Chapter 10 – Dependable systems

Topics covered

- ✧ Dependability properties
- ✧ Sociotechnical systems
- ✧ Redundancy and diversity
- ✧ Dependable processes
- ✧ Formal methods and dependability

System dependability

- The most important system property is the dependability
- Reflect the user's degree of trust in that system.
- Reflect the extent of the user's confidence that it will operate as users expect.
- Cover the related attributes: reliability, availability and security.

3

Importance of dependability

- System failures may have widespread.
- Systems that are not dependable may be rejected.
- The costs of system failure is high if the failure leads to economic losses.
- Undependable systems may cause information loss.

4

Causes of failure

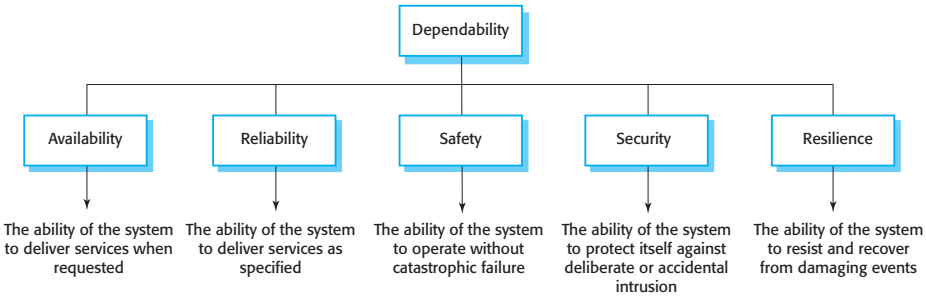
- ✧ Hardware failure
  - Design and manufacturing errors.
- ✧ Software failure
  - Errors in its implementation.
- ✧ Operational failure
  - Human operators make mistakes.

5

Dependability properties

6

The principal dependability properties



7

Principal properties

- ✧ Availability
  - Deliver useful services to users.
- ✧ Reliability
  - Correctly deliver services as expected.
- ✧ Safety
  - Capability of preventing damage to people or its environment.

8

Principal properties

- Security
  - Capability of resisting accidental or deliberate intrusions.
- Resilience
  - A judgment of how well a system can maintain the continuity of its critical services.

9

Other dependability properties

- Repairability
  - Capability of being repaired in the event of a failure
- Maintainability
  - Capability of being adapted to new requirements
- Error tolerance
  - Capability to tolerate failures due to user input errors

10

Dependability attribute dependencies

- ✧ Depend on the system's availability and reliability.
- ✧ Corrupted data by an external attack.
- ✧ Unavailable to conduct denial of service attacks on a system.
- ✧ Malicious system virus infection and damage

11

Dependability achievement

- ✧ Inspect and avoid accidental error introduction.
- ✧ Validation processes to reveal errors.
- ✧ Fault tolerant system to tolerate runtime errors.
- ✧ Protection mechanisms against external attacks.

12

Dependability achievement

- ✧ Correct system configuration.
- ✧ Capabilities to resist cyberattacks.
- ✧ Service recovery mechanisms after a failure.

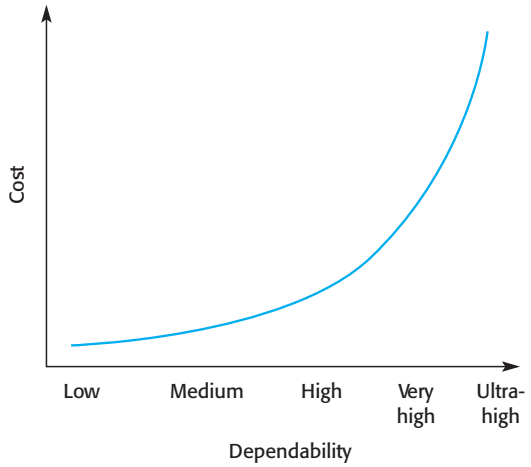
13

Dependability costs

- ✧ Dependability costs increase exponentially.
- ✧ There are two reasons for this
  - Expensive development techniques and hardware for higher levels of dependability.
  - Increased testing and system validation for system clients and regulators.

14

Cost/dependability curve



15

Dependability economics

- ✧ Accepting untrustworthy systems and pay for failure costs may be cost effective.
- ✧ However, it may lose future business depending on social and political factors.
- ✧ Depends on system types that need modest levels of dependability.

