Software Quality Management

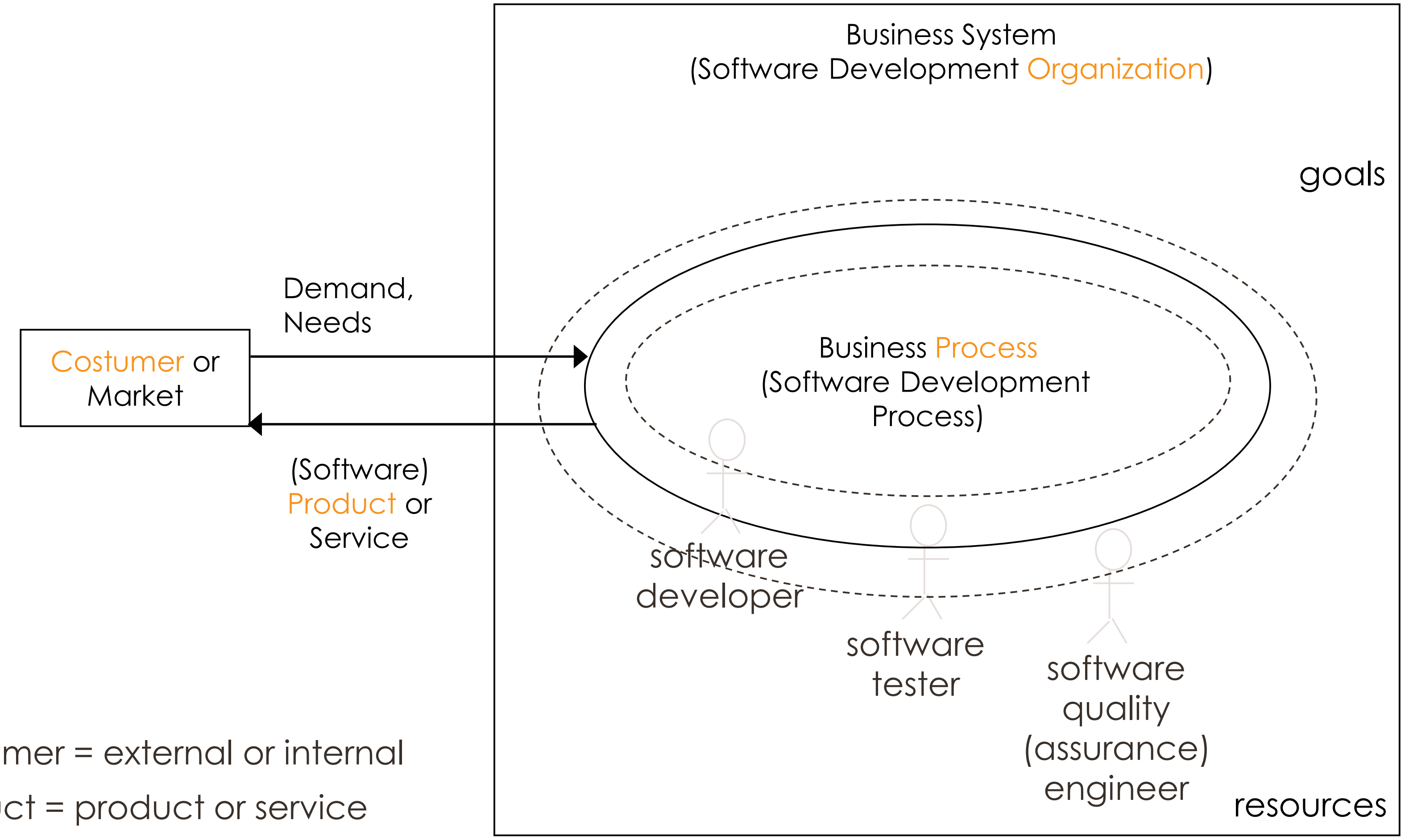
Software Quality

Lecturer: **Nguyễn Ngọc Tú**
Email: *Tu.NguyenNgoc@hoasen.edu.vn*
Web: *sites.google.com/site/QuanLyChatLuongPhanMem/*
Face Group: *www.facebook.com/groups/SoftwareQualityManagement/*
#AdTekDev *#ICoTek* #VNASQ #VNSQA #VNSoftwareTesting

Outline

- Perspectives & Expectations
- Definition of Software Quality
- SQM Activities
- Cost of quality
- Quality Parameters
- Quality Frameworks & ISO-9126
- Quality Assurance
- Initiatives to achieve *Software Quality*

