Chapter 1
Introduction to Effective Software Testing

OXFORD
HIGHER EDUCATION

Objectives

- How the software testing has been evolved?
- What are software testing myths and what is the corresponding truth?
- Software testing is a separate discipline.
- Testing is a complete process.
- What are the goals of Software testing?
- Testing is based on a negative/destructive view rather than optimistic.
- Model for testing process.
- Complete testing is not possible.
- Various Schools of Software testing.

Evolution of Software Testing

- In the early days of software development, Software Testing was considered only as a debugging process for removing the errors after the development of software.
- By 1970, software engineering term was in common use. But software testing was just a beginning at that time.
- In 1978, G.J. Myers realized the need to discuss the techniques of software testing in a separate subject. He wrote the book “The Art of Software Testing” [2] which is a classic work on software testing.
- Myers discussed the psychology of testing and emphasized that testing should be done with the mind-set of finding the errors not to demonstrate that errors are not present.
- By 1980, software professionals and organizations started talking about the quality in software. Organizations started Quality assurance teams for the project, which take care of all the testing activities for the project right from the beginning.

Evolution of Software Testing

- In the 1990s testing tools finally came into their own. There was flood of various tools, which are absolutely vital to adequate testing of the software systems. However, they do not solve all the problems and cannot replace a testing process.
- Gelperin and Hetzel [79] have characterized the growth of software testing with time. Based on this, we can divide the evolution of software testing in following phases:

