



Software Quality – A Quick Peek



Sridhar Chimalakonda



A generic tip

Think *broad*

but do as *specific [precise]* as possible!!!

Which laptop will you buy?

Memory Size

- ☐ Up to 2 GB (135)
- ☐ 4 GB (586)
- ☐ 6 GB (12)
- ☐ 8 GB (158)
- ☐ 12 GB (6)
- ☐ 16 GB & more (35)

Cash On Delivery (What's this?)

- ☐ Eligible for Cash On Delivery (657)

Notebook Type

- ☐ Chromebook (1)
- ☐ Convertible 2 in 1 (9)
- ☐ Notebook (435)
- ☐ Ultrabook (1)

Storage Type

- ☐ Hybrid Drive (19)
- ☐ Mechanical Hard Drive (377)
- ☐ Solid State Drive (33)

Laptop Features

- ☐ Anti Reflective (199)
- ☐ Touchscreen (22)

HDD Size

- ☐ Up to 159 GB (49)
- ☐ 250 - 499 GB (42)
- ☐ 500 - 999 GB (428)
- ☐ 1TB & More (68)

Operating System

- ☐ Windows 10 (205)
- ☐ Windows 8.1 (163)
- ☐ Windows 8 (61)
- ☐ Mac OS (20)

☐ Over ₹50,000

★ Refine by brand/price

FEATURED CATEGORIES

LAPTOPS BY OS



- Windows 10 >
- Mac OS >
- DOS >
- Linux >
- Windows 8 >

LAPTOPS BY USE



- Everyday Use >
- Entertainment >
- Travel & portability >
- Multi-tasking >
- Gaming >

LAPTOPS BY TYPE



- 2-in-1 Detachables >
- 2-in-1 Convertibles >
- Touchscreen laptops >
- Premium laptops >
- Chromebook >

Most Helpful Reviews on Popular Laptops



1 Best Selling

iBall Excelance CompBook 11.6-inch

Most helpful review

4.0 out of 5 stars Pre research required before

Most rec

5.0 out of

Which mobile/speakers will you buy?

- China, Micromax, Lenevo, Xiaomi, Samsung, iPhone,



Beats



Bose



LG

SONY

Sony



Skullcandy



JBL



JayBird



JVC

Which plane do you fly?

Leading Brands in the Aircraft Manufacturing Industry

Large Aircraft



GULFSTREAM



Medium Aircraft



BOMBARDIER
the evolution of mobility

Small Aircraft



BOMBARDIER
the evolution of mobility



Warranty



BRANDS WE CARRY



What looks outside [Customer?] and What happens inside? [Engineering?]



IS

STARTING AT \$37,325*
2.0L I-4 OR 3.5L V6

[BUILD](#) | [CURRENT OFFERS](#)



ES

STARTING AT \$38,100*
268-HP 3.5L V6

[BUILD](#) | [CURRENT OFFERS](#)



ES HYBRID

STARTING AT \$41,020*
40-MPG COMBINED RATING*

[BUILD](#) | [CURRENT OFFERS](#)



GS

STARTING AT \$45,615*
2.0L I-4 OR 3.5L V6

[BUILD](#) | [CURRENT OFFERS](#)



GS HYBRID

STARTING AT \$63,080*
338 TOTAL SYSTEM HP*

[BUILD](#) | [CURRENT OFFERS](#)



GS F

STARTING AT \$84,440*
467-HP* 5.0L V8

[BUILD](#) | [CURRENT OFFERS](#)



LS

STARTING AT \$72,520*
386-HP* 4.6L V8 (RWD)

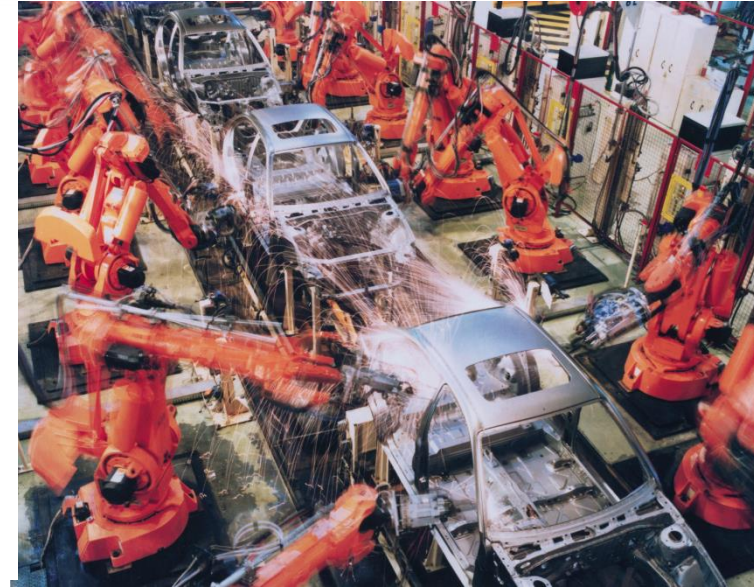
[BUILD](#) | [CURRENT OFFERS](#)



LS HYBRID

STARTING AT \$120,440*
438 TOTAL SYSTEM HP*

[BUILD](#) | [CURRENT OFFERS](#)



Slide Dec 2:

2 Primary differences between Verification and Validation give example for each

Explain the Kano Model of qualities for a software chat application

Draw table to list Garvins 5 perspectives of Software Quality, explain each through example of banking software.
Write an example for each for bookmyshow platform.

What are the 3 broad ways of accessing software quality , write advantages and disadvantages of each approach.

1) Formal Methods

2) Quality Standards

3) Testing

Explain 3 software qualities and how to measure them.(metric)

1) Maintainability-> The property of the code to be modified / customised with relative ease (ease of fixing bugs / adding new features)

Metrics: Quantitative-> # lines of code, Code Complexity

Qualitatively-> Readability , extent of Monolithic Practices , Code Smells etc

2) Performance -> The property of the code to use minimal hardware resources while maintaining low response time , quick functionality + Maintaining overall efficiency.

Metrics: Time Complexity of Code, RAM usage , CPU spikes per minute, Space Complexity

3) Reliability -> How much the software can withstand erroneous user inputs and network disruptions , underlying OS issues etc

metrics: Mean Recovery Time (MTTR), Mean Time Between Failure (MTBF)

What is Quality anyway?

Good is not good, where better is expected.

Thomas Fuller, Church History of Britain, XI.3

Explain various kind of classification and categories of software qualities?

1) Internal VS External

-> External -> visible to users

-> Internal -> concern the developers (affects External Qual)

2) Product VS Process :

-> Our goal is to develop software products

-> Process is the way to do it (affects Product Qual)

Crosby - “Quality is free: the art of making quality certain”

“The first erroneous assumption is that quality means goodness, or luxury or shininess. The word “quality” is often used to signify the relative worth of something in such phrases as “good quality”, “bad quality” and “quality of life” - which means different things to each and every person. As follows quality must be defined as “conformance to requirements” if we are to manage it. Consequently, the nonconformance detected is the absence of quality, quality problems become nonconformance problems, and quality becomes definable.”

Deming – Out of the Crisis

The difficulty in defining quality is to translate future needs of the user into measurable characteristics, so that a product can be designed and turned out to give satisfaction at a price that the user will pay. This is not easy, and as soon as one feels fairly successful in the endeavor, he finds that the needs of the consumer have changed, competitors have moved in etc

- “Quality is neither mind nor matter, but a third entity independent of the two . . . even though Quality cannot be defined, you know what it is.” (R. M. Pirsig, *Zen and the Art of Motorcycle Maintenance*, pp. 185, 213)
- “... a condition of excellence implying fine quality as distinct from poor quality. . . . Quality is achieving or reaching for the highest standard as against being satisfied with the sloppy or fraudulent.” (B. W. Tuchman, “The Decline of Quality,” *New York Times Magazine*, 2 November 1980, p. 38)

