Information System Design

IT60105

Lecture 24

Introduction to System Testing

Lecture #24

- Some recent software failures
- Software quality assurance (QA)
- Causes of software bugs
  - Error vs. failure
- Verification and validation
- System life cycle testing

Some Recent Software Failures

- August 2006
  - A majority of customers got their balances NIL in savings bank account at SBI Kaharagpur branch
- May 2005
  - A bug in *site management software* utilized by companies with a significant percentage of WWW traffic was reported in may 2004. The bug resulted in performance problems for many of sites simultaneously and required disabling of the software until the bug was fixed
- April 2004
  - A bug was determined to be a major contributor to the Northeast (USA) blackout, the worst power system failure in North American history. The failure involved loss of electrical power to 50 million customers for 37 days, forced shutdown of 100 power plants, and economic losses at $16 billion

A Big Challenge!

- QA – better take prevention well in advance
- QA – a big headache to system developers
  - Solving problem is a high-visibility process, preventing problem is a low-visibility process
  - An old Chinese parable:
    - In ancient China there was a family of healers. One of whom was known throughout the land and employed as a physician in King's palace. The physician was asked which of his family was the most skillful healer. He replied
      - I tend to the sick and dying with drastic and dramatic treatments and on occasion someone is cured and **my name gets out among the lords**
      - My elder brother cures sickness when it just begins to take root, and **his skills are known among the local peasants and neighbors**
      - My eldest brother is able to sense the spirit of sickness and eradicate it before it takes form. **His name is unknown outside our home**

Software Quality Assurance

- Software QA involves the entire software development **process**
  - Monitoring and improving the process
  - Making sure that any agreed-upon standards and procedures are followed
  - Ensuring that problems are found and dealt with
- It is oriented to “prevention”
  - It can dealt with any situation for which the system is built with

Causes of Software Errors

- **Miscommunication or no communication**
  - As to specifics of what an application should or should not do (the application's requirement)
- **Software complexity**
  - The complexity of current software applications can be difficult to comprehend for anyone without experience in modern-day software development
  - GUI interfaces, client-server, distributed and web savvy applications, sheer size of users have all contributed to the exponential growth in system complexity
- **Technological advancement**
  - Rapid change in hardware and software technology does not permit to get acquaintance with the advancement

Causes of Software Errors

- **Programming errors**
  - Programmers, like anyone else prone to human errors
- **Changing requirements**
  - If there are many minor or major changes, known and unknown dependencies among parts of project likely to interact and cause problems, and the complexity of coordinating changes may result in errors
- **Time pressure**
  - Scheduling of software projects is difficult at best, often requiring a lot of guess work. When deadline loom and the crunch come, mistakes will be made

Error vs. Failure

- **An error is a defect or bug**
  - A linked list is used whose link field in the last node is not assigned a NULL value
- **A failure is a manifestation of an error**
  - An array is used without its initialization and thus having garbage. Search an item may match a garbage value and the system than fails
- **But the mere presence of an error may not necessarily lead to a failure**
  - Searching over a linked-list may goes on infinitely; system is running but no result.
  - Setting a zero to all balances is an error + failure