Evolution of Software Testing

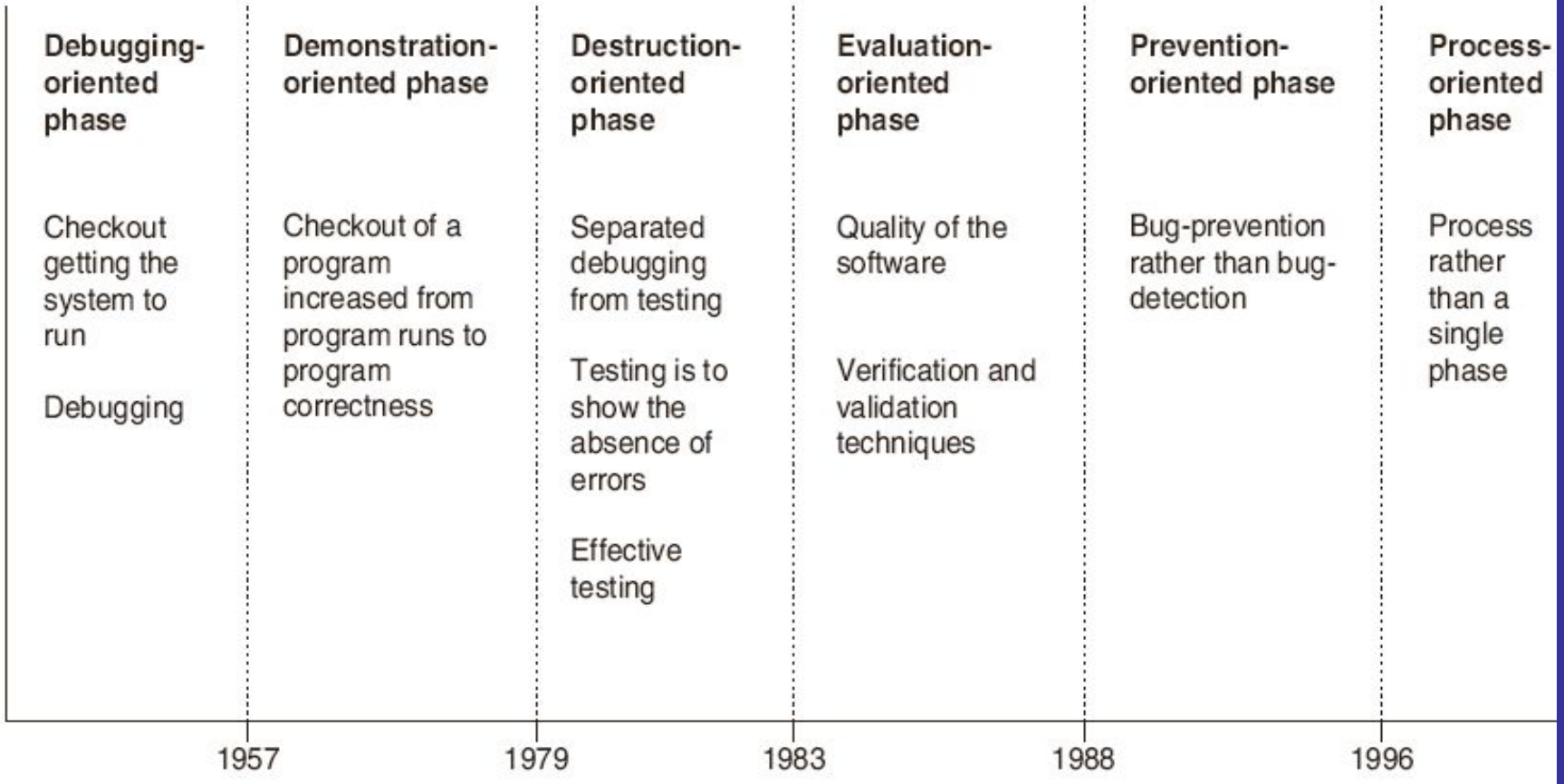


Figure 1.1 Evolution phases of software testing

Evolution of Software Testing

- The evolution of software testing was also discussed by Hung Q. Nguyen and Rob Pirozzi in a white paper [81], in three phases namely Software Testing 1.0, Software Testing 2.0 and Software Testing 3.0. These three phases discuss the evolution same as in the earlier phases we described. According to this classification, the current state-of-practice is Software Testing 3.0. These phases are :

Evolution of Software Testing

Software Testing 1.0

- In this phase, software testing was considered to be just a single phase to be performed after coding of the software in SDLC. No test organization was there. A few testing tools were present but their use was limited due to the high cost. Management was not concerned with the testing as there was no quality goal.

Software Testing 2.0

- In this phase, software testing gained the importance in SDLC and the concept of early testing was also started. Testing was evolving into the direction of planning the test resources. Many testing tools were also available in this phase.

Evolution of Software Testing

OXFORD
HIGHER EDUCATION

Software Testing 3.0

- In this phase, software testing is being evolved in the form of a process which is based on the strategic effort. It means that there should be a process which gives us a road map of the overall testing process. Moreover, it should be driven by the quality goals in mind so that all controlling and monitoring activities can be performed by the managers. Thus, management is actively involved in this phase.

Software Testing Myths

- *Testing is a single phase in SDLC performed after coding*.
- *Testing is easy*.
- *Software development is of more worth as compared to testing*.
- *Complete testing is possible*.
- *The testing starts after the program development*.
- *The purpose of testing is to check the functionality of the software*.
- *Anyone can be a tester*.

