IS2001: Software Engineering

Software Quality Management

Piyumi Kariyawasam

Course Outline

Learning Outcomes

- Be able to identify different quality attributes of a Software product
- Understand the quality management process and the key process activities of quality assurance, quality planning and quality control.
- Understand the importance of standards in the quality management process.
- Understand how measurement may be helpful in assessing some quality attributes.

Questions in last year paper

- 2014
  - Define 4 software quality attributes which can be used to measure the software quality.
- 2015
  - Define Product Standards and Process Standards.
  - Briefly explain following software quality attributes.
    - i. Availability ii. Traceability iii. Resilience iv. Portability
  - Briefly explain the three (3) software quality activities.

Software Quality Management

- Quality management provides an independent check on the software development process. The deliverables from the software process are input to the quality management process and are checked to ensure that they are consistent with organizational standards and goals.
- Quality management should be separated from project management so that quality is not compromised by management responsibilities for project budget and schedule.
- An independent team should responsible for quality management and should report to the management above the project management level.

Software Quality Management

- There must be a management commitment to quality.
- Concerned with ensuring that the required level of quality is achieved in a software product.
- Involves defining appropriate quality standards and procedures and ensuring that these are followed.
- Should aim to develop a ‘quality culture’ where quality is seen as everyone’s responsibility.

Software Quality Management

- Three principal concerns:
  1. At the organizational level, quality management is concerned with establishing a framework of organizational processes and standards that will lead to high-quality software.
  2. At the project level, quality management involves the application of specific quality processes and checking that these planned processes have been followed.
  3. At the project level, quality management is also concerned with establishing a quality plan for a project. The quality plan should set out the quality goals for the project and define what processes and standards are to be used.

What is Quality?

- Quality, simplistically, means how well a software system conforms to a given specification and design and how it meets non-functional requirements that support the delivery of the functional requirements
- This is problematical for software systems
  - There is a tension between customer quality requirements (efficiency, reliability, etc.) and developer quality requirements (maintainability, reusability, etc.);
  - Some quality requirements are difficult to specify in an unambiguous way;
  - Software specifications are usually incomplete and often inconsistent.
- The focus may be 'fitness for purpose' rather than specification conformance.

Fitness for Purpose…

- Have programming and documentation standards been followed in the development process?
- Has the software been properly tested?
- Is the software sufficiently dependable to be put into use?
- Is the performance of the software acceptable for normal use?
- Is the software usable?
- Is the software well-structured and understandable?

Quality in…

- Traditional view
  - quality is about perfection/bug free code. Generally associated with testing at the end of development. Testing cannot introduce quality to a product, it can only reduce the number of defects in the product.

- Modern view (ISO 9000)
  - Good quality is not perfection but fit for the purpose. Build the right product in the right way. Do not over engineer (too expensive). Do not under engineer (not fit for the purpose).

Scope of Quality Management

- Quality management is particularly important for large, complex systems. The quality documentation is a record of progress and supports continuity of development as the development team changes.
- For smaller systems, quality management needs less documentation and should focus on establishing a quality culture.

Software Quality Attributes

- Quality attribute requirements are part of an application's non-functional requirements, which capture the many facets of how the functional requirements of an application are achieved.
- Quality attribute requirements must be specific about how an application should achieve a given need.
  - (*e.g. The transaction should be done efficiently*)
- Quality attributes have been classified under different categories. However taken together they form the non-functional requirements of the system.

Software Quality Attributes – Engineering Aspect

- **Availability**
  - Availability is the proportion of time that the system is functional and working. It can be measured as a percentage of the total system downtime over a predefined period. Availability will be affected by system errors, infrastructure problems, malicious attacks, and system load.
- **Redundancy**
  - Redundancy is the inclusion of extra components which are not strictly necessary to functioning, but in case of failure in other components, then there will be a backup.
- **Disaster Recovery**
  - Disaster Recovery is the ability of the software to continue to function when a Disaster occurs.
- **Security**
  - Security is the ability of the software to remain protected from unauthorized access. This includes both change access and view access.

