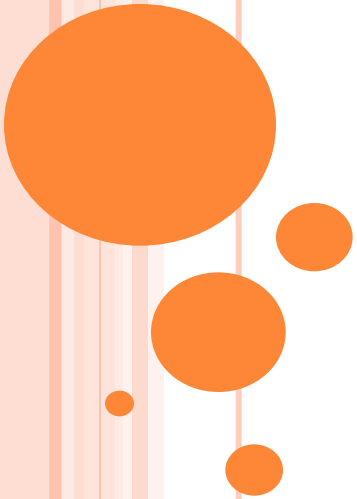
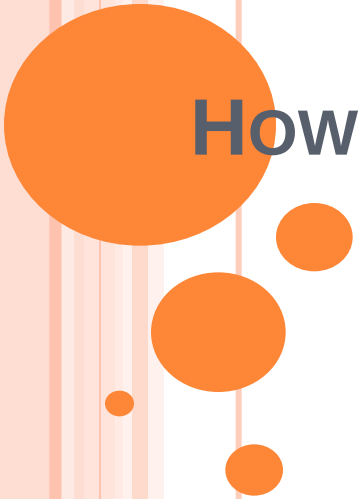
Software Engineering



Lecture-43

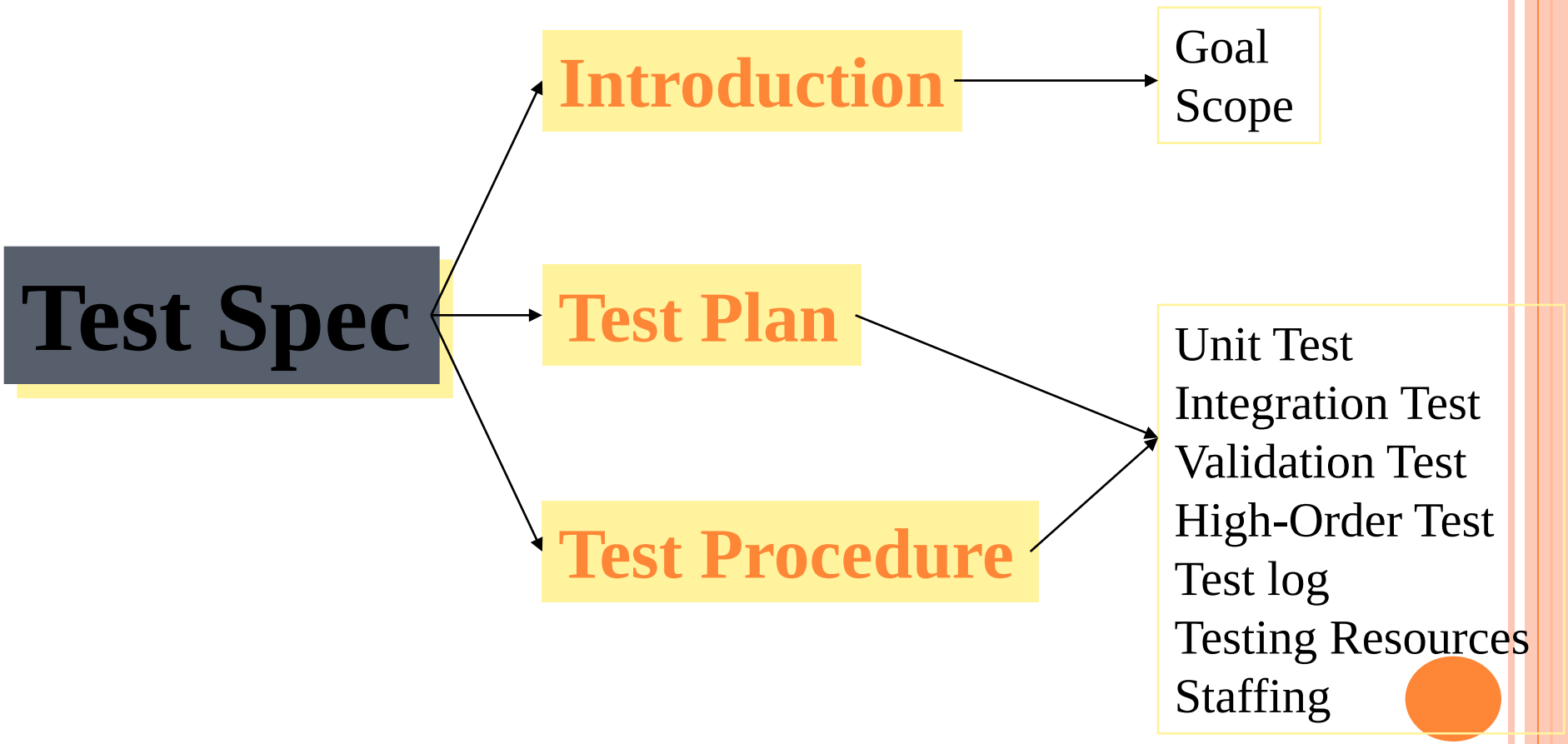


How to Write a Test Plan?

Topics Covered

- Test Specification
- Types of Testing
- Goals of WMITS
- Goals and Objects
- Major Constraints

What Must be Included?



Test Specification - Major Items

1.0 Introduction

This section provides an overview of the entire test document. This document describes both the test plan and the test procedure.

1.1 Goals and objectives

Overall goals and objectives of the test process are described.

1.2 Statement of scope

A description of the scope of software testing is developed. Functionality/features/behavior to be tested is noted. In addition any functionality/features/behavior that is not to be tested is also noted.

1.3 Major constraints

Any business, product line or technical constraints that will impact the manner in which the software is to be tested are noted here.

2.0 Test Plan

This section describes the overall testing strategy and the project management issues that are required to properly execute effective tests.

2.1 Software to be tested

The software to be tested is identified by name. Exclusions are noted explicitly.

2.2 Testing strategy

The overall strategy for software testing is described.

2.2.1 Unit testing

The strategy for unit tested is described. This includes an indication of the components that will undergo unit tests or the criteria to be used to select components for unit test. Test cases are NOT **included here.**

2.2.2 Integration testing

The integration testing strategy is specified. This section includes a discussion of the order of integration by software function. Test cases are NOT included here.