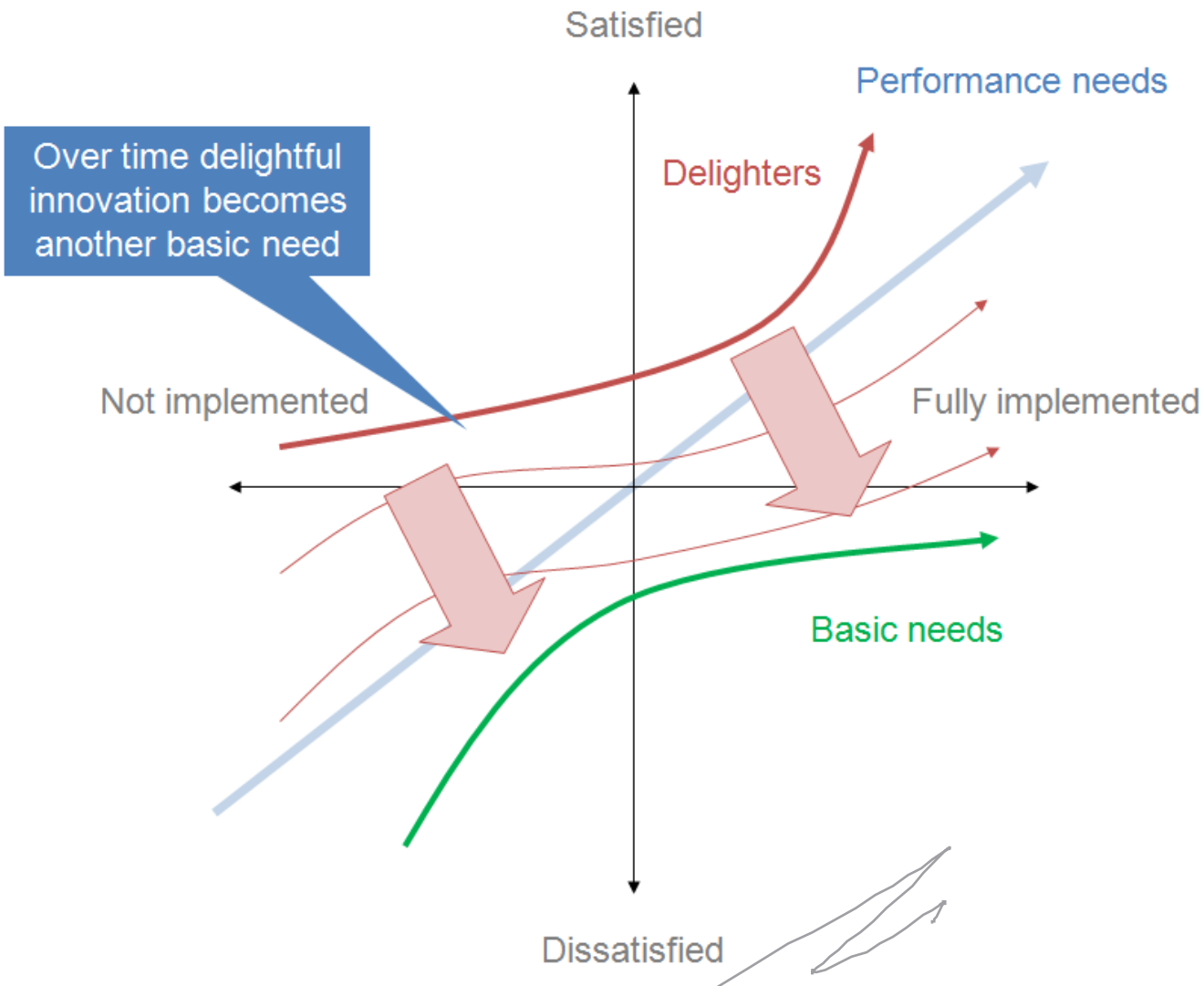
- “Differences in quality amount to differences in the quantity of some desired ingredient or attribute.” (L. Abbott, *Quality and Competition*, pp. 126–127)
- “Quality refers to the amounts of the unpriced attributes contained in each unit of the priced attribute.” (K. B. Laffler, “Ambiguous Changes in Product Quality,” *American Economic Review*, December 1982, p. 956)

- “Quality consists of the capacity to satisfy wants . . .” (C. D. Edwards, “The Meaning of Quality,” *Quality Progress*, October 1968, p. 37)
- “Quality is the degree to which a specific product satisfies the wants of a specific consumer.” (H. L. Gilmore, “Product Conformance Cost,” *Quality Progress*, June 1974, p. 16)
- “Quality is any aspect of a product, including the services included in the contract of sales, which influences the demand curve.” (R. Dorfman and P. O. Steiner, “Optimal Advertising and Optimal Quality,” *American Economic Review*, December 1954, p. 831)

- “Quality [means] conformance to requirements.” (P. B. Crosby, *Quality Is Free*, p. 15)
- “Quality is the degree to which a specific product conforms to a design or specification.” (Gilmore, June 1974, p. 16)

- “Quality is the degree of excellence at an acceptable price and the control of variability at an acceptable cost.” (R. A. Broh, *Managing Quality for Higher Profits*, 1982, p. 3)
- “Quality means best for certain customer conditions. These conditions are (a) the actual use and (b) the selling price of the product.” (A. V. Feigenbaum, *Total Quality Control*, p. 1)

Kano Model



1) Indifferents -> these are the properties / features which impact the user minimally
there presence or absence is of little value to the end user ex (blue bell icon changed to red bell in the website)

- *Dissatisfiers* are properties of the system that are self-evident and taken for granted (subconscious knowledge).
- *Satisfiers* are explicitly demanded system properties (conscious knowledge).
- *Delighters* are system properties that the stakeholder does not know or expect and discovers only while using the system—a pleasant and useful surprise (unconscious knowledge). ...

Examples

Deming's Philosophy

- Process Step - Product Quality Co-relation

Explain the Kano Model of qualities for a software chat application

- 1) Dissatisfiers -> * message being sent to the person even if he is offline(it reaches him when he becomes online) * Safe User Authentication (unhackable) * image video audio sending capabilities * minimal cost
- 2) Satisfiers -> * Provision for internet phone call * fast sending of messages * fast download of messages * minimal downtime of server
- 3) Delighters -> * Custom emojis * unlimited file transfer (free of cost no money for storage) * No 3rd party sharing of messages (ad revenue 0) (like telegram)

How customers make payments?

- Which qualities to do they prefer?
 - Dissatisfiers, Satisfiers, Delighters
- How many bank transactions a minute, an hour, a day?
- The daily average volume is around 1.6 million with a peak volume of 2.6 million so far [NPCI]

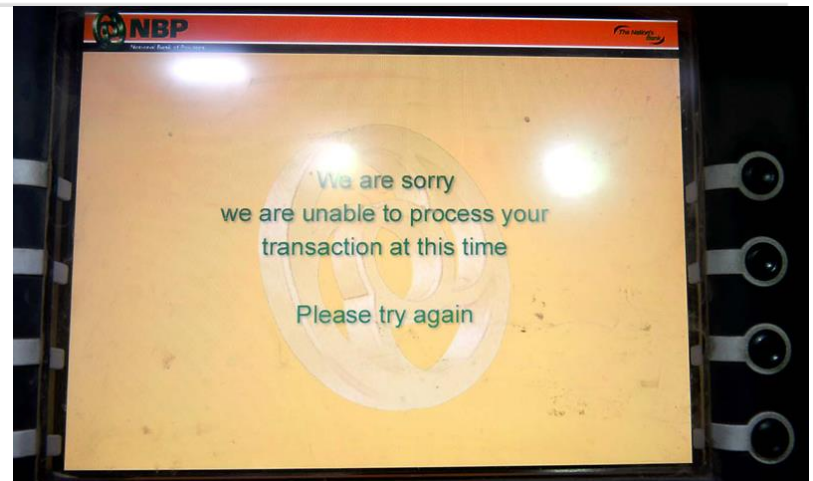


☰ SECTIONS

HOME | ALL INDIA

A Third Of India's ATMs Aren't Working. Banks Ordered To Quick Fix

All India | Written by Rahul Shrivastava | Updated: May 24, 2016 12:49 IST



Banker's Perspective

- Availability, Security, Reliability, Performance, Easy to use...
- *How to design these qualities?*

Inside...

```
function withdrawl{account,  
amount}  
{  
    If(account.balance>amount)  
    {  
        churnout(amount);  
        account_balance-=amount;  
    }  
  
    else  
        Msg:“Minimum balance  
        required”  
}
```

Some examples like this

```
function withdrawl{account,  
amount}  
{  
  
    //secure_check();  
  
    int  
    current_balance=account.b  
    alance;  
  
    If(current.balance>amount)  
        churnout(amount);  
  
    else  
        Msg:“Minimum balance  
        required”  
}
```

Some

A single line of code...