Software Quality Attributes – Engineering Aspect

- **Flexibility**
  - Flexibility is the ability of a software to adapt when external changes occur.
- **Traceability**
  - Traceability is the ability of the Software to offer insight into the inner processing when required. A higher level of traceability is required at time of debugging a problem or at times of new interoperability testing.
- **Maintainability**
  - Maintainability is the ability of a software to adapt to changes, improve over time, correct any bugs and be proactively fixed through preventive maintenance.
- **Testability**
  - Testability is the ability of a software to be tested thoroughly before putting into production.
- **Modifiability**
  - Modifiability is a measure of how easy it may be to change an application to cater for new functional and non-functional requirements.

Software Quality Attributes – Technology Aspect

- **Reliability**
  - Reliability is the measure of how a product behaves in varying circumstances. It is the ability of a system to remain operational over time. Reliability is measured as the probability that a system will not fail to perform its intended functions over a specified time interval.
- **Robustness**
  - Robustness is defined as the ability of a software product to cope with unusual situation.
- **Efficiency**
  - Efficiency is the ability of the software to do the required processing on least amount of hardware.
- **Resilience**
  - Resilience is the ability of a software to absorb sudden bursts of legitimate loads.

Software Quality Attributes – Technology Aspect

- **Compatibility**
  - Compatibility is the ability of the software to work with other systems.
- **Modularity**
  - Modularity is the measure of the extent to which software is composed of separate, interchangeable components, each of which accomplishes one function and contains everything necessary to accomplish this. Modularity increases cohesion and reduces coupling and makes it easier to extend the functionality and maintain the code.
- **Reusability**
  - Reusability is the capability for components and subsystems to be suitable for use in other applications and in other scenarios. Reusability minimizes the duplication of components and also the implementation time.

Software Quality Attributes –Operational Aspect

- **Manageability**
  - Manageability is the ability of a software to be managed from different perspectives.
- **Usability**
  - Usability is the ability of a software to offer its interfaces in a user friendly and elegant way.
- **Accessibility**
  - Accessibility is the ability of a software to be accessible from a multitude of devices and for a number of different types of users. (e.g. users with disabilities)
- **Auditability**
  - Auditability is the ability of a software to keep track of what happened, who did it, from where, how and when.

Software Quality Attributes –Operational Aspect

- **Portability**
  - Portability is the ability of a program (or system) to execute properly on multiple hardware platforms.
- **Interoperability**
  - Interoperability is the ability of a system or different systems to operate successfully by communicating and exchanging information with other external systems written and run by external parties. An interoperable system makes it easier to exchange and reuse information internally as well as externally.
- **Performance**
  - Performance is an indication of the responsiveness of a system to execute any action within a given time interval. It can be measured in terms of latency or throughput. Latency is the time taken to respond to any event. Throughput is the number of events that take place within a given amount of time.

Software Quality Attributes –Commercial Aspect

- **Affordability**
  - Affordability is the ability of a software to keep its total cost within the range of affordability of an Operator. It is mainly about the ability to 'pay as you grow' and It has many dimensions to it .
- **Scalability**
  - Scalability is the ability of the software to cater for heavier processing loads as the needs arise. In addition to its usual meaning of how heavier loads can be handled, scalability also means 'how small can you start'. Scalable architectures allow both the vendor and the operator to follow a 'Pay as your grow' model. Horizontal Scalability is the ability to be able to add more resources to get higher processing. Vertical Scalability is the ability to be able to add bigger resources to get higher processing.

Software Quality Attributes

Bohem’s Classification

- **Current Usefulness**
  - The qualities expected from a software system in user’s point of view.
  - The qualities must be exhibited in the software product
- **Potential Usefulness**
  - The qualities expected from a software system in developer’s point of view.
  - Deals with the product’s lifetime in the future

Bohem's Classification

Current usefulness	Potential usefulness
Efficiency	Maintainability
Reliability	Modularity
Usability	Reusability
Correctness	Portability
User friendliness	
Robustness	

