Software Quality

Software Architecture: to meet functional requirements and quality requirements

We use

- Proffessor Richard Taylor and
- Note on Wikipedia: https://en.wikipedia.org/wiki/Software_quality and https://en.wikipedia.org/wiki/ISO/IEC_9126

1-Issues of qualitative measures

- How big is the program?
  - Huge!!
- How close are you to finishing?
  - We are almost there!!
- Can you, as a manager, make any useful decisions from such subjective information?
- Need information like, cost, effort, size of project.

2-Metric: quantifiable measures

- Quantifiable measures that could be used to measure characteristics of a software system or the software development process
- Required in all phases
- Required for effective management
- Managers need quantifiable information, and not *subjective* information
  - Subjective information goes against the fundamental goal of *engineering)*

3-Type of Metrics

- Product metrics
  - quantify characteristics of the product being developed
    - size, reliability
- Process metrics
  - quantify characteristics of the process being used to develop the software
    - efficiency of fault detection

4-Capability Maturity Model

The model provides a theoretical continuum along which process maturity can be developed incrementally from one level to the next. Skipping levels is not allowed/feasible.

Level 1 - ***Initial (Chaotic)***

It is characteristic of processes at this level that they are (typically) undocumented and in a state of dynamic change, tending to be driven in an *ad hoc*, uncontrolled and reactive manner by users or events. This provides a chaotic or unstable environment for the processes.

Level 2 - ***Repeatable***

It is characteristic of processes at this level that some processes are repeatable, possibly with consistent results. Process discipline is unlikely to be rigorous, but where it exists it may help to ensure that existing processes are maintained during times of stress.

Level 3 - ***Defined***

It is characteristic of processes at this level that there are sets of defined and documented standard processes established and subject to some degree of improvement over time. These standard processes are in place (i.e., they are the AS-IS processes) and used to establish consistency of process performance across the organization.

Level 4 - ***Managed***

It is characteristic of processes at this level that, using process metrics, management can effectively control the AS-IS process (e.g., for software development). In particular, management can identify ways to adjust and adapt the process to particular projects without measurable losses of quality or deviations from specifications. Process Capability is established from this level.

Level 5 - ***Optimizing***

It is a characteristic of processes at this level that the focus is on continually improving process performance through both incremental and innovative technological changes/improvements.

At maturity level 5, processes are concerned with addressing statistical *common causes* of process variation and changing the process (for example, to shift the mean of the process performance) to improve process performance. This would be done at the same time as maintaining the likelihood of achieving the established quantitative process-improvement objectives. There are only a few companies in the world that have attained this level 5.

- Level 4: Managed level
  - Process measurement performed
  - Quality and productivity goals set
  - Continually measured and corrective actions taken
  - Statistical quality controls in place
- Level 5: Optimizing level
  - Statistical quality and process control in place
  - Positive feedback loop used for improvement in productivity and quality

5-Cost Issues

It costs to measure software system with quality metrics

6-Validity of Metric

There may be disagreement about the validity of metric

7-Type of Metric

Basic metrics

- Size (like LOC)
- Cost (in $$$)
- Duration (months)
- Effort (person-months)
- Quality (number of faults detected)

8-Examples of metrics

•Source lines of code.
•Cyclomatic complexity, is used to measure code complexity.
•Function point analysis (FPA), is used to measure the size (functions) of software.
•Bugs per lines of code.
•Code coverage, measures the code lines that are executed for a given set of software tests.
•Cohesion, measures how well the source code in a given module work together to provide a single function.
•Coupling, measures how well two software components are data related, i.e. how independent they are.

The above list is only a small set of software metrics, the important points to note are:-

•They are all measurable, that is they can be quantified.
•They are all related to one or more software quality characteristics.

9-Some metrics that are related to software architecture

- We define ***Change Propagation*** from component A to component B as the probability that a change in A due to corrective/ defective maintenance requires a change in B to maintain the overall function of the system.

-metric for connector reliability: number redundant connectors or backup connectors

10-ISO 9126

Quality Model, Internal Metrics, External Metrics, Quality in Use Metrics

Quality Model [edit]

The quality model presented in the first part of the standard, ISO/IEC 9126-1,[2] classifies software quality in a structured set of characteristics and sub-characteristics as follows:

- **Functionality** - *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
  - Functionality Compliance
- Reliability - *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
  - Reliability Compliance
- Usability - *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
  - Attractiveness
  - Usability Compliance
- Efficiency - *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 Behaviour
  - Resource Utilization
  - Efficiency Compliance
- **Maintainability** - *A set of attributes that bear on the effort needed to make specified modifications.*
  - Analyzability
  - Changeability
  - Stability
  - Testability
  - Maintainability Compliance
- **Portability** - *A set of attributes that bear on the ability of software to be transferred from one environment to another.*
  - Adaptability
  - Installability
  - Co-Existence
  - Replaceability
  - Portability Compliance

The next step is to create/select metrics for the sub-attributes

Internal Metrics [edit]

Internal metrics are those which do not rely on software execution (static measure)

External Metrics [edit]

External metrics are applicable to running software.

