SET B ANSWERS:

3 . (i) Effort estimation and it's importance.

(ii) Software testing in iterative model.

ANS: (i) For making scheduling, resource planning, and budget for a test project, the test manager should make a good effort estimate. Effort estimate should include information such as project size, productivity, and test strategy. While project size and test strategy information comes after consultation with the customer, the productivity figure comes from experience and knowledge of the team members of the project team. The wideband Delphi technique uses brainstorming sessions to arrive at effort estimate figures after discussing the project details with the project team. This is a good technique because the people who will be assigned the project work will know their own productivity levels and can figure out the size of their assigned project tasks from their own experience. Initial estimates from each team member are then discussed with other team members in an open environment. Each person has his own estimate. These estimates are then unanimously condensed into final estimate figures for each project task.

Effort estimation is one area where no test manager can have a good grasp, at the initial stages of the project. This is because not many details are clear about the project. As the project unfolds, after executing some of its related tasks, things become clearer. At that stage, any test manager can comfortably give an effort estimate for the remaining project tasks.

(ii) In an iterative model, each iteration is a short cycle. So the amount of testing in each iteration is also small. Thus, unlike in waterfall model, software testing has a lesser role in the iterative development life cycle.

Generally, software defects tend to increase with the size of software products. Since in iteration mode the software product is small, there will be fewer defects in the product. Although in reality, as the software product grows in size over many iterations, the number of defects per line of software code is bound to increase. In iterative development, regression testing is also a big issue. In each iteration, there will be a large number of regression test cases to run. As the product size increases with iterations, the set of regression test cases also increases. It becomes a liability after a while. Manually running all those regression tests takes a lot of time, which becomes a hindrance for the release schedule. In such cases, the best option is to go for automation of these regression test cases. Automated test cases take much less time (sometimes if the manual running of test cases was taking 5 days, after automation it took only 5 h) to run.

OR

Various types of maintenance models and explain its characteristics with example.

ANS: Software maintenance process models have been defined and Some of the popular ones include the quick fix model, Boehm's model, Osborne's model, iterative enhancement model, and reuse oriented model.

The Quick Fix Model- This is the simplest of maintenance models; whenever any defects with the software products are found they are immediately fixed. There is no planning involved in the whole process and it is mostly an ad hoc approach.

