

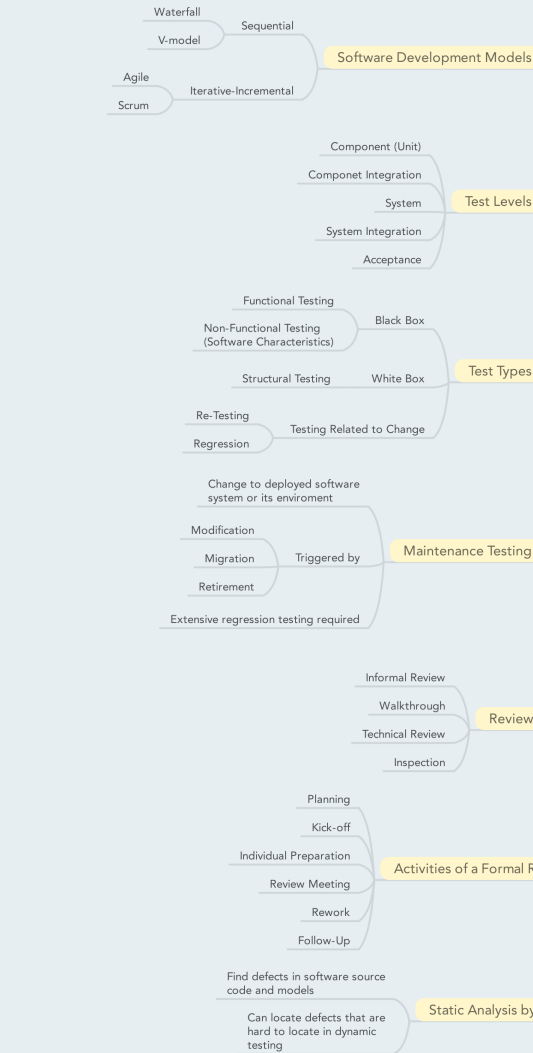
Software Testing

Free Quiz

1 Fundamentals of Testing



2 Testing Throughout the Software Life Cycle

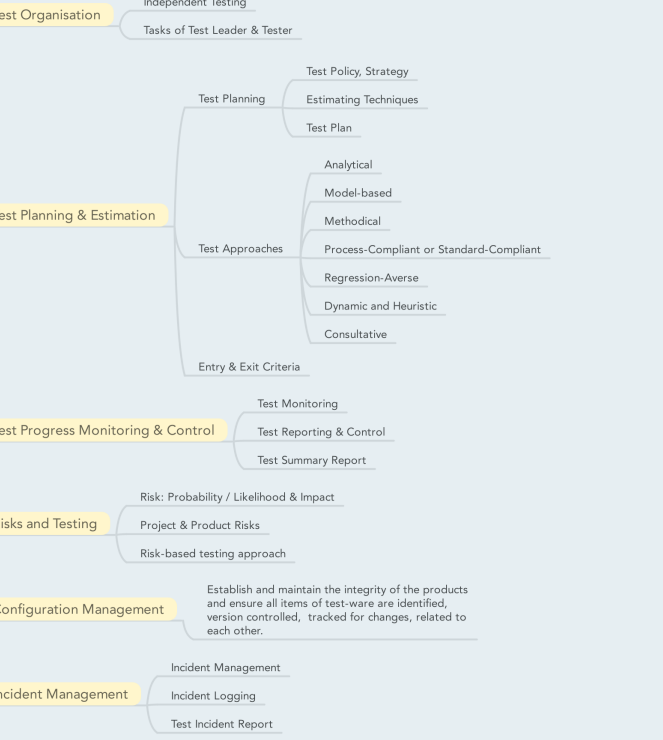


3 Static Techniques

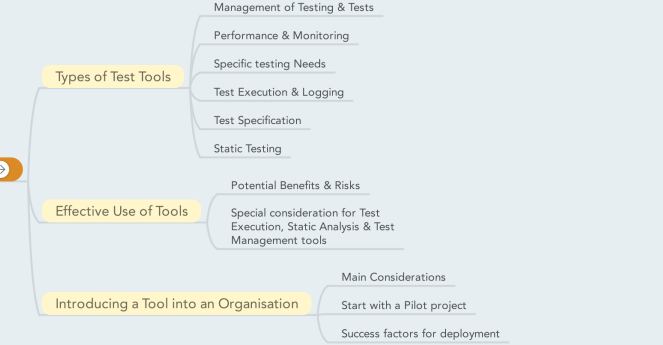
4 Test Design Techniques



5 Test Management



6 Tool Support for Testing



Software Testing

<http://www.rogeriodasilva.com/>

Software testing is an investigation conducted to provide stakeholders with information about the quality of the product or service under test. Software testing can also provide an objective, independent view of the software to allow the business to appreciate and understand the risks of software implementation. Test techniques include, but are not limited to, the process of executing a program or application with the intent of finding software bugs.