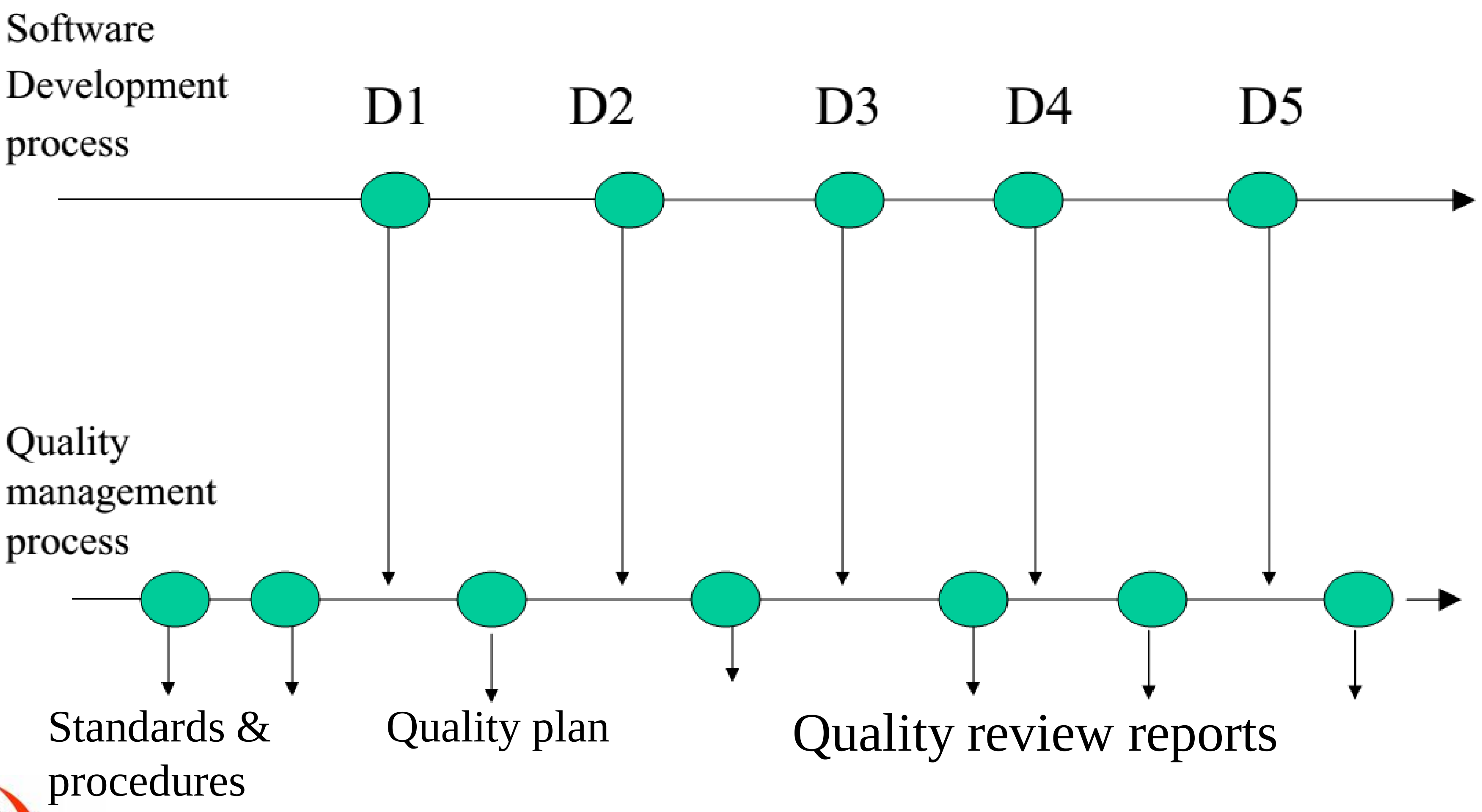
McCall's Classification

- Product Operation
  - Efficiency
  - Correctness
  - User friendliness
  - Usability
  - Reliability
  - Robustness
- Product Revision
  - Maintainability
  - Flexibility
  - Testability
- Product Transition
  - Interoperability
  - Reusability
  - Portability

Quality Conflicts

- It is not possible for any system to be optimized for all of these attributes – for example, improving robustness may lead to loss of performance
- The quality plan should therefore define the most important quality attributes for the software that is being developed
- The plan should also include a definition of the quality assessment process, an agreed way of assessing whether some quality, such as maintainability or robustness, is present in the product