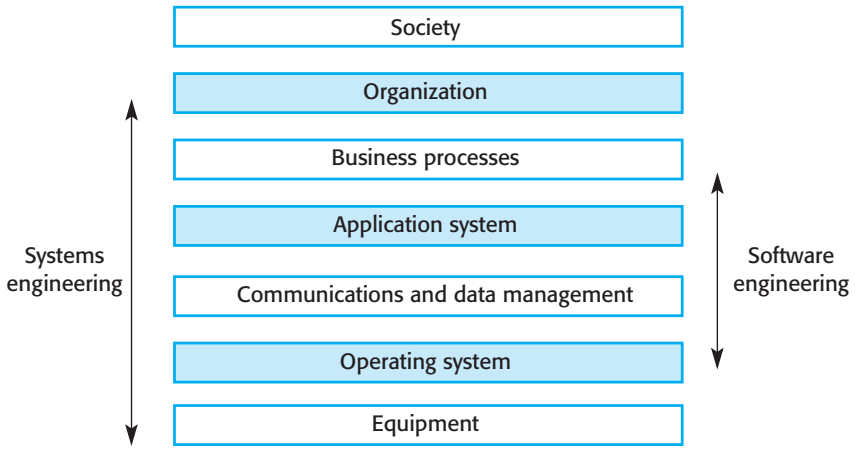
16

Sociotechnical systems (STS)

Systems and software

- ✧ Software engineering is part of system engineering process.
- ✧ Software systems are are essential components of systems based on organizational purposes.
- ✧ Example
  - The wilderness weather system is part of forecasting systems
  - Hardware and software, forecasting processes, the organizations, etc.

The sociotechnical systems (STS) stack



19

Layers in the STS stack

- Equipment
  - Hardware devices, including embedded systems
- Operating system
  - Common facilities for higher level applications.
- Communications and data management
  - Access to remote systems and databases.
- Application systems
  - Functionalities for specific requirements.

20

Layers in the STS stack

- ✧ Business processes
  - Processes involving people and systems
- ✧ Organizations
  - Business activities for system operations
- ✧ Society
  - Laws, regulation and culture

21

Holistic system design

- ✧ Interactions and dependencies between system layers
  - Example: regulation changes causes changes in applications.
- ✧ For dependability, a systems perspective is essential
  - Software failures within the enclosing layers.
  - Failures in adjacent layers affects software systems.

22

Regulation and compliance

- ✧ The general model of economic organization
  - Universal in the world.
  - Offer goods and services and make a profit.
- ✧ Ensure the safety of their citizens
  - Follow standards to ensure that products are safe and secure.

23

Regulated systems

- ✧ Critical systems are regulated systems
  - Approved by an external regulator.
  - E.g., nuclear systems and air traffic control systems
- ✧ A safety and dependability case
  - Approved by the regulator.
  - Create the evidence for systems' dependability, safety and security.

24

Safety regulation

- Regulation and compliance applies to the sociotechnical system.
- Safety-related systems
  - Certified as safe by the regulator.
- Produce safety cases to show systems follow rules and regulations.
- Expensive to document certification.

25

Redundancy and diversity

Aug 29th

Redundancy and diversity

- Redundancy
  - Keep more than a single version.
- Diversity
  - Provide the same functionality in different mechanism.
- Redundant and diverse components should be performed independently
  - E.g., software written in different programming languages.

27

Diversity and redundancy examples

- Redundancy.
  - Backup servers to switch, when failure occurs.
- Diversity.
  - Different servers running on different operating systems.

28

Process diversity and redundancy

- ✧ Process activities
  - Not depend on a single approach, such as testing.
- ✧ Redundant and diverse process activities.
- ✧ Multiple process activities complement each other
  - Cross-checking techniques avoid process errors

29

Problems with redundancy and diversity

- ✧ Adding diversity and redundancy increases complexity.
- ✧ Increase the chances of error
  - Unanticipated interactions between redundant components.
- ✧ Advocate simplicity to decrease software dependability.
  - E.g., an Airbus product is redundant/diverse; a Boeing product has no software diversity

30

Dependable processes

Dependable processes

- A well-defined, repeatable software process to reduce faults.
- A well-defined repeatable process
  - Not depend on individual skills.
- Check whether to use software engineering practice.
- Verification and validation (V&V) activities for fault detection.

