshore outsourcing firm. Finding a reputable vendor can be especially difficult for a small or midsized firm that lacks experience in identifying and vetting contractors.

Many of the ethical issues that arise when considering whether to use H-1B and contingent workers also apply to outsourcing and offshore outsourcing. For example, managers must consider the trade-offs between using offshore outsourcing firms and devoting money and time to retain and develop their own staff. Often, companies that begin offshoring also lay off portions of their own staff as part of that move.

Cultural and language differences can cause misunderstandings among project members in different countries. For example, in some cultures, shaking one's head up and down simply means "Yes, I understand what you are saying." It does not necessarily mean "Yes, I agree with what you are saying." And the difficulty of communicating directly with people over long distances can make offshore outsourcing perilous, especially when key team members speak English as their second language.

The compromising of customer data is yet another potential outsourcing issue. In a study to understand the risks associated with services outsourcing, researchers at Arizona State University analyzed 146 customer data breaches between 2005 and 2010. Of those, 25 were breaches for which the outsourced service provider was responsible. Clearly, organizations that outsource must take precautions to protect private data, regardless of where it is stored or processed.

Another downside to offshore outsourcing is that a company loses the knowledge and experience gained by outsourced workers when those workers are reassigned after a project's completion. Finally, offshore outsourcing does not advance the development of permanent IT workers in the United States, which increases its dependency on foreign workers to build the IT infrastructure of the future. Many of the jobs that go overseas are entry-level positions that help develop employees for future, more responsible positions.

Strategies for Successful Offshore

Outsourcing Successful projects require day-to-day interaction between software development and business teams, so it is essential for the hiring company to take a hands-on approach to project management. Companies cannot afford to outsource responsibility and accountability.

To improve the chances that an offshore outsourcing project will succeed, a company must carefully evaluate whether an outsourcing firm can provide the following:

- Employees with the required expertise in the technologies involved in the project
- A project manager who speaks the employer company's native language
- A pool of staff large enough to meet the needs of the project
- A reliable, state-of-the-art communications network
- High-quality, on-site managers and supervisors

Whistle-Blowing

Whistle-blowing is an effort to attract public attention to a negligent, illegal, unethical, abusive, or dangerous act by an individual or organization. In some cases, whistle-blowers are employees who act as informants on their company, revealing information to enrich themselves or to gain revenge for a perceived wrong. In most cases, however, whistle-blowers act ethically in an attempt to correct what they think is a major wrongdoing, often at great personal risk.

A whistle-blower usually has personal knowledge of what is happening inside the offending organization because of his or her role as an employee of the organization. Sometimes the whistle-blower is not an employee but a person with special knowledge gained from a position as an auditor or business partner.

In going public with the information they have, whistle-blowers often risk their own careers and sometimes even affect the lives of their friends and family. In extreme situations, whistle-blowers must choose between protecting society and remaining silent.

A senior finance manager at Oracle alleged she was fired for refusing to follow what she thought were unlawful accounting practices designed to boost sales revenue figures for the IT company's software-as-a-service and cloud computing services. In a whistle-blower lawsuit filed against the company, the former Oracle employee claimed she was instructed to add millions of dollars of accruals for anticipated business that had not yet materialized. Oracle stock dropped almost 4 percent the day following the filing of the lawsuit. Oracle countered that the employee had worked there for under a year and was fired because she failed to fulfill the responsibilities of her role. Furthermore, Oracle claimed that she had never raised any allegations about unlawful accounting practices internally. The lawsuit between the employee and Oracle was settled in February 2017 but no details were released.

Protection for Whistle-Blowers

Whistle-blower protection laws allow employees to alert the proper authorities to employer actions that are unethical, illegal, or unsafe, or that violate specific public policies. Unfortunately, no comprehensive federal law protects all whistle-blowers from retaliatory acts. Instead, numerous laws protect a certain class of specific whistle-blowing acts in various industries. To make things even more complicated, each law has different filing provisions, administrative and judicial remedies, and statutes of limitations (which set time limits for legal action). Thus, the first step in reviewing a whistle-blower's claim of retaliation is for an experienced attorney to analyze the various laws and determine if and how the employee is protected. Once that is known, the attorney can determine what procedures to follow in filing a claim.