Quality in Use Metrics [edit]

Quality in use metrics are only available when the final product is used in real conditions.

Ideally, the internal quality determines the external quality and external quality determines quality in use.

This standard stems from the GE model for describing software quality, presented in 1977 by McCall et al., which is organized around three types of Quality Characteristics:

- Factors (To specify): They describe the external view of the software, as viewed by the users.
- Criteria (To build): They describe the internal view of the software, as seen by the developer.
- Metrics (To control): They are defined and used to provide a scale and method for measurement.

ISO/IEC 9126 distinguishes between a defect and a nonconformity, a **defect** being *The nonfulfilment of intended usage requirements*, whereas a **nonconformity** is *The nonfulfilment of specified requirements*. A similar distinction is made between validation and verification, known as V&V in the testing trade.

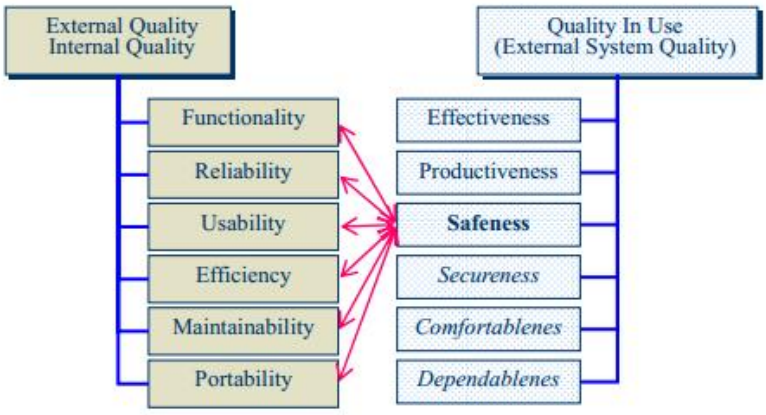

Figure 4: Modified ISO/IEC 9126-1 Quality Model

11-Examples

Requirements for a Stock Exchange Monitoring System

The primary goal of a real-time monitoring system is to capture, to analyze and to broadcast events (data) in real-time. It is a *soft* real-time system, where some of the events may miss their deadline, without affecting the whole system's behavior. The

Architectures proposed for the monitoring system

The proposed architectures based on two different architectural patterns, publisher/subscriber (push model) and repository are shown in Figures 4 and 5 respectively.

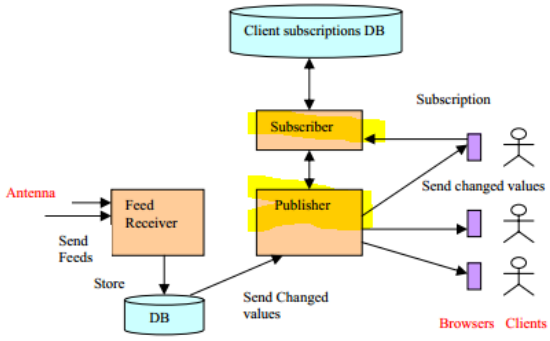

Fig. 4: Architecture based on the publisher/subscriber pattern

The publisher/subscriber memorizes the client subscriptions and the actual values in the Client Subscription DB and the DB respectively.

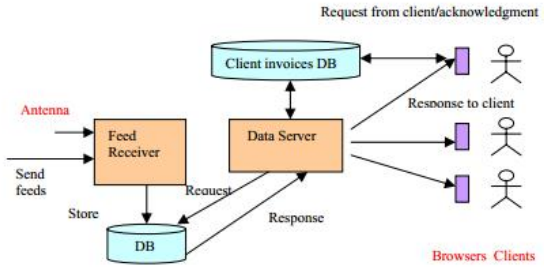

Fig. 5: Architecture based on the repository pattern

The repository memorizes the actual values in the DB and the client requests in the Client invoices DB, for invoicing purposes.

Measure and compare between 2 architecture candidates

Characteristics	Sub-characteristics	Publisher/Subscriber	Repository	Comments and results
Functionality	Suitability	yes	yes	
	Accuracy	=	=	No special computation required
	Interoperability	yes	yes	Communication through browsers
	Security	Mechanism for a subscription	Mechanism for each client request	**Publisher/subscriber is better**
Reliability	Maturity	Maturity (Reception) + Maturity (DB) + Maturity (Publisher) + Maturity (Subscriber) + Maturity (subscriptionDB)	Maturity (Reception) + Maturity (DB) + Maturity (Data Server) + Maturity (Client invoices DB)	**Repository is better**
	Fault Tolerance (availability)	=	=	Depends on additional mechanisms
	Recoverability	=	=	Depends on additional mechanisms
Usability		=	=	Depends on the browser's GUI
Efficiency	Time behavior (time spent from the data reception to the data delivery)	time (Reception)+ time (store in DB)+ time (send changes)+ time (Publisher)+ time (send changed values)	time (Client Request)+ time (Client invoices DB)+ time (Data Server)+ time (request)+time(DB)+ time (reponse Data Server)+time (Response to Client)	**Publisher/subscriber is better**
	Resource utilization (time)	Browser displays always	Browser displays on request	**Repository is better**
	Resource utilization (space)	Size (subscription DB)	Size (invoices DB)	Depends on external issues, such as the volume of client requests
Maintainability		=	=	Depends on the code in modules
Portability		=	=	Depends on additional mechanisms

Example: IT Architecture Process Maturity Levels

http://pubs.opengroup.org/architecture/togaf8-doc/arch/chap27.html

Level 0: None

No IT architecture program. No IT architecture to speak of.

