BITS Pilani
Pilani Campus

BITS Pilani presentation

Dr. Nagesh BS

innovate achieve lead

BITS Pilani
Pilani Campus

SE ZG501
Software Quality Assurance and Testing
Lecture No. 1

Software Quality Assurance

SQA Addresses the global challenge of the improvement of software quality.

It seeks to provide an overview of software quality assurance (SQA) practices for customers, managers, auditors, suppliers, and personnel responsible for software projects, development, maintenance, and software services.

In a globally competitive environment, clients and competitors exert a great deal of pressure on organizations.

Clients are increasingly demanding and require, among other things, software that is of high quality, low cost, delivered quickly, and with impeccable after-sales support.

To meet the demand, quality, and deadlines, the organization must use efficient quality assurance practices for their software activities.

Standards define ways to maximize performance but managers and employees are largely left to themselves to decide how to practically improve the situation.

They face several problems:

- **increasing pressure to deliver quality products quickly**
- **increasing size and complexity of software and of systems**
- **increasing requirements to meet national, international, and professional standards**
- **subcontracting and outsourcing**
- **distributed work teams**
- **ever changing platforms and technologies.**

INTRODUCTION

Software is developed, maintained, and used by people in a wide variety of situations.

Students ,enthusiasts , professionals address quality problems that arise in the software they are working with

The Guide to the Software Engineering Body of Knowledge (SWEBOK) [SWE 14] constitutes the first international consensus developed on the fundamental knowledge required by all software engineers.

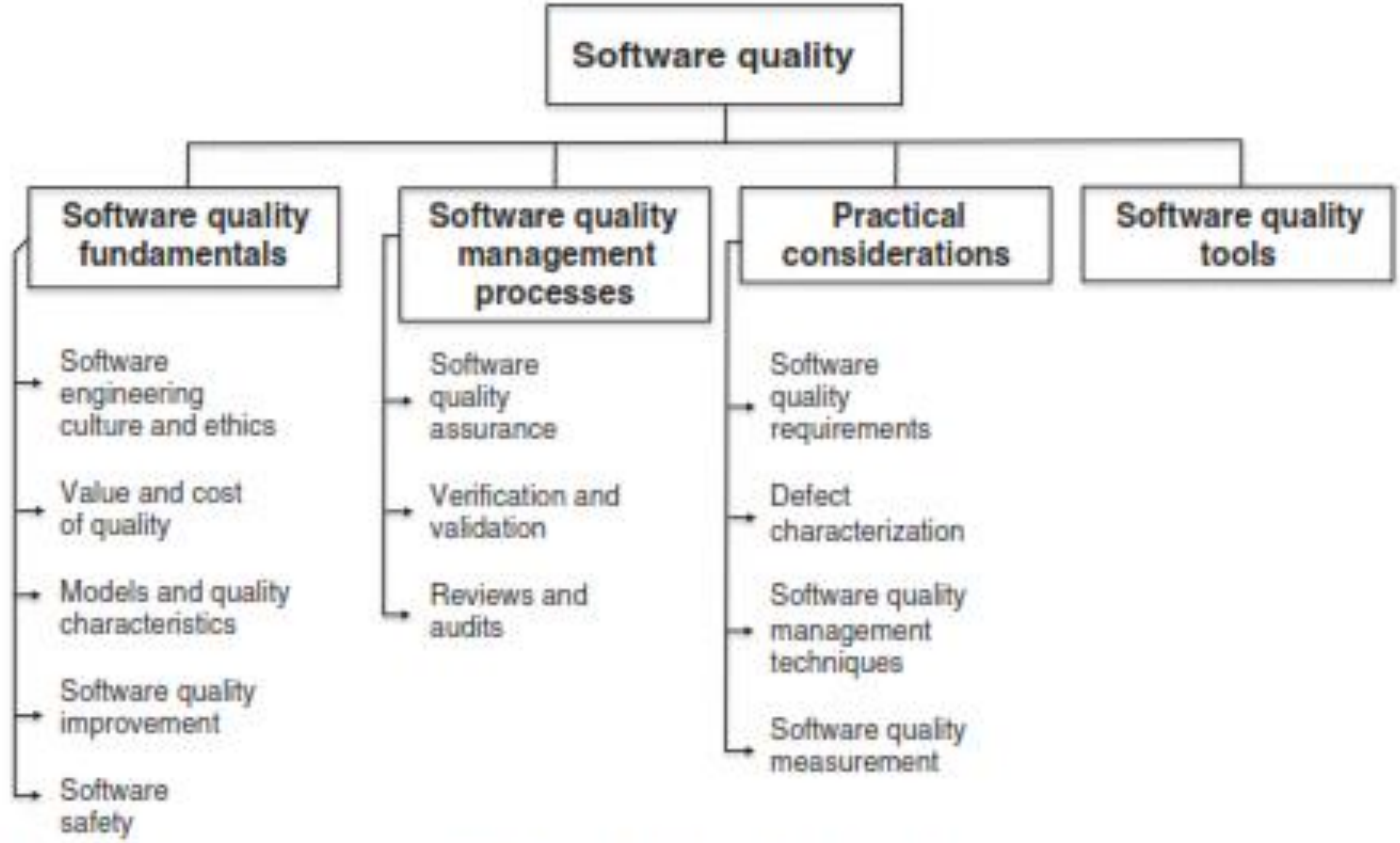

Software quality
Software quality fundamentals
Software quality management processes
Practical considerations
Software quality tools
Software engineering culture and ethics
Value and cost of quality
Models and quality characteristics
Software quality improvement
Software safety
Software quality assurance
Verification and validation
Reviews and audits
Software quality requirements
Defect characterization
Software quality management techniques
Software quality measurement

Figure 1.1 Software Quality in the SWEBOK® Guide [SWE 14].

DEFINING SOFTWARE QUALITY

“software,” “software quality,” and “software quality assurance”

Software [ISO 24765 [ISO 17a]]

1) All or part of the programs, procedures, rules, and associated documentation of an information processing system.
2) Computer programs, procedures, and possibly associated documentation and data pertaining to the operation of a computer system.

Software found in embedded systems is sometimes called microcode or firmware.

Firmware is present in commercial mass-market products and controls machines and devices used in our daily lives.

Firmware

Combination of a hardware device and computer instructions or computer data that reside as read-only software on the hardware device.

SOFTWARE ERRORS, DEFECTS, AND FAILURES

– The system crashed during production.

– The designer made an error.

– After a review, we found a defect in the test plan.

– I found a bug in a program today.

– The system broke down.

– The client complained about a problem with a calculation in the payment report.

– A failure was reported in the monitoring subsystem.