Software Testing Goals

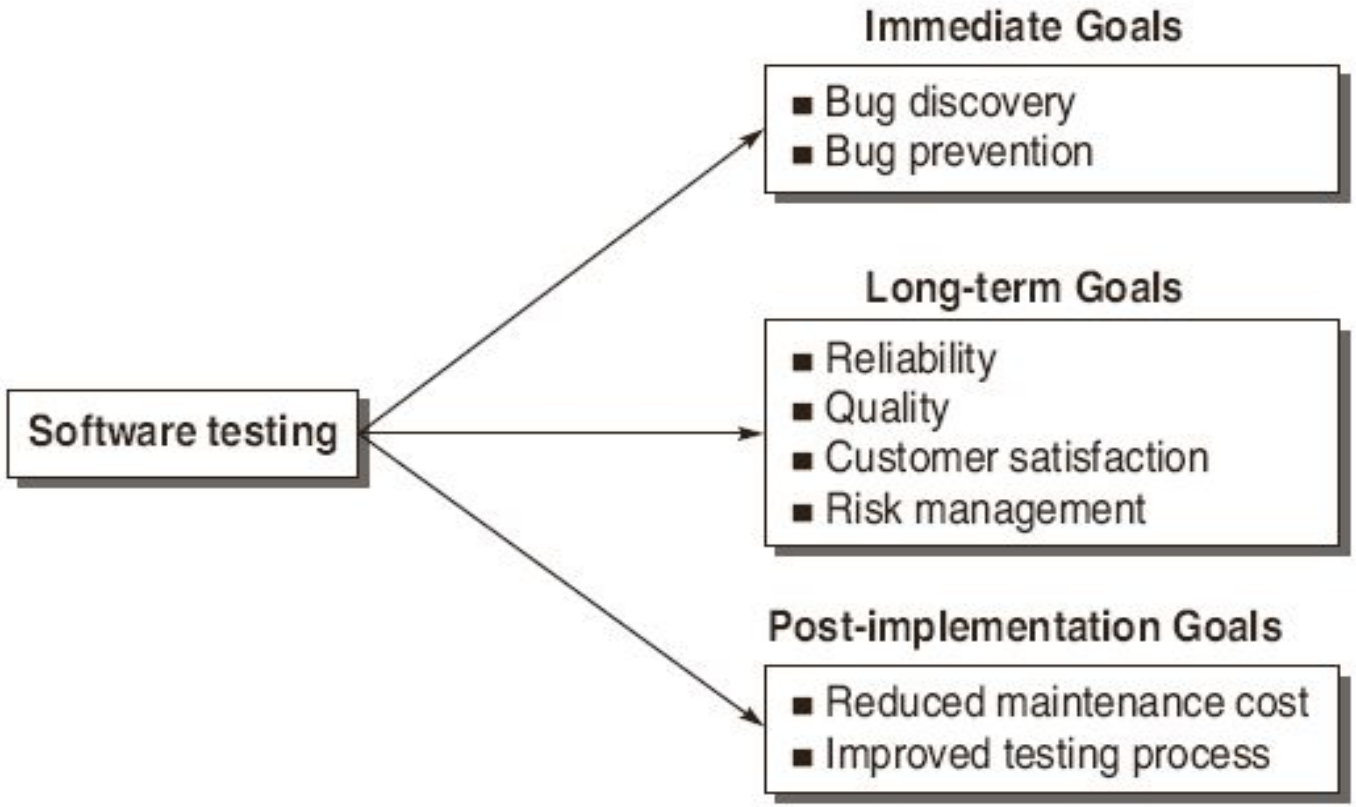
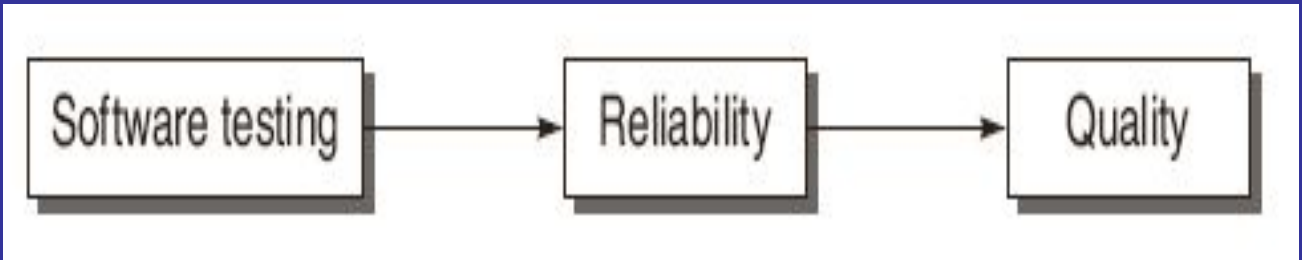
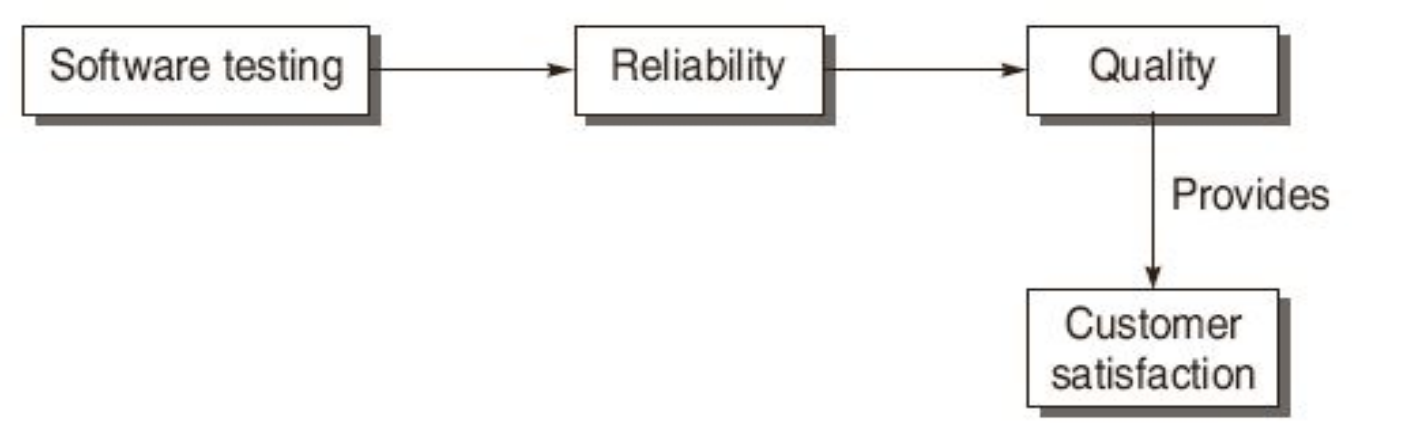