Dependable process characteristics

- ✧ Explicitly defined
  - A defined process model to drive the production process.
  - Data must be collected during the process to prove that the development follows process models.
- ✧ Repeatable
  - Not rely on individual judgment.
  - Can be repeated across projects and with different team members.

33

Attributes of dependable processes

Process characteristic	Description
Auditable	The process should be understandable by people apart from process participants, who can check that process standards are being followed and make suggestions for process improvement.
Diverse	The process should include redundant and diverse verification and validation activities.
Documentable	The process should have a defined process model that sets out the activities in the process and the documentation that is to be produced during these activities.
Robust	The process should be able to recover from failures of individual process activities.
Standardized	A comprehensive set of software development standards covering software production and documentation should be available.

34

Dependable process activities

- ✧ Requirements reviews
  - Check whether to be complete and consistent.
- ✧ Requirements management
  - Ensure that requirement modifications are controlled and understood.
- ✧ Formal specification
  - Analyze mathematical models.
- ✧ System modeling
  - Documentation through graphical models
  - Relationships between system requirements.

35

Dependable process activities

- ✧ Design and program inspections
  - Inspection and checking for systems by different people.
- ✧ Static analysis
  - Automated inspection on the program source code.
- ✧ Test planning and management
  - Design system test suites.
  - Manage to provide enough coverage of system requirements.

36

Dependable processes and agility

- ✧ Produce process and product documentation.
- ✧ Up-front requirements analysis
  - Discover requirements and requirements conflicts for system safety and security
- ✧ These conflict with agile development
  - Minimizing documentation of system requirements

37

Dependable processes and agility

- ✧ Agile process
  - iterative development, test-first development and user involvement in the development team.
- ✧ Agile team follows agile process, actions, and agile methods
- ✧ However, ‘pure agile’ is impractical for dependable systems.

38

Formal methods and dependability

Formal specification

- ✧ Formal methods
  - Development approaches based on mathematical analysis.
- ✧ Formal methods include
  - Formal specification;
  - Specification analysis and proof;
  - Transformational development;
  - Program verification.
- ✧ Reduce programming errors and cost for dependable systems.

Formal approaches

- ✧ Verification-based approaches
  - Different representations of a software system are proved to be equivalent.
  - Demonstrate the absence of errors.
- ✧ Refinement-based approaches
  - A system representation is transformed into a lower-level representation.
  - Correct transformation results in equivalent representations.

41

Use of formal methods

- ✧ The principal benefits
  - Reduce faults or runtime errors.
- ✧ Applicable main area:
  - Dependable systems engineering.
- ✧ Cost-effective formal methods
  - Reduce high system failure costs

42

Classes of error

- ✧ Specification and design errors and omissions.
  - Reveal errors and omissions in requirements.
  - Models generated automatically from source code.
  - Analysis by using model checking find undesirable faults.
- ✧ Inconsistences between a specification and a program.
  - Refinement methods
  - Programmer mistakes of inconsistencies with specification
  - Discover inconsistencies between programs and specifications.

43

Benefits of formal specification

- ✧ Developing a formal specification
  - Analyze system requirements in detail.
  - Detect problems, inconsistencies and incompleteness.
- ✧ Specification expressed in a formal language
  - Discover inconsistencies and incompleteness.
- ✧ A formal method correctly transforms a formal specification into a program.
- ✧ Reduce program testing costs
  - Verify a program formally against its specification.

44

Acceptance of formal methods

- Formal methods limited in practical development:
  - Hard to understand a formal specification
  - Cannot assess if it is an accurate representation.
  - Assess development costs but harder to assess the benefits.
  - Unwilling to invest in formal methods.
  - Unfamiliar with formal method approach.
  - Difficulty in scaling up to large systems.
  - Incompatibility with agile development methods.

45

Key points

- System dependability is important
  - failure of critical systems can lead to economic losses, information loss, physical damage or threats to human life.
- The dependability of a computer system
  - a system property that reflects the user's degree of trust in the system.
  - The most important dimensions are availability, reliability, safety, security and resilience.
- Sociotechnical systems
  - computer hardware, software and people, and are situated within an organization.
  - designed to support organizational or business goals and objectives.

46

Key points

- The use of a dependable, repeatable process
  - essential if faults in a system are to be minimized.
  - verification and validation activities at all stages, from requirements definition through to system implementation.
- The use of redundancy and diversity
  - essential to the development of dependable systems.
- Formal methods,
  - a formal model of a system as a basis for development
  - reduce the number of specification and implementation errors

47