Software Engineering
College of Arts, Media and Technology ,CMU.

SE422 Software Quality Assurance

CH2-Software Quality

Kittitouch S.
1.1-8-11-12

Topics

- What is Software?
- Software Error, Fault and Failure
  - Case study:Therac-25
- Classification of the causes of software errors
- Software Quality
- Software Quality Assurance
- Quality Control
- The objectives of SQA activities

What is Software?

- Software – IEEE definition
  Software is Computer programs, procedures, and possibly associated documentation and data pertaining to the operation of a computer system.

Institute of Electrical and Electronics Engineers (IEEE)

Software….

- Computer programs (the “code”)
- Procedures
- Documentation
- Data necessary for operating the software system.

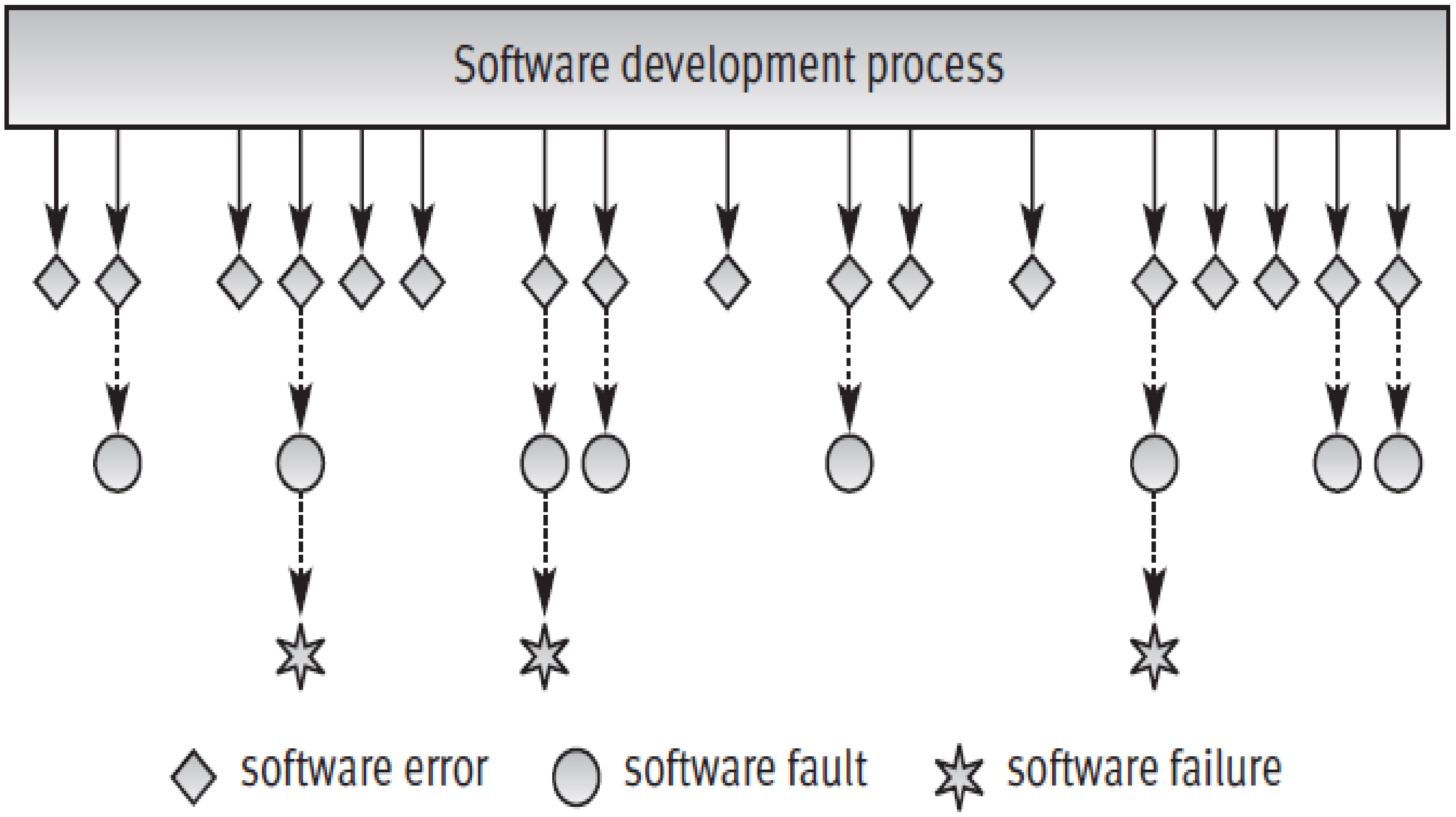
Software Error, Fault and failures

Software failure case study