2.2.3 Validation testing

The validation testing strategy is specified. This section includes a discussion of the order of validation by software function. Test cases are NOT included here.

3.0 Test Procedure

This section describes as detailed test procedure including test tactics and test cases for the software.

3.1 Software to be tested

The software to be tested is identified by name. Exclusions are noted explicitly.

3.2 Testing procedure

The overall procedure for software testing is described.

3.2.1 Unit test cases

The procedure for unit testing is described for each software component (that will be unit tested) is presented. This section is repeated for all components i.

3.2.1.2 Stubs and/or drivers for component i

3.2.2 Integration testing

The integration testing procedure is specified.

3.2.2.1 Testing procedure for integration

3.2.2.2 Stubs and drivers required

3.2.2.3 Test cases and their purpose

3.2.2.4 Expected results

3.2.3 Validation testing

The validation testing procedure is specified.

3.2.3.1 Testing procedure for validation

3.2.3.3 Expected results

3.2.3.4 Pass/fail criterion for all validation tests

3.2.4 High-order testing

The high-order testing procedure is specified. For each of the high order tests specified below, the test

Case Study

- WMITS (Waste Management Inspection Tracking System) Software Requirements Specifications
- 1.1 Goals and Objectives

  The main purpose of WMITS is to help automate the entire process that the Department of Environmental Quality (DEQ) Waste Management Division (WMD) staff members perform throughout an inspection.