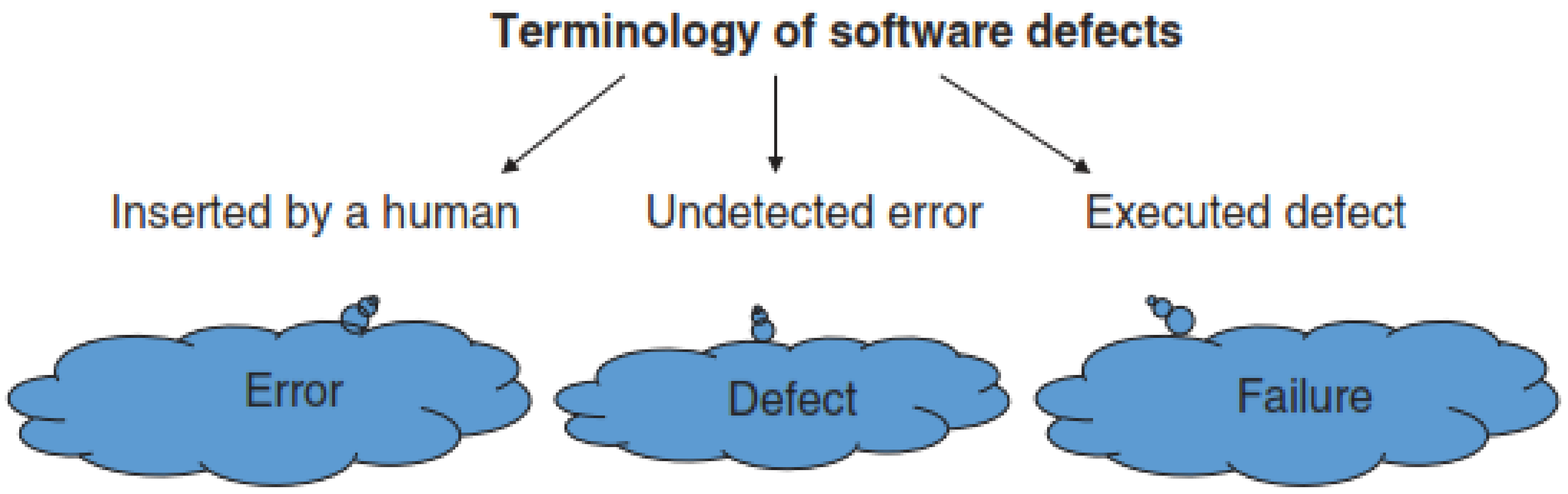
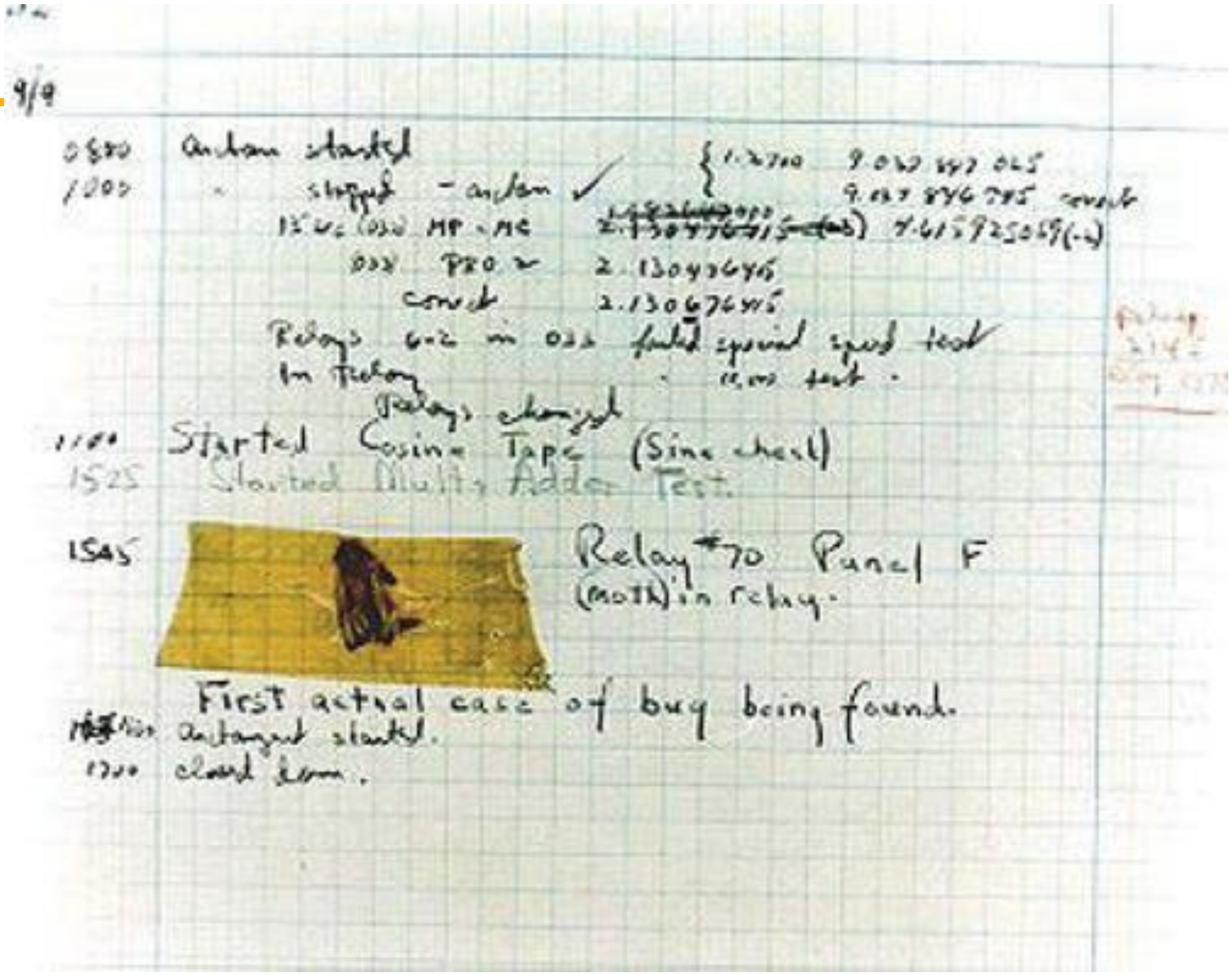


Figure 1.2 Terminology recommended for describing software problems.



9/9

0800 Anstan started
1000 " stopped - anstan ✓ { 1.2700 9.037 847 025
9.037 846 795 correct
13" UC (032) MP - MC ~~1.982147000~~ ~~2.130476415~~ (-3) 4.615925059(-2)
(033) PRO 2 2.130476415
correct 2.130676415
Relays 6-2 in 033 failed special speed test
In Relay " 10.000 test .
Relays changed
1100 Started Cosine Tape (Sine check)
1525 Started Multy Adder Test.
1545 Relay #70 Panel F
(moth) in relay.
First actual case of bug being found.
1630 Antangent started.
1700 closed down.

[illegible]

A failure (synonymous with a crash or breakdown) is the execution (or manifestation) of a fault in the operating environment.

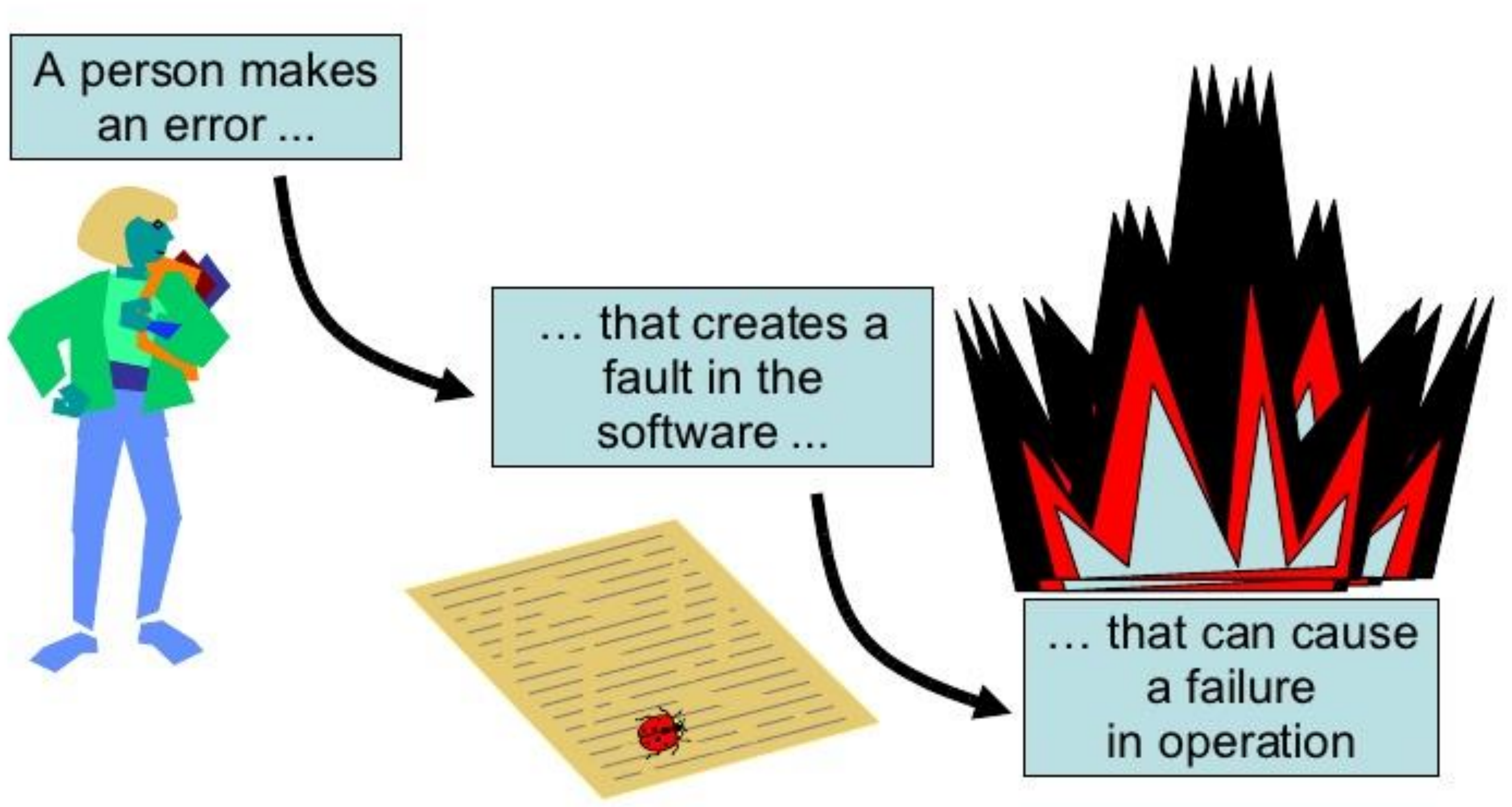
A failure is defined as the termination of the ability of a component to fully or partially perform a function that it was designed to carry out.