Dealing with a Whistle-Blowing Situation

Each potential whistle-blowing case involves different circumstances, issues, and personalities. Two people working together in the same company may have different values and concerns that cause them to react in different ways to a particular situation—and both reactions might be ethical. It is impossible to outline a definitive step-by-step procedure of how to behave in a whistle-blowing situation.

This section provides a general sequence of events and highlights key issues that a potential whistle-blower should consider. Anyone considering becoming a whistle-blower is strongly advised to seek legal counsel.

<u>Assess the Seriousness of the Situation</u>

Before considering whistle-blowing, a person should have specific knowledge that his or her company or a coworker is acting unethically and that the action represents a serious threat to the public interest. The employee should carefully and informally seek trusted resources outside the company and ask for their assessment. Do they also see the situation as serious? Their point of view may help the employee see the situation from a different perspective and alleviate concerns. On the other hand, the outside resources may reinforce the employee's initial suspicions, forcing a series of difficult ethical decisions.

Begin Documentation

An employee who identifies an illegal or unethical practice should begin to compile adequate documentation to establish wrongdoing. The documentation should record all events and facts as well as the employee's insights about the situation. This record helps construct a chronology of events if legal testimony is required in the future. An employee should identify and copy all supporting memos, correspondence, manuals, and other documents before taking the next step. Otherwise, records may disappear and become inaccessible. The employee should maintain documentation and keep it up to date throughout the process.

Attempt to Address the Situation Internally

An employee should next attempt to address the problem internally by providing a written summary to the appropriate managers. Ideally, the employee can expose the problem and deal with it from inside the organization. The focus should be on disclosing the facts and how the situation affects others. The employee's goal should be to fix the problem, not to place blame. Given the potential negative impact of whistle-blowing on the employee's future, this step should not be dismissed or taken lightly.

Fortunately, many problems are solved at this point, and further, more drastic actions by the employee are unnecessary. The appropriate managers get involved and resolve the issue that initiated the whistle-blower's action.

On the other hand, managers who are engaged in unethical or illegal behavior might not welcome an employee's questions or concerns. In such cases, the whistle-blower can expect to be strongly discouraged from taking further action. Employee demotion or termination on false or exaggerated claims can occur. Attempts at discrediting the employee can also be expected. As an extreme example, Dr. Jeffrey Wigand, former vice president of research and development at Brown & Williamson, disclosed wrongdoings involving the use of cancer causing ingredients in the tobacco industry. As a result, he received several anonymous death threats; however, none of the threats could be traced back to its source.

Consider Escalating the Situation within the Company

The employee's initial attempt to deal with a situation internally may be unsuccessful. At this point, the employee may rationalize that he or she has done all that is required by raising the issue. Others may feel so strongly about the situation that they are compelled to take further action. Thus, a determined and conscientious employee may feel forced to choose between escalating the problem and going over the manager's head, or going outside the organization to deal with the problem. The employee may feel obligated to sound the alarm on the company because there appears to be no chance to solve the problem internally.

Going over an immediate manager's head can put one's career in jeopardy. Supervisors may retaliate against a challenge to their management, although some organizations may have an effective corporate ethics officer who can be trusted to give the employee a fair and objective hearing. Alternatively, a senior manager with a reputation for fairness and some responsibility for the area of concern might step in. However, in many work environments, the challenger may be fired, demoted, or reassigned to a less desirable position or job location. Such actions send a loud signal throughout an organization that loyalty is highly valued and that challengers will be dealt with harshly. Whether reprisal is ethical depends in large part on the legitimacy of the employee's issue. If the employee is truly overreacting to a minor issue, then the employee may deserve some sort of reprimand for exercising poor judgment.

If senior managers refuse to deal with a legitimate problem, the employee can decide to drop the matter or go outside the organization to try to remedy the situation. Even if a senior manager agrees with the employee's position and overrules the employee's immediate supervisor, the employee may want to request a transfer to avoid working for the same person.

