Course for Software Testing

BS (Software Engineering)
Spring Semester
February 2014
Foundation University Rawalpindi Campus

Instructor: Sohaib Altaf
sohaib@hybriditservices.com
http://www.hybriditservices.com/course/FU-BSSE8-ST

Lecture 6

Debugging

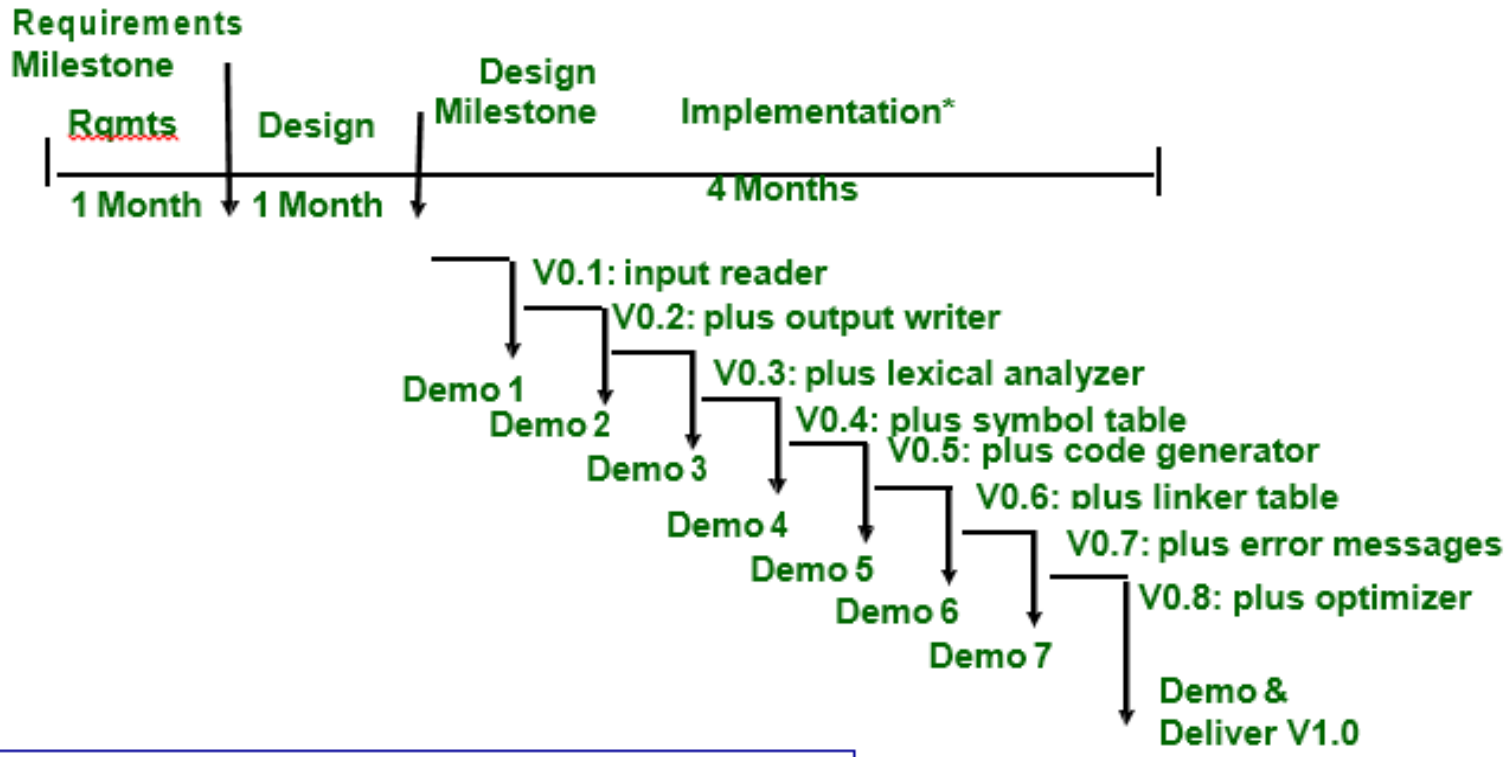
Testing is the process of….
- determining if a program has any errors, further inform it to concerns
- when testing determine an error, the process used to determine the cause of this error and to remove it, is known as debugging.

