Chap 2. Introduction to Software Testing

2.2 Test Management

1. Test Documentation
2. Test Planning
3. Test Status Reporting
4. Defect Reporting

1. Test Documentation

-Maintaining proper documentation is an important aspect of any testing effort.

-Purpose of Test documentation:

÷ Plan and guide the testing activities
÷ Report on test progress
÷ Used to assess product quality and to achieve conformance with standards and regulations
÷ Serve as management decision-making tool

-Test documentation include:

÷Test plan
÷Defect report
÷Test status report
÷(Final) Test report
÷Test case specification and design document

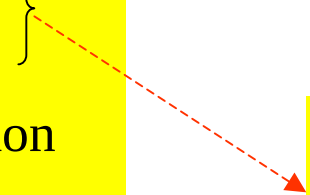
-Various test documentation templates and standards are available:

÷Main issue to consider: the cost of maintaining such documentation, which should be weighed against the product requirements.

IEEE Standard 829 for Software Test Documentation

- IEEE Standard 829 for software test documentation is a standard initially published by the IEEE in 1983 and later approved by the American National Standards Institute (ANSI).
- The standard describes a wide range of types of information that can be included in test documentation.
- The IEEE standard 829 define the following test document templates:

- Test plan
- Test-design specification
- Test-case specification
- Test-procedure specification
- Test-item transmittal report
- Test-log



–Test-case specification identifier
–Test items
–Input specifications
–Environmental needs
–Output specifications
–Special procedural requirements
–Intercase dependencies

Test Case Template

Build Number	
Tester Name	
Test Type	
Test Case Name	
Test Case Number	
Test Case Description	
Items to be tested	
1	
…	
n	
Specifications	
Input	**Expected Output**
Procedural Steps	
1	
…	
m	

2. Test Planning

-Test planning starts at the beginning of the development process; it should represent the first task in any testing process.

-The test plan defines the road map to be followed during the testing process.

÷Provide background information on the system under testing,
÷Specify the test objectives,
÷Identify and describe the test and risks factors involved.
÷Provide a description of the components and functions to be tested, and the nature of the tests to be conducted,
÷Identify the potential members of the test team and their responsibilities.

-Typical test planning involves the following tasks:

1. Identify risks factors, which will guide the testing efforts
2. Identify test phases
3. Map risks factors to test phases
4. Specify for each test phase, the tasks to be done and the workers assigned
5. Identify test resources, environments, and tools.
6. Determine preliminary test schedule and estimate initial test costs

IEEE std 829-1983 Test Plan Template

Test Plan

1. **Introduction**
2. **Test Items**
3. **Features To Be Tested**
4. **Features Not To Be Tested**
5. **Approach**
    - 5.1 Code Inspection
    - 5.2 Unit Testing
    - 5.3 System Testing
    - 5.4 Regression Testing
    - 5.5 Acceptance Testing
6. **Item Pass/Fail Criteria**
7. **Suspension Criteria and Resumption Requirements**
8. **Test Deliverables**
9. **Testing Tasks**
10. **Environmental Needs**
11. **Responsibilities**
12. **Staffing and Training Needs**
13. **Schedule**
14. **Risks and Contingencies**
15. **Approvals**

•To Do List

Revision History

Test Responsibilities and Roles

Test Manager is tasked with the overall responsibility for the test effort's success. He is the the primary person in charge of advocating and assessing product quality.

Test Analyst is responsible for initially identifying and defining the required tests, and subsequently evaluating the results of the test effort.

Test Designer is responsible for defining the test approach and ensuring its successful implementation.

Tester is responsible for the core activities of the test effort, which involves conducting the necessary tests and logging the outcomes of that testing.

Defining the Test Approach

- The test approach (or "testing strategy") specifies the techniques that will be used to accomplish the test mission.
- The test approach also specifies how the techniques will be used.
- A good test approach is:
  - Diversified
  - Risk-focused
  - Product-specific
  - Practical
  - Defensible

2. Test Status Reporting

-Reporting the test results is one of the most important tasks in the software testing process.

÷Give the management an effective decision-making tool.

-The project ***test status report*** addresses issues that are of high interest to management such as:

÷When the final product will be released,

÷Whether or not enough testing has been achieved,

÷How reliable will the system be.

-Since the test (status) report serves as a management decision-making tool, it should be short, concise, and easy to read.

