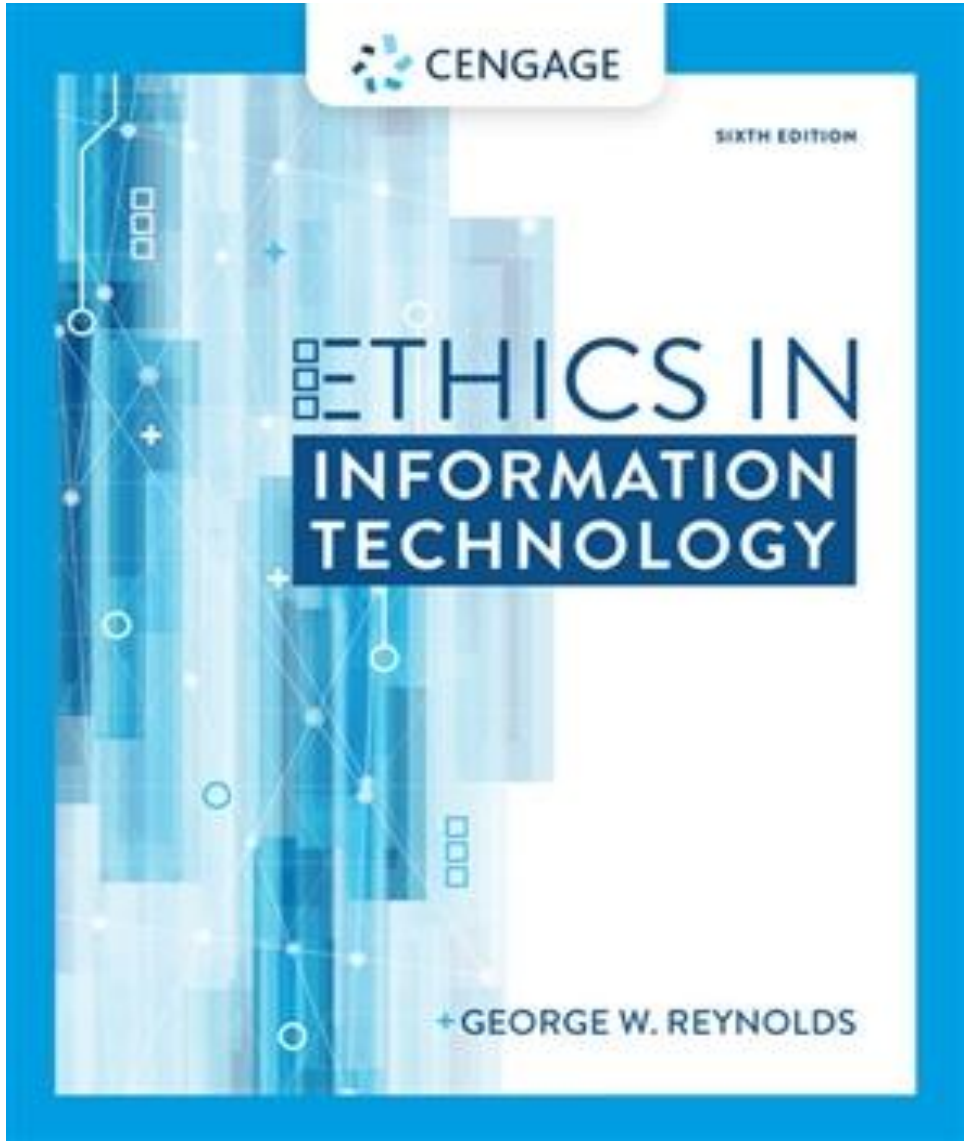
CROWN INSTITUTE

FROM POSSIBILITY TO ACTUALITY

Professional Practice in Information Technology ICT945

Prescribed Text

Reynolds, G. (2018), *Ethics in Information Technology*, 6th Edition, Cengage Learning, Boston, MA



Chapter 7

Ethical Decisions in Software Development

Computers are magnificent tools for the realisation of our dreams, but no machine can replace the human spark of spirit, compassion, love, and understanding.