Perspectives & Expectations



Customer = external or internal

Product = product or service

Test = test and review

Development = development and maintenance

Definition of Software Quality

IEEE

1. The degree to which a system, component, or process meets specified requirements.

2. The degree to which a system, component, or process meets customer or user needs or expectations.

Definition of Software Quality *Assurance*

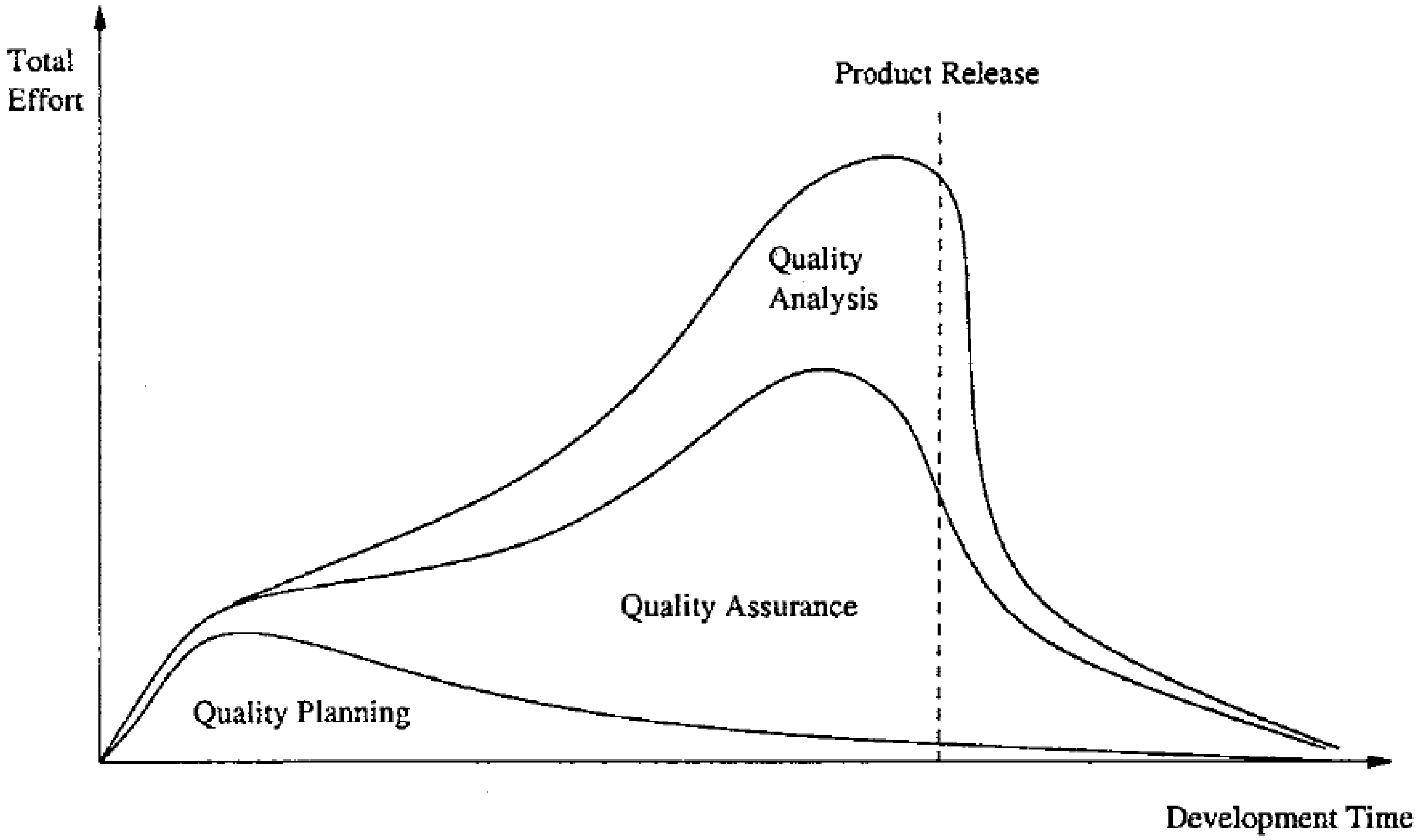
1. A planned and systematic pattern of all actions necessary to provide adequate confidence that an item or product conforms to established technical requirements.

