Seminar 7
Documentation and Testing
Semester 1, 2004

Lecture Objectives

**Documentation**
- What is it?
  - User Documentation
    - Purpose, Report Format
  - Programming Documentation
    - Purpose, Report Format
  - General Principles for Good Documentation

**Testing**
- What it is.
- Testing Process
- Test Planning
- Testing Strategies

**Documenting**
- User Documentation
  - Purpose
  - Report Format (an example)
    - Specify conventions used
    - System Requirements
    - Software Installation
    - Using the system
      - Screen shots with explanation of purpose, inputs, and outputs
      - May have different sub-sections for different system functionality
    - Contact Details
    - Glossary of Terms
    - Trouble Shooting

**User Documentation**
- Purpose
- What it is.
  - ensure successful first encounters with a system which lead to greater acceptance and use of the system
  - reduces the need to refer problems to system developers
  - overcomes users' fears of equipment and software
  - enables users to find what they want and understand it when they find it
  - improves efficiency, as people understand the system they are working with

User Documentation

Purpose
- A document that explains to the user how to use the system. Could be used as a training tool, a reference for non-technical people or a manual for a new user.

Contents:
- what the system is about (narrative);
- how to use the system (start-up, shut-down, solve problems);
- how to carry out tasks - details of manual procedures involved;
- Common mistakes made;
- how to ensure security;
- how to perform backup and recovery.

User Documentation

Report Format (an example)
Program Documentation

Report Format (an example)

• **Operations overview**
  - Specify technology, platforms, hardware, libraries
  - Directory structure (source code, .exe, databases, etc)

• **Data Details**
  - Naming conventions (files, variable, classes, objects, etc)
  - Revised ERD
  - For databases, identify primary keys, composite keys and foreign keys
  - Description of fields, records, tables
  - Tiggers (special functions that are automatically triggered)
  - Data dictionary

• **Program Specification Information**
  - May include DFD’s or UML (Unified Modelling Language – another way of modelling OO System)
  - Ucase, Class Diagrams, State Diagrams, Sequence Diagrams, Object Diagrams, Data Diagrams.

Documentation

Report Writing in general

• **Cover page**
• **TOC, (table of tables, table of Figures)**
• All tables and figures should be labelled
• **Hierarchy of sections**
• **Version Control**
• Pages should be numbered
• **Indexing**

General principles for good documentation

- is written for the intended audience and purpose;
- has a consistent layout that clarifies the structure of the document;
- uses an appropriate layout for the type of material;
- highlights important points;
- avoids jargon, or where jargon is necessary gives definitions or explanations;
- uses clear examples that are easy to visualise;
- is neither wordy and verbose nor too brief and concise;
- has good reference aids (table of contents, thorough index, cross-referencing);
- is easy to update;
- is produced in an easy-to-manage physical format.

Testing

• **What is testing?**
• **The testing process**
• **Test planning**
• **Test strategies**
What is testing?

- Testing is about verification and validation.
  - Verification
    - Are we building the product right?
  - Validation
    - Are we building the right product?

- Testing Objectives
  - The discovery of defects in the system
  - The assessment of whether or not the system is usable in an operational situation

The testing process

- The testing process may consist of five stages:
  - Unit Testing
  - Module testing
  - Sub-system testing
  - System testing
  - Acceptance Testing

The testing process

- Unit Testing
  - Testing of individual components/functions
  - Eg: testing each coded individual level 3 DFD; add author/paper

- Module Testing
  - A module is a collection of individual components such as an object or some looser collection of procedures and functions
  - Eg: testing of coded level 1 DFD; manage paper/author

The testing process

- Sub-system testing
  - Involves testing collections of modules which have been integrated into sub-system.
  - Sub-systems may be independently designed and implemented
  - Most common problems are sub-system interface mismatches
  - Eg: manage authors and generate papers authors submitted

The Testing process

- System Testing
  - The subsystems are integrated to make up an entire system
  - Eg: integration of manage papers/authors, manage reviewers, manage reviews, generate reports

The Testing Process

- Acceptance Testing
  - Final stage of testing before system is accepted for operational use
  - Involves testing the system with data supplied by the system procurer rather than simulated data developed as part of the testing process
  - Reveals errors and omissions in the systems requirements definition
Test Planning
Major Components of a test plan

- The testing process
  - Description of major phases of testing process
- Requirements traceability
  - Test that all system requirements are met
- Tested Items
  - Specify what is tested
- Testing schedule
  - Devise testing schedule – link to project development schedule
- Test recording procedure
  - Systematically record results of all tests; for auditing
- Hardware and software requirements
- Constraints
  - Identify constraints affecting the testing process

[p379, Sommerville]

Test Plan documents

- Are NOT a static document
- Revised regularly as testing is an activity which is dependent on implementation being complete

Testing strategies

- Top-down testing
- Bottom up testing
- Thread testing
- Stress testing

Top-down testing

- Involves starting at the sub-system level with module represented by stubs
- Stubs are simple components which have the same interface as the module
- After sub-system testing is complete each module is tested in the same way. The functions are represented as stubs.
- Finally, the program components are replaced by the actual code and this is tested
- When to do it, Advantages, Disadvantages

Bottom Up testing

- Involves testing modules at the lower levels in the hierarchy, and then working up the hierarchy of modules until the final module is tested
- Need to write test-driver
- When to do it, Advantages, Disadvantages

Thread testing

- Real-time systems are usually made up of a number of cooperating processes and maybe interrupt driven.
- Involves identifying and executing each possible thread
Stress Testing

- Some classes of system are designed to handle a specific load. Eg: bank transaction processing system may be designed to process 100 transactions per sec
- Stress testing continues these tests beyond the max. design load of the system. Loading is steadily increased until system fails.
- This type of testing has a dual function:
  - It tests the failure behaviour of the system
  - It stresses the system and may cause defects to come into light that would not normally manifest themselves

References

  
  See Index