- Louis V. Gerstner, Jr., former CEO and Chairman of board of IBM

Learning Objectives

- 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?
- What are some effective strategies for developing quality systems?

Software Quality

- **High-quality software systems:** Easy to learn and use; perform quickly and efficiently, meet users' needs; and operate safely and reliably, with minimal system downtime
- **Software defect:** Any error that, if not removed, could cause a software system to fail to meet its users' needs
- **Software quality:** The degree to which a software product meets the needs of its users
- **Quality management:** Defining, measuring, and refining the quality of products and the development process
- **Deliverables**: Products of the development process, such as statements of requirements and flowcharts

Causes of Poor Quality Software

- Many developers do not know how to design quality into software; others do not take the time to do so.
  - Developers must define and **follow** a set of **software engineering principles** and be committed to **learning from mistakes**.
  - Developers must **understand the environment** in which their systems will operate and design systems immune to human error.
- Extreme pressure that software companies feel to reduce the time to market for their products
  - Resources and time needed to ensure quality are often cut under the pressure to ship a new product

The Importance of Software Quality

- **Business information system:** A set of interrelated components that collects and processes data and disseminates the output
  - Captures and records business transactions
  - Controls industrial processes and the operation of industrial and consumer products
  - **Decision support system (DSS):** Used to improve decision making
- Mismanaged software can cause missed product deadlines, increased product development costs, and delivery of low-quality products.

Software Product Liability, Part 1

- Use of software introduces product liability issues

- **Product liability:** The liability of manufacturers, sellers, lessors, and others for injuries caused by defective products
  - Claims are based on: **strict liability**, **negligence**, **breach of warranty**, or **misrepresentation**.

Software Product Liability, Part 2

- **Strict liability:** The defendant is held responsible for injury, regardless of negligence or intent.
  - Plaintiff must prove **only** that the software product is defective or unreasonably dangerous and that the defect caused the injury.
- Legal defenses used against strict liability
  - Doctrine of supervening event
    - Original seller is not liable if the software was materially altered after it left the sellers possession, and the alteration caused the injury.
  - Government contractor defense
    - The precise software specifications were provided by the Government and the software conformed to the specs., and the contractor had warned the Govt. of any known defects.
  - Expired statute of limitations
    - Claims must be made within certain time after injury occurs.

Software Product Liability, Part 3

- **Negligence**
  - Failure to do what a reasonable person would do, or doing something that a reasonable person would not do

- **Defense**:
  - **Contributory negligence:** Plaintiffs' own actions contributes to their injuries

Software Product Liability, Part 4

- **Warranty:** Assures buyers or lessees that a product meets certain standards of quality
- **Breach of warranty:** Buyer or lessee can sue the seller or lessor if the product fails to meet the terms of its warranty
  - Difficult to prove because the software supplier writes the warranty to limit liability