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

SQA – SQC

- *Quality Control*
  - activities designed to **evaluate** the quality of a product
- *Quality Assurance*
  - activities to **prevent and correct** errors

SQM Activities



Cost of quality

- Quality cost includes:
  - prevention cost
    - quality planning
    - formal technical reviews
    - testing equipment
    - training
  - appraisal cost
    - in-process and inter-process inspection
    - equipment calibration and maintenance
    - testing

Cost of quality

- failure cost:
  - internal failure cost:
    - rework, repair, and failure mode analysis
  - external failure cost:
    - complaint resolution
    - product return and replacement
    - help line support
    - warranty work

Cost of quality

• *Prevention*	• *Appraisal*
• **Staff training** • **Requirements analysis & early prototyping** • **Fault-tolerant design** • **Defensive programming** • **Usability analysis** • **Clear specification** • **Accurate internal documentation** • **Pre-purchase evaluation of the reliability of development tools**	• **Design review** • **Code inspection** • **Glass box testing** • **Black box testing** • **Training testers** • **Beta testing** • **Usability testing** • **Pre-release out-of-box testing by customer service staff**
• *Internal Failure*	• *External Failure*
• **Bug fixes** • **Regression testing** • **Wasted in-house user time** • **Wasted tester time** • **Wasted writer time** • **Wasted marketer time** • **Wasted advertisements** • **Direct cost of late shipment** • **Opportunity cost of late shipment**	• **Lost sales and lost customer goodwill** • **Technical support calls** • **Writing answer books (for Support)** • **Investigating complaints** • **Supporting multiple versions in the field** • **Refunds, recalls, warranty, liability costs** • **Interim bug fix releases** • **Shipping updated product** • **PR to soften bad reviews** • **Discounts to resellers**

Quality Parameters

- IBM - CUPRIMDSO
  - Capability
  - Usability
  - Performance
  - Reliability
  - Installability
  - Maintainability
  - Documentation
  - Service
  - Overall
- Hewlett-Packard - FURPS
  - Functionality
  - Usability
  - Reliability
  - Performance
  - Serviceability

Quality Frameworks & ISO-9126

- Quality Improvement Paradigm (QIP)
  - Continuous improvement based on a set of evolving goals, and evaluation of these goals

  1 - Characterize the project

  2 - Set the goals

  3 - Choose appropriate process

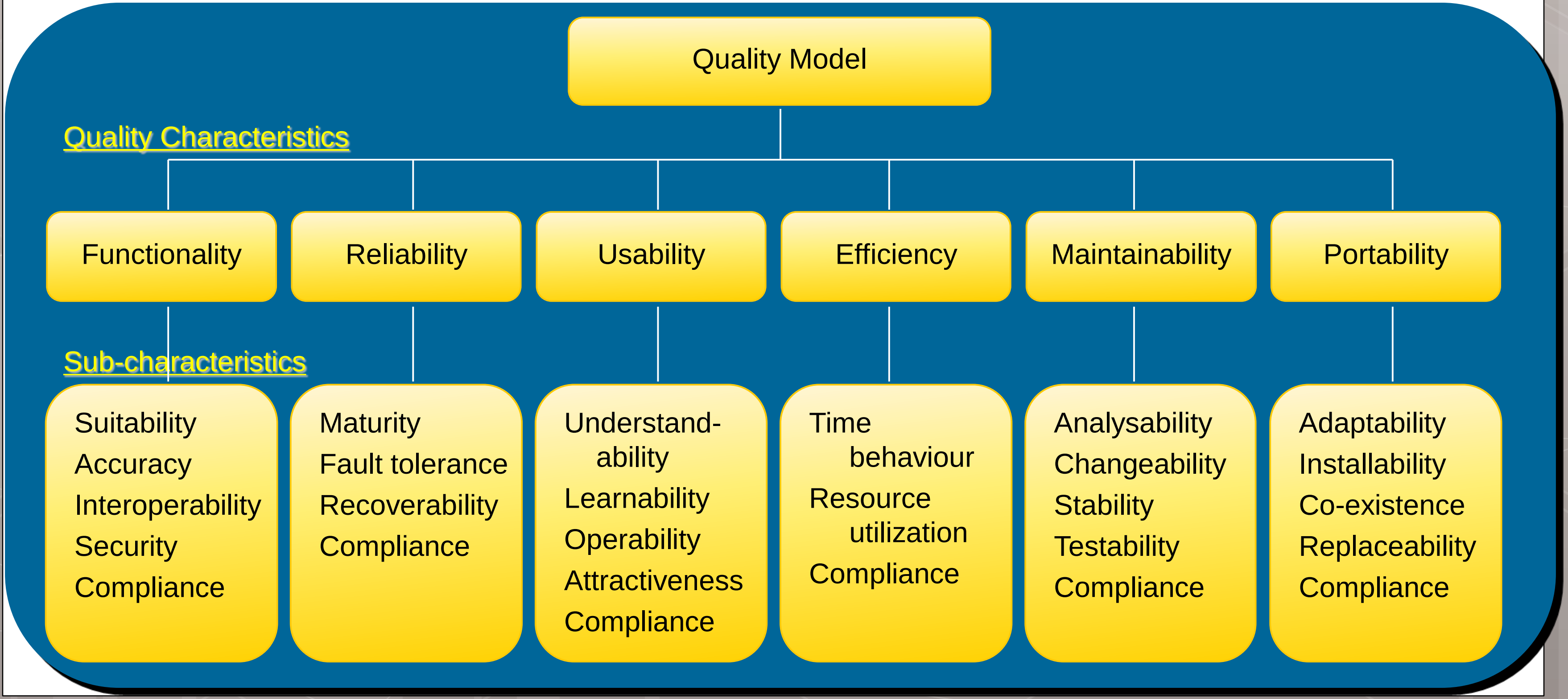
  4 - Execute process (and gather data)