In Debugging, programmer examines the code to determine the cause…
- to fix and check / test the fix, confirm the fix

Further in Testing process
- tester independently re-test the changes and confirm it to concerns
  - if to close the defect or re-open
- [If defects reduces during incremental releases by testing & debugging?] RA
- but if doesn't guarantee 0 defects, it is not proof of correctness

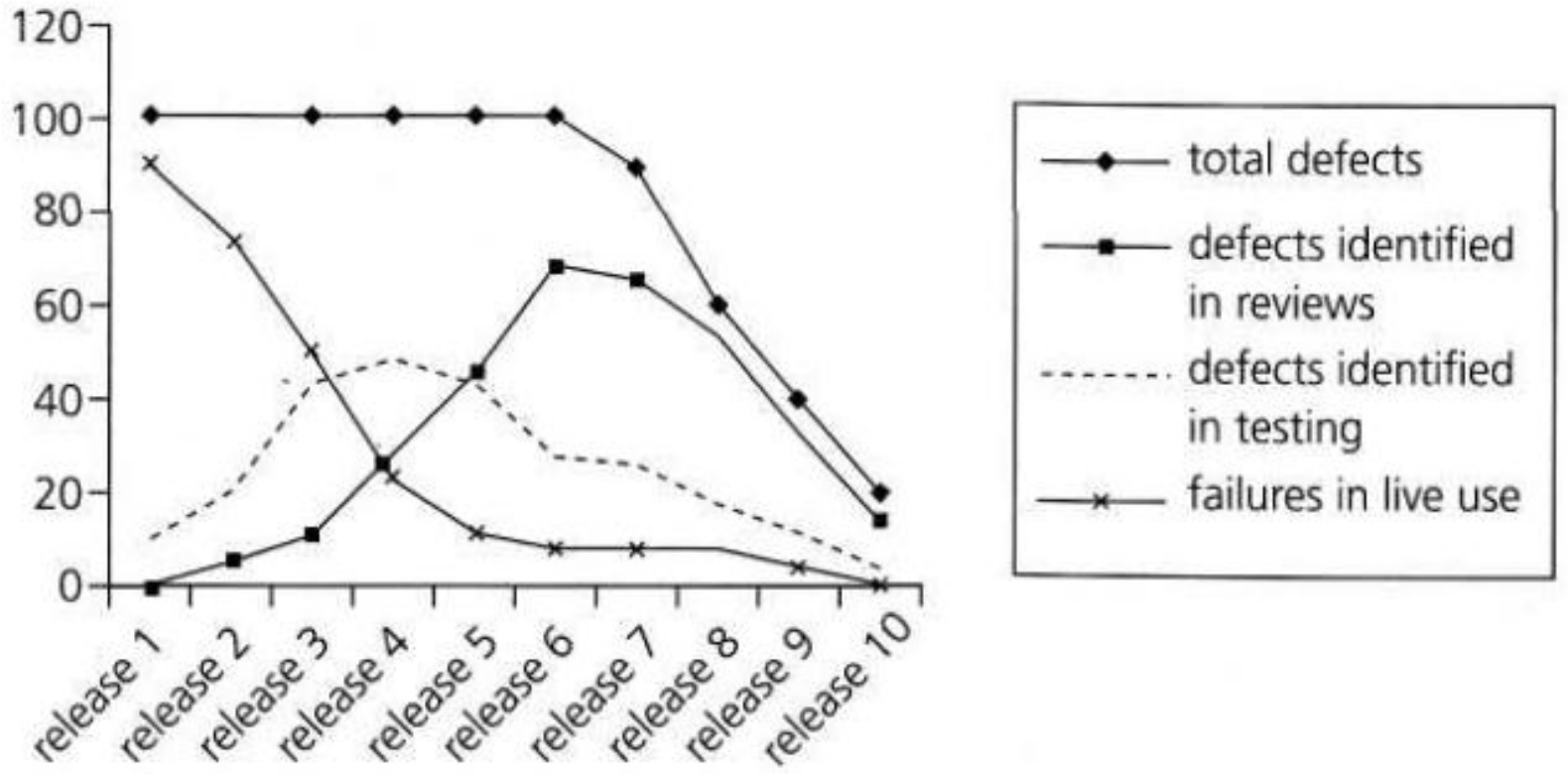
SDLC (Incremental)