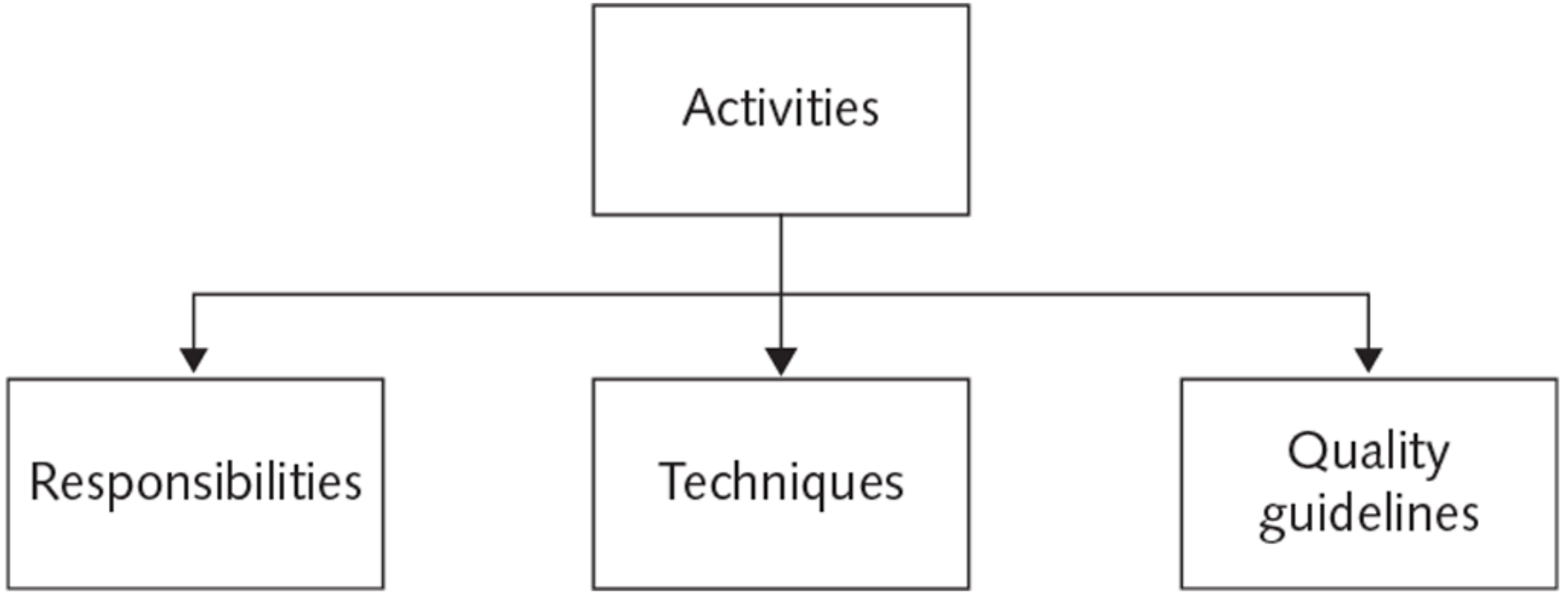
Strategies and Tools for Developing Quality Software

- Software development methodologies
- Capability Maturity Model Integration (CMMI) process-improvement model
- Special techniques for safety-critical systems
- Risk management processes
- Quality management standards

Software Development Methodologies

- **Software development methodology:** Standard, proven work process that enables controlled progress while developing high-quality software
  - Use of an effective methodology can protect software manufacturers from legal liability
    - Reduces the number of software errors
    - Makes negligence harder to prove

Components of a Software Development Methodology



Waterfall System Development Model

- **Waterfall system development model**: A sequential, multistage system development process in which development of the next stage cannot begin until the results of the current stage are approved
  - Stages of waterfall development:
    - Investigation
    - Analysis
    - Design
    - Construction
    - Integration and testing
    - Implementation

Agile Development Methodology

- **Agile development:** A methodology under which a system is developed in **iterations** (often called **sprints**) lasting from one to four weeks
  - Accepts the fact that system requirements are evolving and cannot be fully understood or defined at the start of the project.
- In an agile project, the team evaluates the system every one to four weeks, giving it ample opportunity to identify and implement new requirements.

Pros and Cons of Waterfall

Pros	**Cons**
Formal review at end of each stage allows maximum management control.	Users' needs may go unstated or be miscommunicated. 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.
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.

Pros and Cons of Agile

Pros	**Cons**
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.
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.

The Cost of Removing Defects

- The cost to remove a defect in an early stage of software development can be up to 100 times less than removing a defect in software that has been distributed to customers.
- A product containing inherent defects that harm the user may be the subject of a product liability suit.
- **Quality assurance (QA):** Methods within the development process designed to guarantee reliable operation of a product
  - Should be applied at each stage of development

Software Testing, Part 1

- **Dynamic testing:** Entering test data and comparing the results with the expected results in a process
  - **Black-box testing:** Views the software unit as a device that has expected input and output behaviors, but whose internal workings are unknown
  - **White-box testing**: Treats the software unit as a device that has expected input and output behaviors, and whose internal workings are known