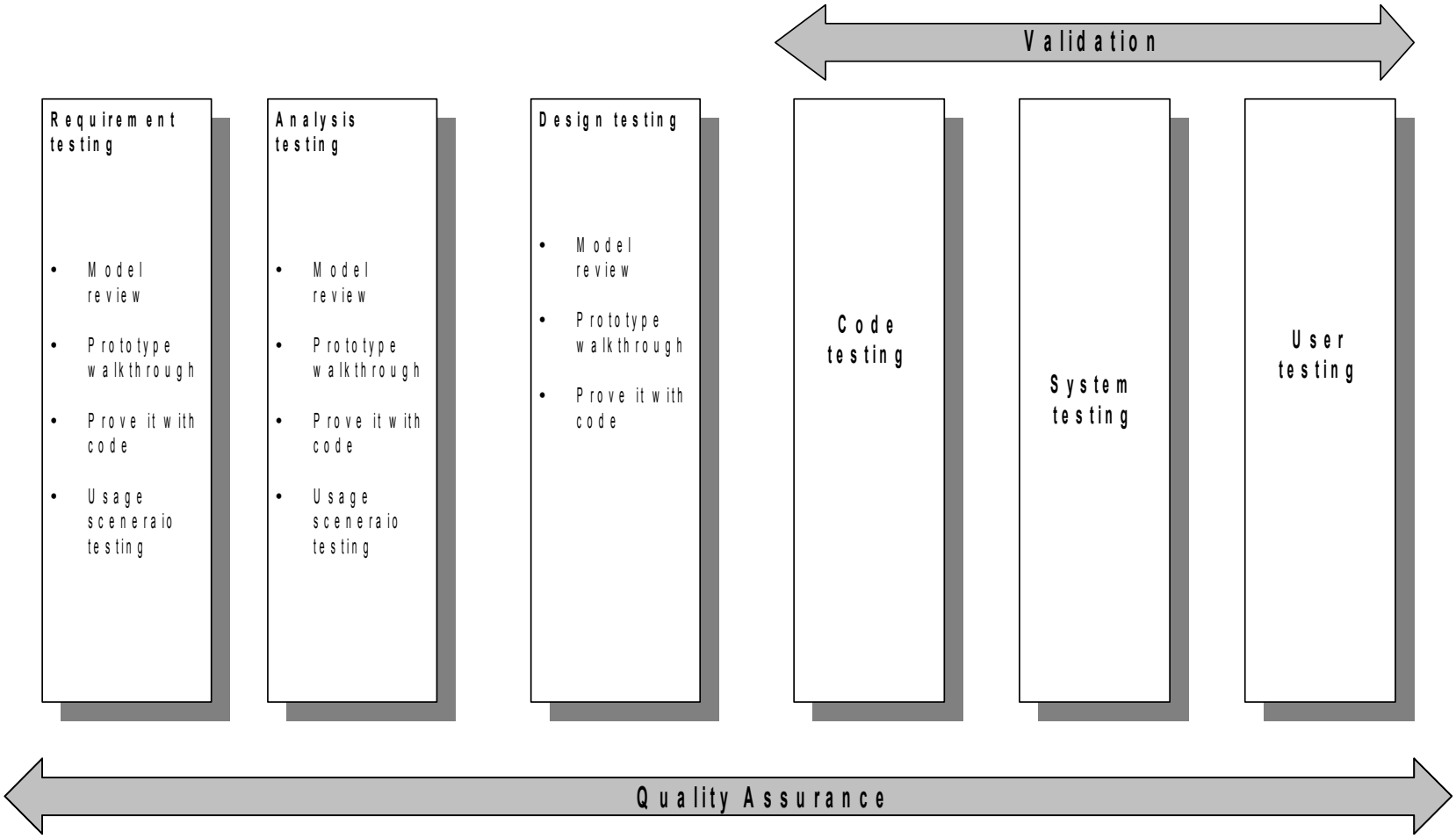
Verification & Validation

- **Verification** is the process of determining whether the output of one phase of software development conforms to that of its previous phase or not
- **Validation** is the process of determining whether a fully developed system conforms to its requirements specification
- **Verification** typically involves reviews and meetings to evaluate documents, plans, code, requirements, and specifications. This can be done with walkthrough and inspection meetings
- **Validation** typically involves actual testing and takes place after verifications are completed

Verification vs. Validation

- **Verification** is concerned with phase containment of errors
- **Validation** is concerned with the error free product
- **Verification** → *Are we doing the right things?*
- **Validation** → *Have we done the right things?*
- The term "IV & V" refers to Independent Verification and Validation

System Life Cycle Testing



Problems to Ponder

- What are the deciding factors for the effort required in
  - QA?
  - Validation?
  - Verification?
- Who are the key persons in QA?
- Trade-off in “Tested OK” and “Quality Guaranteed”