Level 1: Initial

Informal IT architecture process underway.

1. Processes are *ad hoc* and localized. Some IT architecture processes are defined. There is no unified architecture process across technologies or business processes. Success depends on individual efforts.
2. IT architecture processes, documentation, and standards are established by a variety of *ad hoc* means and are localized or informal.
3. Minimal, or implicit linkage to business strategies or business drivers.
4. Limited management team awareness or involvement in the architecture process.
5. Limited operating unit acceptance of the IT architecture process.
6. The latest version of the operating unit's IT architecture documentation is on the web. Little communication exists about the IT architecture process and possible process improvements.
7. IT security considerations are *ad hoc* and localized.
8. No explicit governance of architectural standards.
9. Little or no involvement of strategic planning and acquisition personnel in the enterprise architecture process. Little or no adherence to existing standards.

Level 2: Under Development

IT architecture process is under development.

1. Basic IT architecture process is documented based on OMB Circular A-130 and Department of Commerce IT Architecture Guidance. The architecture process has developed clear roles and responsibilities.
2. IT vision, principles, business linkages, baseline, and Target Architecture are identified. Architecture standards exist, but not necessarily linked to Target Architecture. Technical Reference Model (TRM) and Standards Profile framework established.
3. Explicit linkage to business strategies.
4. Management awareness of architecture effort.
5. Responsibilities are assigned and work is underway.
6. The DoC and operating unit IT architecture web pages are updated periodically and are used to document architecture deliverables.
7. IT security architecture has defined clear roles and responsibilities.
8. Governance of a few architectural standards and some adherence to existing Standards Profile.
9. Little or no formal governance of IT investment and acquisition strategy. Operating unit demonstrates some adherence to existing Standards Profile.

Level 3: Defined

Defined IT architecture including detailed written procedures and TRM.

1. The architecture is well defined and communicated to IT staff and business management with operating unit IT responsibilities. The process is largely followed.
2. Gap analysis and migration plan are completed. Fully developed TRM and Standards Profile. IT goals and methods are identified.
3. IT architecture is integrated with capital planning and investment control.
4. Senior management team aware of and supportive of the enterprise-wide architecture process. Management actively supports architectural standards.
5. Most elements of operating unit show acceptance of or are actively participating in the IT architecture process.
6. Architecture documents updated regularly on DoC IT architecture web page.
7. IT security architecture Standards Profile is fully developed and is integrated with IT architecture.
8. Explicit documented governance of majority of IT investments.
9. IT acquisition strategy exists and includes compliance measures to IT enterprise architecture. Cost benefits are considered in identifying projects.

Level 4: Managed

Managed and measured IT architecture process.

1. IT architecture process is part of the culture. Quality metrics associated with the architecture process are captured.
2. IT architecture documentation is updated on a regular cycle to reflect the updated IT architecture. Business, Data, Applications, and Technology Architectures defined by appropriate *de jure* and *de facto* standards.
3. Capital planning and investment control are adjusted based on the feedback received and lessons learned from updated IT architecture. Periodic re-examination of business drivers.
4. Senior management team directly involved in the architecture review process.
5. The entire operating unit accepts and actively participates in the IT architecture process.
6. Architecture documents are updated regularly, and frequently reviewed for latest architecture developments/standards.
7. Performance metrics associated with IT security architecture are captured.
8. Explicit governance of all IT investments. Formal processes for managing variances feed back into IT architecture.
9. All planned IT acquisitions and purchases are guided and governed by the IT architecture.

Level 5: Optimizing

Continuous improvement of IT architecture process.

1. Concerted efforts to optimize and continuously improve architecture process.
2. A standards and waivers process is used to improve architecture development process.
3. Architecture process metrics are used to optimize and drive business linkages. Business involved in the continuous process improvements of IT architecture.
4. Senior management involvement in optimizing process improvements in architecture development and governance.
5. Feedback on architecture process from all operating unit elements is used to drive architecture process improvements.
6. Architecture documents are used by every decision-maker in the organization for every IT-related business decision.
7. Feedback from IT security architecture metrics are used to drive architecture process improvements.
8. Explicit governance of all IT investments. A standards and waivers process is used to make governance-process improvements.
9. No unplanned IT investment or acquisition activity.