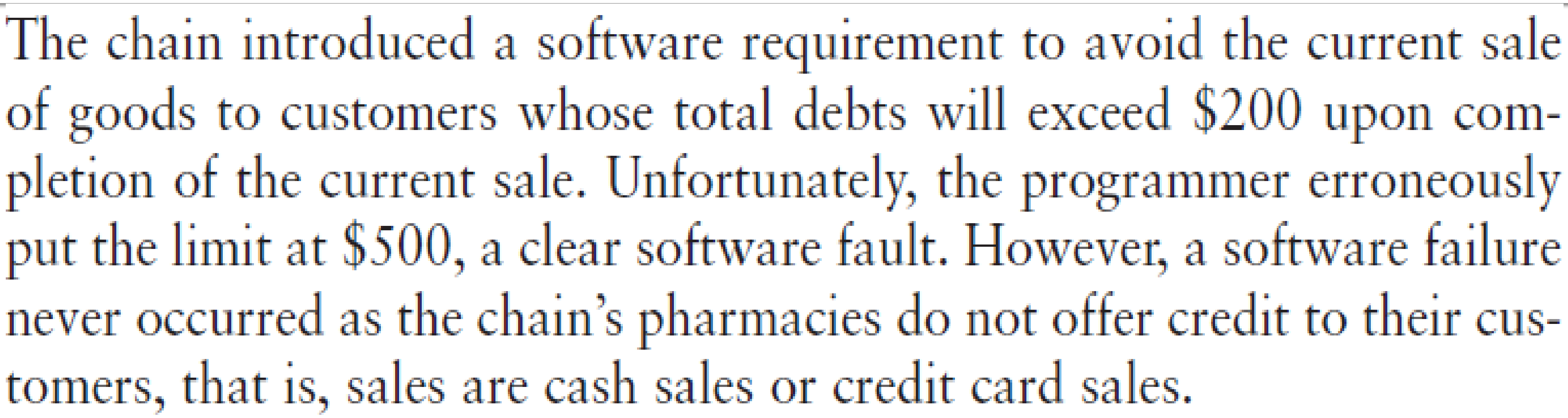
- "We've used the Simplex HR software in our Human Resources Department for about three years and we have never had a software failure."
- "I started to use Simplex HR two months ago; we had so many failures that we are considering replacing the software package."
- "We have been using the same software package for almost four years. We were very satisfied throughout the period until the last few months, when we suddenly faced several severe failures. The Support Center of the software house from which we bought the package claims that they have never encountered failures of the type we experienced even though they serve about 700 customers who utilize Simplex HR."

Software Error, Fault and failures

- Software error: made by programmer.
- Software fault: defect in product.
- Software failure: software fault is activate.

