Chapter 3

Software Quality Factors

Chapter Three

Software Quality Factors

- The need for comprehensive Software Quality Requirements
- Classification of requirements into Software Quality Factors
- Product Operation Factors
- Product Revision Factors
- Product Transition Factors
- Alternative models of software quality factors

- “The new version of our loan contract software is really accurate. We have already processed 1200 customer requests, and checked each of the output contracts. There were no errors. **But** we did face a severe unexpected problem – training a new staff member to use this software takes about two weeks
- “Believe it or not, our software package ‘Blackboard’ for schoolteachers, launched just three months ago, is already installed in 187 schools. **But** we have been suddenly receiving daily complaints from the ‘Blackboard’ maintenance team. They claim that the lack of failure detection features in the software, in addition to the poor programmer’s manual, have caused them to invest more than the time estimated to deal with bugs or adding minor software changes that were agreed as part of purchasing contracts with clients.”
- “Just half a year ago we launched our new product – the radar detector. The firmware RD-8.1, embedded in this product, seems to be the cause for its success. **But**, when we began planning the development of a European version of the product, we found out that though the products will be almost similar, our software development department needs to develop new firmware; almost all the design and programming will be new.”

What is Common in above cases?

- All the software projects satisfactorily fulfilled the basic requirements for correct calculations (correct inventory figures, correct average class’s score, correct loan interest, etc.).
- All the software projects suffered from poor performance in important areas such as maintenance, reliability, software reuse, or training.
- The cause for the poor performance of the developed software projects in these areas was the lack of predefined requirements to cover these important aspects of the software’s functionality

The Requirements Document

- Requirement Documentation (Specification) is one of the **most important elements** for achieving software quality
- Need to explore what constitutes a **good** software requirements document.
- Some SQA Models suggest 11-15 factors categorized; some fewer; some more
- **The need for comprehensive software quality requirements is pervasive in numerous case studies (seen a few in last slide).**
- (Where do the quality factors go??)

Need for Comprehensive Software Quality Requirements

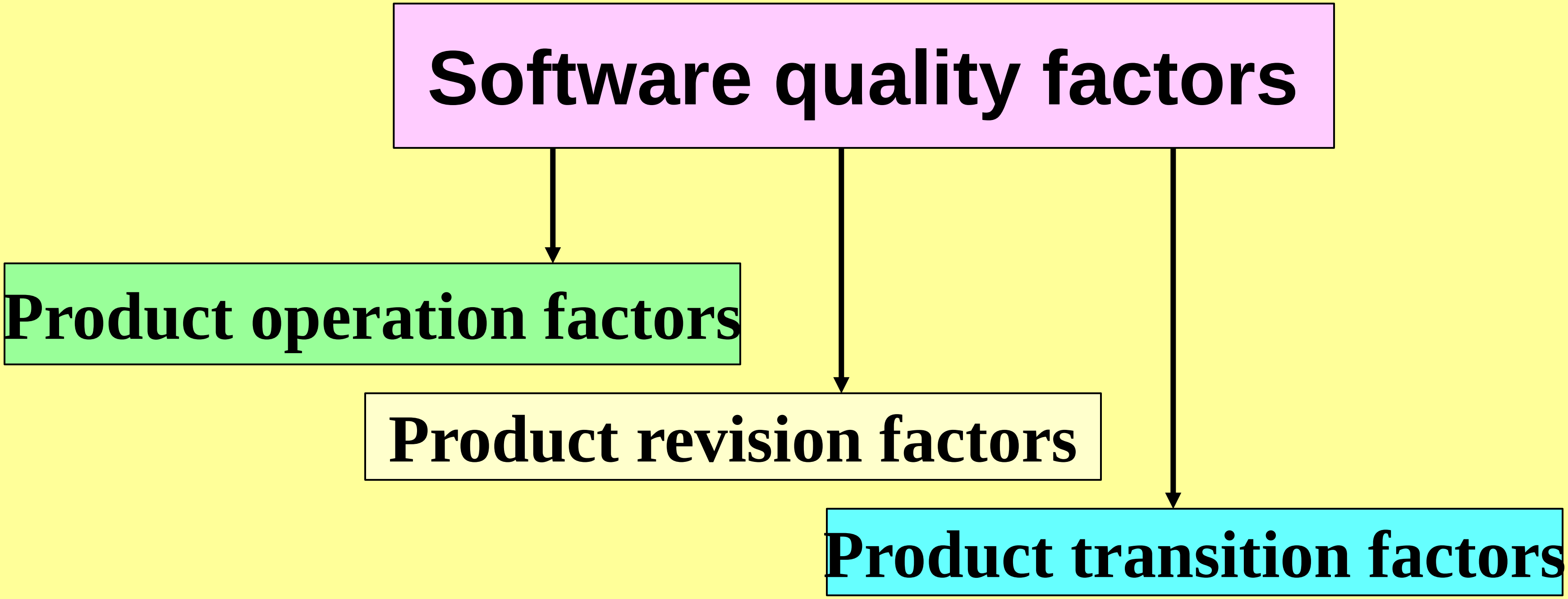
- Need for improving poor requirements documents is widespread
- Frequently lack **quality factors** such as: usability, reusability, maintainability, …
- Software industry groups the long list of related attributes into what we call ***quality factors.*** ***(Sometimes non-functional requirements)***
- Natural to assume an unequal emphasis on all quality factors.
- Emphasis varies from project to project
  - Scalability; maintainability; reliability; portability; etc.
- Let's look at some of the categories…

