Software Engineering Standards

According to the IEEE Comp. Soc. Software Engineering Standards Committee a standard can be:

- An object or measure of comparison that defines or represents the magnitude of a unit.
- A characterisation that establishes allowable tolerances or constraints for categories of items,
- A degree or level of required excellence or attainment.

Why Bother with Standards?

Prevents idiosyncrasy : *e.g.* Standards for primitives in programming languages).

Repeatability : *e.g.* Repeating complex inspection processes.

Consensus wisdom : *e.g.* Software metrics.

Cross-specialisation : *e.g.* Software safety standards.

Customer protection : *e.g.* Quality assurance standards.

Professional discipline : *e.g.* V & V standards.

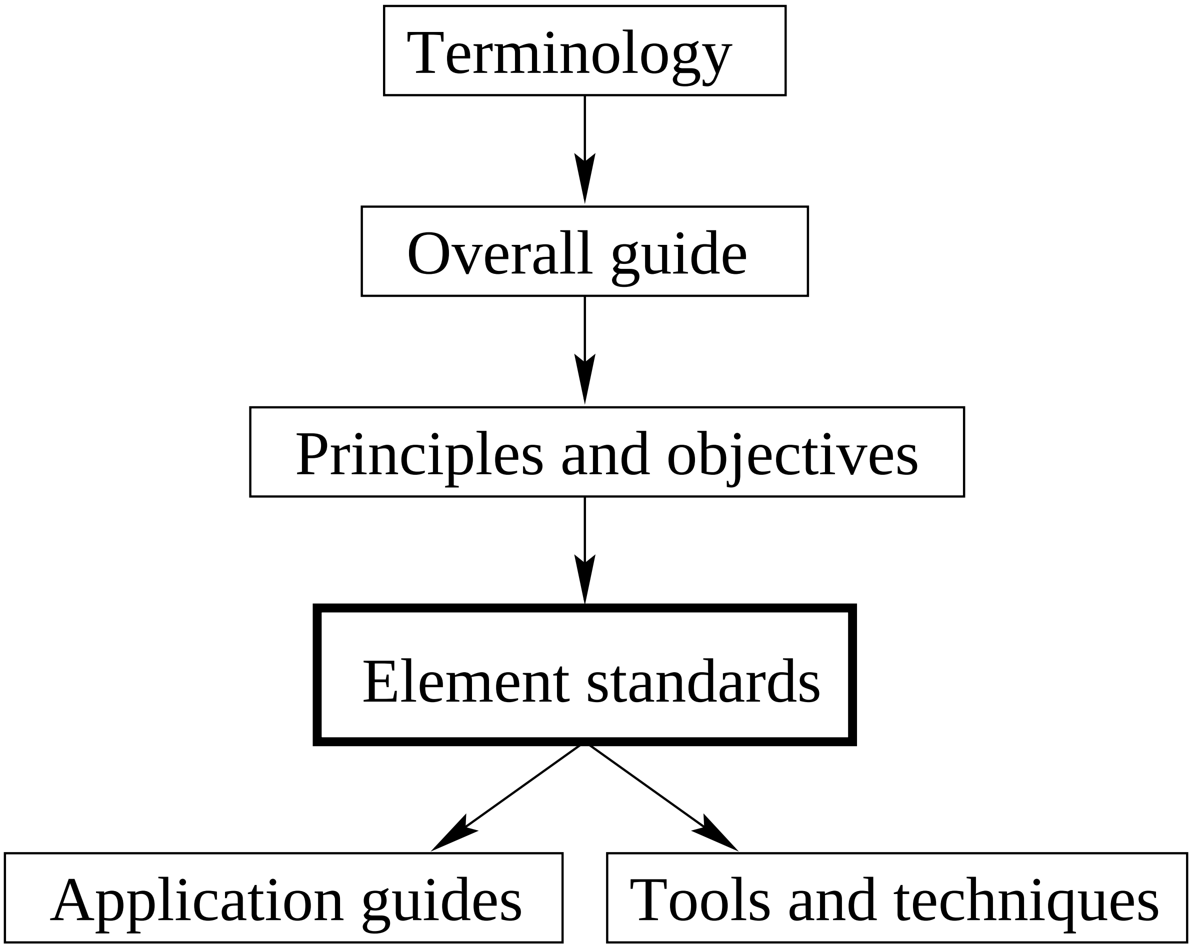
Badging : *e.g.* Capability Maturity Model levels.

Legal Implications

Comparatively few software products are forced by law to comply with specific standards. Most software products have comprehensive non-warranty disclaimers. However:

- For particularly sensitive applications (*e.g.* safety critical) your software will have to meet certain standards before purchase.
- US courts have used voluntary standards to establish a supplier's "duty of care".
- Adherence to standards is a strong defence against negligence claims (admissible in court in most US states).
- There are instances of faults in products being traced back to faults in standards, so
- standards writers must themselves be vigilant against malpractice suits.