It involves the execution of a software component or system component to evaluate one or more properties of interest. In general, these properties indicate the extent to which the component or system under test:

meets the requirements that guided its design and development,

responds correctly to all kinds of inputs,

performs its functions within an acceptable time,

is sufficiently usable,

can be installed and run in its intended environments, and

achieves the general result its stakeholders desire.

As the number of possible tests for even simple software components is practically infinite, all software testing uses some strategy to select tests that are feasible for the available time and resources.

Visit: <http://www.rogeriodasilva.com/>

Fundamentals of Testing

Testing is a process rather than a single activity. This process starts from test planning then designing test cases, preparing for execution and evaluating status till the test closure. So, we can divide the activities within the fundamental test process into the following basic steps: 1) Planning and Control 2) Analysis and Design 3) Implementation and Execution 4) Evaluating exit criteria and Reporting 5) Test Closure activities

Test Design Techniques

Following are the typical design techniques in software engineering: 1. Deriving test cases directly from a requirement specification or black box test design technique. The Techniques include: Boundary Value Analysis (BVA) Equivalence Partitioning (EP) Decision Table Testing State Transition Diagrams Use Case Testing 2. Deriving test cases directly from the structure of a component or system: Statement Coverage Branch Coverage Path Coverage LCSAJ Testing 3. Deriving test cases based on tester's experience on similar systems or testers intuition: Error Guessing Exploratory Testing

Tool Support for Testing

<http://www.rogeriodasilva.com/10-best-free-defect-tracking-tools/>

The Classification Tree Method is a method for test design, as it is used in different areas of software development. It has been developed by Grimm and Grochtmann in 1993. Classification Trees in terms of the Classification Tree Method must not be confused with decision trees.

The classification tree method consists of two major steps:

Identification of test relevant aspects and their corresponding values as well as

Combination of different classes from all classifications into test cases.

The identification of test relevant aspects usually follows the specification of the system under test. These aspects form the input and output data space of the test object.

The second step of test design then follows the principles of combinatorial test design.

While the method can be applied using a pen and a paper, the usual way involves the usage of the Classification Tree Editor, a software tool implementing the classification tree method.

See: 10 Best Free Defect Tracking Tool

<http://www.rogeriodasilva.com/10-best-free-defect-tracking-tools/>

Test Management

- Test Procedure - Manual
- Test Script - Automated

Test management most commonly refers to the activity of managing the computer software testing process. A test management tool is software used to manage tests (automated or manual) that have been previously specified by a test procedure. It is often associated with automation software.

Testing Throughout the Software Life Cycle

There are various software development approaches defined and designed which are used/employed during development process of software, these approaches are also referred as "Software Development Process Models" (e.g. Waterfall model, incremental model, V-model, iterative model, etc.). Each process model follows a particular life cycle in order to ensure success in process of software development. Software life cycle models describe phases of the software cycle and the order in which those phases are executed. Each phase produces deliverables required by the next phase in the life cycle. Requirements are translated into design. Code is produced according to the design which is called development phase. After coding and development the testing verifies the deliverable of the implementation phase against requirements. There are following six phases in every Software development life cycle model: Requirement gathering and analysis Design Implementation or coding Testing Deployment Maintenance

Static Techniques

In integrated circuit design, dynamic logic is a design methodology in combinatory logic circuits, particularly those implemented in MOS technology. It is distinguished from the so-called static logic by exploiting temporary storage of information in stray and gate capacitances. It was popular in the 1970s and has seen a recent resurgence in the design of high speed digital electronics, particularly computer CPUs. Dynamic logic circuits are usually faster than static counterparts, and require less surface area, but are more difficult to design. Dynamic logic has a higher toggle rate than static logic but the capacitive loads being toggled are smaller so the overall power consumption of dynamic logic may be higher or lower depending on various tradeoffs. When referring to a particular logic family, the dynamic adjective usually suffices to distinguish the design methodology, e.g. dynamic CMOS or dynamic SOI design.

Dynamic logic is distinguished from so-called static logic in that dynamic logic uses a clock signal in its implementation of combinational logic circuits. The usual use of a clock signal is to synchronize transitions in sequential logic circuits.

Free Quiz

<http://www.rogeriodasilva.com/istqb-iseb-certified-tester-foundation-level/>

Take the free Quiz. <http://www.rogeriodasilva.com/istqb-iseb-certified-tester-foundation-level/>