An Incremental-build Example
Implementing a Compiler



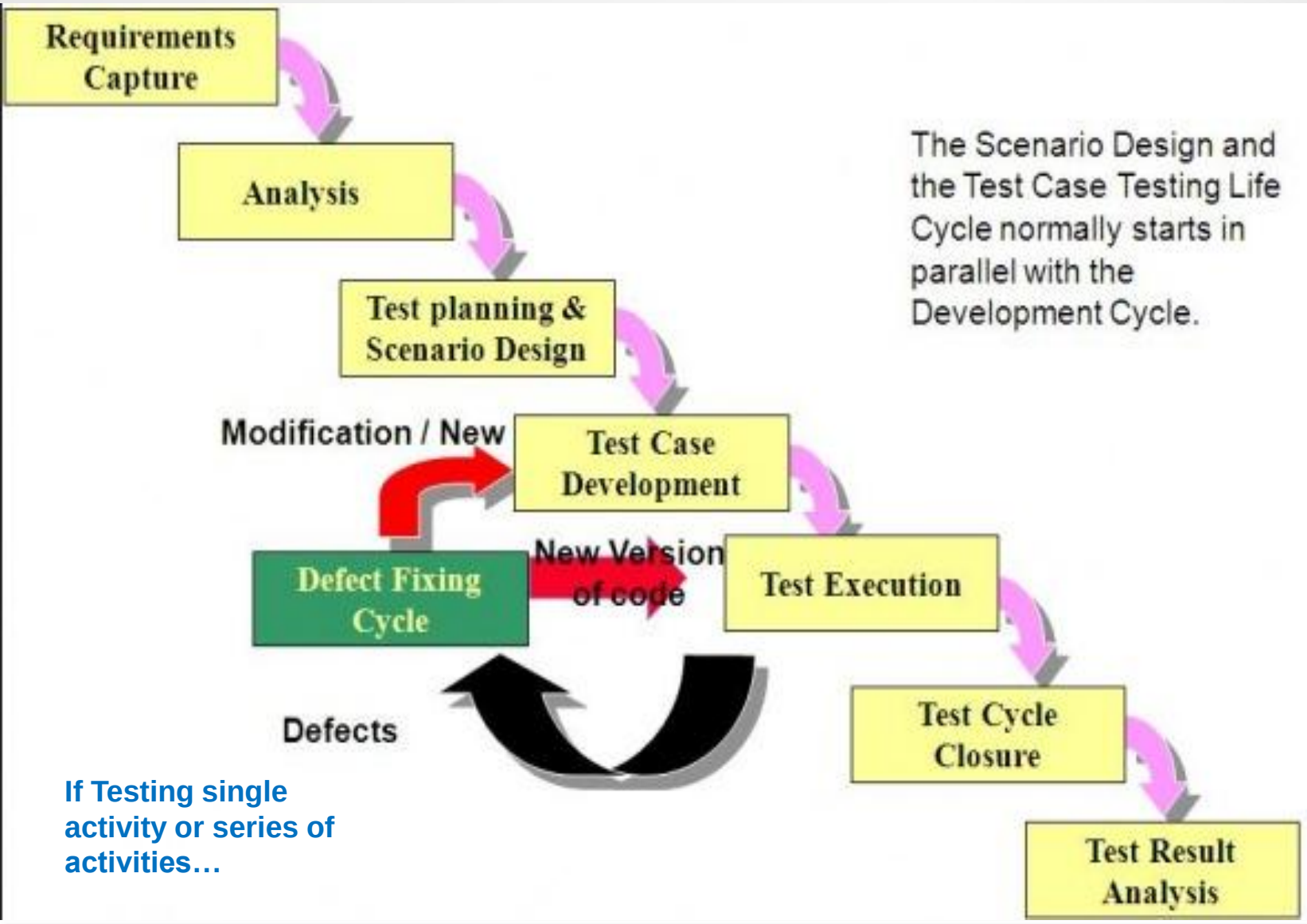
*implementation of each increment includes detailed design, coding, review, integration, testing, and demonstration

No. of Defect over the time



Changes in defect numbers during process improvement

SDLC (Waterfall) & STLC



The Scenario Design and the Test Case Testing Life Cycle normally starts in parallel with the Development Cycle.

If Testing single activity or series of activities…

Fundamental Software Testing Process?

Software Testing Life Cycle (STLC)

- Testing is process rather than single activity. Series of Testing Activities
- Testing activities should start as early as possible in the SDLC
- These takes place throughout the SDLC
- STLC start with test planning and continuous till test closure
- *Planning is process that take place before and after test execution*
- An Analogy… Test process & Code Process:
  - **An Overview of STLC & SDLC…….**

Fundamental Software Testing Process?