The goals of WMITS are:

- To minimize the time span of any inspection
- To minimize the amount of paper work required
- To provide a searchable database of all past inspections
- To provide an automated channel for the public to request information (under Freedom of Information Act)

System Context

- Eventually, multiple users will be using the product simultaneously. Therefore, concurrent connection will be an issue for implementation. In addition, this is a pilot product that hopefully, if successful, can be used in other locations as well. This leads to issues about future support for a larger user base.

1.0 Introduction

- This section gives a general overview of the Test Specification for the Waste Management Inspection Tracking System (WMITS).

1.1 Goals and Objects

- Put it in a simple way, a good product will work perfectly, doing the right thing at the right time. To do that, the software has to go through a series of tests before its final release. Error free software is extremely difficult to achieve. After all, nothing is perfect. Especially for software developed in a short time frame. But high quality can be achieved with a detailed test specification. All (or least most) of the test case will be listed, the development team will follow it step by step, item by item, to test all the necessary objects, data flows, limits, boundaries, and constraints of the software.

1.1 Goals and Objects

- Cyber Rovers would like to have a test specification to counter any difficulties that may impact the development and the future performance of the software. The team's goal is to assist the project team in developing a strategy to deal with any errors. For this, the team will take a look at the most common errors to some very uncommon errors as well.

1.3 Major Constraints

- In this section we will talk about the business, technical or resource related constraint that may keep us from performing all tests necessary.
- 1. The team has limitation on time to test the product at the client's facility. We have access to the facility only during the regular office hours. We also have to set us schedule around the available time of the inspector that is to help us, so time schedule will be a major constraint when we talk about testing at the site.

1.3 Major Constraints

- 2. The team also has got funding for only one hand held PC. This means that we cannot test the software using the PC from some other brand or PC that is of lesser price and lower hardware.
- 3. The team does not know any hacker that can help us test the security problems. So we have to rely on our own knowledge and have to trust the software for the security.

1.3 Major Constraints

- 4. The team also does not have large enough group to have many people use the applications at the same time to perform real stress related testing. So we will not be able to test the product for the larger user base.

- Critique: Each of these constraints represents a significant product quality risk.The team should consider risk mitigation strategies.

2.0 Testing Plan

- We want the product to be bug free. We also want to make sure that there are no defects in the product. So we will be spending large amount of the total software development time on the testing. Below is the description of the testing procedure and strategy. We will also be presenting the timing and scheduled of the tests to be carried out.
  - 2.1 Software to be tested

2.1.1 Interfaces

- Login Window
- We will make use of several different names to log in to the system, so will be testing login window. We will also test OK and Cancel buttons on this window by performing test above.
- DEQ – Microsoft Visual Basic [Design] Window
- This is the main window that we will use to access the database using Visual Basic. We will have several different drop-down menus in this window. File, Facility, Inspection, approve, Reports, Maintenance and Help are the drop down menu that will be available in this window we will try to use all the menus and than different options available in each of the window.

2.2 Testing Strategy

- In the following section we will describe the testing strategy. We will use four different methods to test our product.

2.2.1 Unit Testing

- In the unit test case we will be testing the separate modules of the software. We will carry out white box testing where each module or component of the software is tested individually. We will test the components by passing data through it and we will be monitoring data to find the errors.
- We will be looking for entry and exit conditions of the data. We will make sure that all the components work without any troubles. The test primarily be carried out by the programmer who designed and implemented the module. Lead tester will than carry out test on the modules to finalise the testing.

2.2.2 Integration Testing

- In this method of testing we will implement the software at the clients location and will run it. So we will be testing the product on clients network. As part of testing, will be looking for any signs of the collision between our software components and those of the clients. We want to make sure there is no confusion among the application on the network when they are running simultaneously.