  5 - Analyze data

  6 - Package the experience for reuse

Quality Frameworks & ISO-9126

- provides a ***hierarchical framework*** for quality definition, organized into quality characteristics and sub-characteristic


Quality Model
Quality Characteristics
Functionality
Reliability
Usability
Efficiency
Maintainability
Portability
Sub-characteristics
Suitability Accuracy Interoperability Security Compliance
Maturity Fault tolerance Recoverability Compliance
Understand-ability Learnability Operability Attractiveness Compliance
Time behaviour Resource utilization Compliance
Analysability Changeability Stability Testability Compliance
Adaptability Installability Co-existence Replaceability Compliance

ISO-9126: *Functionality*

[1] p18

- A set of attributes that bear on the existence of a set of functions and their specified properties.
- The functions are those that satisfy stated or implied needs
  - Suitability
  - Accuracy
  - Interoperability
  - Security

ISO-9126: *Reliability*

[1] p18

- A set of attributes that bear on the capability of software to maintain its level of performance under stated conditions for a stated period of time.
  - Maturity
  - Fault tolerance
  - Recoverability

ISO-9126: *Usability*

[1] p18

- A set of attributes that bear on the effort needed for use, and on the individual assessment of such use, by a stated or implied set of users.
  - Understandability
  - Learnability
  - Operability

ISO-9126: *Efficiency*

[1] p18

- A set of attributes that bear on the relationship between the level of performance of the software and the amount of resources used, under stated conditions
  - Time behavior
  - Resource behavior

ISO-9126: *Maintainability*

[1] p18

- A set of attributes that bear on the effort needed to make specified modifications.
  - Analyzability
  - Changeability
  - Stability
  - Testability

ISO-9126: *Portability*

[1] p19

- A set of attributes that bear on the ability of software to be transferred from one environment to another.
  - Adaptability
  - Installability
  - Conformance
  - Replaceability

Alternative frameworks

- Other quality frameworks/mega-models
  - McCall: factors, criteria, and metrics
  - Basili: GQM (goal-question-metric)
  - SEI/CMM: process focus/levels
  - Dromey: component reflects Q-attributes
  - Defect-based view: common in industry
    - cost of defect: by Boehm, NIST, etc.

Quality Assurance

Classification scheme

- Three generic categories
  - Defect **prevention** through error blocking or error source removal
  - Defect **reduction** through fault detection and removal
  - Defect **containment** through failure prevention and containment