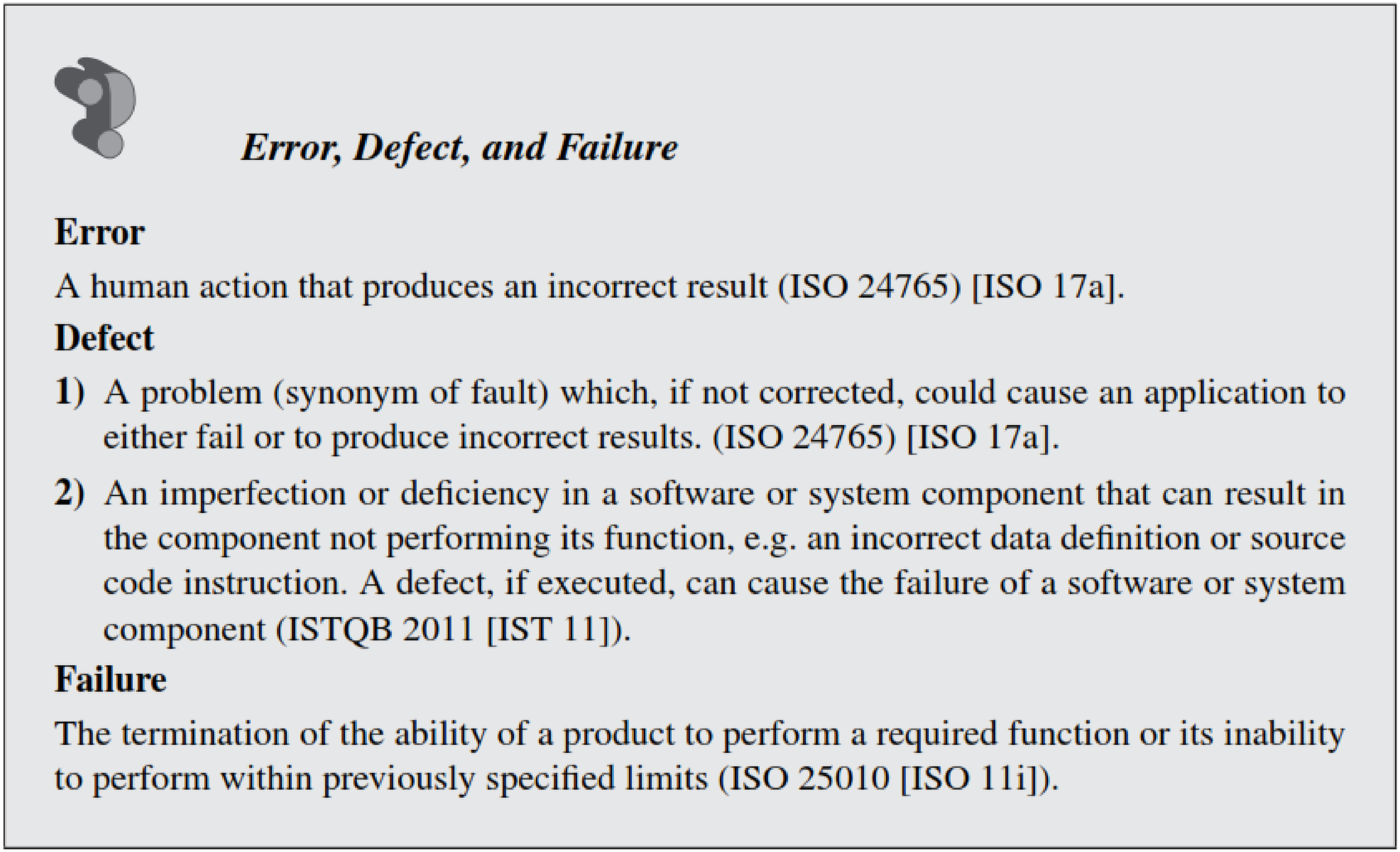
The origin of a failure lies with a defect hidden, that is, not detected by tests or reviews, in the system currently in operation.

Defects (synonym of faults) are human errors that were not detected during software development, quality assurance (QA), or testing.

An error can be found in the documentation, the software source code instructions, the logical execution of the code, or anywhere else in the life cycle of the system.

Error - Fault - Failure


A person makes an error ...
… that creates a fault in the software ...
… that can cause a failure in operation



Error, Defect, and Failure

Error

A human action that produces an incorrect result (ISO 24765) [ISO 17a].

Defect

1) A problem (synonym of fault) which, if not corrected, could cause an application to either fail or to produce incorrect results. (ISO 24765) [ISO 17a].

2) An imperfection or deficiency in a software or system component that can result in the component not performing its function, e.g. an incorrect data definition or source code instruction. A defect, if executed, can cause the failure of a software or system component (ISTQB 2011 [IST 11]).

Failure

The termination of the ability of a product to perform a required function or its inability to perform within previously specified limits (ISO 25010 [ISO 11i]).