Impact on Quality

9 8  Member Banks

2 2 5 4 3 2  ATMs

1 2 2 0 7 6 3  enabled PoS Terminals

1 4 8  Member Banks

2 3 PPIs

1 3 7 0 9 8 7 5 9 MMID issued

5 1  Live Banks

5 6 5 Sub- Member & Others

1 1 0 9  Unique Banks

1 1 0 5  Credit members

8 5 2  Debit members

7 7 0  Live Banks

1 2 5  Live Entities

0 8 8 eKYC Live Entities

2 9 2 2 3 0 5 4 3 Aadhaar Numbers in NPCI Mapper

Let's turn to Software
Quality



Adobe Software License Agreement

English (North America) ▼

ADOBE

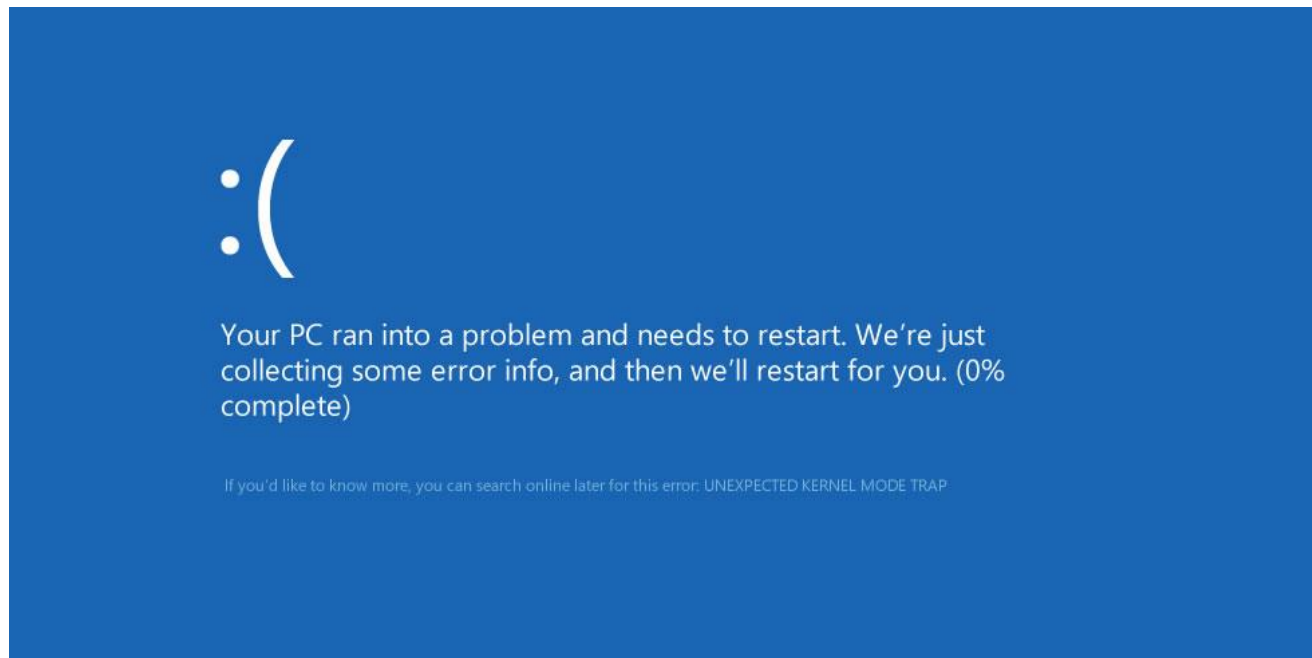
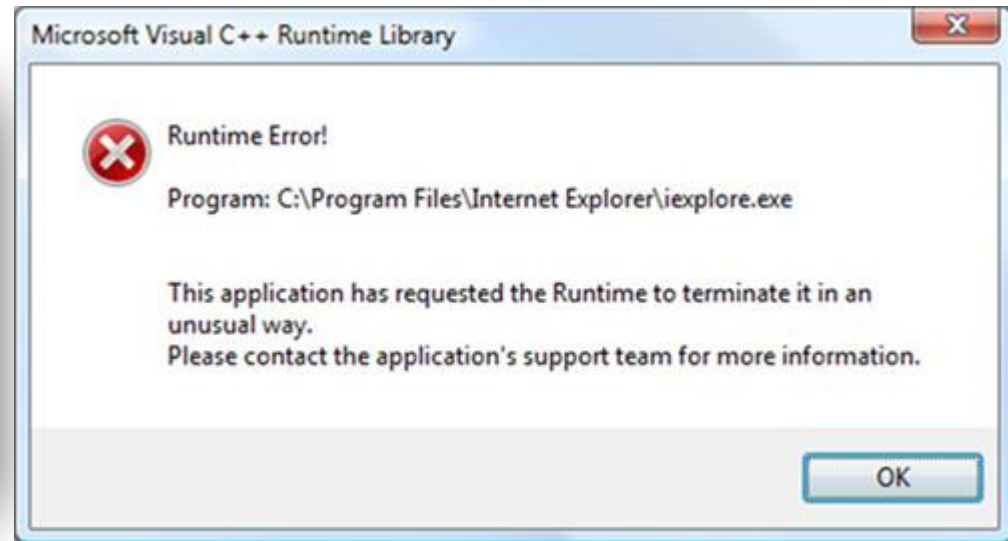
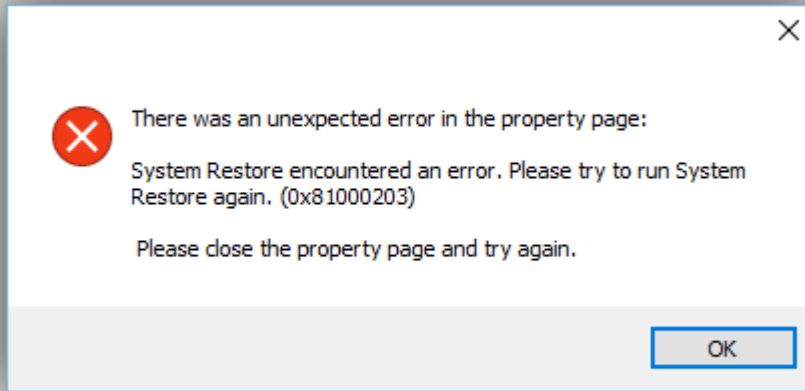
Software License Agreement

PLEASE READ THIS AGREEMENT CAREFULLY. BY COPYING, INSTALLING, OR USING ALL OR ANY PORTION OF THIS SOFTWARE, YOU (HEREINAFTER "CUSTOMER") ACCEPT ALL THE TERMS AND CONDITIONS OF THIS AGREEMENT, INCLUDING, WITHOUT LIMITATION, THE PROVISIONS ON LICENSE RESTRICTIONS IN SECTION 4, LIMITED WARRANTY IN SECTIONS 6 AND 7, LIMITATION OF LIABILITY IN SECTION 8, AND SPECIFIC PROVISIONS AND EXCEPTIONS IN SECTION 16. CUSTOMER AGREES THAT THIS AGREEMENT IS LIKE ANY WRITTEN NEGOTIATED AGREEMENT SIGNED BY CUSTOMER. THIS AGREEMENT IS ENFORCEABLE AGAINST CUSTOMER. IF CUSTOMER DOES NOT AGREE TO THE TERMS OF THIS AGREEMENT, CUSTOMER MAY NOT USE THE SOFTWARE.

Customer may have another written agreement directly with Adobe (e.g., a volume license agreement) that supplements or supersedes all or portions of this agreement. The Software is LICENSED, NOT SOLD, only in accordance with the terms of this agreement. Use of some Adobe and some non-Adobe materials and services included in or accessed through the Software may be subject to additional terms and conditions. Notices about non-Adobe materials are available at <http://www.adobe.com/go/thirdparty>.

Back

Accept



Why software quality is challenging?

- No physical artifacts
- Lack of clarity
- Mind boggling complexity
- Drastic technology changes
- Failures often but not tolerable
- Change is expected rapidly
-
- Explicit Versus Implicit
- Tangible Versus Intangible
- Manageable Complexity Versus Unmanageable Complexity
- Changeable Environment Versus
- Unchangeable Environment
- No Major Changes Versus Major Changes

What is Software Quality anyway?



Draw table to list Garvin's 5 perspectives of Software Quality, explain each through example of banking software.

Garvin's Five Perspectives

- Transcendental view.
 - Quality is something we can recognize but not define.
- User view.
 - Quality is fitness for purpose.
- Manufacturing view
 - Quality is conformance to specification
- Product view
 - Quality is tied to inherent product characteristics
- Value-based view.
 - Quality depends on the amount the customer is willing to pay for it.

Three Key Perspectives of Software Quality

- Customer Perspective

User view + Value Based

- Technology/Engineering Perspective

Manufacturing View

- Management Perspective

Product View

Explain various kind of classification and categories of software qualities? Write an example for each for bookmyshow platform.

How to Design Quality Software?