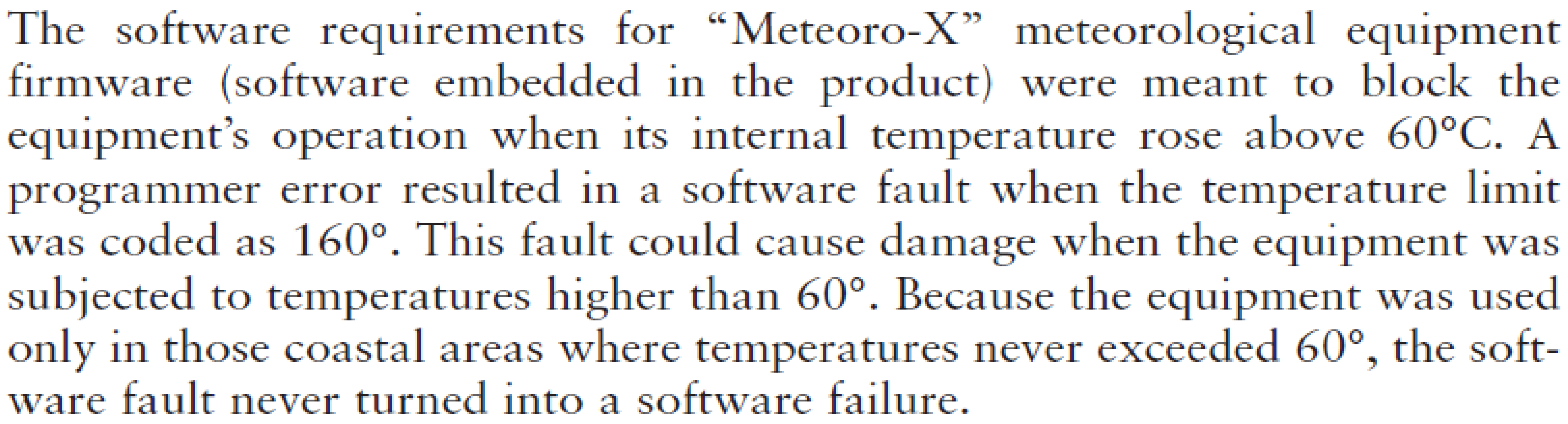

Software development process
software error
software fault
software failure

Software Error, Fault and failures



The chain introduced a software requirement to avoid the current sale of goods to customers whose total debts will exceed $200 upon completion of the current sale. Unfortunately, the programmer erroneously put the limit at $500, a clear software fault. However, a software failure never occurred as the chain's pharmacies do not offer credit to their customers, that is, sales are cash sales or credit card sales.

Example 2: The "Meteoro-X" meteorological equipment firmware



The software requirements for "Meteoro-X" meteorological equipment firmware (software embedded in the product) were meant to block the equipment's operation when its internal temperature rose above 60°C. A programmer error resulted in a software fault when the temperature limit was coded as 160°. This fault could cause damage when the equipment was subjected to temperatures higher than 60°. Because the equipment was used only in those coastal areas where temperatures never exceeded 60°, the software fault never turned into a software failure.