Figure 1.2 Software testing goals

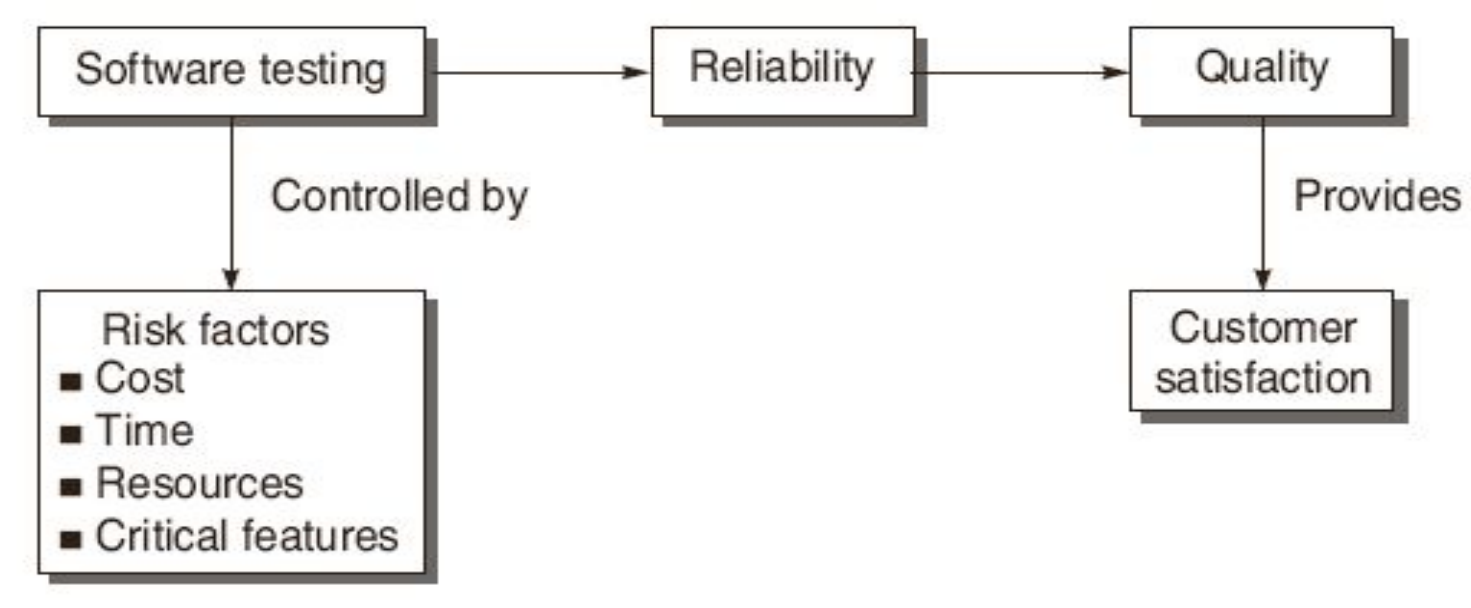
Testing produces Reliability and Quality



Quality leads to customer satisfaction



Testing controlled by Risk factors



Psychology for Software Testing

- Software testing is directly related to human psychology. Though software testing has not been defined till now, but most frequently it is defined as –

  " Testing is the process of demonstrating that errors are not present."
- So the purpose of testing is to show that software performs its intended functions correctly. This definition is correct, but partially.
- If testing is performed keeping this goal in mind, then we cannot achieve the desired goals (described above in the previous section) as we will not be able to test the software as a whole.
- It may hide some bugs.
- If our goal is to demonstrate that a program has errors; we will design the test cases which have higher probability to uncover the bugs instead of showing that software works.