Quality of Requirements

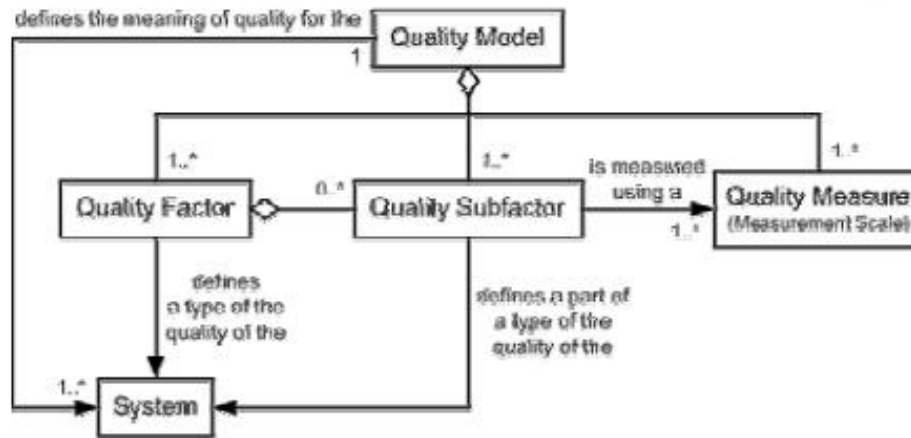


Figure 1: Quality Model

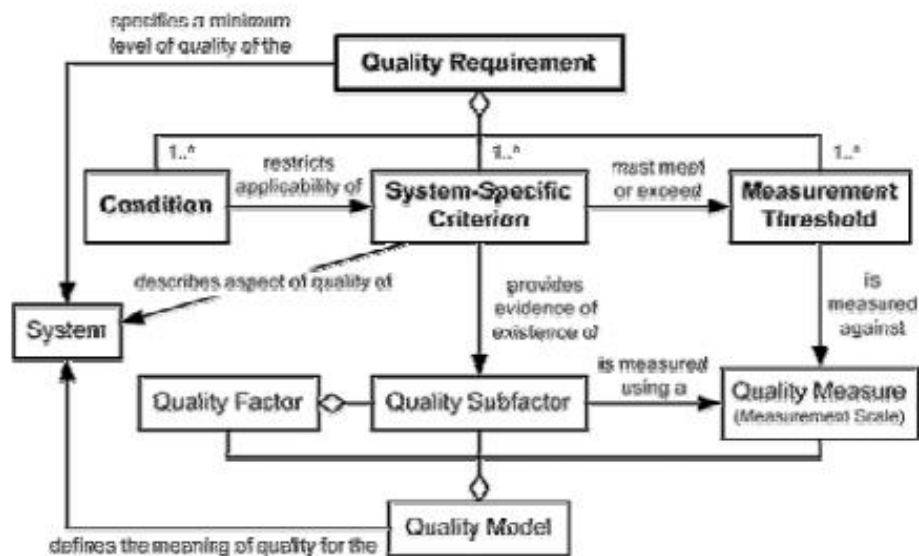


Figure 2: Quality Requirements

Architecture Quality

- Architectural Patterns

- Reference Architecture
- MVC
- Distributed
- Event-driven
- Batch
- Pipes and filters
- Repository-centric
- Blackboard
- Rule-based

- Design Patterns

- Distributed
- Event-driven
- Frame-based
- Batch
- Pipes and filters
- Repository-centric
- Blackboard
- Rule-based

Design for Software Quality

- **Creational patterns** focus on the “creation, composition, and representation of objects, e.g.,
 - **Singleton pattern**: Control the creation of instances to just one
 - **Abstract factory pattern**: centralize decision of what **factory** to instantiate
 - **Factory method pattern**: centralize creation of an object of a specific type choosing one of several implementations
- **Structural patterns** focus on problems and solutions associated with how classes and objects are organized and integrated to build a larger structure, e.g.,
 - **Adapter pattern**: 'adapts' one interface for a class into one that a client expects
 - **Aggregate pattern**: a version of the **Composite pattern** with methods for aggregation of children
- **Behavioral patterns** address problems associated with the assignment of responsibility between objects and the manner in which communication is effected between objects, e.g.,
 - **Chain of responsibility pattern**: Command objects are handled or passed on to other objects by logic-containing processing objects
 - **Command pattern**: Command objects encapsulate an action and its parameters
 - **Observer pattern**: Enable loose coupling between publishers and subscribers

Principles of Good Design

- Divide and Conquer
- Increase cohesion (keep related stuff together)
- Decrease coupling (minimize dependencies between modules)
- Increase the level of abstraction wherever possible
 - When two modules interact, create abstract interfaces so that modules don't have to know specific low-level details about other modules
- Design for Quality of service (Testability, Flexibility, Modifiability, etc.)
- Design by Contract

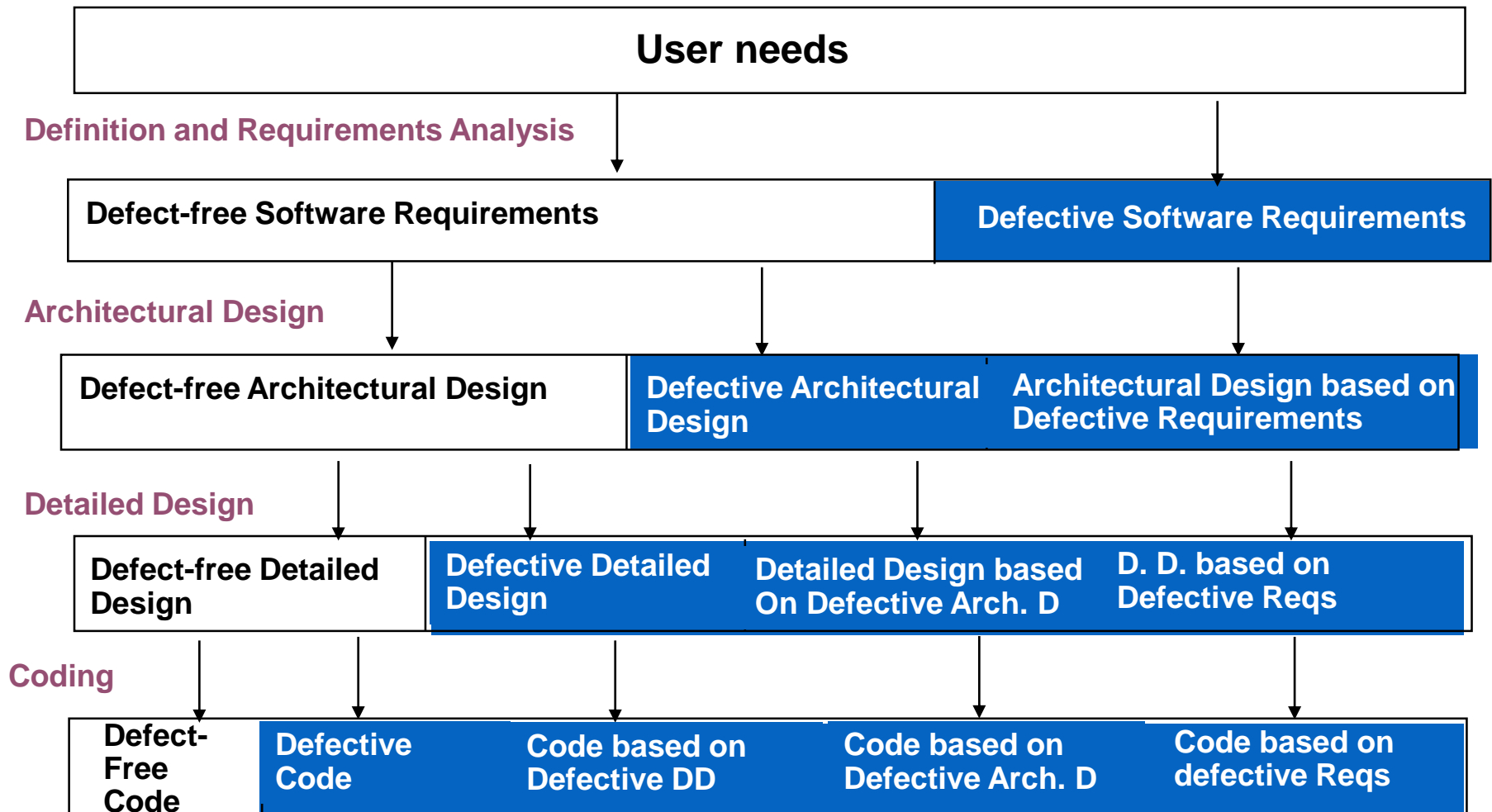
Code Quality

- *External* quality mean all the properties of the software as a product that users can experience and enjoy :conformity to their expectations (and evolution thereof)
- reliability
- accuracy
- ease of use and comfort (including response delay)
- robustness (or adaptability to some unforeseen condition of use)
- openness (or adaptability to future extensions or evolutions)
- *Internal* quality mean all the properties of the software *as seen by the developers* that are desirable in order to facilitate the process of creating a good product
- code do not suffer from duplication
- cohesion : each [module|class|routine] does one thing and does it well
- low coupling : minimal interdependencies and interrelation between objects
- simplicity
- generality : the problem domain bounds are known and stated
- clarity : the code enjoys a good documentation level