Extra Thoughts

- Seems like in Software Engineering we concentrate on capturing, designing, implementing, and deploying with emphasis on **functional requirements.**
- Little (not none!) emphasis on the **non-functional requirements** (quality factors).
- More and more emphasis now placed on quality factors
- Can be a critical factor in satisfying overall requirements.

- In the RUP (**Rational Unified Process – A s/w devp. process from rational a IBM division)**, non-functional requirements are captured in the Software Requirements Specification (SRS); functional requirement usually captured in Use Case stories.

McCall's software quality factors model



Software quality factors

Product operation factors

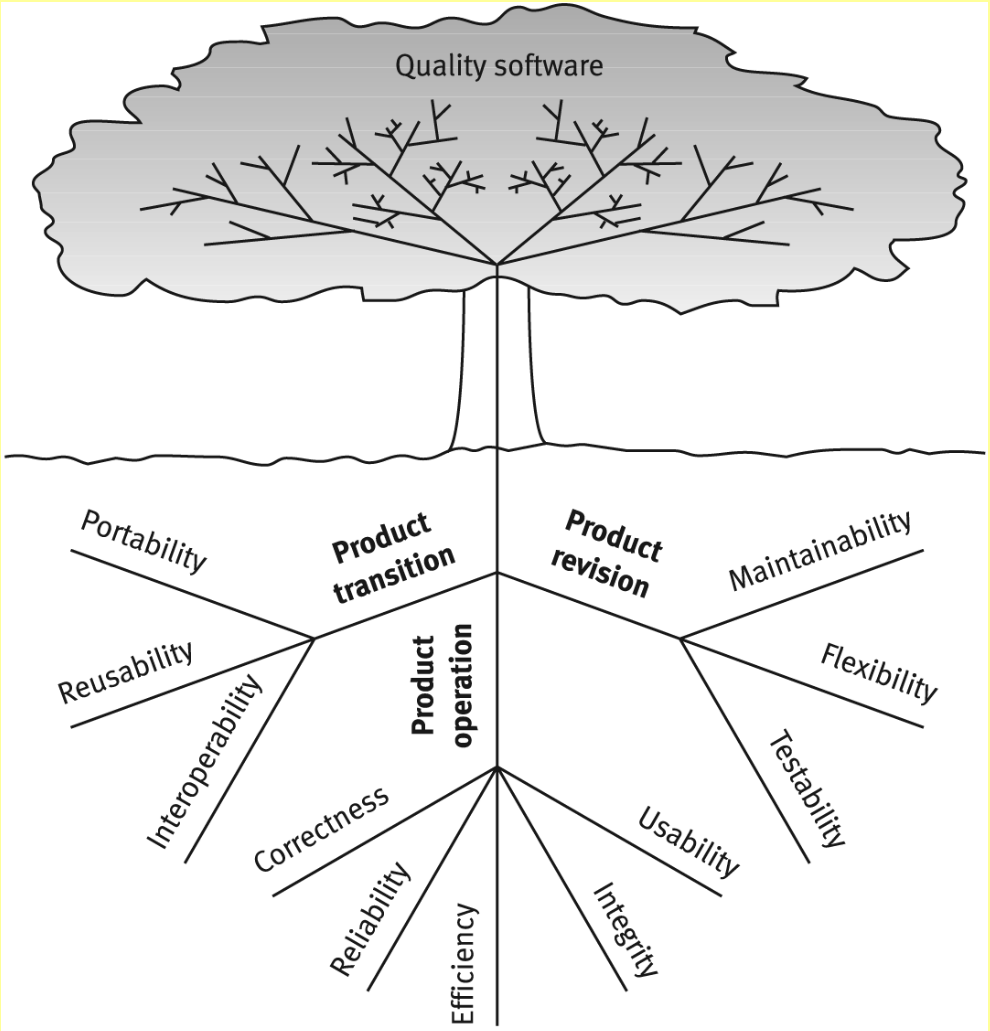
Product revision factors

Product transition factors

McCall’s Quality Factors

- McCall has 11 factors; Groups them into categories.
  - 1977; others have added, but this still prevail.
- Three categories:
  - **Product Operation Factors**
    - How well it runs….
    - Correctness, reliability, efficiency, integrity, and usability
  - **Product Revision Factors**
    - How well it can be changed, tested, and redeployed.
    - Maintainability; flexibility; testability
  - **Product Transition Factors**
    - How well it can be moved to different platforms and interface with other systems
    - Portability; Reusability; Interoperability
- Since these underpin the notion of quality factors and others who have added, reworded or added one or two, we will spend time on these factors.

McCalls factor model tree



Product operation factors

- Correctness
- Reliability
- Efficiency
- Integrity
- Usability

How well does it run and ease of use.

McCall’s Quality Factors

Category: Product **Operation** Factors

- **1. Correctness**.
- **Please note that we are asserting that ‘correctness’ issues are arising from the requirements documentation and the specification of the outputs…**
- **Examples include:**
  - **The output mission** (e.g., sales invoice printout, and red alarms when temperature rises above 250°F)
  - **Required accuracies for these outputs**, that could be affected by inaccurate data or faulty calculations;
  - **The completeness of the outputs** provided, which can be impacted by incomplete data (often done)
  - **The up-to-dateness of the output** (time between event and its consideration by the software system)
  - **The availability of the information** (the reaction time, defined as the time needed to obtain the requested information)
  - **The standards** for coding and documenting the software system

Example: The correctness requirements of a club membership information system consisted of the following:

- **The output mission:** A defined list of 11 types of reports, four types of standard letters to members and eight types of queries, which were to be displayed on the monitor on request.