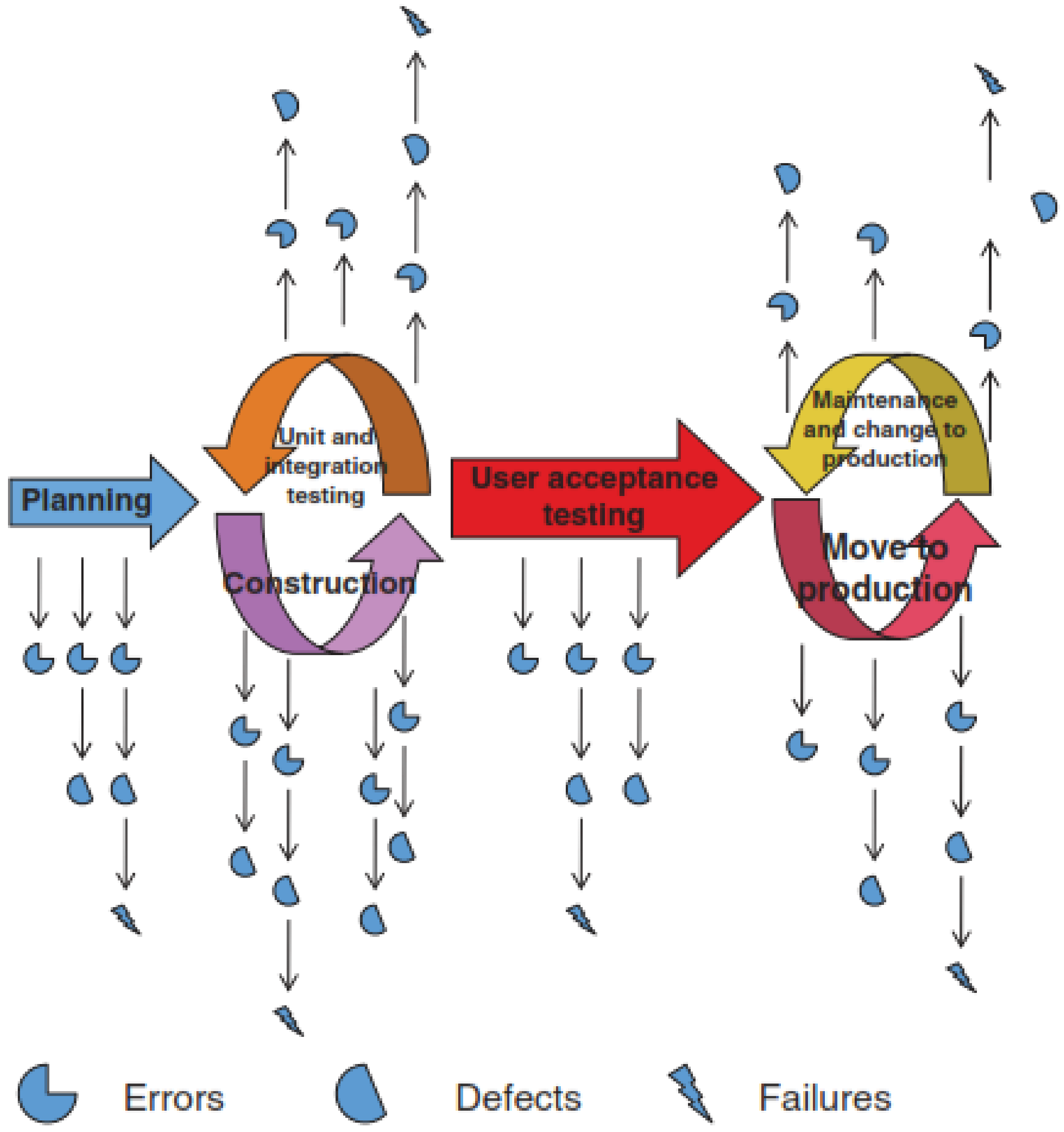


Figure 1.3 Errors, defects, and failures in the software life cycle.

Case 1:

A local pharmacy added a software requirement to its cash register to prevent sales of more than $75 to customers owing more than $200 on their pharmacy credit card.

The programmer did not fully understand the specification and created a sales limit of $500 within the program. This **defect** never caused a failure since no client could purchase more than $500 worth of items given that the pharmacy credit card had a limit of $400.

Case 2: In 2009, a loyalty program was introduced to the clients of American Signature, a large furniture supplier. The specifications described the following business rules: a customer who makes a monthly purchase that is higher than the average amount of monthly purchases for all customers will be considered a Preferred Customer.

The Preferred Customer will be identified when making a purchase, and will be immediately given a gift or major discount once a month.

The defect introduced into the system (due to a poor understanding of the algorithm to set up for this requirement) involved only taking into account the average amount of current purchases and not the customer’s monthly history. At the time of the software failure, the cash register was identifying far too many Preferred Clients, resulting in a loss for the company.

Development Life Cycle

Software life cycle process that contains the activities of requirements analysis, design, coding, integration, testing, installation, and support for acceptance of software products.

Depending on the business model of your organization, you will have to allow for varying degrees of effort in identifying and correcting defects.

Airbus, Boeing, Bombardier, and Embraer to have identified and corrected all the defects in the software for their airplanes before we board them!

Research Studies have come to the following conclusions:

– The scope of most defects is very limited and easy to correct.

– Many defects occur outside of the coding activity (e.g., requirement, architecture activities).

– Poor understanding of the design is a recurrent problem in programming error studies.

– It is a good idea to measure the number and origin of defects in your organization to set targets for improvement.

SQA need to develop a classification of the causes of software error by category.

1) problems with defining requirements;
2) maintaining effective communication between client and developer;
3) deviations from specifications;
4) architecture and design errors;
5) coding errors (including test code);
6) non-compliance with current processes/procedures;
7) inadequate reviews and tests;
8) documentation errors.

SOFTWARE QUALITY

- *Conformance to established software requirements; the capability of a software product to satisfy stated and implied needs when used under specified conditions.*
- *The degree to which a software product meets established requirements; however, quality depends upon the degree to which those established requirements accurately represent stakeholder needs, wants, and expectations*

SOFTWARE QUALITY ASSURANCE

Software Engineering

The systematic application of scientific and technological knowledge, methods, and experience for the design, implementation, testing, and documentation of software.

Quality Assurance

- **1) a planned and systematic pattern of all actions necessary to provide adequate confi**dence that an item or product conforms to established technical requirements;
- **2) a set of activities designed to evaluate the process by which products are developed or** manufactured;
- **3) the planned and systematic activities implemented within the quality system, and** demonstrated as needed, to provide adequate confidence that an entity will fulfil requirements for quality.

Software Quality Assurance

Definition

A set of activities that define and assess the adequacy of software processes to provide evidence that establishes confidence that the software processes are appropriate for and produce software products of suitable quality for their intended purposes.

A key attribute of SQA is the objectivity of the SQA function with respect to the project.

The SQA function may also be organizationally independent of the project; that is, free from technical, managerial, and financial pressures from the project.

SQA Elements

- The need to plan the quality aspects of a product or service;
- Systematic activities that tell us, throughout the software life cycle, that certain corrections are required;
- The quality system is a complete system that must, in the context of quality management, allow for the setting up of a quality policy and continuous improvement;
- QA techniques that demonstrate the level of quality reached so as to instill confidence in users; and lastly,
- Demonstrate that the quality requirements defined for the project, for the change or by the software department have been met.

BUSINESS MODELS AND THE CHOICE OF SOFTWARE ENGINEERING PRACTICES

Business Model

A business model describes the rationale of how an organization creates, delivers, and captures value (economic, social, or other forms of value).

The essence of a business model is that it defines the manner by which the business enterprise delivers value to customers, entices customers to pay for value, and converts those payments to profit.

Choice of Software Practices

As expected, people from different business sectors chose software engineering practices that would lower the probability of their worst fears(quality and deadlines).

Since their apprehensions are different, their practices are also different.

SUCCESS FACTORS
Foster Software Quality

1) SQA techniques adapted to the environment.
2) Clear terminology with regards to software problems.
3) An understanding and specific attention to each major category of software error sources.
4) An awareness of the SQA body of knowledge of the SWEBOK as a guide for SQA.

Factors that may Adversely Affect Software Quality

1) A lack of cohesion between SQA techniques and environmental factors in your organization.
2) Confusing terminology used to describe software problems.
3) A lack of understanding or interest for collecting information on software error sources.
4) Poor understanding of software quality fundamentals.
5) Ignorance or non-adherence with published SQA techniques.

Software quality assurance vs. software quality control

Quality control **is defined as "a set of activities designed to evaluate the** quality of a developed or manufactured product"

The main objective of quality assurance **is to minimize the cost of guaranteeing** quality by a variety of activities performed throughout the development and manufacturing processes/stages.

These activities prevent the causes of errors, and detect and correct them early in the development process.

Quality assurance activities substantially reduce the rate of products that do not qualify for shipment and, at the same time, reduce the costs of guaranteeing quality in most cases.

The objectives of SQA activities

Software development (process-oriented):

Software maintenance (product-oriented):

Software development (process-oriented):

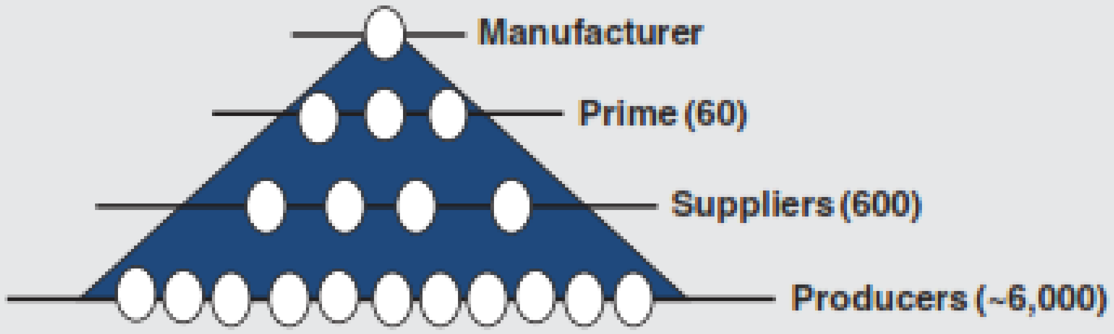
- 1. Assuring an acceptable level of confidence that the software will conform to functional technical requirements.
- 2. Assuring an acceptable level of confidence that the software will conform to managerial scheduling and budgetary requirements.
- 3. Initiating and managing of activities for the improvement and greater efficiency of software development and SQA activities. This means improving the prospects that the functional and managerial requirements will be achieved while reducing the costs of carrying out the software development and SQA activities.

Software maintenance (product-oriented):

- **1. Assuring with an acceptable level of confidence that the software maintenance activities will conform to the functional technical requirements.**
- **2. Assuring with an acceptable level of confidence that the software maintenance activities will conform to managerial scheduling and budgetary requirements.**
- **3. Initiating and managing activities to improve and increase the efficiency of software maintenance and SQA activities. This involves improving the prospects of achieving functional and managerial requirements while reducing costs.**

Quality Culture

A large Japanese electronic product manufacturing company uses a considerable number of suppliers. The supplier pyramid is made up of a first level with some sixty suppliers, a second level with a few hundred suppliers, and a third level with a few thousand very small suppliers.