  "Testing is the process of executing a program with the intent of finding errors."

Psychology for Software Testing

OXFORD
HIGHER EDUCATION

- We should not have a guilty feeling for our errors and for human beings.
- This psychology factor brings the concept that we should concentrate on discovering and preventing the errors and not to feel guilt about them.
- Therefore, testing cannot be a joyous event unless you cast out your guilt.

Software Testing Definitions

- *"Testing is the process of executing a program with the intent of finding errors."*

  - Myers [2]

- *"A successful test is one that uncovers an as-yet-undiscovered error."*

  - Myers [2]

- *"Testing can show the presence of bugs but never their absence."*

  - W. Dijkstra [125].

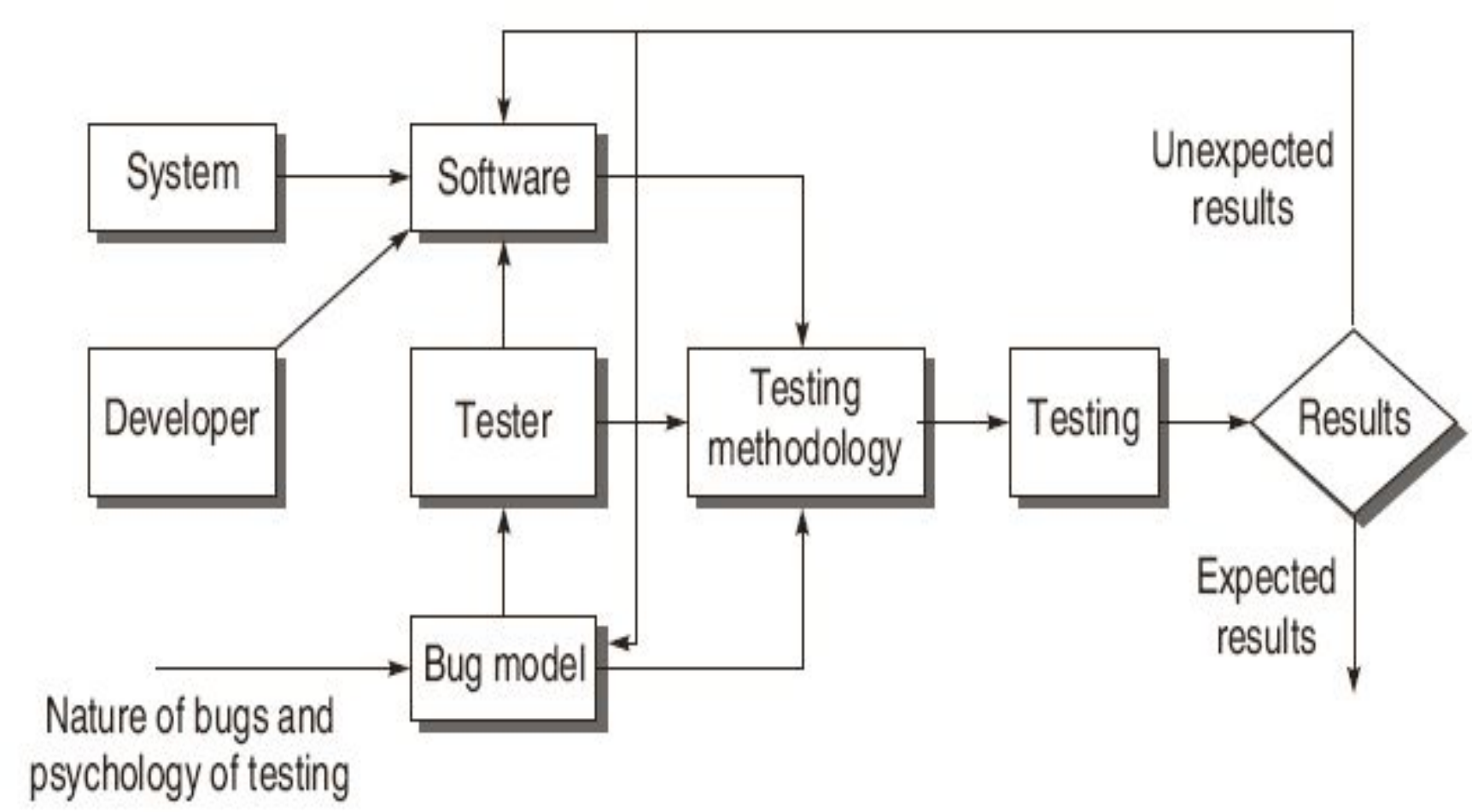
Software Testing Definitions

- "*Testing is a concurrent lifecycle process of engineering, using and maintaining testware (i.e. testing artifacts) in order to measure and improve the quality of the software being Tested.*"

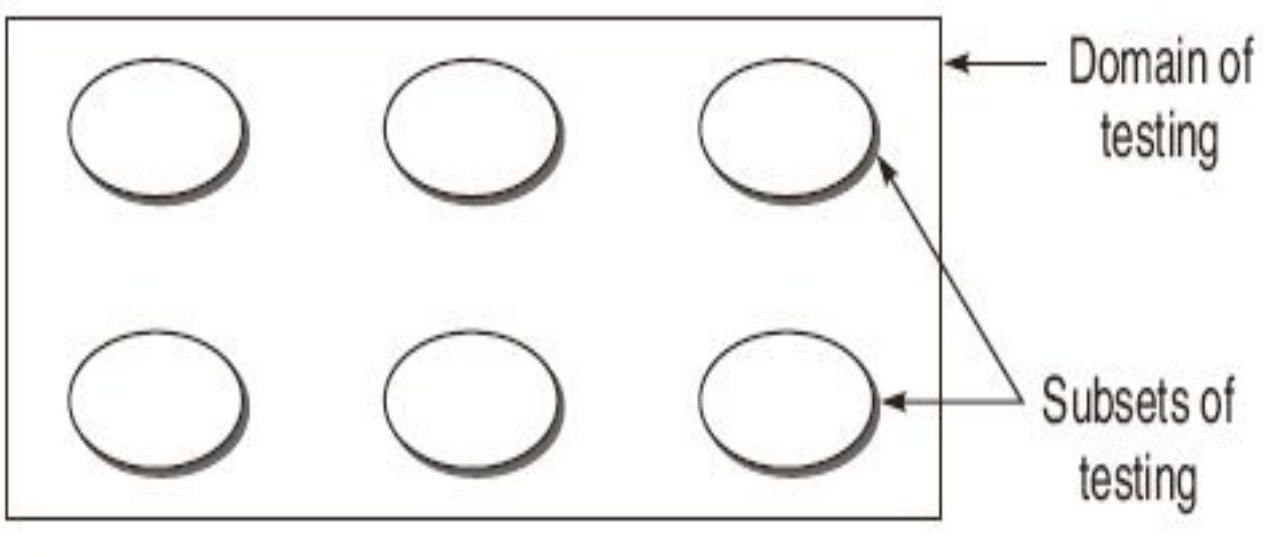
  - *Craig* [117]

- *"Software testing is a process that detects important bugs with the objective of having better quality software."*