2.3 Classification of the causes of software errors

1. Faulty definition of requirements

2.3 Classification of the causes of software errors

2. Client–developer communication failures

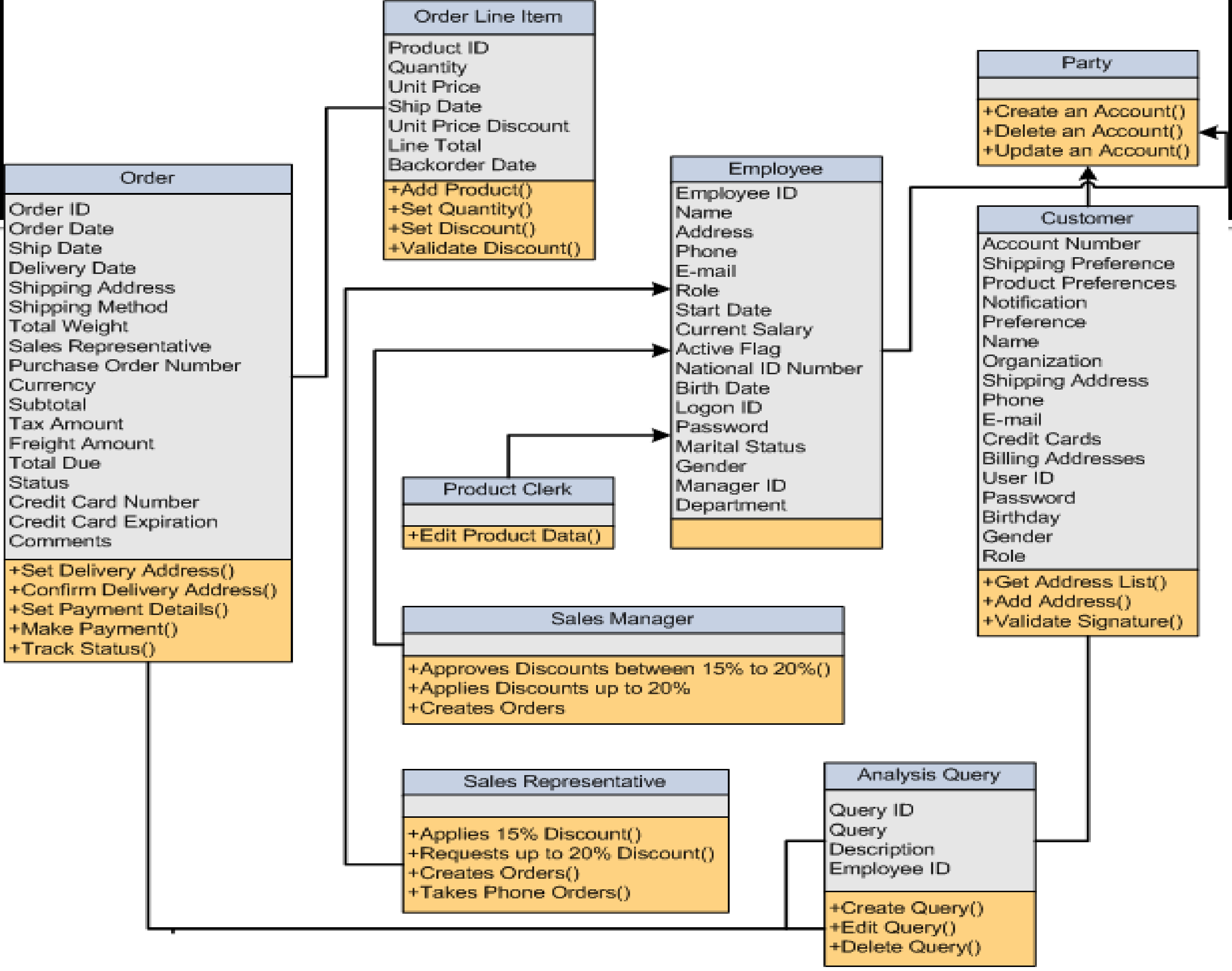
- Misunderstanding requirement, change, design,…,etc.

2.3 Classification of the causes of software errors

3. Deliberate deviations from software requirements

- The developer reuses software modules taken from an earlier project without sufficient analysis of the changes and adaptations needed to correctly fulfill all the new requirements.

- Due to time or budget pressures, the developer decides to omit part of the required functions in an attempt to cope with these pressures.