- **The required accuracy of the outputs:** The probability for a non-accurate output, containing one or more mistakes, will not exceed 1%.
- **The completeness of the output information:** The probability of missing data about a member, his attendance at club events, and his payments will not exceed 1%.
- **The up-to-dateness of the information:** Not more than two working days for information about participation in events and not more than one working day for information about entry of member payments and personal data.
- **The availability of information:** Reaction time for queries will be less than two seconds on average; the reaction time for reports will be less than four minutes.
- **The required standards and guidelines:** The software and its documentation are required to comply with the client’s guidelines.

McCall’s Quality Factors
Category: Product Operation Factors

- **2. Reliability Requirements**. (remember, this quality factor is specified in the **specs**!)
- Reliability requirements deal with the failure to provide service.
  - They determine the maximum allowed software system failure rate, either overall or to required functions.
- Example specs:
  - The failures frequency of a heart-monitoring unit that will operate in a hospital’s intensive care ward is required to be less than one in 20 years. Its heart attack detection function is required to have a failure rate of less than one per million cases.
  - One requirement of the new software system to be installed in the main branch of Independence Bank, which operates 120 branches, is that it will not fail, on average, more than 10 minutes per month during the bank’s office hours.

3. Efficiency Requirements

■ Efficiency requirements deal with the hardware resources needed to perform all the functions of the software system in conformance to all other requirements.

■ Here we consider MIPS, MHz (cycles per second); data storage capabilities measured in MB or TB or communication lines (usually measured in KBPS, MBPS, or GBPS).

■ Example :

1. A chain of stores is considering two alternative bids for a software system. Both bids consist of placing the same computers in the chain's headquarters and its branches. The bids differ solely in the storage volume: 20 GB per branch computer and 100 GB in the head office computer (Bid A); 10 GB per branch computer and 30 GB in the head office computer (Bid B). There is also a difference in the number of communication lines required: Bid A consists of three communication lines of 28.8 KBPS between each branch and the head office, whereas Bid B is based on two communication lines of the same capacity between each branch and the head office. In this case, it is clear that Bid B is more efficient than Bid A because fewer hardware resources are required.