Model for Software Testing



Effective Software Testing vs Exhaustive Software Testing

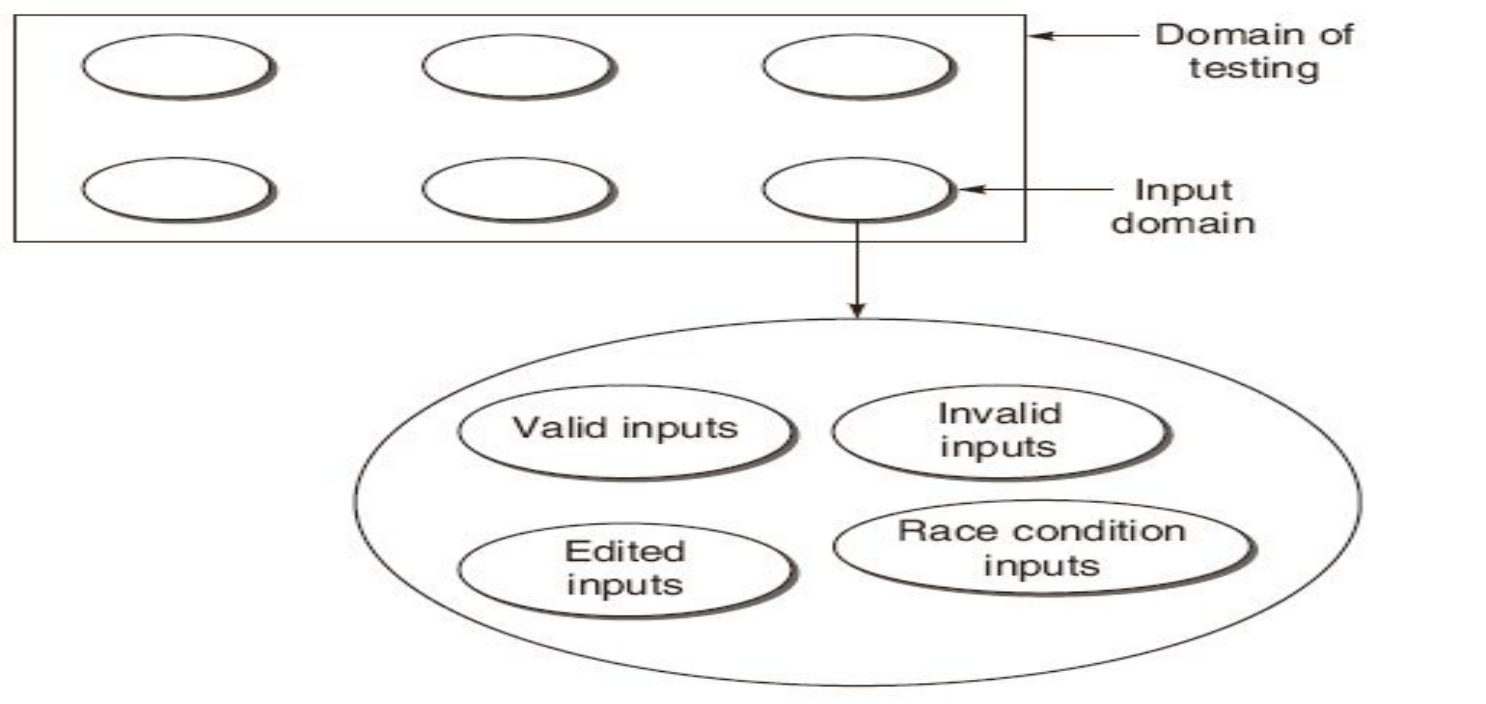


Effective Software Testing vs Exhaustive Software Testing

- The domain of possible inputs to the software is too large to test.

- Valid Inputs
- Invalid Inputs
- Edited Inputs
- Race Conditions

Effective Software Testing vs Exhaustive Software Testing



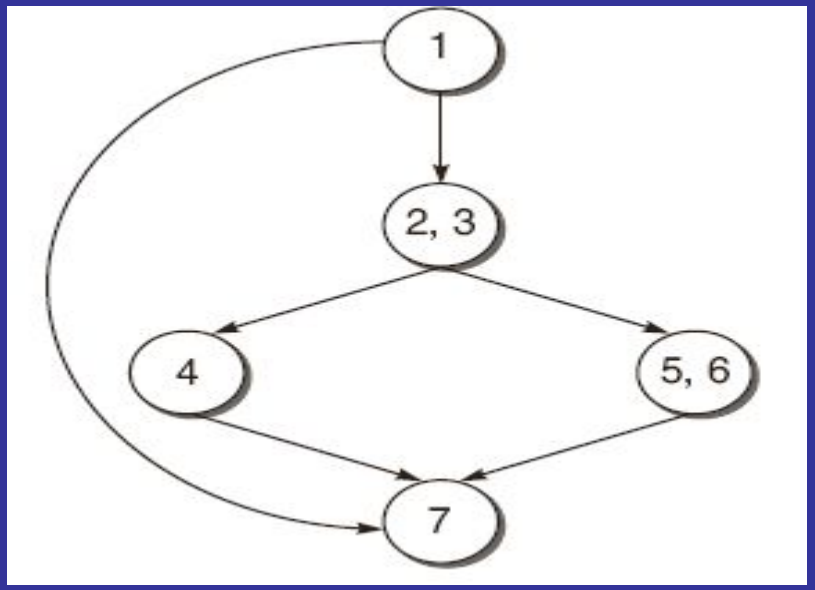
Effective Software Testing vs Exhaustive Software Testing