Software Testing, Part 2

- **Static testing:** Software-testing technique in which software is tested without actually executing the code
- **Unit testing:** Individual components of code (subroutines, modules, and programs) are tested
- **Integration testing:** Software units are combined into an integrated subsystem and tested to ensure the linkages among the various subsystems work
- **System testing:** Various subsystems are combined to test the entire system
- **User acceptance testing:** Trained end users conduct independent testing

Capability Maturity Model Integration (CMMI)

- **CMMI models:** Collections of best practices that help organizations improve processes
  - **Best practice:** A method that has consistently shown results superior to those achieved with other means
  - **CMMI-Development (CMMI-DEV):** Frequently used to assess and improve software development processes
    - Defines five levels of software development maturity:
      - Initial
      - Managed
      - Defined
      - Quantitatively managed
      - Optimizing

Developing Safety Critical Systems, Part 1

- **Safety-critical system:** A system whose failure may cause injury or death
  - Safe operation relies on the flawless performance of software
- Key assumption when developing safety-critical systems:
  - Safety will not automatically result from following the organization's standard development methodology
    - All development tasks require:
      - Additional steps
      - More thorough documentation
      - Vigilant checking and rechecking

Developing Safety Critical Systems, Part 2

- **System safety engineer:** Has responsibility for system's safety; uses a logging and monitoring system to track hazards from a project's start to its finish
- *When designing, building, and operating a safety-critical system, a formal risk analysis must be conducted*

- **Reliability:** A measure of the rate of failure in a system that would render it unusable over its expected lifetime
- **System-human interface:** One of the most important and difficult areas of safety-critical system design
  - Design of the system should not allow for erroneous judgment on the part of the operator

Risk

- **Risk:** The potential of gaining or losing something of value
  - Quantified by three elements:
    - A risk event
    - The probability of the event happening
    - The impact (positive or negative) on the business outcome

Annualized Loss Expectancy

- **Annualized rate of occurrence (ARO):** An estimate of the probability that this event will occur over the course of a year
- **Single loss expectancy (SLE):** The estimated loss that would be incurred if the event happens
- **Annualized loss expectancy:** The estimated loss from this risk over the course of a year
- ARO × SLE = ALE

Risk Management

- **Risk management:** 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**.
  - Strategies for addressing a particular risk:
    - Acceptance
    - Avoidance
    - Mitigation
    - Redundancy
    - Transference

Quality Management Standards

- **ISO 9001 family of standards:** A set of standards that serves as a guide to quality products, services, and management; updated every five years
  - To become ISO 9001 certified, an organization must submit to an examination by an external assessor
- **Failure mode and effects analysis (FMEA):** A technique used to develop ISO 9001-compliant quality systems
  - Evaluates reliability and determines the effects of system and equipment failures
  - **Failure mode:** Describes how a product or process could fail to perform the functions described by the customer

Manager's Checklist For Improving Software Quality, Part 1

QUESTION	YES	NO
• Has senior management made a commitment to develop quality software? • Have you used CMMI to evaluate your organization's software development process? • Has your company adopted a standard software development methodology? • Does the methodology place a heavy emphasis on quality management and address how to define, measure, and refine the quality of the software development process and its products? • Are software project managers and team members trained in the use of this methodology? • Are software project managers and team members held accountable for following this methodology?		

Manager's Checklist For Improving Software Quality, Part 2

QUESTION	YES	NO
• Is a strong effort made to identify and remove errors as early as possible in the software development process? • Are both static and dynamic software testing methods used? • Are white-box testing and black-box testing methods used? • Has an honest assessment been made to determine if the software being developed is safety critical? • If the software is safety critical, are additional tools and methods employed, and do they include the following: a project safety engineer, hazard logs, safety reviews, formal configuration management systems, rigorous documentation, risk analysis processes, and the FMEA technique?		

