CS5030

Dependability

Learning objectives

- On completing this lecture and associated reading, you should
  - Understand why dependability is an important characteristic of a software system
  - Understand the five principal dimensions of dependability
  - Be aware of the specialised terminology that is used when discussing dependability
  - Understand how dependability may be achieved in software

Motivation

- Increasing use of software and reliance on it
- Increasing instances of software failure
  - Costs ranging from inconvenience to potential loss of life
- Increasing need to develop and maintain dependable systems
- Extent of dependability required of a system depends on its use

Dependability

- Reflects the extent of the user's confidence that the system will operate as users expect and that it will not fail during normal use

- For many computer systems, this is often the most important property

- Dependability is subjective
  - Depends on the judgement of stakeholders
  - What is a failure to one stakeholder may be acceptable behaviour to another

Dependability and system specifications

- Dependability can only be defined formally with respect to a system specification
  - A failure is a deviation from this specification
- However, many specifications are incomplete or incorrect
  - A system that conforms to its specification may fail from the perspective of system users
- Users don’t typically read specifications so don’t know how the system is supposed to behave
  - Therefore, perceived dependability is more important in practice

Dependability – conceptual framework

- Dependability tree

- Dependability
  - Attributes / dimensions / properties
  - Threats
  - Means

Dimensions of dependable systems (1)

- Original
  - Availability, reliability, safety & security

- Revised
  - Availability, reliability, safety, security and resilience

Dimensions of dependable systems (2)

- Availability
  - The ability of the system to deliver services when requested
- Reliability
  - The ability of the system to deliver services as requested
- Safety
  - The ability of the system to operate without catastrophic failure
- Security
  - The ability of the system to protect itself against deliberate or accidental intrusion
- Resilience
  - The ability of the system to resist and recover from damaging events

Other dependability properties

- Repairability
  - Reflects the extent to which the system can be repaired in the event of a failure
- Maintainability
  - Reflects the extent to which the system can be adapted to new requirements
- Error tolerance
  - Reflects the extent to which user input errors can be avoided and tolerated

Dependencies among dependability dimensions - examples

- Safe system operation depends on the system being available and operating reliably
- A system may be unreliable because its data has been corrupted by an external attack
- Denial of service attacks on a system are intended to make it unavailable
- If a system is infected with malware, you cannot be confident of its reliability or safety

Threats to dependability

- Failures
  - A failure is an event that occurs when the delivered service deviates from correct service
- Errors
  - An error is a deviation of at least one system state from the correct service state
- Faults
  - A fault is an adjudged or hypothesised cause of an error

Causes of failures

- Hardware
  - Design and manufacturing errors or components reaching the end of their natural life
- Software
  - Errors in its specification, design or implementation
- Operational
  - Errors made by human operators

- Causes may be related

Consequences of failures

- System failures may have widespread effects with large numbers of people affected by the failure
- Systems that are not dependable and are unreliable, unsafe or insecure may be rejected by their users
- The costs of system failure may be very high if the failure leads to economic losses or physical damage
- Undependable systems may cause information loss with a high recovery cost

Means to achieve dependability

- Fault prevention / avoidance
  - Means to prevent the occurrence or introduction of faults
- Fault tolerance
  - Means to avoid service failures in the presence of faults
- Fault detection
  - Means to detect faults before the system goes into service
- Fault removal
  - Means to reduce the number and severity of faults

Costs of dependability

- Costs can increase exponentially as increasing levels of dependability are required
  - Use of more expensive development techniques and hardware required to achieve higher levels of dependability
  - Increased testing and system validation required to convince clients and regulators that the required levels of dependability have been achieved

- Because of high costs of dependability, it may be more cost effective to accept untrustworthy systems and pay for failure costs
  - Possibility depends on social, political and domain factors

Regulated systems

- Many critical systems are regulated systems
  - Nuclear systems
  - Air traffic control systems
  - Medical devices

- Their use must be approved by an external regulator before the systems go into service
  - A safety and dependability case has to be approved by the regulator
  - The development team has to create the evidence to convince a regulator that the system is dependable, safe and secure

Tactics for dependability (1)

- Redundancy
  - Create and maintain more than a single version of critical components so that if one fails then a backup is available
- Diversity
  - Provide the same functionality in different ways in different components so that they will not fail in the same way

Tactics for dependability (2)

- Redundant and diverse components should be independent so that they will not suffer from common-mode failures
  - For example, components implemented in different programming languages means that a compiler fault will not affect all of them
- Redundancy and diversity apply to development processes as well as software
  - Process activities, such as validation, should not depend on a single approach, such as testing, to validate the system
  - Explicitly defined, repeatable processes are required

Challenges

- Adding diversity and redundancy to a system increases the system complexity
- This can increase the chances of error because of unanticipated interactions and dependencies between the redundant system components
- Some engineers therefore advocate simplicity and extensive verification and validation as a more effective route to software dependability

Dependable process activities

- Requirements reviews
- Requirements change management
- Formal specification
- Documentation of software design along with links to requirements
- Design and software inspections
- Static analysis of software
- Test planning and management

Dependable processes and agility

- Dependable software often requires certification
- More upfront planning, documentation and analysis are therefore required
  - Conflict with pure agile methodology
- A hybrid, custom-defined agile methodology incorporating dependable techniques may be used

Chaos engineering

- Deliberate introduction of failures in production
  - Testing the resilience of systems against failures
    - Infrastructure, network, software
    - Control blast radius
  - Pioneered by Netflix
    - Automated tool that randomly chose a server and disabled it

- Resources
  - Chaos Monkey and Simian Army by Netflix
  - Phoenix Servers by Martin Fowler
  - Chaos Engineering by Gremlin

Key points

- System dependability is important because failure of critical systems can lead to economic losses, information loss, physical damage or threats to human life
- The dependability of a computer system is a system property that reflects the user's degree of trust in the system
- The most important dimensions of dependability are availability, reliability, safety, security and resilience
- The use of dependable, repeatable processes is essential if faults in a system are to be minimised
- The use of redundancy and diversity in hardware, software processes and software systems is essential to the development of dependable systems