Assess the Implications of Becoming a Whistle-Blower

If whistle-blowers feel they have made a strong attempt to resolve the problem internally without results, they must stop and fully assess whether they are prepared to go forward and blow the whistle on the company. Depending on the situation, an employee may incur significant legal fees in order to bring charges against an agency or company that may have access to an array of legal resources as well as a lot more money than the individual employee. An employee who chooses to proceed might be accused of having a grievance with the employer or of trying to profit from the accusations. The employee may be fired and may lose the confidence of coworkers, friends, and even family members. A potential whistle-blower must attempt to answer many ethical questions before making a decision on how to proceed:

- Given the potentially high price, do I really want to proceed?
- Have I exhausted all means of dealing with the problem? Is whistle-blowing all that is left?
- Am I violating an obligation to be loyal to my employer and work for its best interests?
- Will the public exposure of corruption and mismanagement in the organization really correct the underlying cause of these problems and protect others from harm?

From the moment an employee becomes known as a whistle-blower, a public battle may ensue. Whistle-blowers can expect attacks on their personal integrity and character as well as negative publicity in the media. Friends and family members will hear these accusations, and ideally, they should be notified beforehand and consulted for advice before the whistle-blower goes public. This notification helps prevent friends and family members from being surprised at future actions by the whistle-blower or the employer.

The whistle-blower should also consider consulting support groups, elected officials, and professional organizations. For example, the National Whistleblowers Center provides referrals for legal counseling and education about the rights of whistle-blowers.

Use Experienced Resources to Develop an Action Plan

A whistle-blower should consult with competent legal counsel who has experience in whistleblowing cases. He or she will determine which statutes and laws apply, depending on the agency, the employer, the state involved, and on the nature of the case. Counsel should also know the statute of limitations for reporting the offense, as well as the whistle-blower's protection under the law. Before blowing the whistle publicly, the employee should get an honest assessment of the soundness of his or her legal position and an estimate of the costs of a lawsuit.

Execute the Action Plan

A whistle-blower who chooses to pursue a matter legally should do so based on the research and guidance of legal counsel. If the whistle-blower wants to remain unknown, the safest course of action is to leak information anonymously to the press. The problem with this approach, however, is that anonymous claims are often not taken seriously. In most cases, working directly with appropriate regulatory agencies and legal authorities is more likely to get results, including the imposition of fines, the halting of operations, or other actions that draw the offending organization's immediate attention.

Live with the Consequences

Whistle-blowers must be on guard against retaliation, such as being discredited by coworkers, threatened, or set up; for example, management may attempt to have the whistle-blower transferred, demoted, or fired for breaking some minor rule, such as arriving late to work or leaving early. To justify their actions, management may argue that such behavior has been ongoing. The whistle-blower might need a good strategy and a good attorney to counteract such actions and take recourse under the law.

Green Computing

Electronic devices such as computers and smartphones contain hundreds—or even thousands—of components, which are, in turn, composed of many different materials, including some (such as beryllium, cadmium, lead, mercury, brominated flame retardants (BFRs), selenium, and polyvinyl chloride) that are known to be potentially harmful to humans and the environment. Electronics manufacturing employees and suppliers at all steps along the supply chain and manufacturing process are at risk of unhealthy exposure to these raw materials. Users of these products can also be exposed to these materials when using poorly designed or improperly manufactured devices. Care must also be taken when recycling or destroying these devices to avoid contaminating the environment.

Green computing is concerned with the efficient and environmentally responsible design, manufacture, operation, and disposal of IT-related products, including all types of computing devices (from smartphones to supercomputers), printers, printer materials such as cartridges and toner, and storage devices.

Green computing has three goals:

(1) reduce the use of hazardous material,
(2) allow companies to lower their power-related costs, and
(3) enable the safe disposal or recycling of computers and computer-related equipment.

Many business organizations recognize that going green is in their best interests in terms of public relations, employee safety, and the community at large. These organizations also recognize that green computing presents an opportunity to substantially reduce total costs over the life cycle of their IT equipment.