OXFORD
HIGHER EDUCATION

There are too many possible paths through the program to test.

```
1for (int i = 0; i < n; ++i)
2{
3              if (m >=0)
4                 x[i] = x[i] + 10;
5              else
6                 x[i] = x[i] - 2;
7}...
```

Effective Software Testing vs Exhaustive Software Testing

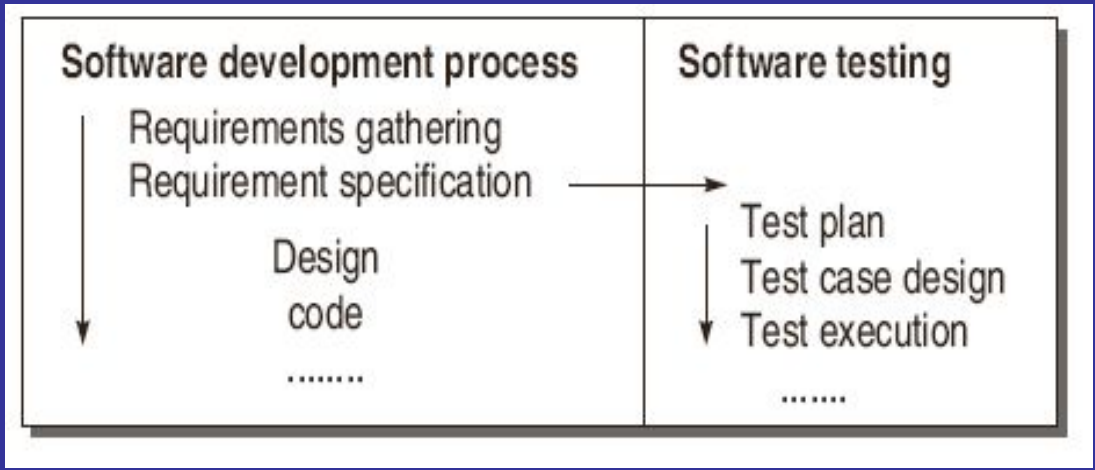


- if n is 20, then the number of paths will be $2^{20} + 1$, i.e. 1048577.

Effective Software Testing vs Exhaustive Software Testing

- Every Design error cannot be found.

Software Testing as a Process



Software Testing as a Process

OXFORD
HIGHER EDUCATION

- An organization for the better quality software must adopt a testing process and consider the following points:
- Testing process should be organized such that there is enough time for important and critical features of the software.
- Testing techniques should be adopted such that these techniques detect maximum bugs.
- Quality factors should be quantified so that there is clear understanding in running the testing process. In other words, process should be driven by the quantified quality goals. In this way, process can be monitored and measured.
- Testing procedures and steps must be defined and documented.
- There must be scope for continuous process improvement.

Schools of Software Testing

The idea of schools of testing was given by Bret Pettichord [82]. He has proposed the following schools:

- **Analytical School of Testing**

  This school defines software testing as a branch of computer science and mathematics.

Schools of Software Testing

OXFORD
HIGHER EDUCATION

- **Standard School of Testing**

The emphasis is on measurement of testing activities to track the development progress.

This school defines software testing as a managed process.

Schools of Software Testing

OXFORD
HIGHER EDUCATION

Quality School of Testing

This school defines software testing as a branch of software quality assurance.

The implications of this school are:

- It prefers “Quality Assurance” term over “Testing”
- Testing is a stepping stone to “process improvement”.

Schools of Software Testing

OXFORD
HIGHER EDUCATION

- **Context-driven School of Testing**

  This school is based on the concept that testing should be performed according to the context of the environment and project. The testing solutions cannot be same for every context.

- **Agile School of Testing**

  This type of school is based on the testing of software which is being developed iterative method of development and delivery. In this type of process model, software is delivered in a short span of time and based on the feedback more features and capabilities are added. The focus is on satisfying the customer by delivering working software quickly with minimum features and then improvising on it based on the feedback.