Software Testing Life Cycle (STLC)

- Thought process of designing tests early in the life cycle can help to prevent defects from being introduced into **code.**
  - We'll build the right software, correctly and at a lower cost overall.
  - The later in the life cycle we find bugs, the more expensive they are to fix.

- Review Software products and related work products
  - Don't just test code.
    - Review (static test) the requirements, design specifications,
    - & related documents i.e. such as operation, user and training material.
    - Static and Dynamic testing, analogy with driving test, cold test, dry & actual run…

Sample Software Testing Life Cycle

- Test plan
- Test Preparation
- Test Evaluation (Implementation & Execution)
- Recording the Observation
- Towards Closing (Test Closure Activities)
- Preparing the Test Report
- *Test Control (Management Perspective)*

- These activities are logically sequential, but, in a particular project, may overlap, take place concurrently and even be repeated.

Sample Software Testing Life Cycle

- Test Plan
- Capture the activities take place before & after test execution.
- Manage Testing
  - Plan what we want to do
  - Control the test activities
  - Report on Testing Progress
  - Report Status of Software Under Test
  - Finalize Close Testing, Close Phase
- Test Plan is a document that defines the overall testing approach.

- Test Preparation
- Need to select what testing we will do
  - Selecting test conditions and **Designing test cases**

Sample Software Testing Life Cycle

- Test Preparation (Cont..)
- Document that defines what is selected to test, and describes the expected result is called Test Design.
- Set of procedures in order to find out the defects.

- Test Evaluation
- Executing the test cases.
  - We must check the results against what is required, if it satisfied
  - & evaluate the software under test and the completion criteria,
  - & decide whether we have finished testing and whether the software product has passed the tests.
  - Develop Confidence, If fit for purpose? Analogy of driving test
    - Both the examiner and the tester are not evaluating for perfection, but for meeting sufficient of the attributes required to pass the test.

Sample Software Testing Life Cycle

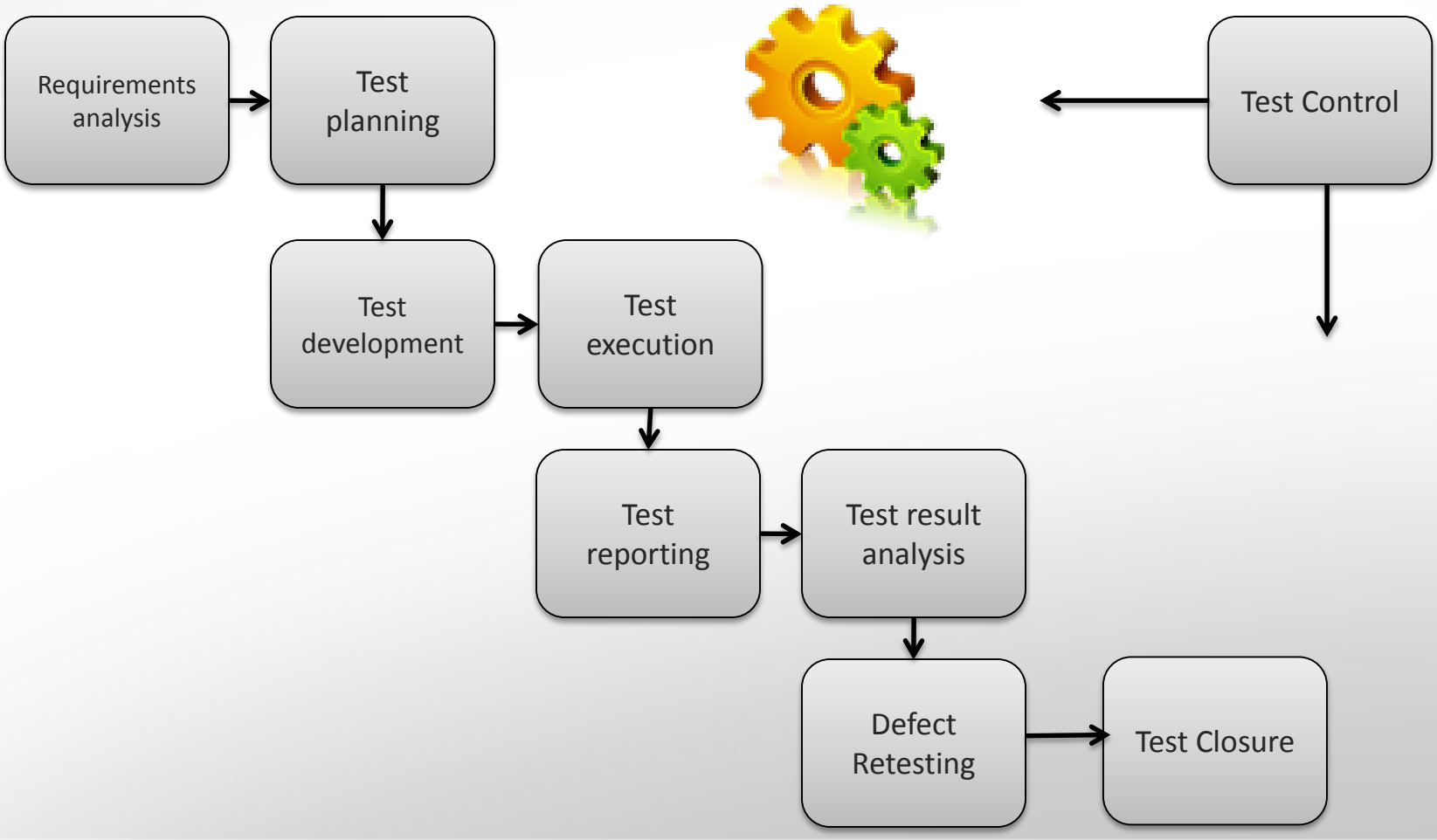
- Recording the Observation
- Record the test results
- Pass and fail information will be stored in this **Test Log**