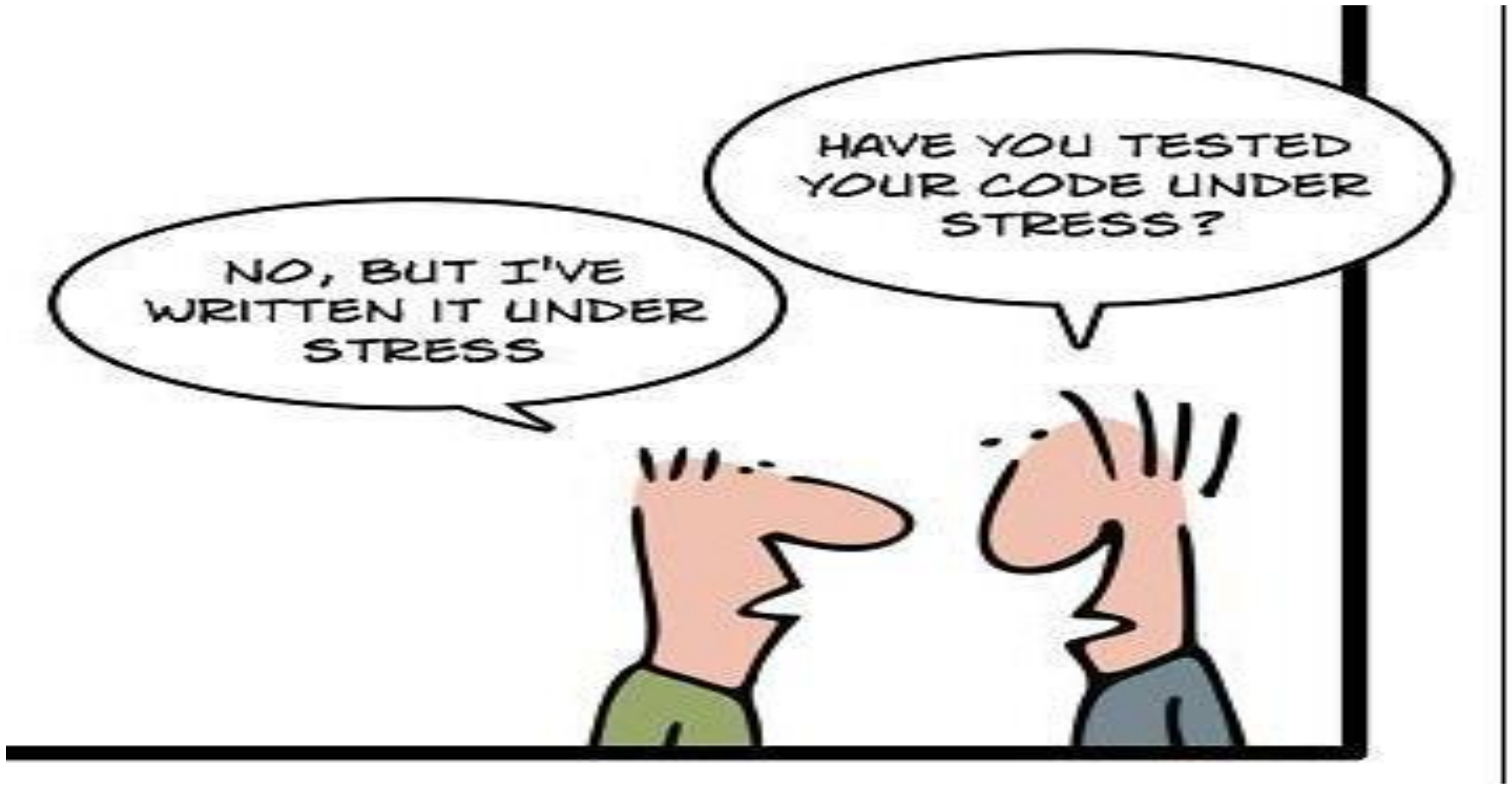

Order Line Item
Product ID
Quantity
Unit Price
Ship Date
Unit Price Discount
Line Total
Backorder Date
+Add Product()
+Set Quantity()
+Set Discount()
+Validate Discount()
Party
+Create an Account()
+Delete an Account()
+Update an Account()
Order
Order ID
Order Date
Ship Date
Delivery Date
Shipping Address
Shipping Method
Total Weight
Sales Representative
Purchase Order Number
Currency
Subtotal
Tax Amount
Freight Amount
Total Due
Status
Credit Card Number
Credit Card Expiration
Comments
+Set Delivery Address()
+Confirm Delivery Address()
+Set Payment Details()
+Make Payment()
+Track Status()
Employee
Employee ID
Name
Address
Phone
E-mail
Role
Start Date
Current Salary
Active Flag
National ID Number
Birth Date
Logon ID
Password
Marital Status
Gender
Manager ID
Department
Customer
Account Number
Shipping Preference
Product Preferences
Notification
Preference
Name
Organization
Shipping Address
Phone
E-mail
Credit Cards
Billing Addresses
User ID
Password
Birthday
Gender
Role
+Get Address List()
+Add Address()
+Validate Signature()
Product Clerk
+Edit Product Data()
Sales Manager
+Approves Discounts between 15% to 20%()
+Applies Discounts up to 20%
+Creates Orders
Sales Representative
+Applies 15% Discount()
+Requests up to 20% Discount()
+Creates Orders()
+Takes Phone Orders()
Analysis Query
Query ID
Query
Description
Employee ID
+Create Query()
+Edit Query()
+Delete Query()