Quality Management & Software Development



Quality Management & Software Development

- Quality management provides an independent check on the software development process
- The deliverables from the software process are input to the quality management process and are checked to ensure that they are consistent with organizational standards and goals
- Quality management should be separated from project management so that quality is not compromised by management responsibilities for project budget and schedule
- There must be management commitment towards quality

Software Quality Activities

1. **Quality assurance**
   – Establish organizational procedures and standards for quality

2. **Quality planning**
   – Select applicable procedures and standards for a particular project and modify these as required

3. **Quality control**
   – Ensure that procedures and standards are followed by the software development team

1. Quality Assurance

- Quality assurance activities define a framework for achieving software quality

- There are two types of standards that may be established as a part of the quality assurance process;
  1. Product standards
  2. Process standards

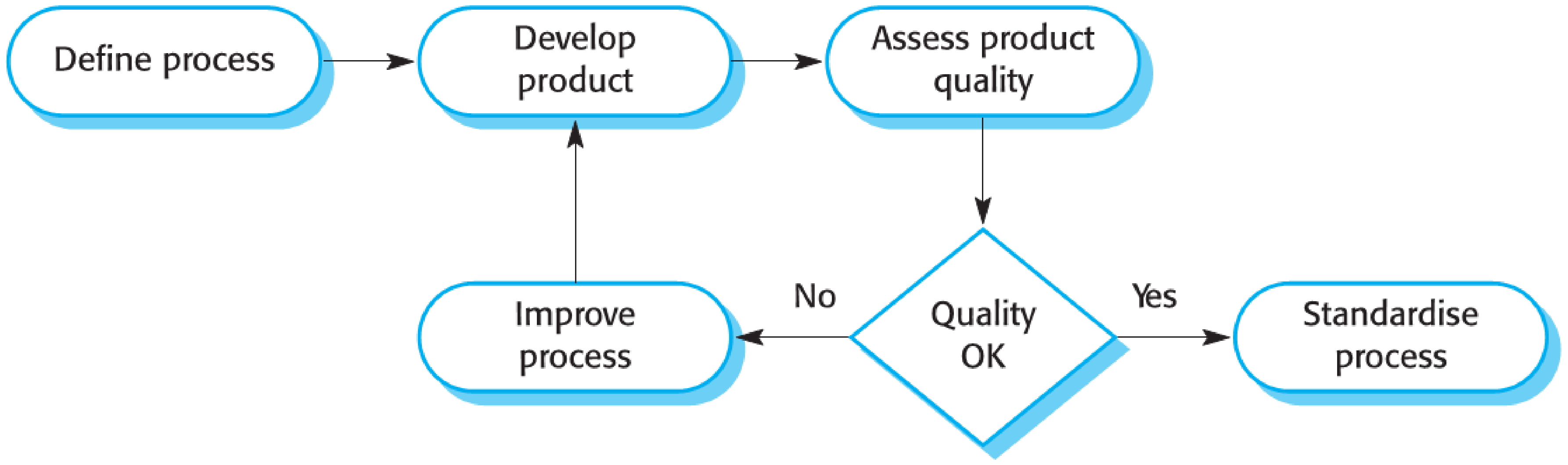
1. Quality Assurance

- **Product Standard**
  - These are standards that apply to the software product being developed.
  - They include standards such as document standards, coding standards and user interface standards.
  - Product quality includes reusability, usability, portability, maintainability ... etc.

1. Quality Assurance

- **Process Standard**
  - These are standards that define the processes which should be followed during software development .
  - They may include definitions of specification, design and validation processes and a description of the documents which must be generated in the course of these processes.