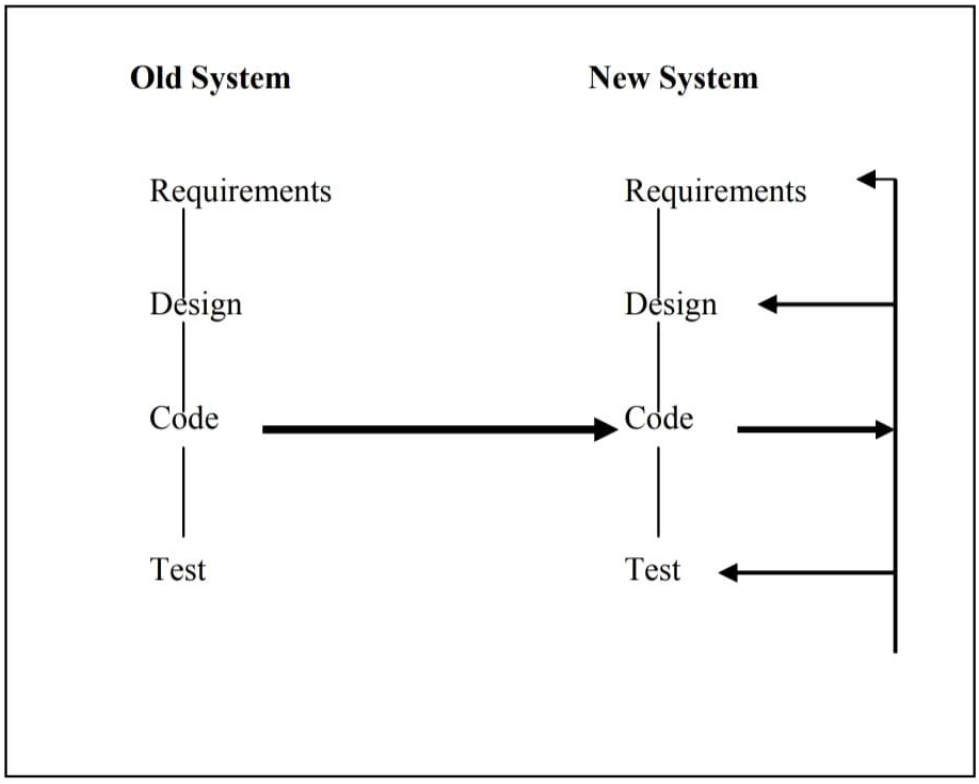
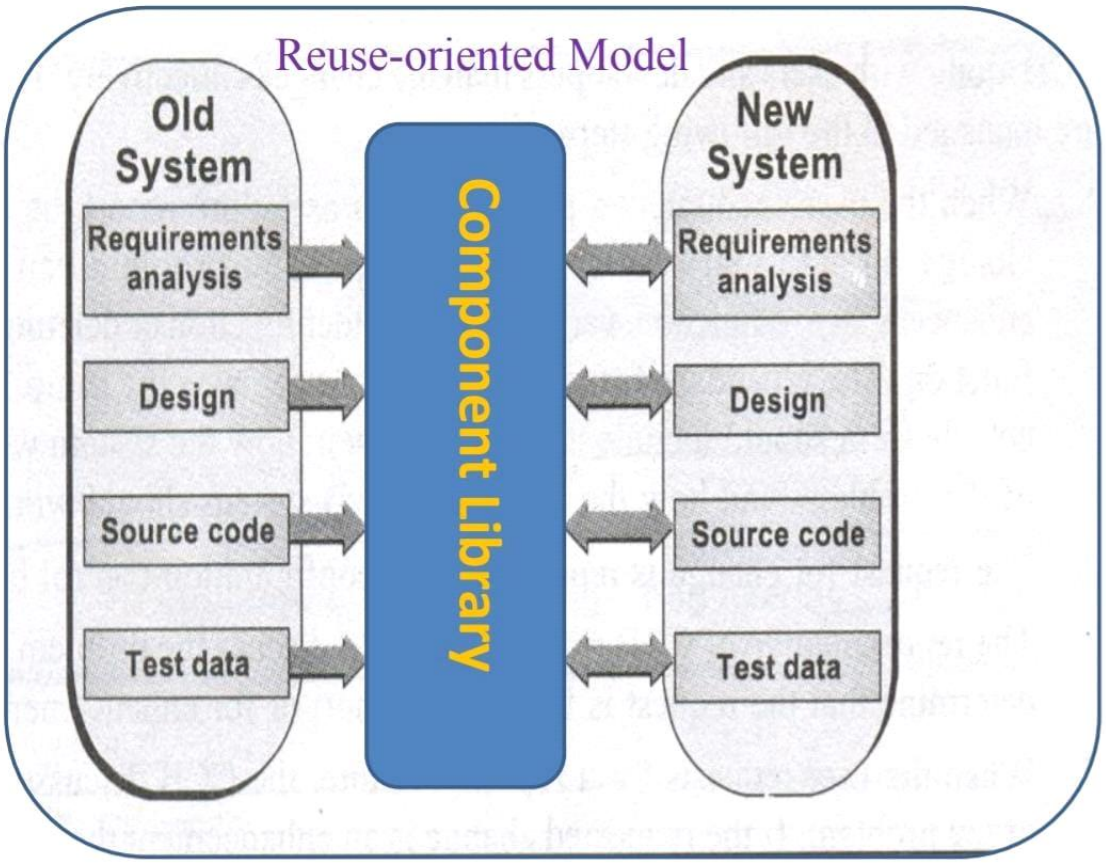


Figure 1: Quick-fix process model

Reuse Oriented model- This type of process is adopted for component-based software products. For fixing any defects, existing components are analyzed and then the appropriate changes are made.