Levels of Standards



Some Standards Organisations

ANSI : American National Standards Institute. Does not itself make standards but approves them.

AIAA : American Institute of Aeronautics and Astronautics (*e.g.* AIAA R-013-1992 Recommended Practice for Software Reliability).

EIA : Electronic Industries Association (*e.g.* EIA/IS-632 Systems Engineering)

IEC : International Electrotechnical Commission (*e.g.* IEC 61508 Functional Safety - Safety-Related Systems)

IEEE : Institute of Electrical and Electronics Engineers Computer Society Software Engineering Standards Committee (*e.g.* IEEE Std 1228-1994 Standard for Software Safety Plans)

ISO : International Organisation for Standardisation (*e.g.* ISO/IEC 2382-7:1989 Vocabulary-Part 7: Computer Programming)

Computer Science Standards

Surprisingly few CS standards exist, although one could argue this is because CS is pervasive in others.

Examples:

Terminology : IEEE Std 610.12:1990
Standard Glossary of Software Engineering Terminology.

Techniques : ISO/IEC 8631:1989 Program
Constructs and Conventions for their Representation.

Quality Assurance Standards

Differing views of quality standards: taking a systems view (that good management systems yield high quality); and taking an analytical view (that good measurement frameworks yield high quality).

Examples:

Quality management : ISO 9000-3 Quality Management and Quality Assurance Standards - Part 3: Guidelines for the application of 9001 to the development, supply, installation and maintenance of computer software.

Quality measurement : IEEE Std 1061-1992 Standard for Software Quality Metrics Methodology.

Project Management Standards

These are concerned with how general principles of good management are applied to specific areas of software engineering.

Examples:

General project management :
IEE Std 1058.1-1987 Standard for Software Project Management Plans.

Producing plans : IEEE Std 1059-1993 Guide for Software Verification and Validation Plans.

Systems Engineering Standards

Particular application domains develop sophisticated interactions between system and software engineering, so standardising from a systems point of view can be beneficial.

Examples:

Lifecycle : ISO/IEC WD 15288 System Life Cycle Processes

Requirements : IEEE Std 1233-1996 Guide for Developing System Requirements Specifications

Dependability Standards

As hardware dependability has improved, software has received more attention as a dependability risk. Dependability of software isn't just a question of internal measures (*e.g.* availability, reliability) but also broader issues (*e.g.* maintainability, system context).

Dependability standards often set integrity levels necessary to maintain system risks within acceptable limits.

Examples:

Dependability management : IEC 300-1(1993) Dependability management Part 1: Dependability programme management.

Risk analysis : IEC 1025(1990) Fault Tree Analysis

Reliability : AIAA R-013-1992 Recommended Practice for Software Reliability

Safety Standards

These traditionally come out of specific industrial sectors (*e.g.* American Nuclear Society, UK Ministry of Defence), since safety requires deep analysis of the domain as well as the technology.

Examples:

Safety plans : IEEE Std 1228-1994 Standard for Software Safety Plans.

Functional safety : IEC 61508 Functional Safety - Safety-Related Systems.

Nuclear domain : IEE 603 Criteria for Safety Systems of Nuclear Plants

Resources Standards

Although software engineering is in flux, it is possible to standardise on some forms of resources which are used widely across applications.

Examples:

Terminology : IEEE 610,12-1990 Standard Glossary of Software Engineering terminology

Semantics : IEEE P1320.1 Standard Syntax ans Semantics for IDEF0.

Re-use libraries : AIAA G-010-1993 Guide for Reusable Software: Assessment Criteria for Aerospace Application

Tools : ISO/IEC 14102:1995 Guideline for the Evaluation and Selection of CASE tools.

Product Standards

These focus on the products of software engineering, rather than on the processes used to obtain them. Perhaps surprisingly, product standards seem difficult to obtain.

Examples:

Product evaluation : ISO/IEC 14598 Software product evaluation.

Packaging : ISO/IEC 12119:1994 Software Packages - Quality Requirements and Testing.

Process Standards

A popular focus of standardisation, partly because product standardisation is elusive and partly because much has been gained by refining process. Much of software engineering is in fact the study of process.

Examples:

Life cycle : ISO/IEC 12207:1995 Information Technology - Software Life Cycle Processes.

Acquisition : ISO/IEC 15026 System and software Integrity Levels.

Maintenance : IEEE Std 1219-1992 Standard for Software Maintenance.

Productivity : IEE Std 1045-1992 Standard for Software Productivity Metrics.

Company Guidelines

Specific companies may develop their own guidelines for system/software design. These define good practice within a company. They often conform to more general standards.

Example: Shell UK Code of Practice: Fire and Gas Detection and Alarm Systems for Offshore Installations. Describes what a fire and gas alarm system must do; prescribes properties of that system; sets goals for achieving those properties; gives examples of typical design solutions.

Trends

- Concern about absence of scientific foundation for standards.
- Recognition that standards usually aren't isolated.
- Questioning of software (non)warranty agreements.