A software defect for a component produced by a third-level supplier caused a loss of over $200 million for this manufacturer.

Adapted from Shintani (2006) [SHI 06]

Setting up a quality culture and software quality assurance (SQA) principles, as stipulated in the standards, could help solve these problems.

Quality, influenced by organization’s senior management and the organization’s culture, has a cost, has a positive effect on profits, and must be governed by a code of ethics.

COST OF QUALITY

One of the major factors that explains the resistance to implementing quality assurance is the perception of its high cost.

As software engineers, responsible for informing administrators of the risks that a company takes when not fully committed to the quality of its software.

A practical manner of beginning the exercise is to identify the costs of non-quality.

It is easier to identify potential savings by studying the problems caused by software.

Costs of a project

(1) Implementation costs

(2) Prevention costs

(3) Appraisal costs

(4) The costs associated with failures or anomalies.

- If all development activities are error free, then 100% of costs could be implementation costs.
- Given that we make mistakes, we need to be able to identify them. The costs of detecting errors are appraisal costs (e.g., testing).
- Costs due to errors are anomaly costs.
- When we want to reduce the cost of anomalies, we invest in training, tools, and methodology. These are prevention costs.

The “cost of quality” is not calculated in the same way in all organizations.

- There is a certain amount of ambiguity between the notion of the cost of quality, the cost of non-quality, and the cost of obtaining quality.
- Most commonly used model at this time, costs of quality take into account the following five perspectives:

(1) prevention costs, (2) appraisal costs, (3–4) failure costs (internal during development, external on the client’s premises), and (5) costs associated with warranty claims and loss of reputation caused by non-quality.

calculation of the cost of quality in this model is as follows:

Quality costs = Prevention costs

+ Appraisal or evaluation costs

+ Internal and external failure costs

+ Warranty claims and loss of reputation costs

Prevention costs: This is defined as the cost incurred by an organization to prevent the occurrence of errors in the various steps of the development or maintenance process.

- For example, the cost of training employees, the cost of maintenance to ensure a stable manufacturing process, and the cost of making improvements.

Appraisal costs: The cost of verifying or evaluating a product or service during the different steps in the development process. Monitoring system costs (their maintenance and management costs).

Internal failure costs: The cost resulting from anomalies before the product or service has been delivered to the client. Loss of earnings due to non-compliance (cost of making changes, waste, additional human activities, and the use of extra products).

External failure costs: The cost incurred by the company when the client discovers defects. Cost of late deliveries, cost of managing disputes with the client, logistical costs for storing the replacement product or for delivery of the product to the client.

Warranty claims and loss of reputation costs: Cost of warranty execution and damage caused to a corporate image as well as the cost of losing clients.

The first objective of the SQA is to convince management that there are proven benefits to SQA activities.

“identifying an error early in the process can save a lot of time, money and effort.”

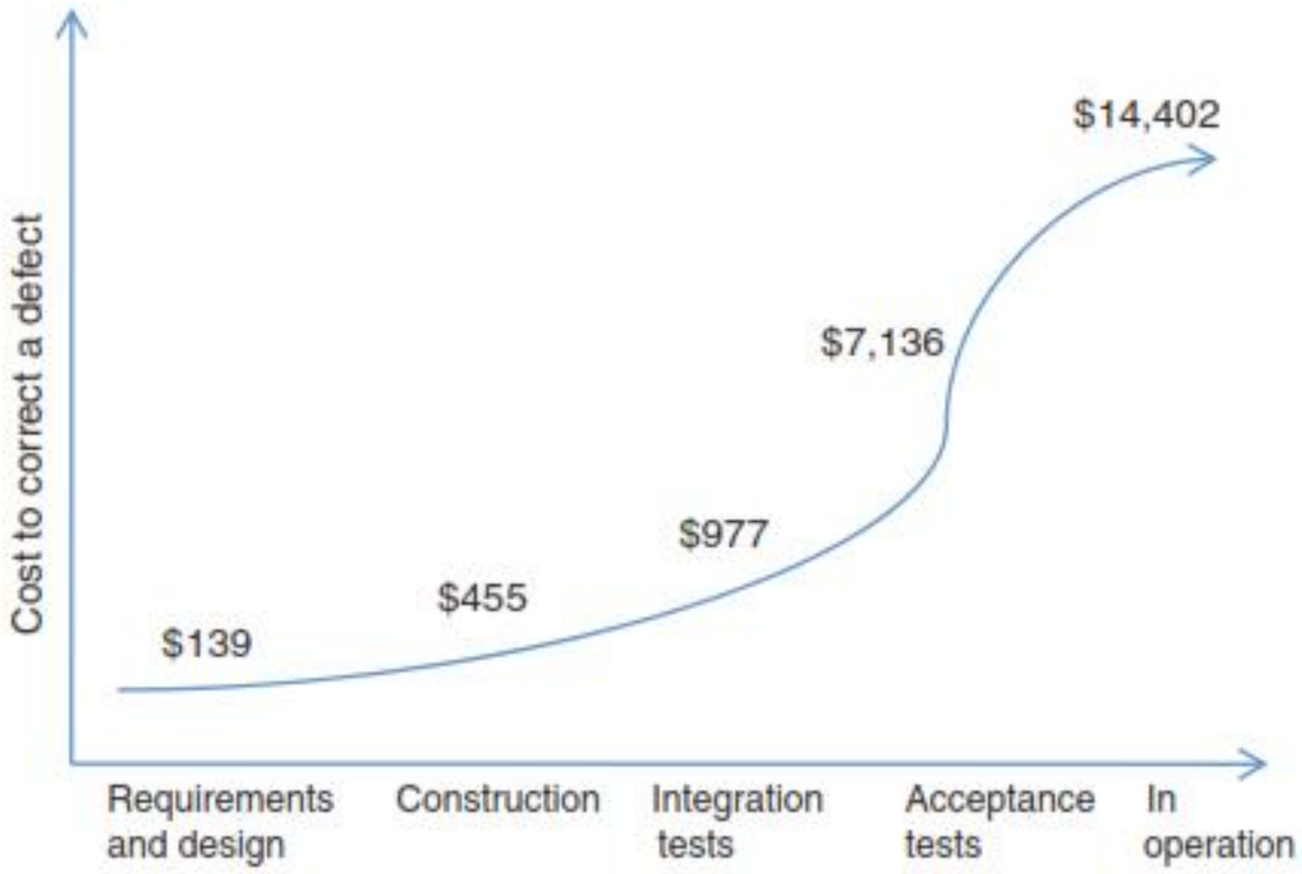


Figure 2.2 Costs of propagating an error[1] [JON 00].