- Towards Closing (Building Confidence)
- Before closing the phase / project, follow the status, which items are successful, *Tracking*
- All the failed items should be traced (**track:** *not passed, yet failed or* failure after pass, wrong report i.e. typing error or any unethical intention)

Sample Software Testing Life Cycle

- Preparing the Test results
- Reports are generated during the STLC. Final report for test summary. (generated at the end of testing for making decisions, measuring)

Sample Testing Cycle



Software Testing Life Cycle (STLC)

A **Test Cycle** is often guided by a test plan, and continue through to test closure.

The Test Plan

- Who (Team Foundation)
- What (Development Plan, Most Critical & Complex Module)
- When (Tentative Plan, Model: Waterfall or Incremental)
- Where (in-house or independent, or alpha, or beta, bugs propagated from past)
- How (From SRS Document, Manual, Tool-based, Both partial, Types)

Test Plan Considerations

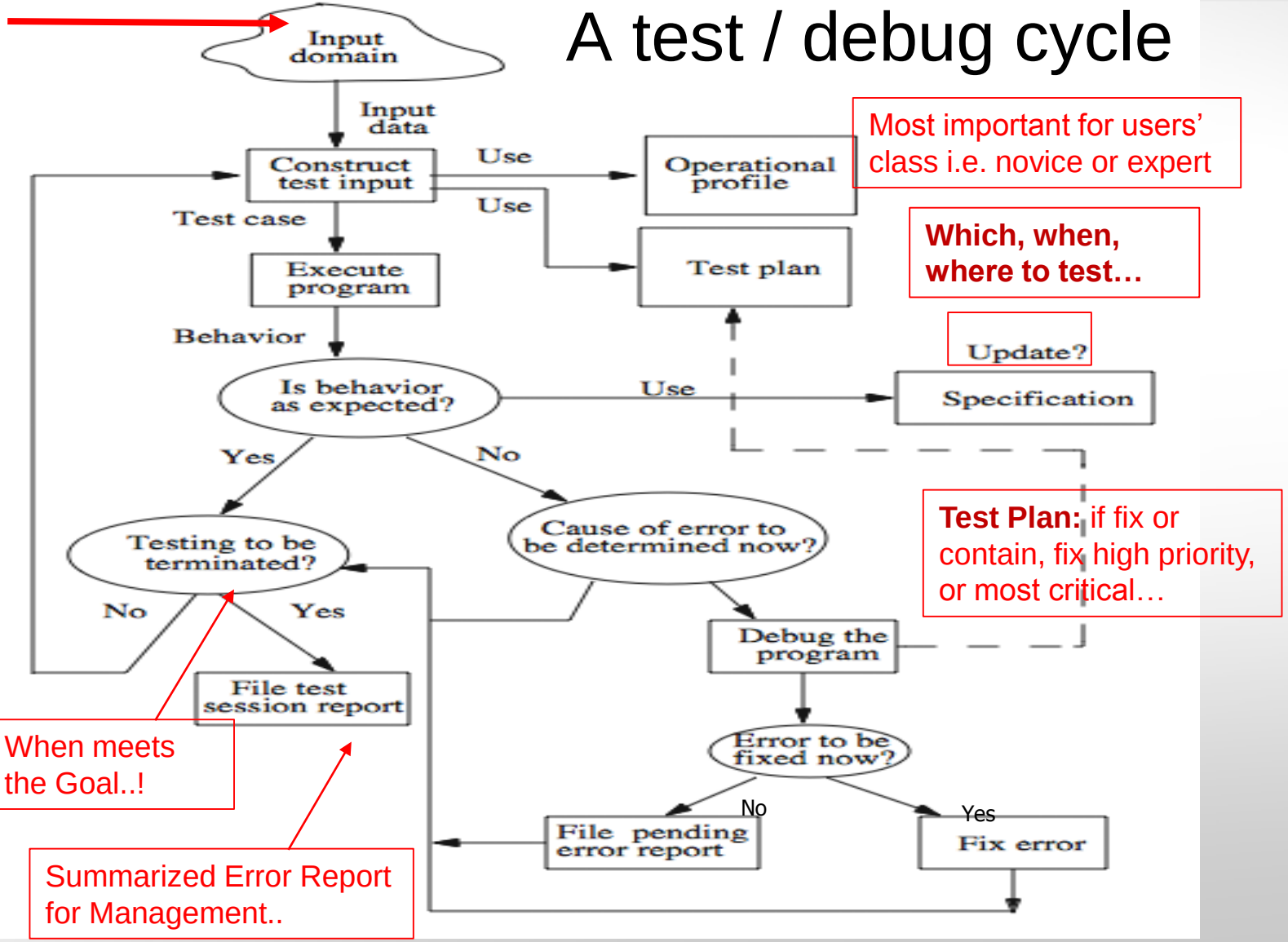
- What are the critical or most complex modules?
  - make sure they get integration tested first
  - probably deserve white-box attention
- Where have you had problems in the past?
- Third-Party delivered components?
- What training is required?
  - conducting formal reviews
  - use of testing tools
  - defect report logging