- **4. Integrity Requirements**

■Integrity requirements deal with the software system security, that is requirements to present access to an authorize person,

[to distinguish between the majority of personnel allowed to see the information (“read permit”) and a limited group who will be allowed to add and change data (“write permit”), and so forth.]

■ Example : GIS S/W allowed citizens access to its GIS through Internet only for viewing and copying data but not to insert changes.

- **5. Usability Requirements**

- ■ Usability deals with learnability, utility, usability, and more
- ■ Usability requirements deal with the scope of staff resources needed to train a new employee and to operate the software system.
- ■ **Example** : The software usability requirements document for the new help desk system initiated by a home appliance service company lists the following specifications :
  - a) A staff member should be able to handle at least 60 service calls a day
  - b) Training a new employee will take no more than two days, immediately at the end of which the trainee will be able to handle 45 service calls a day.

Product revision factors

- **Maintainability**
- **Flexibility**
- **Testability**

Can I fix it easily, retest, version it, and deploy it easily?

McCall’s Quality Factors

Category: Product **Revision** Software Factors

- These deal with requirements that affect the complete range of software maintenance activities:
  - – **Corrective maintenance** (correction of software faults and failures)
  - – **Adaptive maintenance** (adapting the current software to additional circumstances and customers without changing the software)
  - – **Perfective maintenance** (enhancement and improvement of existing software with respect to locally limited issues. Includes adding features that can enhance user experience and removing features that are not effective and functional.)

1. Maintainability Requirements

- – The degree of effort needed to identify reasons (find the problem) for software failure and to correct failures and to verify the success of the corrections.
- – Deals with the modular structure of the software, internal program documentation, programmer manual, architectural and detail design and corresponding documentation
- – **Example** :
  a) The size of a software module will not exceed 30 statements.
  b) The programming will adhere to the company coding standards and guidelines.

McCall’s Quality Factors

Category: Product Revision Software Factors

2. **Flexibility Requirements** – The capabilities and efforts required to support adaptive maintenance activities are covered by the flexibility requirements.

- These include the resources (in man-days) required to adapt a software package to a variety of customers of the same trade, of various extents of activities, of different ranges of products, and so on.
- This factor’s requirements also support perfective maintenance activities.

– May also involve a little perfective maintenance to perhaps do a little better due to the customer’s perhaps slightly more robust environment.

– Different clients exercise software differently and this is big !

- **Example:** TSS (teacher support software) deals with the documentation of pupil achievements, the calculation of final grades, the printing of term grade documents, and the automatic printing of warning letters to parents of failing pupils. The software specifications included the following flexibility requirements:

  (a) The software should be suitable for teachers of all subjects and all school levels (elementary, junior and high schools).

  (b) Non-professionals should be able to create new types of reports according to the teacher's requirements and/or the city's education department demands.

3. **Testability Requirements –** Testability requirements deal with the testing of an information system as well as with its operation.

- Testability requirements for the ease of testing are related to special features in the programs that help the tester, for instance by providing predefined intermediate results and log files.
  - Are intermediate results of computations predefined to assist testing?
  - Are log files created?
  - Backup?
  - Does the software diagnose itself prior to and perhaps during operations?

Product transition factors

- **Portability**
- **Reusability**
- **Interoperability**

Can I move the app to different hardware?

Interface easily with different hardware / software systems;

Can I reuse major portions of the code with little modification to develop new apps?

McCall's Quality Factors

Category: Product **Transition** Software Quality Factors

1. Portability Requirements: If the software must be ported to different environments (different hardware, operating systems, …) and still maintain an existing environment, then portability is a must.

Example: A software package designed and programmed to operate in a Windows 7 environment is required to allow low-cost transfer to Linux environment

2. Reusability Requirements: Are we able to reuse **parts** of the app for new applications?

- Can save immense development costs due to errors found / tested.
- Certainly, higher quality software and development more quickly.
- Very big deal nowadays.

McCall's Quality Factors

Category: Product Transition Software Quality Factors

3. Interoperability Requirements:

- Interoperability requirements focus on creating interfaces with other software systems or with other equipment firmware.
- Interoperability requirements can specify the name of the software or firmware for which interface is required.

