

Configuration Management

- “A discipline applying technical and administrative direction and surveillance to: identify and document the functional and physical characteristics of a configuration item, control changes to those characteristics, record and report change processing and implementation status and verify compliance with requirements.”

Configuration Management

- Revision Control
- Change Management
- Release Control

- Configuration items:
 - product artifacts: executable, source, user documentation
 - project artifacts: project plan, specification, test cases...
 - acquired elements
- Identification methods:
 - revision control tools
 - standard naming/numbering
 - well-named folder hierarchies
 - document maps
- Storage: where artifacts reside, tools used, how to recover/add elements...

Change Management

- Change driven by users/customers, developers...
- Change request form
- Assess
 - impact of change to existing project (i.e., how can it be implemented)
 - value of change (i.e., do you need it)
 - cost of change
 - ⇒ determine whether to accept it

Artifact Lifecycle

- Draft Stage – changes made informally using revision control
- => Baseline (reviewed and accepted)
- Accepted Stage – changes through formal change control
- => Release
- Maintenance Stage - Maintenance changes
- => Retire

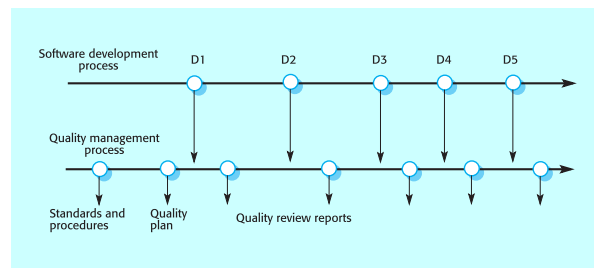
What is quality?

- Quality, simplistically, means that a product should meet its specification.
- Testing is only part of quality... includes reviews, verification and validation...
- Quality Management should apply to all artifacts:
 - product: executable (via testing), source code (programming style, headers, lines of code, variable/method names...), documents (templates, format...)
 - process: SRS, SDS, project plan, test cases... (are standards followed, adequacy criteria met, plan reviews and updates...)

The quality compromise

- 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.
- We cannot wait for specifications to improve before paying attention to quality management.
- We must put quality management procedures into place to improve quality in spite of imperfect specification.

Quality management and software development



Quality management activities

- Quality assurance
 - Establish organisational procedures and standards for quality.
- Quality planning
 - Select applicable procedures and standards for a particular project and modify these as required.
- Quality control
 - Ensure that procedures and standards are followed by the software development team.
- Quality management should be separate from project management to ensure independence (but still included in project planning).

Practical process quality

- Define process standards such as how reviews should be conducted, configuration management, etc.
- Monitor the development process to ensure that standards are being followed.
- Report on the process to project management and software procurer.
- Don't use inappropriate practices simply because standards have been established.

Quality assurance and standards

- Standards are the key to effective quality management.
- They may be international, national, organizational or project standards.
- **Product standards** define characteristics that all components should exhibit e.g. a common programming style.
- **Process standards** define how the software process should be enacted.

Product and process standards

Product standards	Process standards
Design review form	Design review conduct
Requirements document structure	Submission of documents to CM
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 manual work is often involved to maintain the documentation associated with the standards.

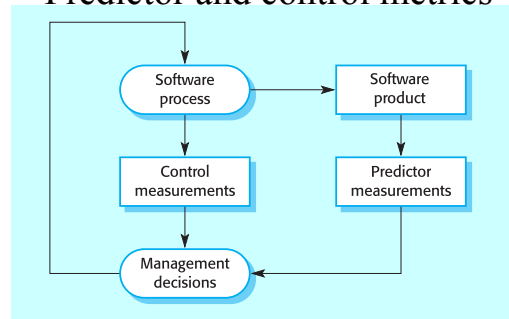
Software measurement and metrics

- Software measurement is concerned with deriving a numeric value for an attribute of a software product or process.
- This allows for objective comparisons between techniques and processes.
- Although some companies have introduced measurement programmes, most organisations still don't make systematic use of software measurement.
- There are few established standards in this area.

Software metric

- Any type of measurement which relates to a software system, process or related documentation
 - Lines of code in a program, the Fog index, number of person-days required to develop a component.
- Allow the software and the software process to be quantified.
- May be used to predict product attributes or to control the software process.
- Product metrics can be used for general predictions or to identify anomalous components.

Predictor and control metrics



Metrics assumptions

- A software property can be measured.
- The relationship exists between what we can measure and what we want to know. We can only measure internal attributes but are often more interested in external software attributes.
- This relationship has been formalised and validated.
- It may be difficult to relate what can be measured to desirable external quality attributes.