Logical design error case study

- Erroneous definition of boundary conditions. For example, the client's requirements stated that a special discount will be granted to customers who make purchases more than three times in the same month. The analyst erroneously defined the software process to state that the discount would be granted to those who make purchases three times or more in the same month.

2.3 Classification of the causes of software errors

5. Coding error.


HAVE YOU TESTED YOUR CODE UNDER STRESS?
NO, BUT I'VE WRITTEN IT UNDER STRESS

2.3 Classification of the causes of software errors

6. Non-compliance with documentation and coding instructions.

- Team members who need to coordinate their own codes with code modules developed by "non-complying" team members can be expected to encounter more than the usual number of difficulties when trying to understand the software developed by the other team members.
- Individuals replacing the "non-complying" team member (who has retired or been promoted) will find it difficult to fully understand his or her work.

2.3 Classification of the causes of software errors

7. Shortcomings of the testing process.

- Incomplete test plans leave untreated portions of the software or the application functions and states of the system.
- Failures to document and report detected errors and faults.
- Failure to promptly correct detected software faults as a result of in appropriate indications of the reasons for the fault.
- Incomplete correction of detected errors due to negligence or time pressures.

2.3 Classification of the causes of software errors

9. Documentation errors.

- Omission of software functions.
- Errors in the explanations and instructions given to users, resulting in "dead ends" or incorrect applications.
- Listing of non-existing software functions, that is, functions planned in the early stages of development but later dropped, and functions that were active in previous versions of the software but cancelled in the current version.

2.3 The 9 causes of software errors

1. Faulty requirements definition
2. Client–developer communication failures
3. Deliberate deviations from software requirements
4. Logical design errors
5. Coding errors
6. Non-compliance with documentation and coding instructions
7. Shortcomings of the testing process
8. Documentation errors

End of 2-1

Assignment 2-1

- 3 students/team.
- Reading George Wise story.
- Discuss in team to find answer.

Software quality

- IEEE definition
  - The degree to which a system, component, or process meets specified requirements.
  - The degree to which a system, component, or process meets customer or user needs or expectations.

Software Quality Assurance

- IEEE. Definition.
  - A set of activities designed to evaluate the process by which the products are developed or manufactured. Contrast with quality control.

