Software Issues

Definitions

- Software- computer programs made up of a logical sequence of commands to perform a task.
- The software producer/developer creates computer programs to meet either general or specific needs of the consumer
- A buyer gets the benefits of a computer program to solve a specific task/problem.Whenever there is a software there are producers and consumers.

- There is, therefore, a relationship between software producers and users made up of: user expectations and developer limitsFor a healthy relationship all the following must be agreed on:

(1) Standards – universally accepted level of confidence

(2) Reliability – software reliability does not depend on age and wear and tear like hardwareSoftware reliability - is the probability that the software does not encounter an input sequence resulting into failure.

(3) security- software is secure if it does not contain trapdoors through which an intruder can access the system.

(4) Safety – the safety of a software product means the absence of a likelihood of an accident, a hazard, or a riskA number of life critical systems depend on software, therefore, software safety is important.

(5) Quality- a software product has quality if it maintains a high degree of excellence in standards, security, safety,and dependability.

Software standard

- A **software standard** is a standard, protocol, or other common format of a document, file, or data transfer accepted and used by one or more software developers while working on one or more than one computer programs. Software standards enable interoperability between different programs created by different developers.
- Software standards consist of certain terms, concepts, data formats, document styles and techniques agreed upon by software creators so that their software can understand the files and data created by a different computer program. To be considered a standard, a certain protocol needs to be accepted and incorporated by a group of developers who contribute to the definition and maintenance of the standard.
- Some developers prefer using standards for software development because of the efficiencies it provides for code development[1] and wider user acceptance and use of the resulting application.[2]

Software reliability

- Software reliability is the probability of failure-free operation of a computer program for a specified period in a specified environment.
- Reliability is a customer-oriented view of software quality. It relates to operation rather than design of the program, and hence it is dynamic rather than static.
- It accounts for the frequency with which faults cause problems. Measuring and predicting software reliability has become vital for software managers, software engineers, managers and engineers of products that include software, and to users of these products.

- It is difficult to define the term objectively.
- Difficult to measure user expectations,
- Difficult to measure environmental factors.
- It’s not enough to consider simple failure rate:
  – Not all failures are created equal; some have much more serious consequences. – Might be able to recover from some failures reasonably

Failures and Faults

- A failure corresponds to unexpected runtime behavior observed by a user of the software.
- A fault is a static software characteristic which causes a failure to occur.

Not every fault causes a failure: – Code that is "mostly" correct. – Dead or infrequently-used code. – Faults that depend on a set of circumstances to occur. • If a tree falls ... – If a user doesn't notice wrong behavior, is it a fault? – If a user expects "wrong" behavior, is it a fault?

Improving Reliability

- Primary objective: Remove faults with the most serious consequences.
- Secondary objective: Remove faults that are encountered most often by users.

Fixing N% of the faults does not, in general, lead to an N% reliability improvement. • 90-10 Rule: 90% of the time you are executing 10% of the code. • One study showed that removing 60% of software “defects” led to a 3% reliability improvement.

Software Security

- Software security refers to a set of practices that help protect software applications and digital solutions from attackers. Developers incorporate these techniques into the software development life cycle and testing processes. As a result, companies can ensure their digital solutions remain secure and are able to function in the event of a malicious attack.
- Secure software development is incredibly important because there are always people out there who seek to exploit business data. As businesses become more reliant on software, these programs must remain safe and secure. With strong software security protocols in place, you can prevent attackers from stealing potentially sensitive information such as credit card numbers and trade secrets, and build trust among users.

Major Concerns with Software Security

- **Phishing**: Phishing happens when an attacker poses as someone else in an attempt to gain personal information, such as software credentials.
- **Distributed denial of service (DDoS) Attacks**: A DDoS attack happens when an attacker overloads servers with packets, causing the software to crash.
- **Cloud service attacks**: Companies are increasingly relying on cloud-based services to support remote workers. Some cloud infrastructure has vulnerabilities hackers can exploit.
- **Software supply chain attacks**: Some pieces of software are critical in the business supply chain, especially for e-commerce. A software supply chain attack happens when hackers exploit a third-party service to access data about a business.