Status Reporting

-Allows assessing the progress of the *test mission* and the status of the *test effort.*

- Key questions: How far are we? How much is left to do?
  - Experienced test managers may have very different answers

- Complex, multidimensional question
  - –Many types of data explain “extent of testing”
  - –Simple metrics are often profoundly misleading
  - –The best status reports consider several dimensions together

The Overall Structure of a Status Report

- Here's one structure that some managers find works well for them:
  - The report has four parts, each part starts a separate page.
  - Part 1 Risks and responsibilities
  - Part 2 Progress against plan
  - Part 3 Project bug metrics
  - Part 4 Deferred and no-fix bugs to approve

- *Part 1: Risks and responsibilities*
  - Highlights current problems, such as:
    - Artifacts due into testing but not arrived
    - Artifacts that due out of testing but not yet completed
    - Staff turnover that threatens the schedule
    - Equipment acquisition problems that might threaten the schedule.

The Overall Structure of a Common Report

- *Part 2: Progress against plan*

Can be documented in the following table:

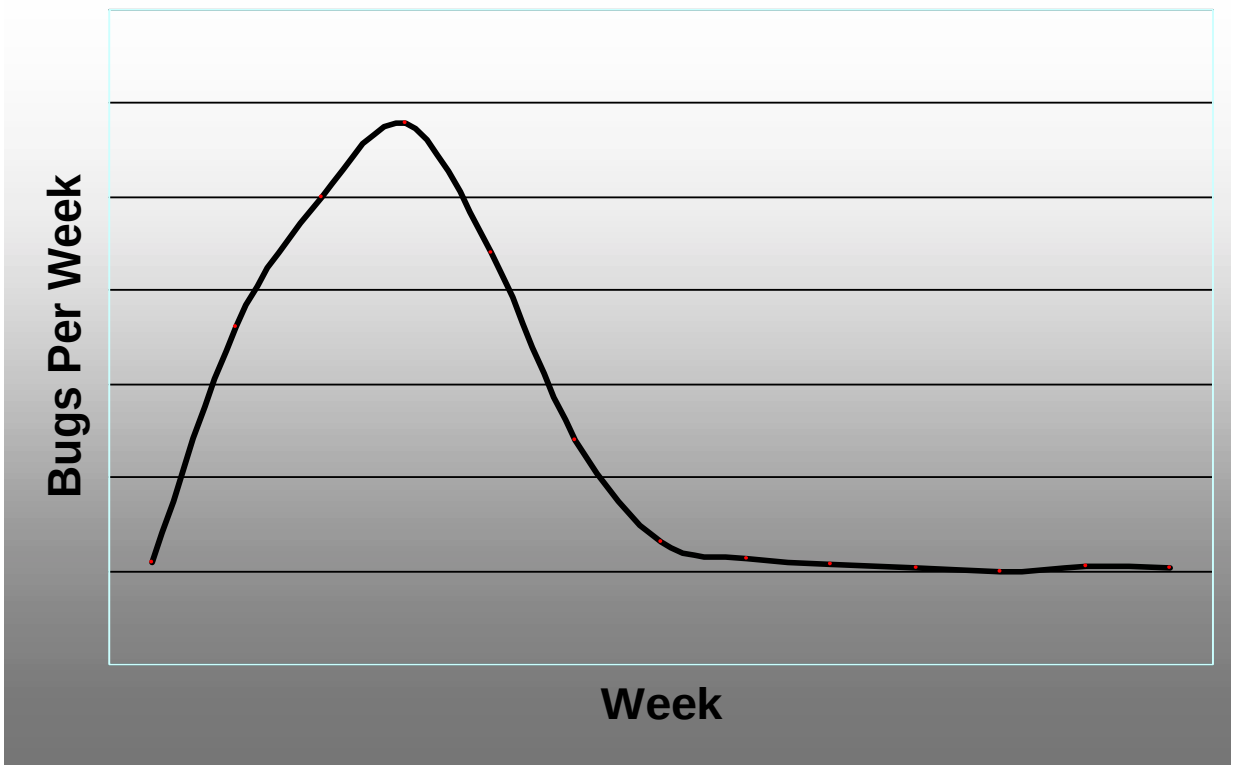
Component	Test Type	Tester	Total Tests Planned / Created	Tests Passed / Failed / Blocked	Time Budget	Time Spent	Projected effort for Next Build	Notes

- Note how this covers progress against a plan, risks/obstacles, effort and results, all in one chart

The Overall Structure of a Common Report

- *Part 3: Project bug metrics*
  - These charts show find / fix rates over the course of the project.
  - Useful to give a sense of the rate at which problems are being repaired.
  - If the repair rate near the end of the project is slow compared to the find rate, the schedule is at risk.
  - It is too easy to over-interpret these charts

Bug Counts and Extent of Testing?