It is estimated that in the United States, 51.9 million computers, 35.8 million monitors, and 33.6 million hard copy devices (printers, faxes, etc.)—representing a total of 1.3 million tons of waste—were disposed of in 2010 alone. E-waste is the fastest growing municipal waste stream in the United States. Because it is impossible for manufacturers to ensure safe recycling or disposal, the best practice would be for them to eliminate the use of toxic substances, particularly since recycling of used computers, monitors, and printers has raised concerns about toxicity and carcinogenicity of some of the substances. However, until manufacturers stop using these toxic substances, safe disposal and reclamation operations must be carried out carefully to avoid exposure in recycling operations and leaching of materials, such as heavy metals, from landfills and incinerator ashes. In many cases, recycling companies export large quantities of used electronics to companies in undeveloped countries. Unfortunately, many of these countries do not have strong environmental laws, and they sometimes fail to recognize the potential dangers of dealing with hazardous materials. In their defense, these countries point out that the United States and other first-world countries were allowed to develop robust economies and rise up out of poverty without the restrictions of strict environmental policies.

Electronic Product Environmental Assessment Tool (EPEAT) is a system that enables purchasers to evaluate, compare, and select electronic products based on a total of 51 environmental criteria. Products are ranked in EPEAT according to three tiers of environmental performance: Bronze (meets all 23 required criteria), Silver (meets all 23 of the required criteria plus at least 50 percent of the optional criteria), and Gold (meets all 23 required criteria plus at least 75 percent of the optional criteria). EPEAT was first implemented in 2006 with Computer and Displays (IEEE 1680.1 standard) and has now expanded to Imaging Equipment, under the IEEE 1680.2 standard from January 2013. EPEAT is managed by the Green Electronics Council and currently evaluates more than 4,400 products from more than 60 manufacturers across 43 countries.

The European Union's Restriction of Hazardous Substances Directive, which took effect in 2006, restricts the use of many hazardous materials in computer manufacturing. The directive also requires manufacturers to use at least 65 percent reusable or recyclable components, implement a plan to manage products at the end of their life cycle in an environmentally safe manner, and reduce or eliminate toxic material in their packaging.

The state of California has passed a similar law, called the Electronic Waste Recycling Act. Because of these acts, manufacturers had a strong motivation to remove brominated flame retardants from their PC casings. By the start of 2010, the Apple iPad was free of arsenic, mercury, PVC (polyvinyl chloride), and BFRs. In addition, according to Apple, the iPad's aluminum and glass enclosure is "highly recyclable."

Lenovo is a Chinese manufacturer of personal computers, tablets, smartphones, workstations, servers, electronic storage devices, and printers. Since 2007, the company's product development teams have been using increasing amounts of recycled plastics to meet new customer requirements, satisfy corporate environmental objectives and targets, and achieve EPEAT Gold registrations for its products. The company's efforts have resulted in the avoidance of up to 248 million pounds of CO_2 emissions since 2007.

What key legal and ethical issues are associated with the use of contingent workers, H-1B visa holders, and offshore outsourcing companies?

- Contingent work is a job situation in which an individual does not have an explicit or implicit contract for long-term employment.
- Organizations can obtain contingent workers through temporary staffing firms, employee leasing organizations, and professional employment organizations.
- Temporary staffing firms recruit, train, and test job seekers in a wide range of job categories and skill levels, and then assign them to clients as needed.
- In employee leasing, the subscribing firm transfers all or part of its workforce to the leasing firm, which handles all human-resource-related activities and costs such as payroll, training, and the administration of employee benefits. The subscribing firm leases these workers, but they remain employees of the leasing firm.
- A coemployment relationship is one in which two employers have actual or potential legal rights and duties with respect to the same employee or group of employees.
- A PEO is a business entity that hires the employees of its clients and then assumes all responsibility for all human resource management functions, including administration of benefits. The client company remains responsible for directing and controlling the daily activities of the employees. The client maintains a long-term investment and commitment to the employees, but uses the PEO as a means to outsource the human resource activities.
- The gig economy refers to a work environment in which temporary positions are common and organizations contract with independent workers for short-term engagements.