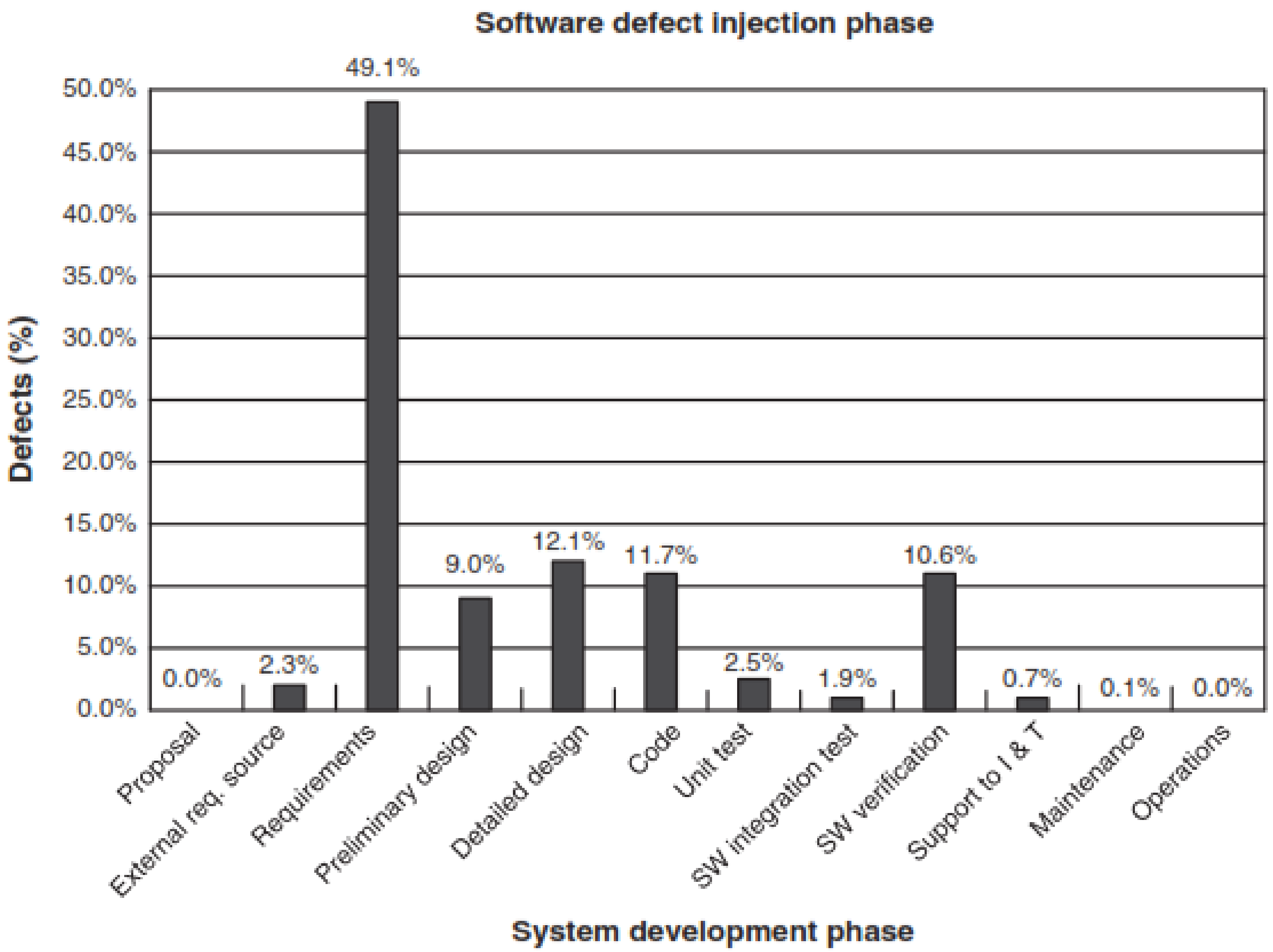


Figure 2.3 Defect injection distribution during the life cycle [SEL 07].

QUALITY CULTURE

Tylor [TYL 10] defined **Human Culture as**

“that complex whole which includes knowledge, belief, art, morals, law, custom, and any other capabilities and habits acquired by man as a member of society.”

It is culture that guides the behaviors, activities, priorities, and decisions of an individual as well as of an organization.

Wiegers (1996) [WIE 96], in his book “*Creating a Software Engineering Culture,”* illustrates the interaction between the software engineering culture of an organization, its software engineers, and its projects

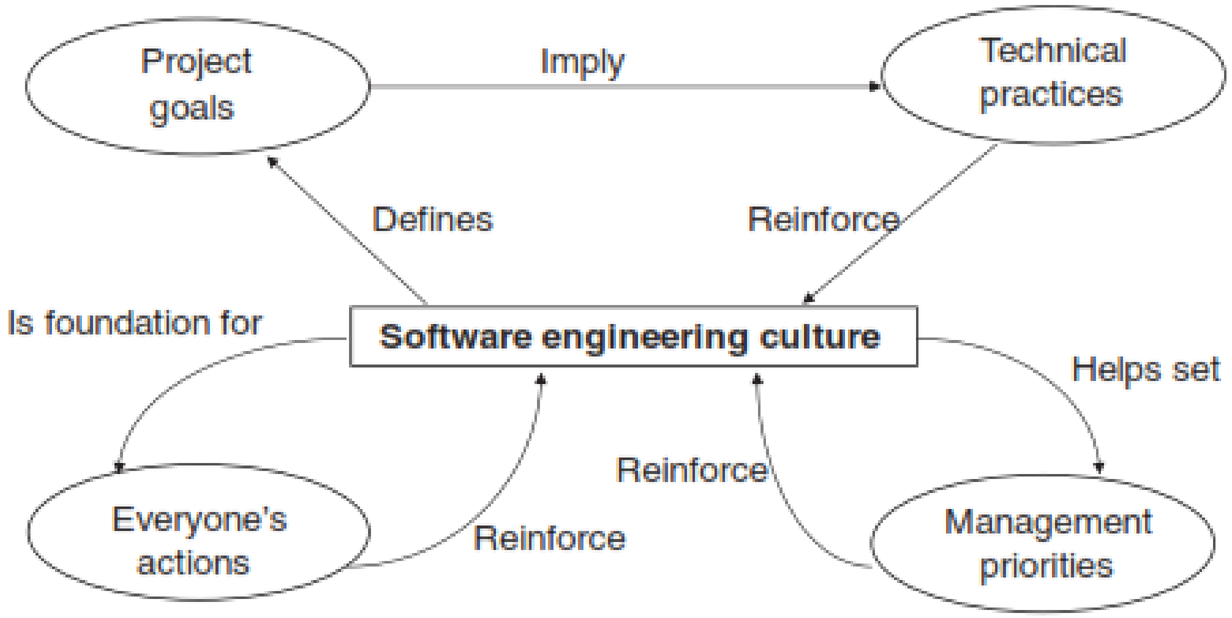


Figure 2.5 Software engineering culture.
Source: Adapted from Wiegers 1996 [WIE 96].

A healthy culture is made up of the following elements:

- The personal commitment of each developer to create quality products by systematically applying effective software engineering practices.
- The commitment to the organization by managers at all levels to provide an environment in which software quality is a fundamental factor of success and allows each developer to carry out this goal.
- The commitment of all team members to constantly improve the processes they use and to always work on improving the products they create.

Figure 2.6 Start coding … I'll go and see what the client wants!
Source: Reproduced with permission of CartoonStock ltd.

