SOFTWARE ENGINEERING CONCEPTS

An Introduction for Computational Scientists

18 May 2003

Dr Michael Papasimeon

SOFTWARE ENGINEERING

- Foundations in System Science and Systems Engineering
- The systematic application of science and management to build and deliver software systems with sufficient quality, that satisfy the requirements and are built on time and budget.
- Software systems are the most complex systems built by humans.
- Software Engineering aims to manage this complexity.

LIFECYCLE PHASES

- Business/Domain Modelling
- Requirements
- Analysis and Design (Architectural and Detailed)
- Implementation (Coding and Integration)
- Testing
- Deployment
- Configuration Management
- Project Management

LIFECYCLE MODELS

- Waterfall
- Spiral
- Agile
- Modern software engineering processes are:
    - Iterative and incremental
    - Requirements/functionality driven
    - Architecture driven (importance of design)

FAILURE RATE: HARDWARE VS SOFTWARE

IMPACT OF CHANGE

SOFTWARE QUALITY FACTORS

- Correctness
- Reliability
- Robustness
- Usability
- Efficiency
- Modularity
- Reusability
- Traceability
- Maintainability
- Extendibility

Measuring Software Quality Factors

- Directly Measurable (e.g. correctness, reliability)
- Indirectly Measurable (e.g. usability)

ASSURING SOFTWARE QUALITY

- Sofware Engineering Methods
- Formal Techncial Reviews
- Software Configuration Managment
- Software Quality Assurance
- Verification and Validation
- Standards and Procedures
- Testing

SOFTWARE REUSE

- **Ad-Hoc:** Incidental reuse that occurs through movement of staff through an organisation.
- **Opportunistic:** Organisation taking advantage of reuse opportunities through smart/intelligent design.
- **Integrated:** When opportunities are sought out as part of an organisation's development process.
- **Leveraged:** Assumes that a given product is part of product family, the members of which have something in common. Construct domains that take in the product family.
- **Anticipated:** Construct domains that anticipate needs

COUPLING AND COHESION

Inter-Module Coupling

The measure of the interdependence of one module to another. Modules should have low coupling. Low coupling minimizes the "ripple effect" where changes in one module cause errors in other modules.

Intra-Module Cohesion

The measure of strength of the association of elements within a module. Modules whose elements are strongly and genuinely related to each other are desired. A module should be highly cohesive.

WHY IS QUALITY IMPORTANT?

The level of quality required depends on the level of impact when risks occur.

- Mission Critical Systems
  - Safety Critical
  - Security Critical
- Decision Critical Systems
  - Longer term, safety, security and financial issues
- Customer/Financial/Enterprise Critical
  - Shorter Term Critical Systems

VERIFICATION AND VALIDATION

Verification: “Are we building the system right?”

The process of determining that a model implementation accurately represents the developer's conceptual description of the model and the solution to the model.

Validation: “Are we building the right system?”

The process of determining the degree to which a model is an accurate representation of the real world from the perspective of the intended uses of the model.

Verification and Validation Documentation

- **SVVP** Software Verification and Validation Plan
- **SVVR** Software Verificatioi and Validation Report

COMPUTATIONAL SCIENCE AND SOFTWARE ENGINEERING

COMPUTATIONAL SCIENCE AND SOFTWARE ENGINEERING (2)

CLIENT INTERACTION