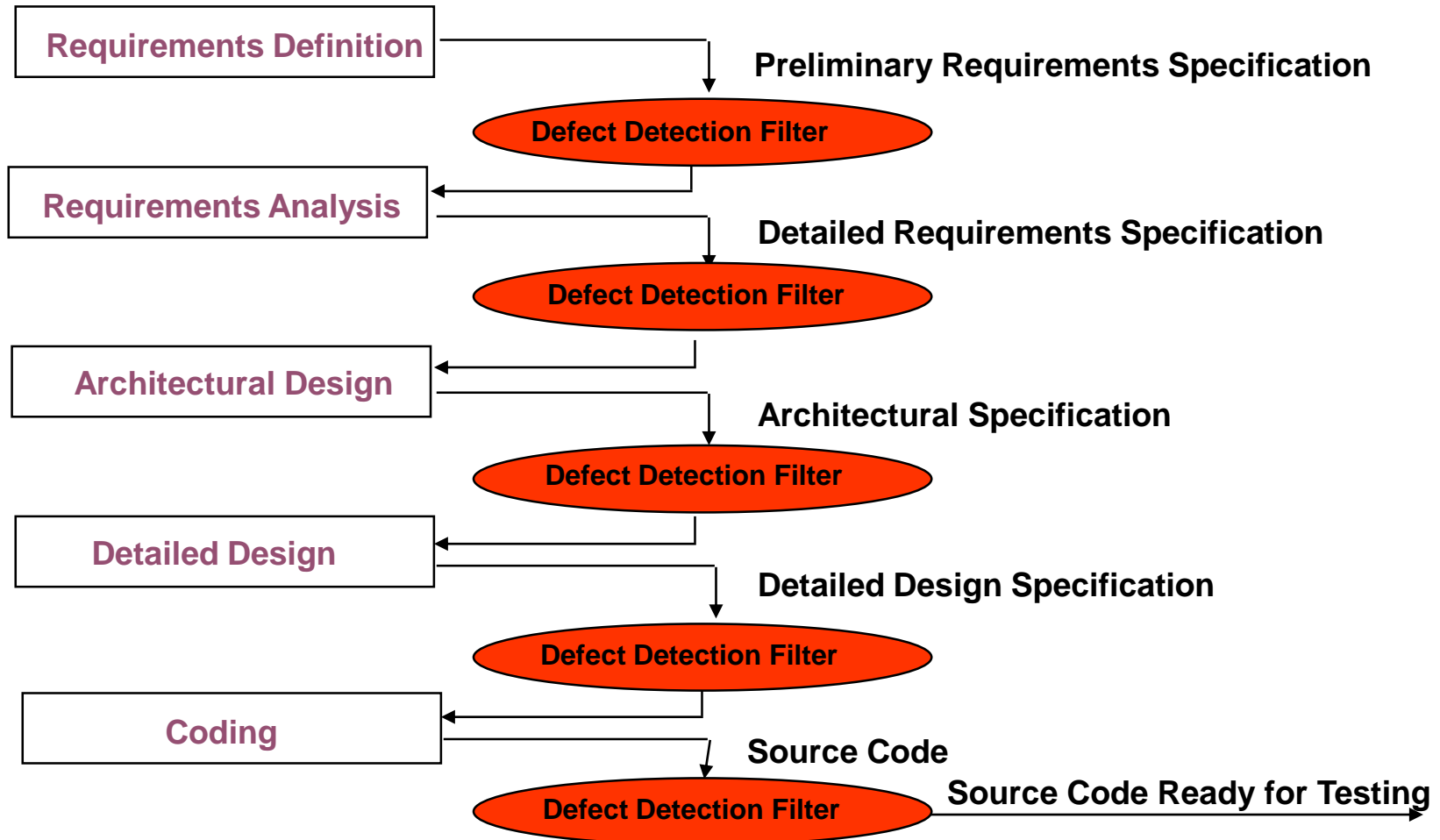
Code Smells Within Classes

- **Conditional Complexity**
 - large conditional logic blocks, particularly blocks that tend to grow larger or change significantly over time.
 - Consider alternative object-oriented approaches such as decorator, strategy, or state.
- **Combinatorial Explosion**
 - Lots of code that does *almost* the same thing.. but with tiny variations in data or behavior.
 - This can be difficult to refactor-- perhaps using generics or an interpreter?
- **Large Class**
 - Large classes, like long methods, are difficult to read, understand, and troubleshoot.
 - Large class can be restructured or broken into smaller

Error Introduction & Propagation



Multiple Stages of Defect Removal



Software

Its Nature and Qualities

Outline

- Software engineering (SE) is an intellectual activity and thus human-intensive
- Software is built to meet a certain functional goal and satisfy certain qualities
- Software processes also must meet certain qualities
- Software qualities are sometimes referred to as “ilities”

Software product

- Different from traditional types of products
 - intangible
 - difficult to describe and evaluate
 - malleable
 - human intensive
 - involves only trivial “manufacturing” process

Classification of sw qualities "ilities"

- Internal vs. external
 - External → visible to users
 - Internal → concern developers
- Product vs. process
 - Our goal is to develop software products
 - The process is how we do it
- Internal qualities affect external qualities
- Process quality affects product quality

Correctness

- Software is correct if it satisfies the functional requirements specifications
 - assuming that specification exists!
- If specifications are formal, since programs are formal objects, correctness can be defined formally
 - It can be proven as a theorem or disproved by counterexamples (testing)

The limits of correctness

- It is an absolute (yes/no) quality
 - there is no concept of “degree of correctness”
 - there is no concept of severity of deviation
- What if specifications are wrong?
 - (e.g., they derive from incorrect requirements or errors in domain knowledge)

Categories of Software Qualities:

Reliable Robots Use Interopen Performance Resuing Portable Mentainable Verified Understanding

Reliable -> ...

Robots -> Robustness

Use -> Usability

Interopen -> Interoperability

Performance ->....

Reusing -> ...

Portable ->...

Mentainable ->....

Verified ->....

Understandability ->...

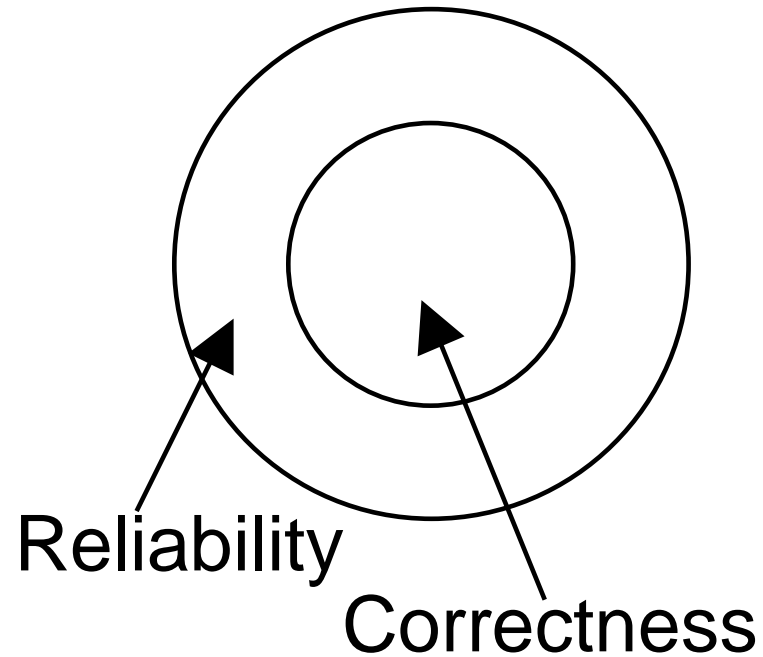
Reliability

- Reliability
 - informally, user can rely on it
 - can be defined mathematically as “probability of absence of failures for a certain time period”
 - if specs are correct, all correct software is reliable, but not vice-versa (in practice, however, specs can be incorrect ...)