Summary, Part 1

- **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 systems: Easy to learn and use; perform quickly and efficiently; and operate safely and reliably, with minimal downtime
  - Software defect: Any error that, if not removed, could cause a software system to fail to meet users' needs
  - Software quality: The degree to which a software product meets the needs of its users

Summary, Part 2

- **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?**
  - Quality management: Defining, measuring, and refining the quality of products and the development process
  - Resources and time needed to develop high-quality software are often cut under pressure to ship new product
  - High-quality software: Required in fields such as air traffic control, automobile safety, and health care

Summary, Part 3

- **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?**
  - A business information system: A set of interrelated components that collects and processes data and disseminates the output.
  - Software product liability claims are based on strict liability, negligence, breach of warranty, or misrepresentation.
    - Strict liability: The defendant is held responsible for injury regardless of negligence or intent

Summary, Part 4

- **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?**
  - Warranty: Assures buyers or lessees that a product meets certain standards of quality; it 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.

Summary, Part 5

- **What are some effective strategies for developing quality systems?**
  - Software development methodology: A standard, proven, work process that enables a team to make controlled and orderly progress in developing high-quality software
    - Defines activities and responsibilities; recommends specific techniques; and offers guidelines for managing the quality of the products during the various development stages

Summary, Part 6

- **What are some effective strategies for developing quality systems?**
  - Waterfall system development model: A sequential, multistage system process in which development of the next stage cannot begin until the results of the current stage are approved
  - Agile development methodology: A development process in which a system is developed in iterations (aka sprints), lasting from one to four weeks
    - Accepts that system requirements are evolving and cannot be fully understood at the start of the project

Summary, Part 7

- **What are some effective strategies for developing quality systems?**
  - An effective development methodology:
    - Enables a manufacturer to produce high-quality software, forecast project-completion milestones, and reduce development and support costs
    - Helps protect software manufacturers from legal liability for defective software by:
      - Reducing the number of software errors that could cause damage
      - Making negligence more difficult to prove

Summary, Part 8

- **What are some effective strategies for developing quality systems?**
  - Cost to remove a defect early in the development process can be up to 100 times less than removing a defect in software that has been distributed to customers
  - Quality assurance (QA): Methods within the development process designed to guarantee reliable operation of a product
    - Should be applied at each stage of the development cycle
  - Types of software testing:
    - Black-box and white-box dynamic testing, static testing, unit testing, integration testing, system testing, and user acceptance testing

Summary, Part 9

- **What are some effective strategies for developing quality systems?**
  - Capability Maturity Model Integration (CMMI) models: Collections of best practices that help organizations improve their processes
    - Best practice: A method or technique that has consistently shown results superior to those achieved with other means
  - CMMI-Development (CMMI-DEV): Used to assess and improve software development practices
    - Defines five levels of software development maturity: initial, managed, defined, quantitatively managed, and optimizing

Summary, Part 10

- **What are some effective strategies for developing quality systems?**
  - Safety-critical system: One whose failure may cause human injury or death
    - Key assumption: Safety will not automatically result from following an organization's standard software development methodology.
    - Must go through much more rigorous development and testing
    - Project safety engineer, a hazard log, and risk analysis
  - Risk: The potential of gaining or losing something of value
    - Quantified by three elements: a risk event, the probability of the event happening, and the impact if the risk occurs

Summary, Part 11

- **What are some effective strategies for developing quality systems?**
  - Annualized rate of occurrence (ARO): An estimate of the probability that an event will occur over the course of a year
  - Single loss expectancy (SLE): The estimated loss that would be incurred if the event happens
  - Annualized loss expectancy (ALE): The estimated loss from this risk over the course of a year
  - ARO × SLE = ALE

Summary, Part 12

- **What are some effective strategies for developing quality systems?**
  - ISO standards: Require organizations to develop formal quality management systems focused on identifying and meeting the needs of customers
    - The ISO 9001 standards serve as a guide to quality products, services, and management
    - Many businesses and government agencies specify that vendors must be ISO 9001 certified
  - Failure mode and effects analysis (FMEA): A technique used to develop ISO 9001–compliant quality systems
    - Evaluates reliability and determines the effects of system and equipment failures