SOFTWARE QUALITY ASSURANCE

Basic Principles

SW Quality Factors

- basic quality characteristics
  - for whom?
  - depend on the phase of the life-cycle
  - quality measures?
- consistency of the transformation process
  - mapping between phases
- completeness and adequacy of the product
  - can the end-user use the product

The Cost of SW Quality [B. Boehm]

- quality costs time & money during development:
  - code and unit test: 8%
  - integration test: 7%
  - re-testing (after each fix): 20%
  - rework (in earlier phases): 10%

  ====> about 45% of your budget
  goes into quality related activities

Cost Ratio of Errors

Requirements : Maintenance
= 1 : 200

====> detect quality problems early

The Benefits of SW Quality

- lower maintenance cost
- stable and useful product
- happy customers
- better chances for continuing releases
- build corporate culture & identity
- build know-how and expertise
- better chances for SW and design reuse
**SW Quality Management**

- SRS must include quality specification
  - be precise & clear (*"measurable"*)
  - make clear what your priorities are
- check at each milestone in your life-cycle
  - use reviews & tests
  - identify risks
  - decide on how to proceed

**SW Quality Characteristics**
[B. Boehm, A. Davis]

- portability
- reliability
- efficiency
- human engineering
- maintainability
  - testability, understandability, modifiability

**Portability**

- The degree to which software running in one environment can easily be converted to run in another environment.
- Specification & test:
  - very hard to quantify (time?)
  - specify indirectly by
    - limiting languages, compilers, tools, ...
    - insisting on good development methodology (cohesion & coupling)
    - isolating environment dependent modules in your design
- To whom and when is it important?

**Reliability**

- The ability of software to behave consistently in a user-acceptable manner (when running in stable and pre-planned environment).
- Specification & test:
  - measure as a percentage: 99% reliability
    - of operational (time), functions, accuracy/integrity,...
  - MTTF (mean time to failure):
    - does software degrade?
  - counting bugs (ratio):
    - a "post mortem"?
  - Bring "reliability" into perspective (cost).

**Efficiency**

- The level at which software uses and supports (scarce) resources.
- Specification & test:
  - resource = capacity: specify in numbers
  - resource = time: specify what time-frame you measure
    - stimulus - response
    - stimulus - stimulus ...
  - make sure your system is
    - device efficient and accountable
  - give rules for degradation of service
- First design, then optimise.
Human Engineering

- User-friendliness and consistency in external behaviour. The degree to which software reflects and integrates customer needs and culture.
- Specification & test:
  - very hard to quantify
  - establish an authorised test/benchmark team
  - specify indirectly:
    - establish criteria for the user interface by providing rules for external system behaviour, execution sequences, robustness, adaptability, ...
    - limit interactive styles, tools, languages, ...

Maintainability

- Based on three major qualities:
  - testability
  - understandability
  - modifiability
- Specification & test:
  - specify indirectly by
    - insisting on good development methodology
    - evaluate your design with cohesion & coupling
    - provide for structured implementation environment: limit languages, compilers, tools, ...
  - A shared interest between customer and supplier.