Definitions plus 1 or 2 examples

Idealized situation

- Requirements are correct



Robustness

- Robustness
 - software behaves “reasonably” even in unforeseen circumstances (e.g., incorrect input, hardware failure)

Performance

- Efficient use of resources
 - memory, processing time, communication
- Can be verified
 - complexity analysis
 - performance evaluation (on a model, via simulation)
- Performance can affect scalability
 - a solution that works on a small local network may not work on a large intranet

Usability

- Expected users find the system easy to use
- Other term: user-friendliness
- Rather subjective, difficult to evaluate
- Affected mostly by *user interface*
 - e.g., visual vs. textual

Verifiability

- How easy it is to verify properties
 - mostly an internal quality
 - can be external as well (e.g., security critical application)

Maintainability

- Maintainability: ease of maintenance
- Maintenance: changes after release
- Maintenance costs exceed 60% of total cost of software
- Three main categories of maintenance
 - **corrective**: removing residual errors (20%)
 - **adaptive**: adjusting to environment changes (20%)
 - **perfective**: quality improvements (>50%)

Categories with examples

Maintainability

- Can be decomposed as
 - Repairability
 - ability to correct defects in reasonable time
 - Evolvability
 - ability to adapt sw to environment changes and to improve it in reasonable time

Reusability

- Existing product (or components) used (with minor modifications) to build another product
 - (Similar to evolvability)
- Also applies to process
- Reuse of standard parts measure of maturity of the field

Portability

- Software can run on different hw platforms or sw environments ex cross platform app dev
- Remains relevant as new platforms and environments are introduced (e.g. digital assistants)
- Relevant when downloading software in a heterogeneous network environment

Understandability

- Ease of understanding software
- Program modification requires program understanding



Interoperability

- Ability of a system to coexist and cooperate with other systems
 - e.g., word processor and spreadsheet

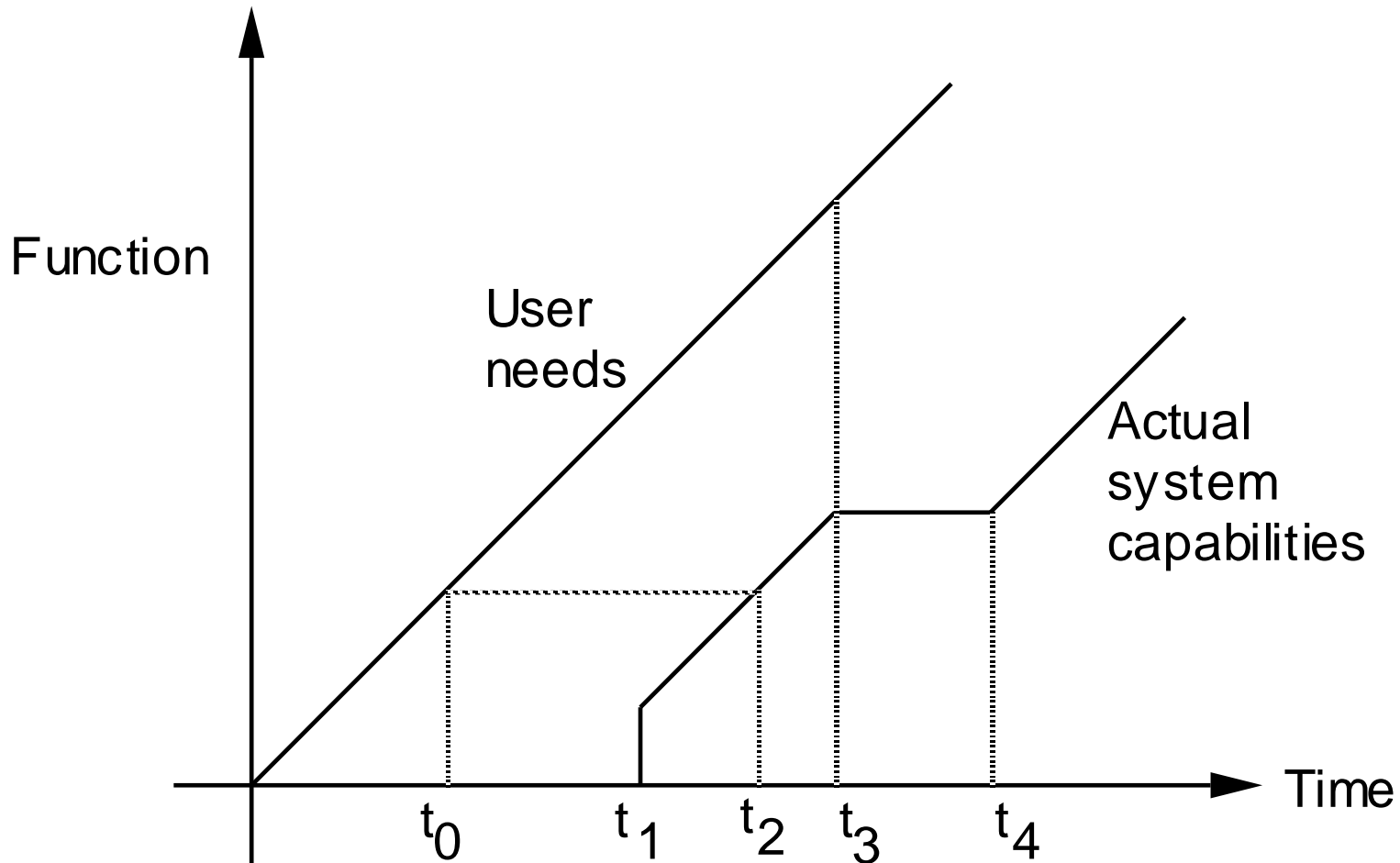
Typical process qualities

- Productivity
 - denotes its efficiency and performance
- Timeliness
 - ability to deliver a product on time
- Visibility
 - all of its steps and current status are documented clearly

Timeliness: issues

- Often the development process does not follow the evolution of user requirements
- A mismatch occurs between user requirements and status of the product

Timeliness: a visual description of the mismatch



Application-specific qualities

- E.g., information systems
 - Data integrity
 - Security
 - Data availability
 - Transaction performance.

Quality measurement

- Many qualities are subjective
- No standard metrics defined for most qualities

How can we assess quality?

There is little evidence that conformance to process standards guarantees good products.

Verification & Validation

- Formal Methods
- Testing
- Standards Compliance

- Quality Control [Product]
- Quality Assurance [Process]

Differences with examples



Verification - Are we building the product *right*?

Validation - Are we building the *right* product?


Formal Methods

- Proofs | Model Checking
- Predicate Logic

“Formal methods are mathematical approaches to software and system development which support the rigorous specification, design and verification of computer systems.”

alloy: a language & tool for relational models

Seven Myths of Formal Methods

1. Formal methods can guarantee that software is perfect.
2. Work by proving that programs are correct.
3. Only highly critical systems benefit from their use. 
4. They involve complex math.
5. They increase the cost of development.
6. They are incomprehensible to clients.
7. Nobody uses them for real projects.

Correctness

3.1 Correctness

Section 3.1 is further subdivided into 3.1.1 on Formal Correctness (or, how to prevent bugs from ever arising) and 3.1.2 on Debugging Techniques (or, how to catch them when you have got them.)