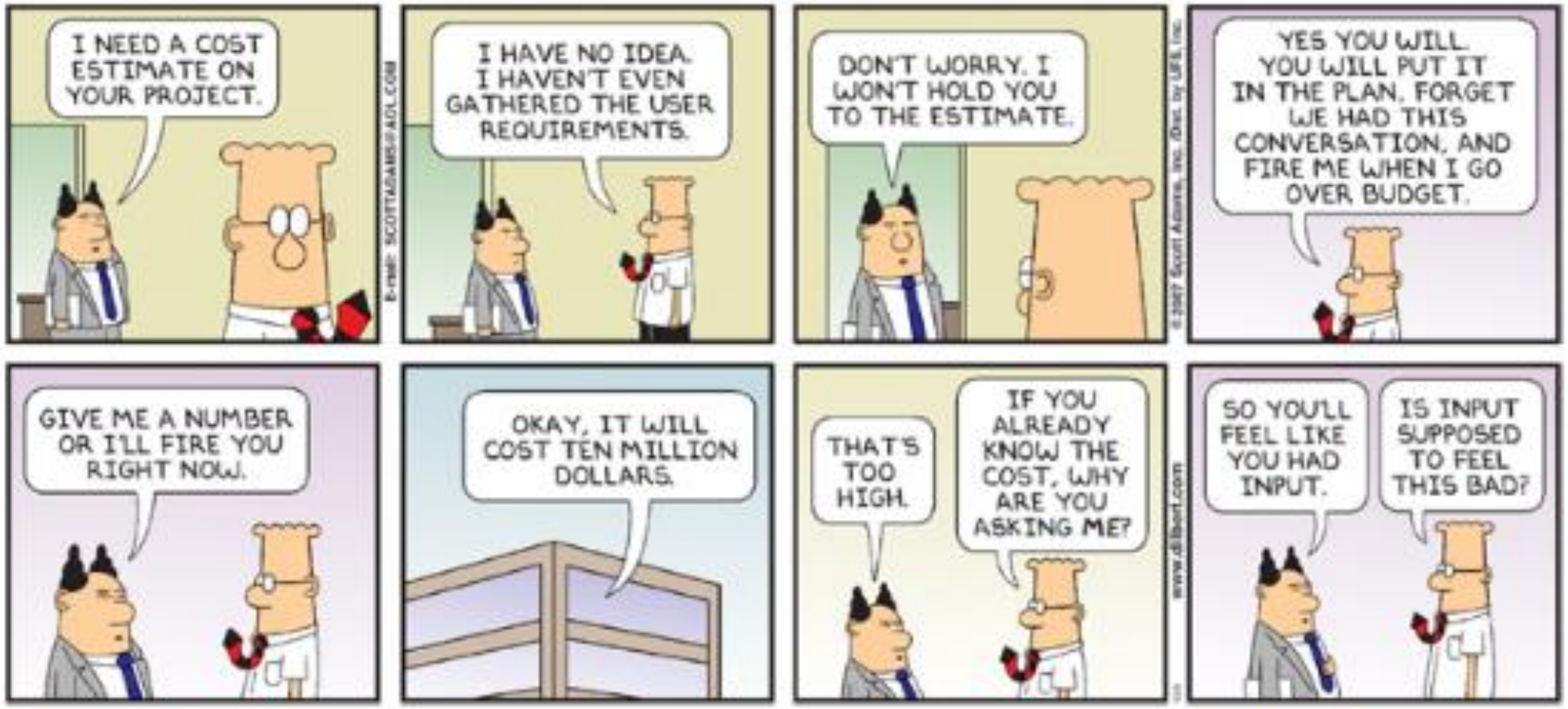

I NEED A COST ESTIMATE ON YOUR PROJECT.
I HAVE NO IDEA. I HAVEN'T EVEN GATHERED THE USER REQUIREMENTS.
DON'T WORRY. I WON'T HOLD YOU TO THE ESTIMATE.
YES YOU WILL. YOU WILL PUT IT IN THE PLAN, FORGET WE HAD THIS CONVERSATION, AND FIRE ME WHEN I GO OVER BUDGET.
GIVE ME A NUMBER OR I'LL FIRE YOU RIGHT NOW.
OKAY, IT WILL COST TEN MILLION DOLLARS.
THAT'S TOO HIGH.
IF YOU ALREADY KNOW THE COST, WHY ARE YOU ASKING ME?
SO YOU'LL FEEL LIKE YOU HAD INPUT.
IS INPUT SUPPOSED TO FEEL THIS BAD?
www.dilbert.com

Figure 2.7 Dilbert is threatened and must provide an estimate on the fly. DILBERT © 2007 Scott Adams. Used By permission of UNIVERSAL UCLICK. All rights reserved.

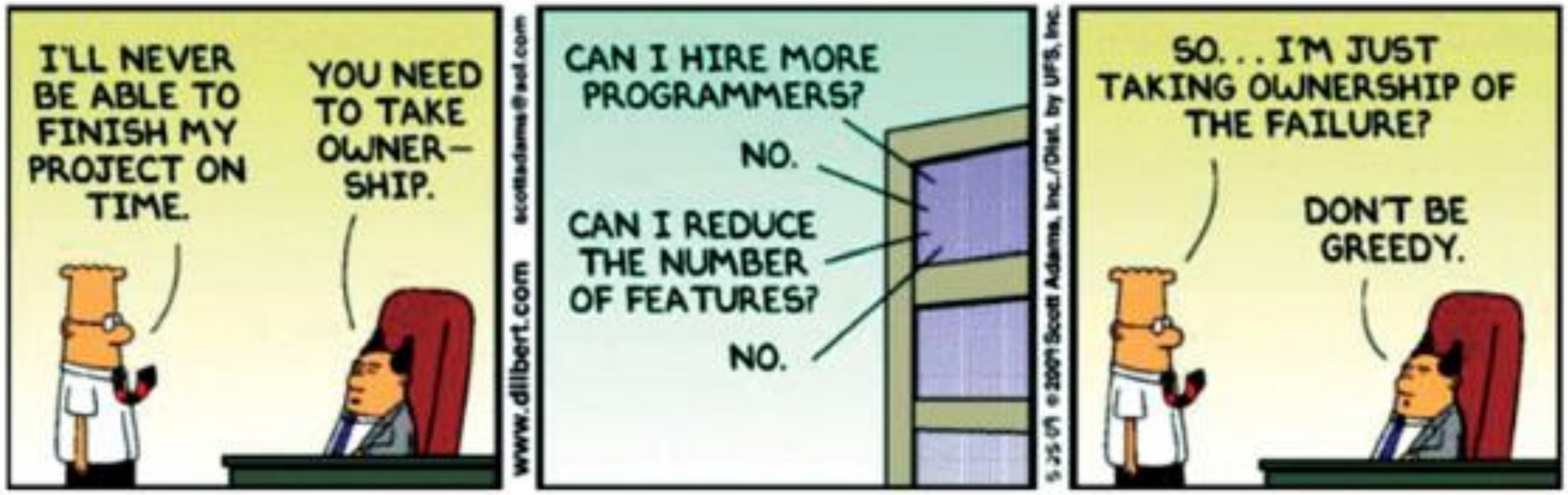

I'LL NEVER BE ABLE TO FINISH MY PROJECT ON TIME.
YOU NEED TO TAKE OWNER-SHIP.
CAN I HIRE MORE PROGRAMMERS?
NO.
CAN I REDUCE THE NUMBER OF FEATURES?
NO.
SO... I'M JUST TAKING OWNERSHIP OF THE FAILURE?
DON'T BE GREEDY.
www.dilbert.com scottadams@aol.com
5-25-09 © 2009 Scott Adams, Inc./Dist. by UFS, Inc.

Figure 2.8 Dilbert tries to negotiate a change in his project. DILBERT © 2009 Scott Adams. Used By permission of UNIVERSAL UCLICK. All rights reserved.

Fourteen principles to follow to develop a culture that fosters quality

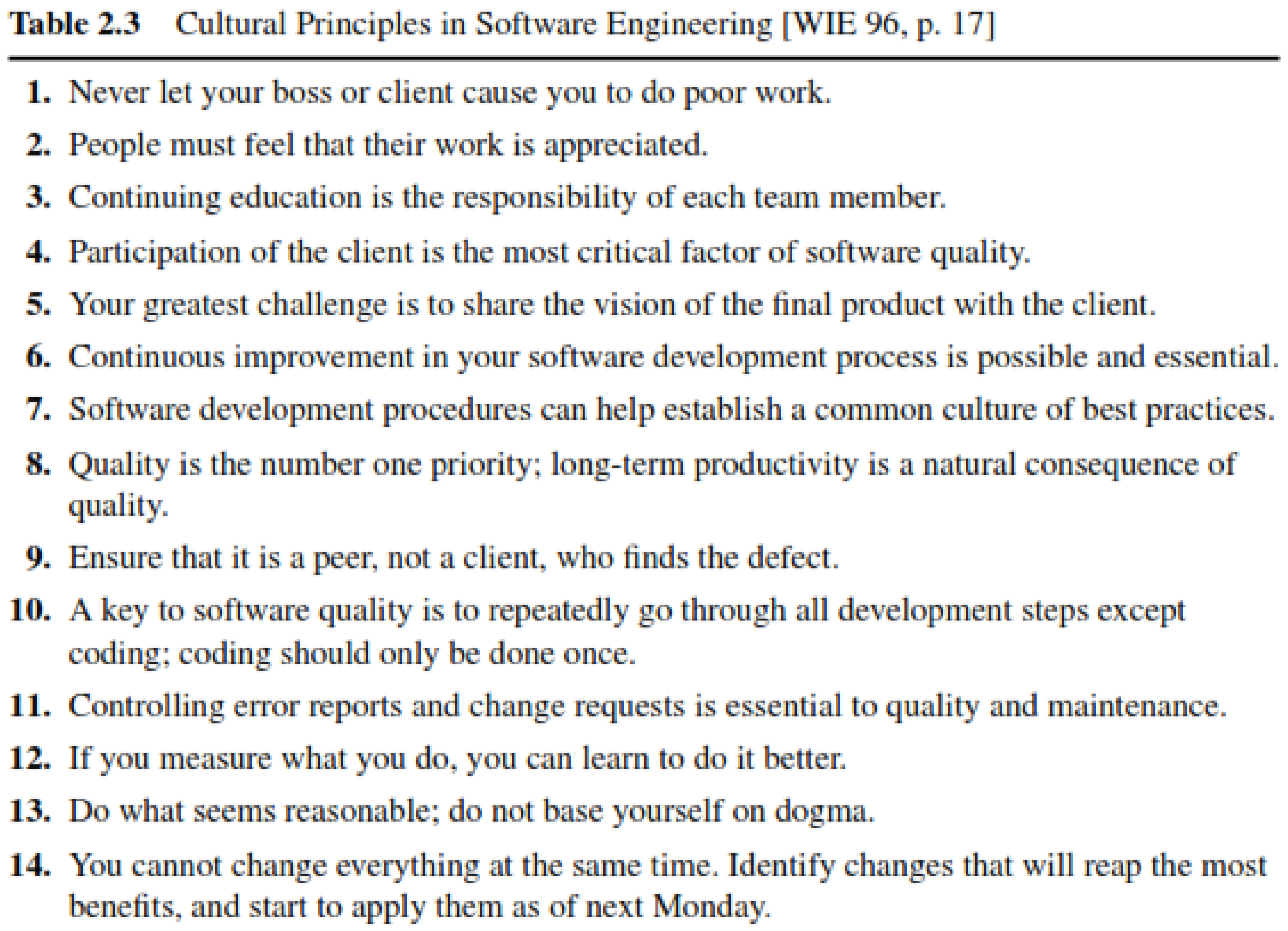


Table 2.3 Cultural Principles in Software Engineering [WIE 96, p. 17]