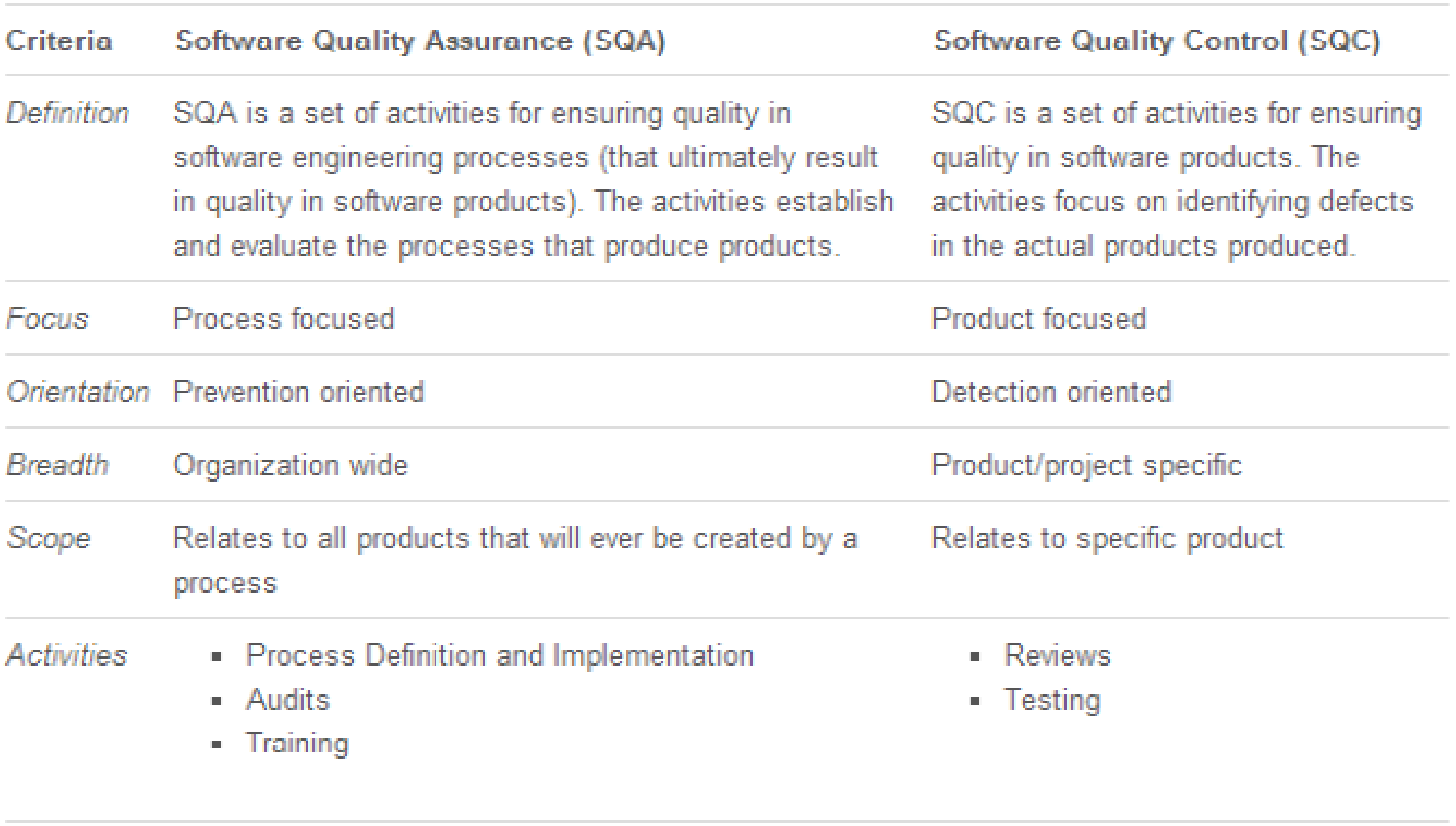
Quality Control

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

- *Reviews*
  - Requirement Review
  - Design Review
  - Code Review
  - Deployment Plan Review
  - Test Plan Review
  - Test Cases Review

- *Testing*
  - Unit Testing
  - Integration Testing
  - System Testing
  - Acceptance Testing

SQA Vs SQC



Criteria	Software Quality Assurance (SQA)	Software Quality Control (SQC)
Definition	SQA is a set of activities for ensuring quality in software engineering processes (that ultimately result in quality in software products). The activities establish and evaluate the processes that produce products.	SQC is a set of activities for ensuring quality in software products. The activities focus on identifying defects in the actual products produced.
Focus	Process focused	Product focused
Orientation	Prevention oriented	Detection oriented
Breadth	Organization wide	Product/project specific
Scope	Relates to all products that will ever be created by a process	Relates to specific product
Activities	▪ Process Definition and Implementation ▪ Audits ▪ Training	▪ Reviews ▪ Testing

The objectives of SQA activities

- **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.

Referrence

- Chapter 2:Daniel Galin. SOFTWARE QUALITY ASSURANCE From theory to implementation. Pearson Education Limited,2004.
- Therac-25:An Engineering Disaster Therac-25 Joanne Lim., October 1998