- Attempt to measure testing progress by plotting a project's bug numbers against a theoretical curve of expected find rates over time.

Potential Side Effects of Defect Curves

- Earlier in testing: Pressure is to increase bug counts
  - –Run tests of features known to be broken or incomplete.
  - –Run multiple related tests to find multiple related bugs.
  - –Look for easy bugs in high quantities rather than hard bugs.
  - –Less emphasis on infrastructure, automation architecture, tools and more emphasis of bug finding. (Short term payoff but long term inefficiency.)
- Later in testing: Pressure to decrease find rate
  - –Run lots of already-run regression tests
  - –Don’t look as hard for new bugs.
  - –Shift focus to appraisal, status reporting.
  - –Postpone bug reporting until after the measurement checkpoint (milestone).
  - –Report bugs informally, outside of tracking system
  - –More bugs are rejected.
- So, sometimes bug curves can be counterintuitive
  - Sometimes, a drop in bug find rate reflects the declining efficiency of a given style of testing or over reliance on a specific technique.
  - Perhaps the better solution, as bug rates drop, is to switch to a more powerful technique.

The Overall Structure of a Common Report

- *Part 4: Deferred and no-change change requests*
  - Every project team fixes some bugs and rejects or defers others.
    - At some point, there must be management review of the collection of problems that will not be fixed.
  - Rather than save up the list for the end of the project, list the new not-to-be-fixed change requests every week.

Software Test Report (STR) Template

-Test status reports are produced during the test process

-At the end of the process, an overall test report summarizing and analyzing the results should be produced as well.

1. SCOPE
- 1.1 IDENTIFICATION
- 1.2 SYSTEM OVERVIEW
- 1.3 DOCUMENT OVERVIEW

2. REFERENCED DOCUMENTS

3. OVERVIEW OF TEST RESULTS
- 3.1 OVERALL ASSESSMENT OF THE SOFTWARE TESTED
- 3.2 IMPACT OF TEST ENVIRONMENT
- 3.3 RECOMMENDED IMPROVEMENTS

4. DETAILED TEST RESULTS
- 4.1 [PROJECT-UNIQUE IDENTIFIER OF A TEST]
  - 4.1.1 Summary of Test Results
  - 4.1.2 Problems Encountered
    - 4.1.2.1 [Project-Unique Identifier of a Test Case]
  - 4.1.3 Deviations from Test Cases/ Procedures
    - 4.1.3.1 [Project-Unique identifier of a test case]

5. TEST LOG

6. NOTES
- 6.1 ABBREVIATIONS AND ACRONYMS

7. APPENDIX A. [ADDITIONAL DATA]

3. Defect Reporting

-A defect report is a tool that you use to convince someone to allocate time and energy to fix a bug.

-Terminologies:

- –*Change request*: any report of an incident, defect or potential enhancement, that is intended as a request for a change to the software under development
- –*Defect report*: a change request reporting a (suspected) defect or error in the product
- –*Bug*: some aspect of the product under test, that in the eyes of a stakeholder, unnecessarily reduces its value; possibly a suspected defect

Defect Report Form

Software system being tested:

Date:

Defect found (name/type):

Location found (unit/module):

Severity of Defect: Critical
Major
Minor

Type of Defect: Missing
Wrong
Extra

Test Data/Script Locating Defect:

Origin of Defect/Phase of Development:

Date Corrected:

Date for Retest:

Result of Retest:

Software s/system tested: name of the software being tested

Date: date on which the test occurred

Defect found (Name/Type): the name and type of a single defect found in the software being tested.

Location found (Unit/Module): The individual unit or system module in which the defect was found.

Severity of defect: Critical, major, or minor.

-Critical: the system cannot run without correction;

-Major: the defect will impact the accuracy of operation;

-Minor: it will not impact the operation.

Type of defect: whether the defect represents something missing, something wrong, or something extra.

Defect Report Form (ctd.)

Software system being tested:

Date:

Defect found (name/type):

Location found (unit/module):

Severity of Defect: Critical
Major
Minor

Type of Defect: Missing
Wrong
Extra

Test Data/Script Locating Defect:

Origin of Defect/Phase of Development:

Date Corrected:

Date for Retest:

Result of Retest:

Test data/script locating defect: which test was used to uncover the defect

Origin of defect/phase of development: the phase in which the defect occurred.

Date corrected: defect correction date.

Retest date: the date on which the testers were scheduled to validate whether the defect had been corrected.

Result of retest: whether the software system functions correctly and the defect no longer exists; or if additional correction and testing will be required. If so, the “To be added later” section will need to be reentered.