Software security techniques

- **Patching your software:** This is the process of fixing software vulnerabilities as they are discovered.
- **Using a firewall:** A firewall is software or hardware that sits between your computer and the internet and helps protect your computer from unauthorized access.
- **Restricting administrative privileges:** Limiting the privileges of users who can access sensitive data can help with reducing attack surface, minimizing the risk of a data breach.
- **Encrypting your data:** Data encryption is a common cybersecurity practice that involves transforming readable data into an unreadable format. Decryption reverses this transformation.
- **Two-factor authentication:** Two-factor authentication requires you to provide two pieces of information, such as a password and a code generated by a mobile app, in order to log in to your account.
- **Employee training:** Employee training is essential for software security. Employees need to be aware of the risks associated with using software and how to protect themselves and their company's data.
-

What is software quality?

- The quality of software can be defined as the ability of the software to function as per user requirement. When it comes to software products it must satisfy all the functionalities written down in the SRS document.
  Key aspects that conclude software quality include,
- **Good design** – It's always important to have a good and aesthetic design to please users
- **Reliability** – Be it any software it should be able to perform the functionality impeccably without issues
- **Durability-** Durability is a confusing term, In this context, durability means the ability of the software to work without any issue for a long period of time.
- **Consistency** – Software should be able to perform consistently over platform and devices
- **Maintainability** – Bugs associated with any software should be able to capture and fix quickly and news tasks and enhancement must be added without any trouble
- **Value for money –** customer and companies who make this app should feel that the money spent on this app has not fone to waste.
- **Portability:**
  A software is claimed to be transportable, if it may be simply created to figure in several package environments, in several machines, with alternative code merchandise, etc.
- **Usability:**
  A software has smart usability if completely different classes of users (i.e. each knowledgeable and novice users) will simply invoke the functions of the merchandise.
- **Reusability:**
  A software has smart reusability if completely different modules of the merchandise will simply be reused to develop new merchandise.
- **Correctness:**
  A software is correct if completely different needs as laid out in the SRS document are properly enforced.


Product quality model
Functional suitability
Functional completeness
Functional correctness
Functional appropriateness
Performance efficiency
Time behavior
Resource use
Capacity
Compatibility
Coexistence
Interoperability
Usability
Appropriateness recognizability
Learnability
Operability
User-error protection
User-interface aesthetics
Accessibility
Reliability
Maturity
Availability
Fault tolerance
Recoverability
Security
Confidentiality
Integrity
Nonrepudiation
Accountability
Authenticity
Maintainability
Modularity
Reusability
Analyzability
Modifiability
Testability
Portability
Adaptability
Installability
Replaceability

Causes of Software Failures

- here are factors that contribute to software failures:
- Human factors
- Nature of software
- Safety critical systems – these are software systems with real-time control components that can have a direct life-threatening impact
- Examples of critical systems:
- Nuclear reactors
- Missile systems
- Aircraft and air control systems

Consumer Protection and the Law

Buyer's rights:

- Replacement
- Refunds
- Updates

Understanding software complexity- software as:

- Product
- Service
- Mix

Consumer Protection and the Law…

Costumer protection tools:

(1) contract (used with products):

- Express warranties
- Implied warranties
- Third-party beneficiary
- Breach of contract – lack of compliance

(2) Tort (used with services):

- Intentional
- Unintentional

Consumer Protection and the Law…

Torts include:

- Negligence – careless, lack of competence, etc..
- Malpractice
- Strict liability
- Misrepresentation

Improving Software Quality

- The safety and reliability of a software product defines the quality of that software
- Software quality can only be improved during the development cycle
- The following techniques done during the software development phase can improve software quality:
- Final review
- Inspection
- Walk-throughs
- Phased-inspection

Producer Protection and the Law

Protection against:

- Piracy
- Illegal copying/downloading of copyrighted software
- Fraudulent lawsuits by customers
- Seek protection from the courts