1. Never let your boss or client cause you to do poor work.
2. People must feel that their work is appreciated.
3. Continuing education is the responsibility of each team member.
4. Participation of the client is the most critical factor of software quality.
5. Your greatest challenge is to share the vision of the final product with the client.
6. Continuous improvement in your software development process is possible and essential.
7. Software development procedures can help establish a common culture of best practices.
8. Quality is the number one priority; long-term productivity is a natural consequence of quality.
9. Ensure that it is a peer, not a client, who finds the defect.
10. A key to software quality is to repeatedly go through all development steps except coding; coding should only be done once.
11. Controlling error reports and change requests is essential to quality and maintenance.
12. If you measure what you do, you can learn to do it better.
13. Do what seems reasonable; do not base yourself on dogma.
14. You cannot change everything at the same time. Identify changes that will reap the most benefits, and start to apply them as of next Monday.

THE SOFTWARE ENGINEERING CODE OF ETHICS

The first draft of the software engineering code of ethics was developed in cooperation with the Institute of Electrical and Electronics Engineers (IEEE) Computer Society and the Association for Computing Machinery (ACM)

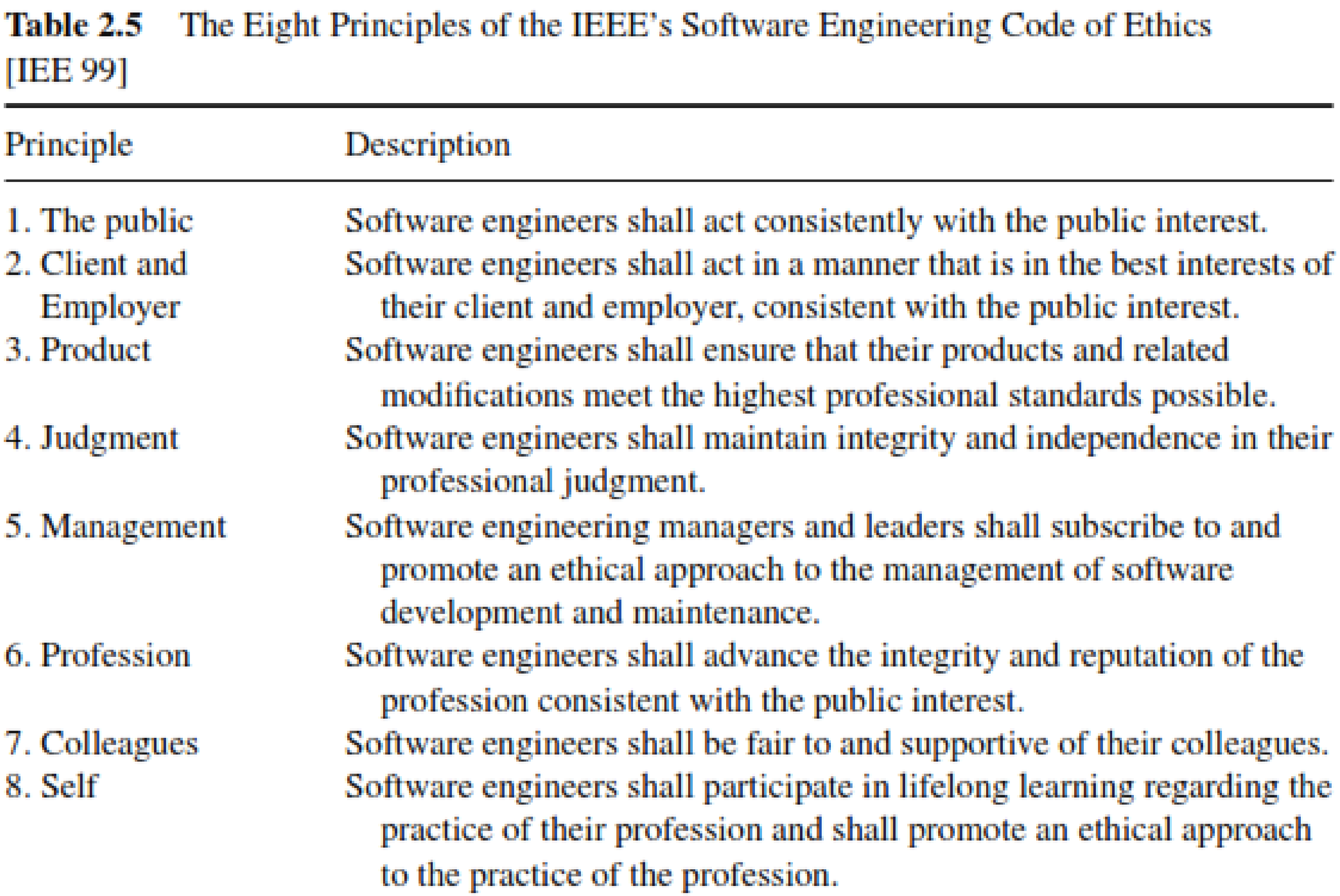


Table 2.5 The Eight Principles of the IEEE's Software Engineering Code of Ethics [IEE 99]

Principle	Description
1. The public	Software engineers shall act consistently with the public interest.
2. Client and Employer	Software engineers shall act in a manner that is in the best interests of their client and employer, consistent with the public interest.
3. Product	Software engineers shall ensure that their products and related modifications meet the highest professional standards possible.
4. Judgment	Software engineers shall maintain integrity and independence in their professional judgment.
5. Management	Software engineering managers and leaders shall subscribe to and promote an ethical approach to the management of software development and maintenance.
6. Profession	Software engineers shall advance the integrity and reputation of the profession consistent with the public interest.
7. Colleagues	Software engineers shall be fair to and supportive of their colleagues.
8. Self	Software engineers shall participate in lifelong learning regarding the practice of their profession and shall promote an ethical approach to the practice of the profession.

Role of SQA in software development life cycle

The Software Development Life Cycle is split into six main phases.

1. Planning
2. Design
3. Implementation
4. Testing
5. Deployment
6. Maintenance

In the planning phase of a software project, Quality Assurance is responsible for making sure that the project meets all quality requirements, such as scope, budget, timeline, and compliance with standards.

QA can also review user requirements and analyze them to determine if they fit within the scope of the project. This helps ensure that expectations are properly set at the beginning of a project, and that resources are allocated appropriately.

QA is involved in the design phase, they can identify aspects of the design that might cause problems ,while they’re still in-progress. This enables the designer or wireframes creator to make changes on the fly.

The role QA plays in the implementation stage:

- Code Reviews
- System Integration Testing
- User Acceptance Testing

The Role of QA in Testing

QA focuses on various aspects, such as functionality, usability, reliability, performance, and compliance with industry standards. In order to ensure that the requirements of the application are met before it is released to its users.

The Role of QA in Deployment

In deployment stage QA ensures that all the elements of specific Custom software development and its properly implemented, tested, verified, and deployed. This allows the product to be released with confidence and without any issues or bugs.

The Role of QA in Maintenance

- Verifying software updates to confirm they are working properly
- Testing changes to make sure they are as expected
- Identifying potential problems with updates
- Following up with customers to ensure they are satisfied with changes and updates
- Documenting all issues related to releases, so that future versions can be improved upon accordingly.
- By investing in QA during maintenance, companies can be sure their products remain reliable and bug-free for customers for the long haul!

innovate achieve lead

THANK YOU