- An independent contractor is an individual who provides services to another individual or organization according to terms defined in a written contract or within a verbal agreement.
- Organizations that use contingent workers must be extremely careful how they pay and treat these workers, or run the risk of getting dragged into a class action lawsuit over misclassification of workers.
- When a firm employs a contingent worker, it does not usually have to provide benefits, can easily adjust the number of workers to meet its business needs, and does not incur training costs.
- Contingent workers may have a low level of commitment to the company and its projects. The skills and knowledge a contingent worker gains while working for a particular are lost when the worker departs at a project's completion.
- An H-1B is a temporary work visa granted by the U.S. Citizenship and Immigration Services (USCIS) for people who work in specialty occupations—jobs that require at least a four year bachelor's degree in a specific field, or equivalent experience.
- Employers hire H-1B workers to meet critical business needs or to obtain essential technical skills or knowledge that cannot be readily found in the United States. H-1B workers may also be used when there are temporary shortages of needed skills.
- Congress sets an annual cap on the number of H-1B visas to be granted—although the number of visas issued often varies greatly from this cap due to various exceptions.
- Companies applying for H-1B visas must offer a wage that is at least 95 percent of the average salary for the occupation. Because wages in the IT field vary substantially, unethical companies can get around the average salary requirement.
- Companies that employ H-1B workers are required to declare that they will not displace American workers; however, they are exempt from that requirement if 15 percent of more of their workers are on H-1B visas and the H-1B workers are paid at least $60,000 a year.
- Many U.S. companies complain they have trouble finding enough qualified workers and urge that the cap on visas be raised. Unemployed and displaced IT workers challenge whether the United States needs to continue importing tens of thousands of H-1B workers each year.
- The number of degrees awarded in the field of computer and information sciences at postsecondary institutions in the United States reached 130,000 in 2015. The Bureau of Labor Statistics has projected an increase of 53,000 new U.S. tech jobs per year from 2014 to 2024.
- Opinions vary as to whether or not the hiring of H-1B workers affects job opportunities and wages.
- Outsourcing is a long-term business arrangement in which a company contracts for services with an outside organization that has expertise in providing a specific function.
- Offshore outsourcing is a form of outsourcing in which the services are provided by an organization whose employees are in a foreign country.
- Outsourcing and offshore outsourcing are used to meet staffing needs while potentially reducing costs and speeding up project schedules.
- Many of the same ethical issues that arise when considering whether to hire H-1B and contingent workers apply to outsourcing and offshore outsourcing.
- Successful offshoring projects require day-to-day interaction between software development and business teams, so it is essential for the hiring company to take a hands-on approach to project management.

What is whistle-blowing, and what ethical issues are associated with it?

- Whistle-blowing is an effort to attract public attention to a negligent, illegal, unethical, abusive, or dangerous act by a company or some other organization.
- Whistle-blower protection laws allow employees to alert the proper authorities to employer actions that are unethical, illegal, or unsafe, or that violate specific public policies. Unfortunately, no comprehensive federal law protects all whistle-blowers from retaliatory acts.
- A potential whistle-blower must consider many ethical implications prior to going public with his or her allegations, including whether the high price of whistle-blowing is worth it; whether all other means of dealing with the problem have been exhausted; whether whistleblowing violates the obligation of loyalty that the employee owes to his or her employer; and whether public exposure of the problem will actually correct its underlying cause and protect others from harm.
- An effective whistle-blowing process includes the following steps:
  (1) assess the seriousness of the situation,
  (2) begin documentation,
  (3) attempt to address the situation internally,
  (4) consider escalating the situation within the company,
  (5) assess the implications of becoming a whistle-blower,
  (6) use experienced resources to develop an action plan,
  (7) execute the action plan, and
  (8) live with the consequences.

What is green computing, and what are organizations doing to support this initiative?

- Green computing is concerned with the efficient and environmentally responsible design, manufacture, operation, and disposal of IT-related products.
- Green computing has three goals:
  (1) reduce the use of hazardous material,
  (2) allow companies to lower their power-related costs, and
  (3) enable the safe disposal or recycling of computers and computer-related equipment.
- Electronic Product Environmental Assessment Tool (EPEAT) is a system that enables purchasers to evaluate, compare, and select electronic products based on 51 environmental criteria.
- The European Union passed the Restriction of Hazardous Substances Directive to restrict the use of many hazardous materials in computer manufacturing, require manufacturers to use at least 65 percent reusable or recyclable components, implement a plan to manage products at the end of their life cycle in an environmentally safe manner, and reduce or eliminate toxic material in their packaging.