2.2.2 Integration Testing

- We will install the software at the clients site and will run it. We will have several different other applications open as well. This applications will be the once that have to interact with our software on normal bases. We will make sure that all the data is saved correctly and there is no loss of data or data base anomalies in the product.
- We will start from the login window and will go through all the all the software functions and will test the. We will be carefully looking for any sort of collision between several different applications

2.2.3 Validation Testing

- In this method of the test we will be working with the customer to find out if the software developed in valid for the clients. We want to make sure that the client is getting what he asked for. We will look at the software requirement document in the case of conflict or misunderstanding with client regarding software components.
- We will perform the black box testing where the software is completed and we test all the software components together. We will have several input data or test data that we will derive results for. We will insert this data in the software and will get results form the software. We will compare the results from the software with the results that we derived. This way will check for the validation of the software.

2.2.3 Validation Testing

- In case there are problems with the software we will create a deficiency list and will record all the problems in there. We will test all the components and subcomponents of the software to perform validation test.
- We have and will try our best so that we don’t have to create deficiency list. This is necessary because if the errors are found at this stage of the software development we cannot fix them by the time we reach the software deliverance date. In this case we have to negotiate with the customer to give us extension on the project.

2.2.4 High-order Testing

- In this test method we will combine several different other types of the testing. We will test for several different conditions by following several different test methods.
- • Recovery testing
  - Here we are concerned with ability of the software to retrieve lost data. We want to make sure that the software is fault tolerance and does not loose data in case of system shutdown or if the system ceases.

2.2.4 High-order Testing

- Security Testing
  - In this method of the test we want to make sure that the security checks are working and no one is able to temper with the data. This is crucial since our software is design to track the activity that is not legal.
- Stress Testing
  - In this test method we want to monitor stress caused to system and the software due to simultaneous use. We want to make sure that the system does not breack down under the extreme use conditions.
- Performance Testing
  - Performance bounds are set during the design part of the software development. These bounds will help us in determining the effectiveness of the software. It will also help us to minimize stress level that is caused to user because of our software.

Test Schedule

- Following is the tentative schedule for the testing of the WMITS.
- Project Test Plan
- –2/9/2000 – 2/15/2000
- This part is straightly theory stage. No any real actions will be performing.
- System Testing
- –3/6/2000 – 3/10/2000
- Generate Testing Report
- –3/7/2000 – 4/7/2000

Testing Resources and Staffing

- We need to have large number of resources available to us in order for us to test the entire software properly.
- Resources
- We will take help of the DEQ staff of the Waste Management Division to help us test the product. We will allow DEQ staff member or members to test the product as part of validation testing. We will allow the DEQ staff to record any errors found in the software and will correct them before the delivery of the software.