Classification scheme – prevention

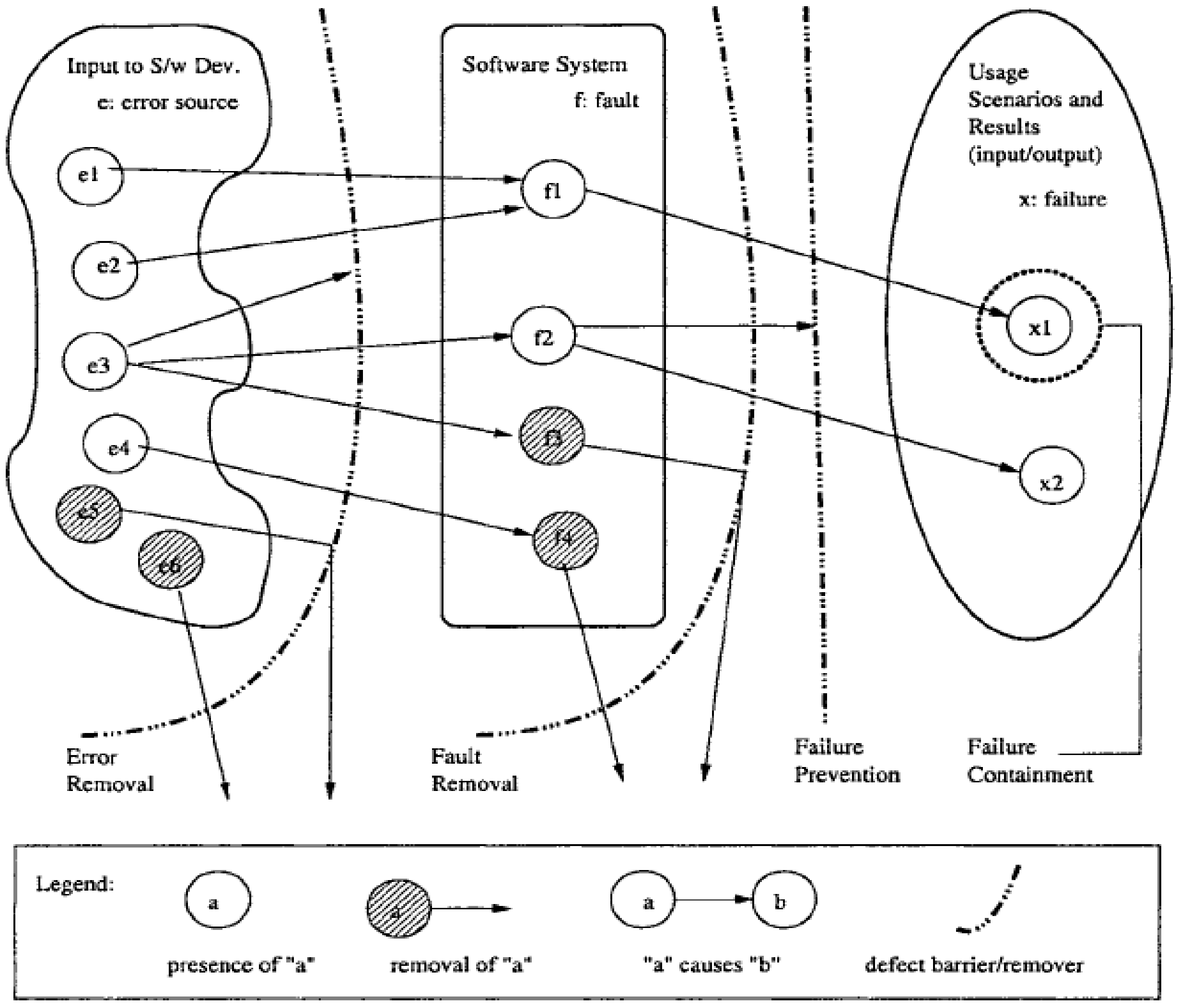
- *prevent certain types of faults from being injected into the software*
- Two generic ways
  - Eliminating certain error sources,
    - such as eliminating ambiguities or correcting human misconceptions, which are the root causes for the errors.
  - Fault prevention or blocking by directly correcting or blocking these missing or incorrect human actions.
    - This group of techniques breaks the causal relation between error sources and faults through the use of certain tools and technologies, enforcement of certain process and product standards, etc.

Classification scheme – reduction

- *Detect and remove certain faults once they have been injected into the software systems.*
- most traditional QA activities fall into this category
- *Ex.*
    - *Inspection directly detects and removes faults from the software code, design, etc*
    - *Testing removes faults based on related failure observations during program execution.*

Classification scheme – *containment*

- ***focus on the failures*** *by either containing them to local areas so that there are no global failures observable to users, or limiting the damage caused by software system failures*


Input to S/w Dev.
e: error source
e1
e2
e3
e4
e5
e6
Software System
f: fault
f1
f2
f3
f4
Usage Scenarios and Results (input/output)
x: failure
x1
x2
Error Removal
Fault Removal
Failure Prevention
Failure Containment
Legend:
a
presence of "a"
a
removal of "a"
a
b
"a" causes "b"
defect barrier/remover

Initiatives to achieve *Software Quality*

- Plan Software Quality Activities
- Define the Metrics
- Implement activities
- Monitor success
- ***Identify Quality improvements needed***

Q/A ?!