  – Frequently these will be known ahead of time and plans can be made to provide for this requirement during design time.

  •Sometimes these systems can be quite different; different platforms, different databases, and more

  – Also, industry or standard application structures in areas can be specified as requirements

- **Example:** The firmware of a medical laboratory's equipment is required to process its results (output) according to a standard data structure that can then serve as input for a number of standard laboratory information systems.

Alternatives

- ■ Formal comparison of the alternative models (Evans M 1987, Deutsch & Willis 1988)
- ■ Both alternative models exclude only one of McCall’s 11 factors, namely the testability factor.
- ■ The Evans and Marciniak factor model consists of 12 factors that are classified into three categories.
- ■ The Deutsch and Willis factor model consists of 15 factors that are classified into four categories.
- ■ Taken together, five new factors were suggested
  - • Evans and Marciniak offer two ‘new’ ones:
    - – Verifiability and Expandability
  - • Deutsch and Willis offer three ‘new’ ones.
    - – Safety, Manageability, and Survivability

McCall's factor model and alternative models

No.	Software quality factor	McCall’s classic model	Alternative factor models	
			Evans and Marciniak model	Deutsch and Willis model
1	Correctness	+	+	+
2	Reliability	+	+	+
3	Efficiency	+	+	+
4	Integrity	+	+	+
5	Usability	+	+	+
6	Maintainability	+	+	+
7	Flexibility	+	+	+
8	Testability	+		
9	Portability	+	+	+
10	Reusability	+	+	+
11	Interoperability	+	+	+
12	Verifiability		+	+
13	Expandability		+	+
14	Safety			+
15	Manageability			+
16	Survivability			+

Alternatives

Evans and Marciniak offer Verifiability and Expandability

- **1. Verifiability Requirements** define design and programming features that enable efficient verification of the design and programming. Most verifiability requirements refer to modularity, to simplicity, and to adherence to documentation and programming guidelines.
- **2. Expandability Requirements** refer to future efforts that will be needed to serve larger populations, improve service, or add new applications in order to improve usability. The majority of these requirements are covered by McCall’s flexibility factor.

Alternatives

Deutsch and Willis offer Safety, Manageability, and Survivability

- **1. Safety Requirements** are meant to eliminate conditions hazardous to operators of equipment as a result of errors in process control software. These errors can result in inappropriate reactions to dangerous situations or to the failure to provide alarm signals when the dangerous conditions to be detected by the software arise.

- **Example:** In a chemical plant, a computerized system controls the flow of acid according to pressure and temperature changes occurring during production. The safety requirements refer to the system's computerized reactions in cases of dangerous situations and also specify what kinds of alarms are needed in each case

- **2. Manageability Requirements** refer to the administrative tools that support software modification during the software development and maintenance periods, such as configuration management, software change procedures, and the like.

- **Example:** “Chemilog” is a software system that automatically logs the flows of chemicals into various containers to allow for later analysis of the efficiency of production units. The development and issue of new versions and releases of “Chemilog” are controlled by the Software Development Board, whose members act according to the company’s software modifications procedure.

– **3. Survivability Requirements** refer to the continuity of service. These define the minimum time allowed between failures of the system, and the maximum time permitted for recovery of service, two factors that pertain to service continuity. Although these requirements may refer separately to total and to partial failures of services, they are especially geared to failures of essential functions or services. Significant similarity exists between the survivability factor and the reliability factor described in McCall’s model.

– **Example:** Taya operates a national lottery, held once a week. About 400000 to 700000 bets are placed weekly. The new software system the customer (the Taya National Lottery) has ordered will be highly computerized and based on a communication system that connects all the betting machines to the central computer. To its other high reliability requirements, Taya has added the following survivability requirement: The probability that unrecoverable damage to the betting files will occur in case of any system failure is to be limited to less than one in a million.

Comparisons

- Expandability and Survivability, actually resemble factors already included in McCall’s factor model, though under different names, Flexibility and Reliability.
- In addition, McCall’s Testability factor can be considered as one element in his own Maintainability factor.
- That is, the alternative factor models add only three “new” factors to McCall’s model:
  - Both models do add **Verifiability**,
  - The Deutsch and Willis model adds the factors **Safety** and **Manageability**.