Iterative Enhancement Model-This model is based on the similar concept of iterative software development. All software defects and change requests are logged and then a small set from this list is taken for making fixes. This set is prepared based on the priority of changes required. High priority fixes are done before low priority fixes.

Iterative Enhancement Model

This model requires complete documentation as starting point of each iteration

Analyze Existing System

It is similar to evolutionary development paradigm

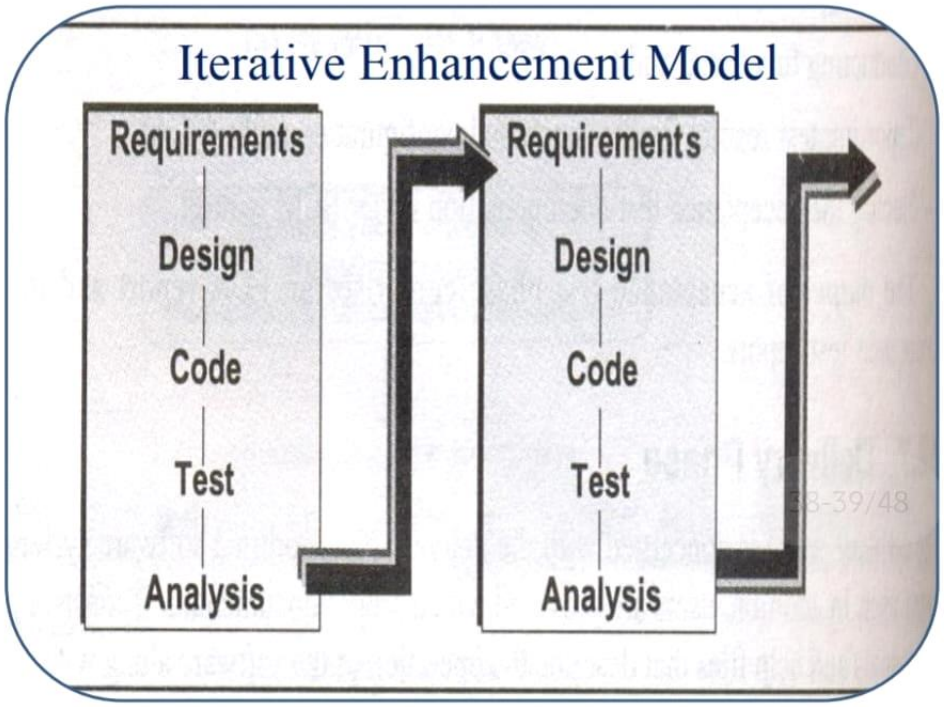
It is a three stage cycle

Redesign Current Version & Implement

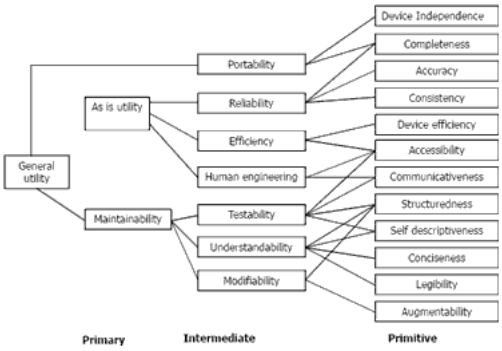
Characterize Proposed Modifications

Existing documentation for each stage is:

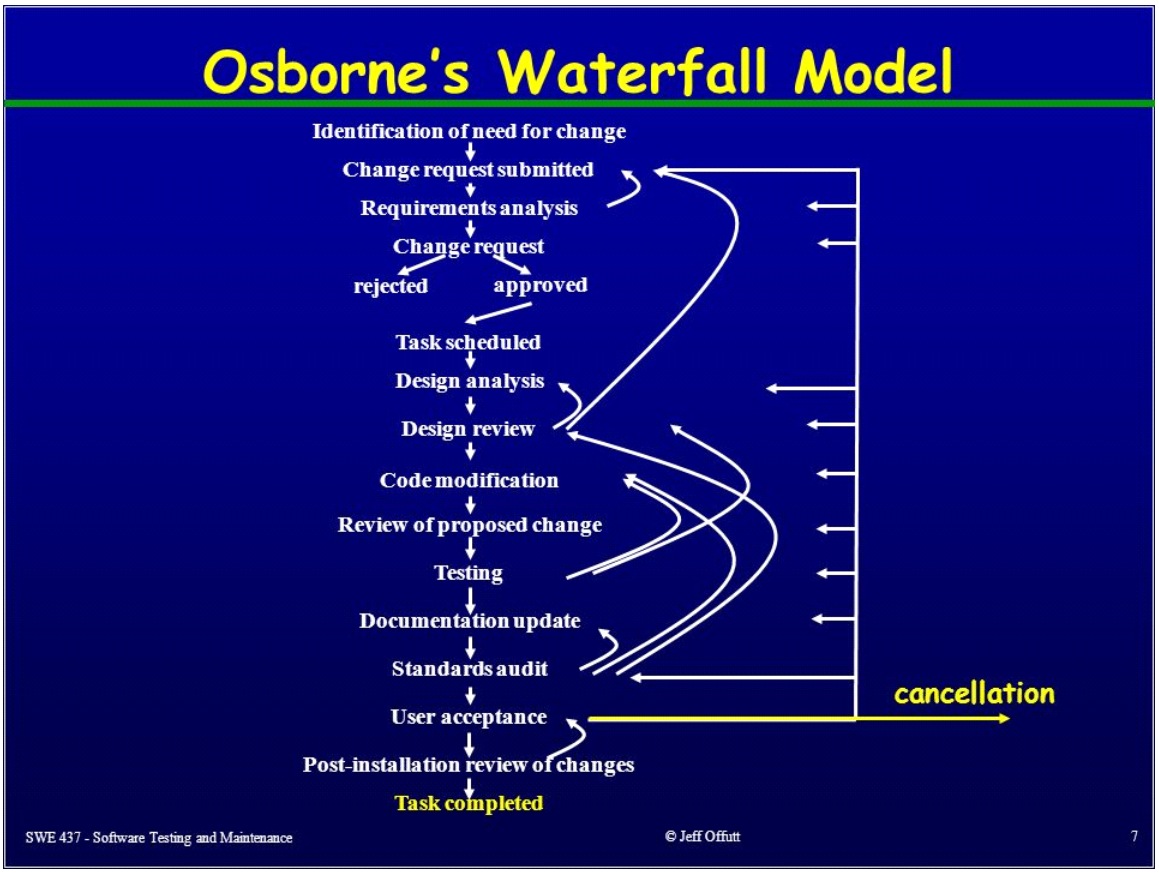
- Requirement documentation, Design Documentation, Source code documentation, & Test documentation



Boehm's model- Boehm's model is based on economic models and often involves calculating ROI, for any planned maintenance. If ROI turns out to be good, then it is carried out or else it is dropped.



Osborne's model- Osborne realized that difficulties in carrying out maintenance work are due to gaps in communication. He proposed four steps to prevent this situation. He stated that a maintenance plan should include all change requests in the form of maintenance requirements. A quality assurance plan should accompany the maintenance plan. Matrix should be developed to measure and assess quality of work carried out during maintenance.