3.1.1 Formal Correctness

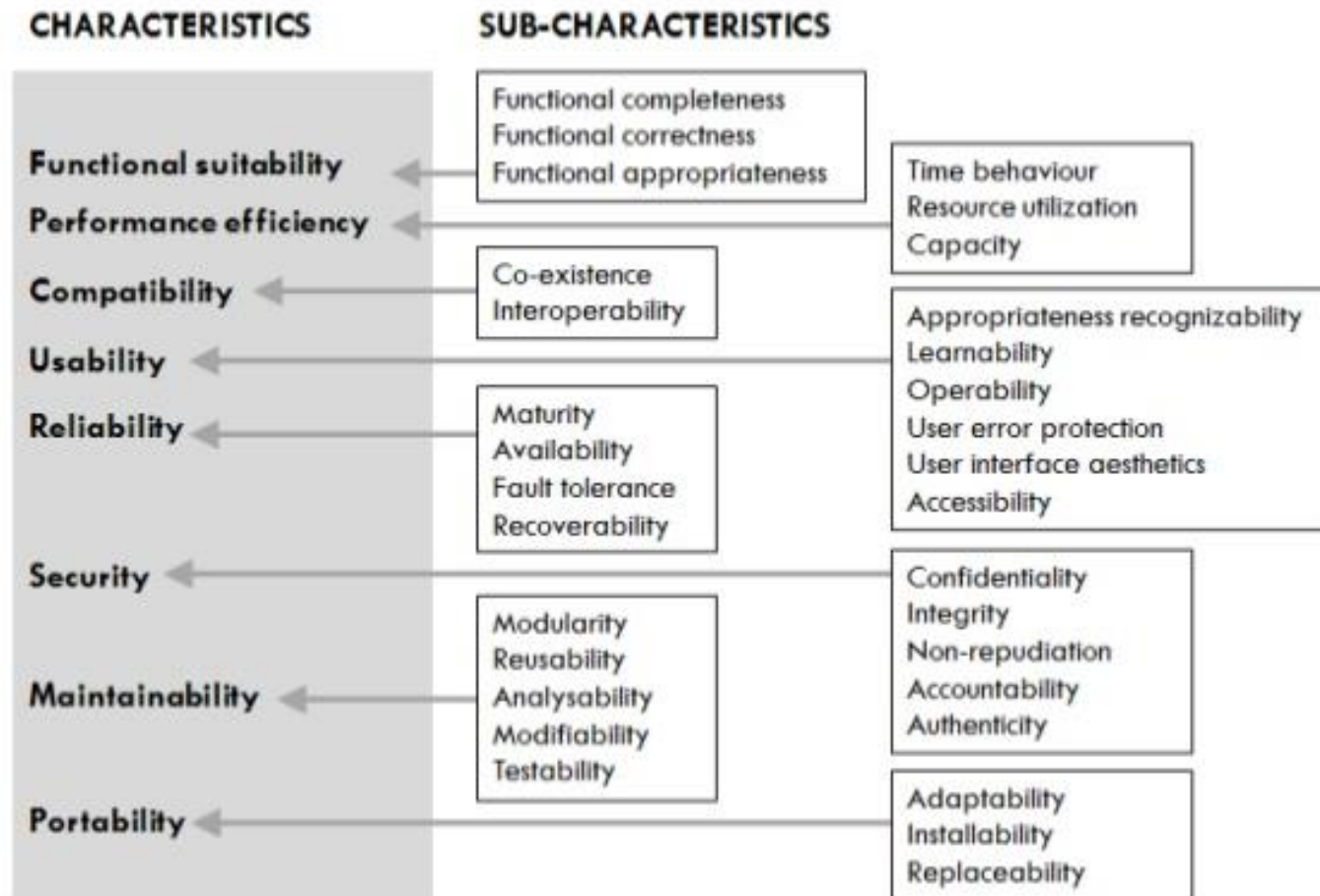
Wirth: (from “The programming language PASCAL and its design criteria”) “I believe that the art of computer programming is based on a relatively small number of fundamental concepts. It became clear during the development of PASCAL that most of these concepts are already present in mathematics in some form, usually surrounded by a terminology different from the one used by programmers. It is necessary to demonstrate this duality in the expression of concepts and to make the mathematician’s knowledge of handling structures available to programmers. This knowledge **21** will be particularly useful for the development of techniques for program verification, i.e. for proving the correctness of programs. In fact, the idea of program verification has influenced the design of PASCAL considerably. Several commonly used features of other languages have been deliberately omitted from PASCAL, because they appear to be too sophisticated for presently known methods of proof. It became clear that those features which are least tractable to proof techniques are exactly the ones which are most difficult to define rigorously and which in practice present the most dangerous sources of programming error.”

Hoare: One can construct convincing proofs quite readily of the ultimate futility of exhaustive testing of a program and even of testing by sampling. So how can one proceed? The role of testing, in theory, is to establish the base propositions of an inductive proof. You should convince yourself, or other people, as firmly as possible that if the program works a certain number of times on specified data, then it will always work on any data. This can be done by an inductive approach to the proof. Testing of the base cases could sometimes be automated. At present, this is mainly theory; note that the tests have to be designed at the same time as the program and the associated proof is a vital part of the documentation. This area of theoretical work seems to show a possibility of practical results, though proving correctness is a laborious and expensive process. Perhaps it is not a luxury for certain crucial areas of a program.

Perlis: Much of program complexity is spurious and a number of test cases properly studied will exhaust the testing problem. The problem is to isolate the right test cases, not to prove the algorithm, for that follows after the choice of the proper test cases.

Dijkstra: Testing shows the presence, not the absence of bugs.

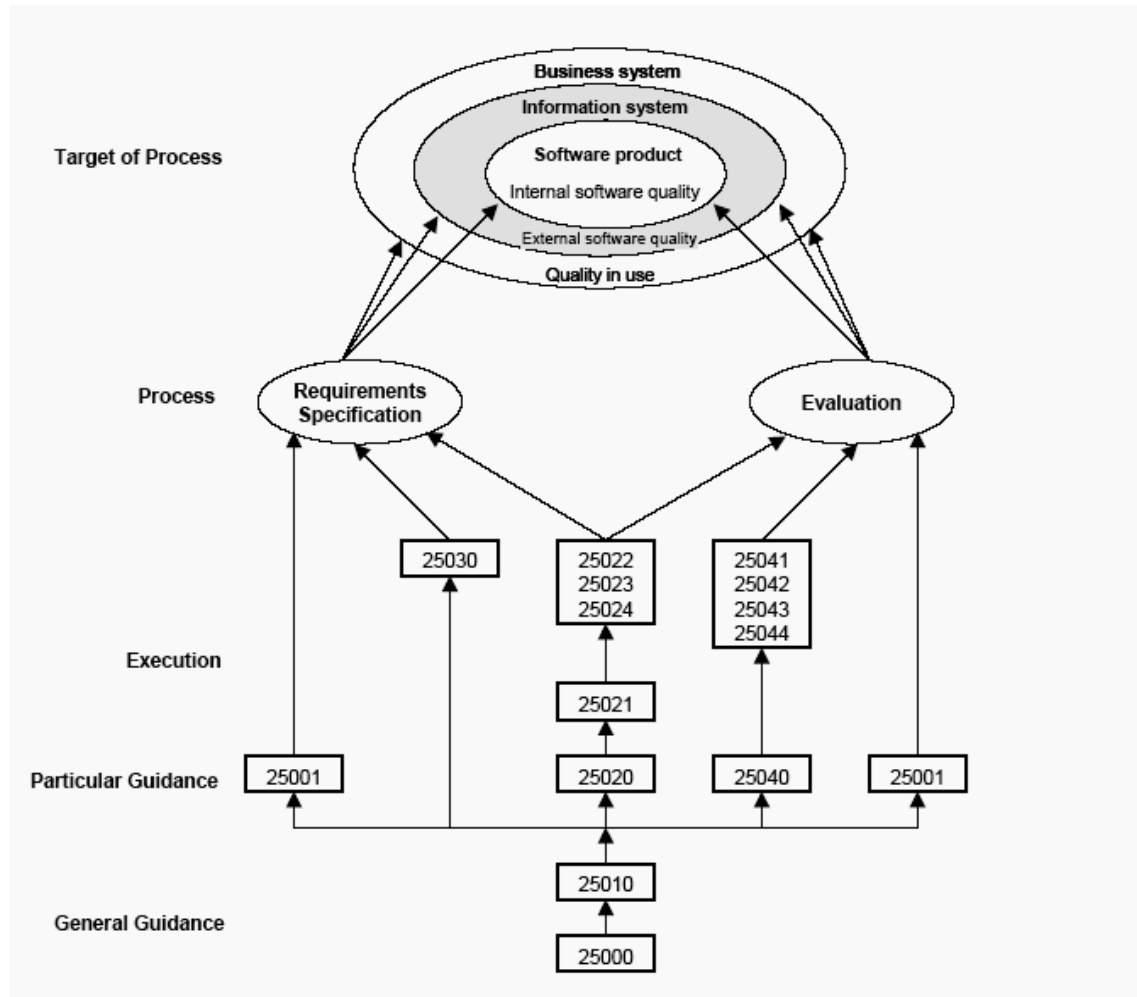
ISO/IEC 25010 Product Quality Model



Software Quality Engineering in practice (I)