Bug Resource Reports

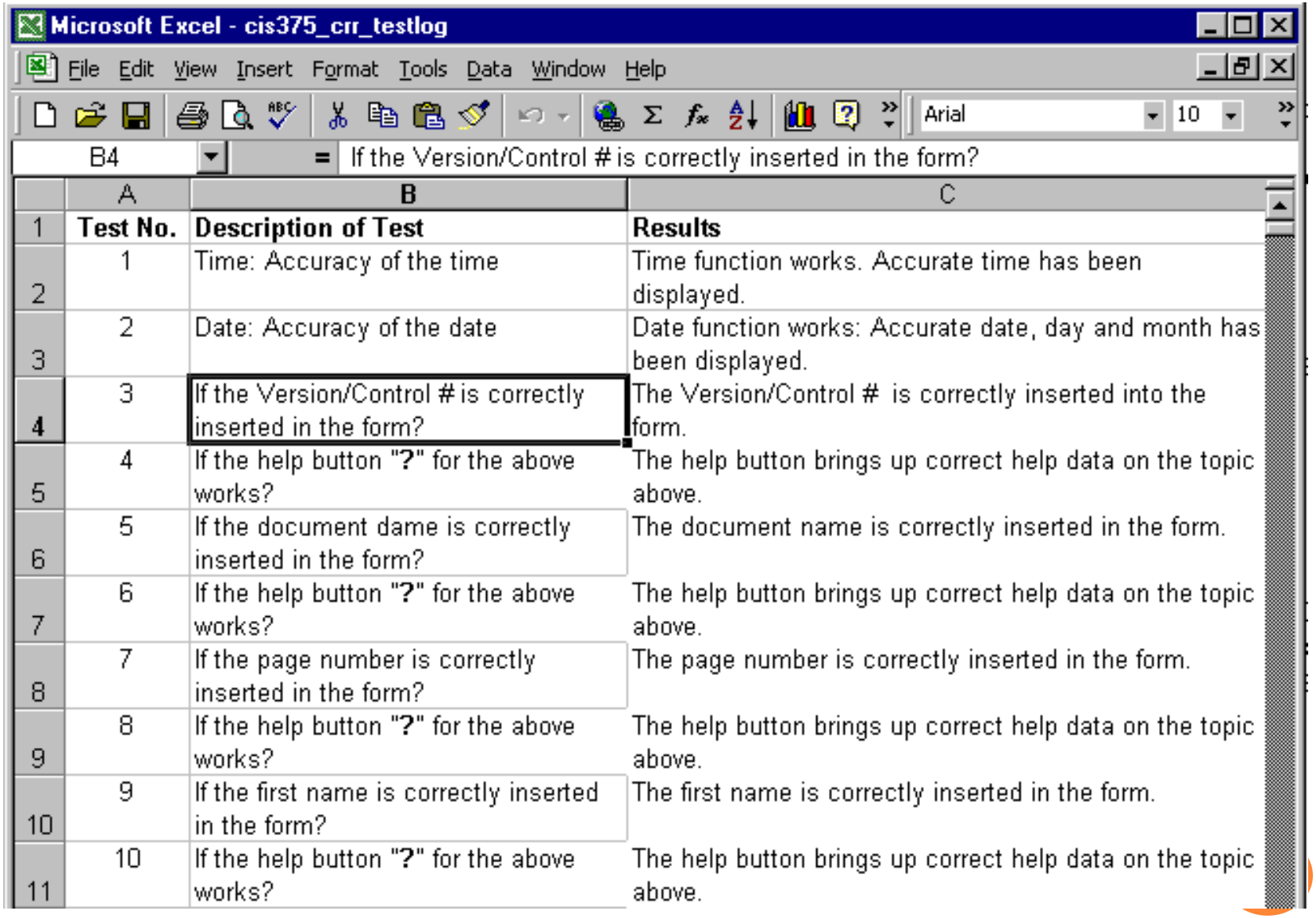
- We will use Bug Resource Report where we will identify the bugs found during the testing and will try to identify the reasons for their occurrence. This will help teams that may work on the product latter to identify the soft spots for the bugs and will help them to come up with way to design products so that bugs are avoided.

Staffing

- We have decided to use simple method for staffing people for the testing. Each program will test the components or functions created by him separately and will hand them over to lead tester. Lead tester will test each component and will make a note of the result in test result table. Once the product is completely developed we all the member of the software project team will test the software with combined effort. DEQ staff will also assist in testing the software.

Test Record Keeping and Log

We will use table created in excel to log all the test, describe them and to record the results of the tests. Below is the example of such table.



	A	B	C
1	**Test No.**	**Description of Test**	**Results**
2	1	Time: Accuracy of the time	Time function works. Accurate time has been displayed.
3	2	Date: Accuracy of the date	Date function works: Accurate date, day and month has been displayed.
4	3	If the Version/Control # is correctly inserted in the form?	The Version/Control # is correctly inserted into the form.
5	4	If the help button "**?**" for the above works?	The help button brings up correct help data on the topic above.
6	5	If the document dame is correctly inserted in the form?	The document name is correctly inserted in the form.
7	6	If the help button "**?**" for the above works?	The help button brings up correct help data on the topic above.
8	7	If the page number is correctly inserted in the form?	The page number is correctly inserted in the form.
9	8	If the help button "**?**" for the above works?	The help button brings up correct help data on the topic above.
10	9	If the first name is correctly inserted in the form?	The first name is correctly inserted in the form.
11	10	If the help button "**?**" for the above works?	The help button brings up correct help data on the topic above.