2. Software Maintenance lifecycle and its functions.

ANS: Software maintenance is a part of Software Development Life Cycle. Its main purpose is to modify and update software application after delivery to correct faults and to improve performance. Software Maintenance is the process of modifying a software product after it has been delivered to the customer. The main purpose of software maintenance is to modify and update software applications after delivery to correct faults and to improve performance.

Need for Maintenance –

- Software Maintenance must be performed in order to:
- Correct faults.
- Improve the design.
- Implement enhancements.
- Interface with other systems.

- Migrate legacy software.
- Retire software.

This software maintenance life cycle has functions:

- Problem identification
- Analysis
- Design implementation
- System testing
- Acceptance testing
- Delivery phase.

Explain each.

OR

Product release management with suitable example.

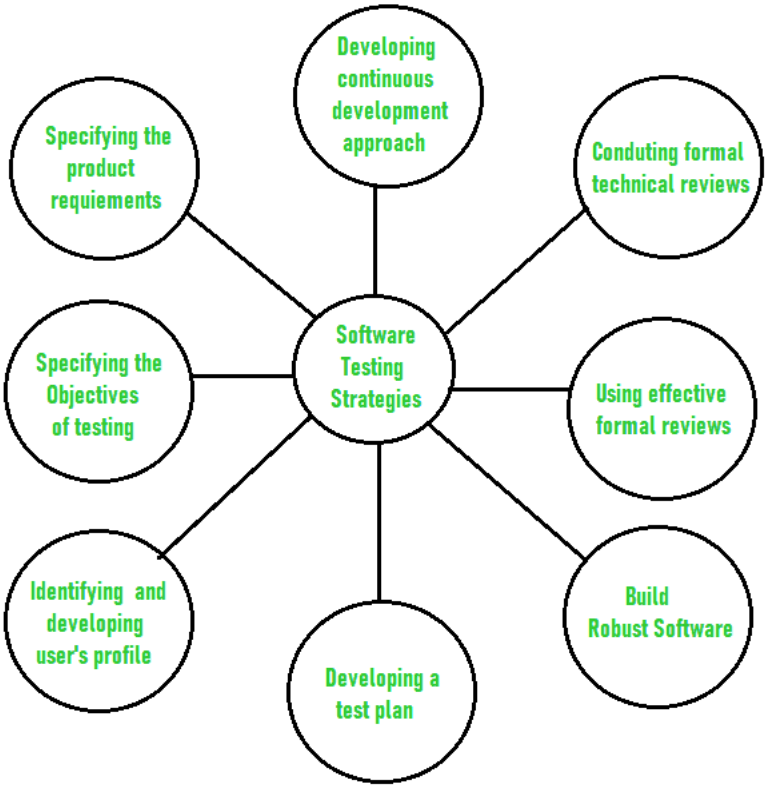
ANS: Same as set 1 qn 3

1. Test strategy and planning with neat diagram.

ANS:

Test Plan	Test Strategy
• A test plan for software project can be defined as a document that defines the scope, objective, approach and emphasis on a software testing effort	• Test strategy is a set of guidelines that explains test design and determines how testing needs to be done
• Components of Test plan include- Test plan id, features to be tested, test techniques, testing tasks, features pass or fail criteria, test deliverables, responsibilities, and schedule, etc.	• Components of Test strategy includes- objectives and scope, documentation formats, test processes, team reporting structure, client communication strategy, etc.
• Test plan is carried out by a testing manager or lead that describes how to test, when to test, who will test and what to test	• A test strategy is carried out by the project manager. It says what type of technique to follow and which module to test
• Test plan narrates about the specification	• Test strategy narrates about the general approaches
• Test plan can change	• Test strategy cannot be changed
• Test planning is done to determine possible issues and dependencies in order to identify the risks.	• It is a long-term plan of action.You can abstract information that is not project specific and put it into test approach
• A test plan exists individually	• In smaller project, test strategy is often found as a section of a test plan
• It is defined at project level	• It is set at organization level and can be used by multiple projects

TEST STRATEGY DIAGRAM:



