Chapter 20: The Software-Quality Landscape

- This is a focus on quality and quality assurance
    - Big picture issues rather than hands on techniques

Characteristics of Software Quality

- External characteristics: ones the user of the software product is aware of
    - Correctness
        - The degree to which a system is free from faults in specification, design and implementation
    - Usability
        - The ease with which users can learn and use a system
    - Efficiency
        - Minimal use of system resources, including memory and execution time
    - Reliability
        - The ability of a system to perform its required functions under states conditions whenever required
        - Aka long time between failure
    - Integrity
        - Degree to which a system prevents unauthorized or improper access to its programs and data
        - Also ensure data is accessed properly, and data itself is legit
            - Correctly formatted dates
            - Tables in parallel updated at the same time
    - Adaptability
        - The extend to which a system can be used, without modification, in applications or environments other than those for which it was specifically designed
    - Accuracy
        - The degree to which a system is free from error
    - Robustness
        - Degree to which a system continues to function in the presence of invalid inputs or stressful environment conditions
- Internal characteristics: ones the programmer cares about in addition to external ones
    - Maintainability
        - The ease with which you can modify a software system to change or add capabilities, improve performance, or correct defects
    - Flexibility
        - The extent to which you can modify a system for uses or environments other than those for which it was specifically designed
    - Portability
        - The ease with which you can modify a system to operate in an environment different from that for which it was specifically designed
    - Reusability

        - The extent to which and the ease with which you can use parts of a system in other systems
    - Readability
        - The ease with which you can read and understand the source code of a system, especially at the detailed-statement level
    - Testability
        - The degree to which you can unit-test and system-test a system; the degree to which you can verify that the system meets its requirements
    - Understandability
        - The ease with which you can comprehend a system
        - More at the general level and how coherent the system is

Techniques for Improving Software Quality

- Software quality objectives
    - Set explicit external and internal characteristic goals
- Explicit quality assurance activity
    - "Global Gary" who gets programs "complete" but are littered with code defects
        - Who typically gets praised
    - "High Quality Henry" who writes excellent programs and makes sure they are usable before being released
    - Need to make quality a priority
- Testing strategy
    - Execution testing can provide a detailed assessment of the products reliability
- Software engineering guidelines for
    - Problem definition
    - Requirements development
    - Architecture
    - Construction
    - System testing
- Informal technical reviews
    - Desk checking
    - Walking through code with a few peers
- Formal technical reviews
    - Need to catch problems when the least amount of time has been invested into a project
    - **Use "Quality Gates" when transitioning between**
        - **Requirements development and architecture**
        - **Architecture and construction**
        - **Construction and system testing**
        - **Gates don't mean section needs to be 100% complete, just to make sure current level is good enough before starting the next level**
- External Audits

Development Process

- Change-control procedures
    - Big obstacle to achieving good quality is uncontrolled changes
    - Need to formalize, there is a huge host of issues associated with not following this
- Measurement of results
    - Unless results of a quality assurance plan are measured, you'll have no way if its working…
        - Correctness
        - Usability
        - Efficiency
        - Etc…
- Prototyping
    - Development of realistic models of a systems key functions
    - Prototypes lead to
        - Better designs
        - Better matches to user needs
        - Improved maintainability

Setting Objectives

- Explicitly setting quality objectives is simple, but easily overlooked
- People do well in what you ask them too
    - Programmers have high achievement motivation →
        - They will work to the objectives specified
        - BUT must be told what the objectives are

"If builders built buildings the way programmers wrote programs, then the first woodpecker that came along would destroy civilization" – Gerald Weinberg

Defect Detecting Techniques

- **Informal design reviews**
- **Formal design reviews**
- **Informal code reviews**
- **Formal code inspections**
- **Modeling or prototyping**
- **Personal desk-checking of code**
- **Unit tests**
- **New component testing**
- **Integration testing**
- **Regression testing**
- **System testing**
- **USE ALL OF THESE, DONE BY MULTIPLE PEOPLE**
- **Code reading also detects more errors than rigid testing**