Process-based Quality



Importance of Standards

- Encapsulation of best practice
  - avoids repetition of past mistakes
- Standards provide a framework for defining what quality means in a particular setting.
  - i.e. that organization's view of quality
- Standards provide continuity
  - new staff can understand the organisation by understanding the standards that are used

Product and Process Standards

Product standards	Process standards
Design review form	Design review conduct
Requirements document structure	Submission of new code for system building
Method header format	Version release process
Java programming style	Project plan approval process
Project plan format	Change control process
Change request form	Test recording process

Problems with Standards

- They may not be seen as relevant and up-to-date by software engineers
- They often involve too much bureaucratic form filling
- If they are unsupported by software tools, tedious form filling work is often involved to maintain the documentation associated with the standards

Standards Development

- Involve practitioners in development
  - Engineers should understand the rationale underlying a standard
- Review standards and their usage regularly
  - Standards can quickly become outdated and this reduces their credibility amongst practitioners
- Detailed standards should have specialized tool support
  - Excessive clerical work is the most significant complaint against standards
  - Web-based forms are not good enough

ISO Standards

- ISO standards are an international set of standards for quality management
- These are applicable to a range of organizations from manufacturing to service industries
- ISO 9001 is applicable to organizations which design, develop and maintain products
- This is a generic model of the quality process that must be instantiated for a particular organization
- ISO 9000 – 3 interprets ISO 9001 for software development

ISO 9000 Certification Process

- Quality standards and procedures should be documented in an organisational quality manual
- External body may certify that the organisational quality manual conforms to ISO 9000 standards

Capability Maturity Model (CMM)

- CMM was developed at the Software Engineering Institute (SEI) in Pittsburgh, and it is very much a rival to ISO9001 for software.
- CMM is a scheme to classify a software development organization according to its capability.
- It identifies five different maturity levels for software developing organizations

Capability Maturity Model (CMM)

1. **Initial**
   – The software development is run informally, and depends on the competence of some persons
2. **Repeatable**
   – There is a common system for project management and control
3. **Defined**
   – There is a common system for the software engineering activities
4. **Managed**
   – The software development process is stable and gives a consistent product quality. Measurements are used to keep the process and product under control
5. **Optimizing**
   – The software development process contain its own improvement process. Process improvement is budgeted and planned and is an integral part of the organization

2. Quality Planning

- A quality plan sets out the desired product qualities and how these are assessed and defines the most significant quality attributes
- The quality plan should define the quality assessment process
- It should set out which organisational standards should be applied and, where necessary, define new standards to be used

Quality Plan Structure

- **Product introduction**
  - Description of the product, its intended market and the quality expectations for the product
- **Product plans**
  - Critical release dates and plans for distribution and maintenance
- **Quality goals**
  - Quality goals in detail including an identification of the critical product quality attributes
- **Process descriptions**
  - Development processes which should be used for product development to achieve the quality goals
- **Risks and risk management**
  - The key risks which might affect product quality and the actions to address these risks

3. Quality Control

- Checking the software development process to ensure that procedures and standards are being followed

- Two approaches to quality control
  1. Quality reviews
  2. Automated software assessment and software measurement

Software Metrics

- Software metric is any type of measurement which relates to a software system, process or related documentation
  - Examples are Lines of code, Cyclomatic Complexity, Length of identifiers, Depth of conditional nesting, Fog index, number of person days … etc.
- Allow the software and the software process to be quantified and compared objectively
- Measures of the software process or product may be used to predict product attributes or to control the software process

Software Metric	Description
Length of code	This is a measure of the size of a program. Generally, larger the size of a code component, the more complex and error prone that code component is likely to be.
Cyclomatic Complexity	This is measure of the control complexity of a program.
Length of identifiers	This is a measure of the average length of distinct identifiers in a program. The longer the identifiers, the more likely they are to be meaningful.
Depth of conditional nesting	Deeply nested if statements are hard to understand and are potentially error prone.
Fog index	This is the measure of the average length of words and sentences in documents. The higher the value of fog index, the more difficult the document may be to understand.

Key Points

- Software quality management is concerned with ensuring that software has a low number of defects and that it reaches the required standards of maintainability, reliability, portability and so on
- SQM includes defining standards for processes and products and establishing processes to check that these standards have been followed
- Software standards are important for quality assurance as they represent an identification of ‘best practice’
- Quality management procedures may be documented in an organizational quality manual, based on the generic model for a quality manual suggested in the ISO 9001 standard
- Product quality metrics are particularly useful for highlighting anomalous components that may have quality problems