TEST PLAN DIAGRAM:

ADDITIONALLY:

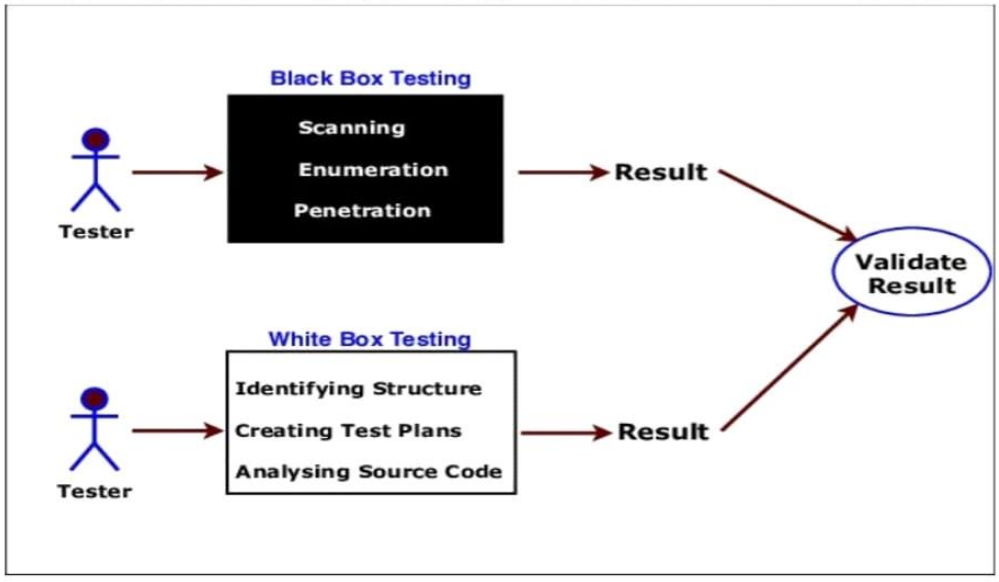
Black-box testing

- Knowing the specified function that a product has been designed to perform, test to see if that function is fully operational and error free
- Includes tests that are conducted at the software interface
- Not concerned with internal logical structure of the software

White-box testing

- Knowing the internal workings of a product, test that all internal operations are performed according to specifications and all internal components have been exercised
- Involves tests that concentrate on close examination of procedural detail
- Logical paths through the software are tested
- Test cases exercise specific sets of conditions and loops

BLACK BOX VS WHITE BOX



OR

(i) Risk management

(ii) Test point analysis

ANS: (i) The test manager should also do plan for all known risks that could impact the test project. If proper risk mitigation planning is not done, and a mishap occurs, then the test project schedule could be jeopardized, costs could escalate, and/or quality could go down.

Some of the risks that can have severe, adverse impact on a test project include an unrealistic schedule, resource unavailability, skill unavailability, frequent requirement changes, etc. The test team has to revise its schedule for additional work as well as to assess impact of the change on the test cases they have to recreate.

For test professional resources, a good alternative resource planning is required.

For scheduling problems, the test manager has to ensure in advance that schedules do not get affected. the test manager has to ensure that the schedule is not unrealistic and also has to load his test engineers appropriately. If some test engineers are not loaded adequately, then project costs may go higher. For this reason, if any test professionals do not have enough assignments on one project, they should be assigned work from other projects.

(ii) Test point analysis (TPA) is a technique to measure the black box test effort estimation. The estimation is for system and acceptance testing. The technique uses-function point analysis in estimation.

TAP calculates the test efforts estimation in test points for highly important functions, according to user and for whole system. TTP is used for calculation of primary test hours (PTH) PTH is effort estimation for primary test hour's activity such as preparation for primary test hour's activity such as preparation, specification and execution.

TTH is calculated by adding some allowance to secondary activities and PTH.

Thus TTH is final effort estimation for testing activities