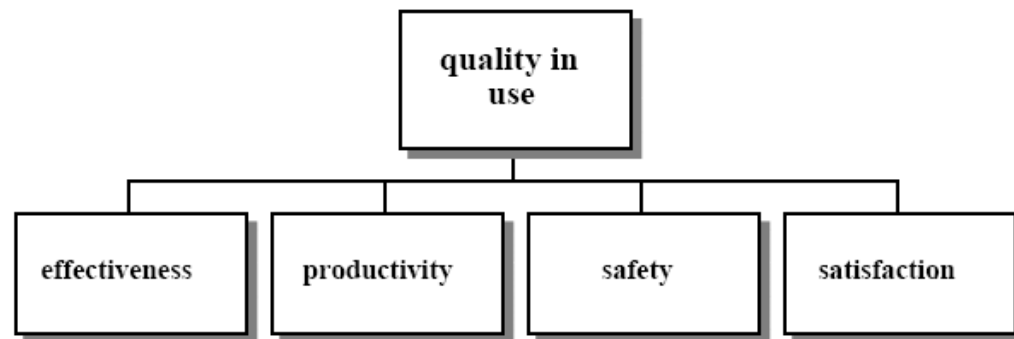
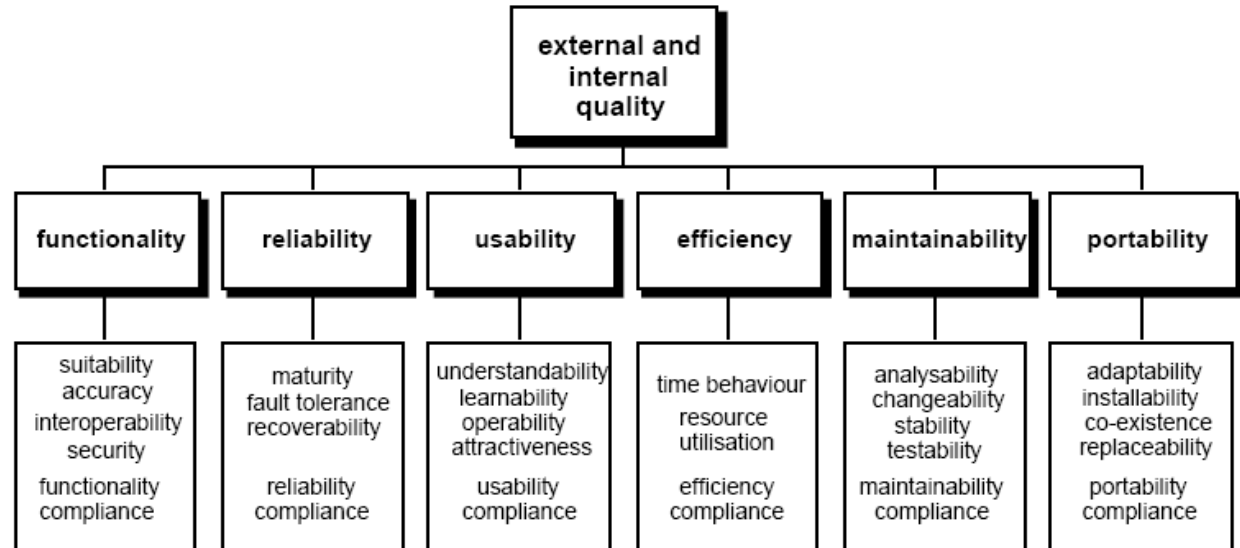
- Engineering issues to consider
 - Quality model
 - Quality requirements
 - Quality design
 - Quality measurement and evaluation
- Management issues to consider
 - Process model
 - Conflict management
 - Financial and maturity constraints

ISO/IEC 25000 SQuaRE or Software Product Quality Requirements and Evaluation



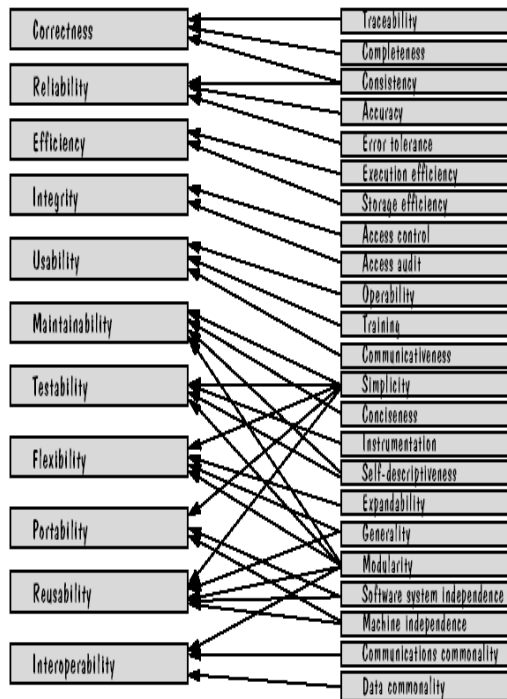
Software Quality Engineering in ISO

ISO/IEC 25000
SQuaRE
quality model

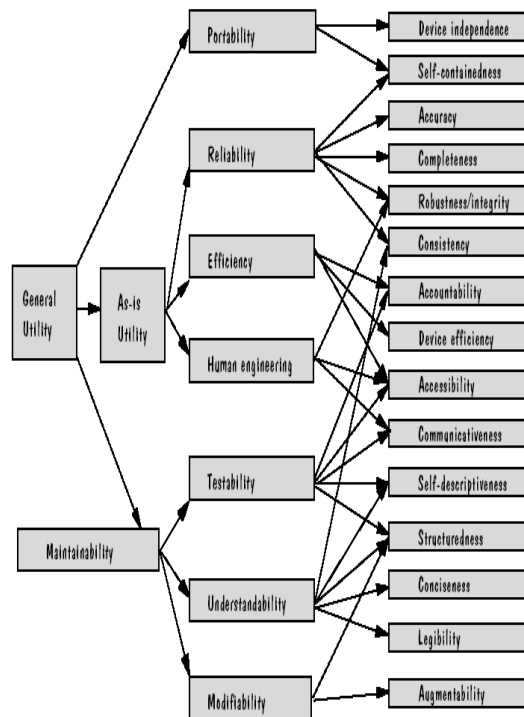


Software Quality Engineering in practice

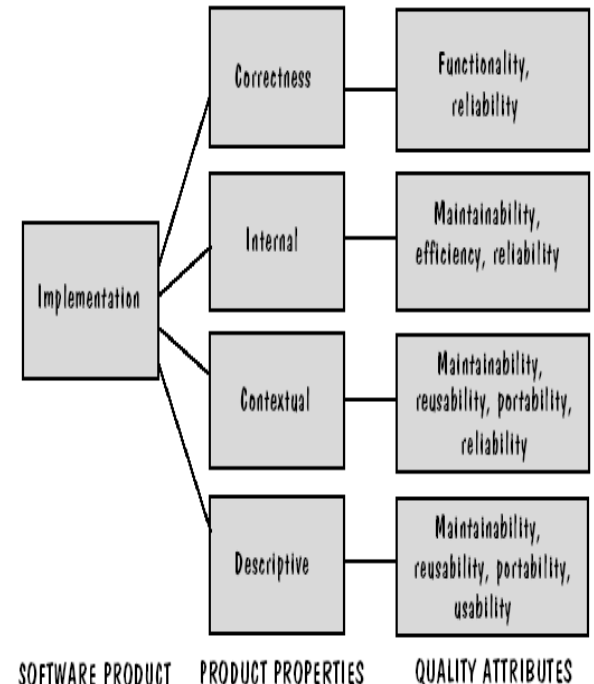
Quality model



McCall



B.Boehm



Dromey

Quality and Open Source

- How can we assess quality of open source software systems?

People and Quality

- What is the effect of people on software quality?

- 1) Functional Quality ->

- How well is the software meeting the needs of the people.

- Its about conforming to the functional & non Functional requirements + specification under given constraints + user budget, a software with less downtime , Fast error recovery time , Usability , Portability , Performance etc are all + points when accessing Functional Quality

- 2) Structural Quality ->

- How well the code itself is structured. It includes aspects of Maintainability , Readability , understandability, Verifiability , modifyability , Security (ex careful use of pointers (no dangling pointers) etc.

- 3) Process quality ->

- Quality of development process directly affects the code quality + Functional quality (adhock development of code vs Systematic Development ... the end product may have same functionality but there is large diff in structural quality + functional quality (reliability , usability etc). A systematic process attracts investors too!! Some aspects include:

- 1) Planning

- 2) Delivering on time

- 3) End User Testing (Beta relases etc)

- Additional measures for checking quality include seeing if its

Process Assessment and Improvement

- **Standard CMMI Assessment Method for Process Improvement (SCAMPI)** — provides a five step process assessment model that incorporates five phases: initiating, diagnosing, establishing, acting and learning. TCS -> CMMI level 5
- **CMM-Based Appraisal for Internal Process Improvement (CBA IPI)**—provides a diagnostic technique for assessing the relative maturity of a software organization
- **SPICE**—The **SPICE (ISO/IEC15504)** standard defines a set of requirements for software process assessment.
- **ISO 9001:2000 (ISO:25010) for Software**—a generic standard that applies to any organization that wants to improve the overall quality of the products, systems, or services that it provides

Software Professionals – What are their roles?



Lead Software Engineer



Software Architect



Testing Engineer



Quality Analyst



Software Engineer



Domain Expert



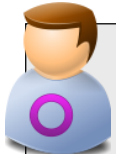
Software Developer



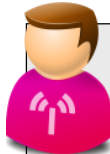
Requirements Engineer



Usability Expert



Support Engineer



Systems Analyst



Business Analyst

Who is a Software Engineer?

- Software Engineer
 - Programmer
 - Developer
 - Database Engineer
-
- There are 5 things a software engineer should be good at: **Programming, Design, Process, Communication & Team work** Managing

Programming Community ki
Management(process) me Desi Team.

Exercise

- Quality of open source systems
- How quality requirements are documented?
- How quality has been built into them?

Scenario based: list quality requirements

Book my show
Netflix
Flipkart/Amazon
Zomato

Take Away

- Quality is critical but comes with trade-offs
- Quality has multiple perspectives
- Design for Quality, not afterthought!