Test Plan Considerations

Goals vs. Objective

- Goals :
  - Define during planning phase
  - When met; point of point of termination from testing
  - Goals are an expected or desired outcome of a planning process. Goals are usually broad, general expressions of the guiding principles and aspirations of a community.
  - Objectives are precise targets that are necessary to achieve goals. Objectives are detailed statements of quantitatively or qualitatively measurable results the plan hopes to accomplish.

A test / debug cycle



Most important for users' class i.e. novice or expert

Which, when, where to test…

Test Plan: if fix or contain, fix high priority, or most critical…

When meets the Goal..!

Summarized Error Report for Management..

Test Plan

- Small Program --> Testing is informal, in testers' mind

- **Example: Test Plan for Sort:** The sort program is to be tested to meet the requirements given earlier. Specifically, the following needs to be done.

    1. Execute sort on at least two input sequences, one with ``A" and the other with ``D" as request characters.

Test plan (contd.)

2 Execute the program on an empty input sequence.

3 Test the program for robustness against erroneous inputs such as ``R'' typed in as the request character.

4 All failures of the test program should be recorded in a suitable file using the Company Failure Report Form.

Constructing Test Data / Case

A test case is a pair consisting of test data to be input to the program and the expected output. The test data is a set of values, one for each input variable.

A test set is a collection of zero or more test cases.

Program requirements & the test plan help in construction of test data

Execution of program on test data might begin after all or few test cases

To continue for more test case it depends on results

Constructing Test Data / Case

Sample test case for sort:

Test Case 1: Test data: <"A' 12 -29 32. >
Expected output: -29 12 32

Test Case 2: Test data: <'‘D' 12 -29 32. >
Expected output: 32 12 -29

TC 1 & 2 are derived in response to item no. 1 in test plan

Test Case 3: Test data: <‘‘A’ . >
No Input to be sorted in ascending order

Test Case 4: Test data: <‘‘D’ . >
No Input to be sorted in descending order

TC 3 & 4 are derived in response to item no. 2 in test plan

No expected output is defined in requirements in above case, so an arbitrary decision is taken

Constructing Test Data / Case

Sample test case for sort:

Test Case 5: Test data: <'“R'” 3 17. >
Expected output: Invalid request character
Valid characters: “A” & “D”

Test Case 6: Test data: <'“A'” C 17. >
Expected output: Invalid number
Valid Output: “all integers 1, 2, 3 ..33K”