2008 Fall

Software Modeling & Analysis

Part 1. Overview

- Introduction to Software Engineering
- Socio-Technical Systems
- Critical Systems

Lecturer: JUNBEOM YOO
jbyoo@konkuk.ac.kr

※ This lecture note is based on materials from Ian Sommerville 2006.

Chapter 1.
Introduction to Software Engineering

Objectives

- To introduce software engineering
- To explain software engineering's importance
- To set out the answers to key questions about software engineering

Software Engineering

- Software engineering
  - is concerned with theories, methods and tools for professional software development.
  - is concerned with cost-effective software development.

FAQs about Software Engineering

1. What is software?
2. What is software engineering?
3. What is the difference between software engineering and computer science?
4. What is the difference between software engineering and system engineering?
5. What is a software process?
6. What is a software process model?
7. What are the costs of software engineering?
8. What are software engineering methods?
9. What is CASE (Computer-Aided Software Engineering) ?
10. What are the attributes of good software?
11. What are the key challenges facing software engineering?

1. What is Software?

- Computer programs and associated documentation such as requirements, design models and user manuals

- Software products may be developed for a particular customer or may be developed for a general market.
- Software products may be
  - Generic - developed to be sold to a range of different customers e.g. PC software such as Excel or Word.
  - Bespoke (custom) - developed for a single customer according to their specification.

- New software can be created by developing new programs, configuring generic software systems or reusing existing software.

2. What is Software Engineering?

- Software engineering is <u>an engineering discipline</u> that is concerned with all aspects of software production.

- Software engineers should
  - adopt a systematic and organised approach to their work
  - use appropriate tools, techniques depending on the problem to be solved, the development constraints, and the resources available.

3. What is the Difference between Software Engineering and Computer Science?

- Computer science is concerned with theory and fundamentals.
- Software engineering is concerned with the practicalities of developing and delivering useful software.

- Computer science theories are still insufficient to act as a complete underpinning for software engineering (unlike e.g. physics and electrical engineering).

4. What is the Difference between Software Engineering and System Engineering?

- System engineering is concerned with all aspects of computer-based systems development including hardware, software and process engineering.

- Software engineering is part of this process concerned with developing the software infrastructure, control, applications and databases in the system.

5. What is a Software Process?

- A set of activities whose goal is the development or evolution of software.

- Generic activities in all software processes are:
  - Specification : what the system should do and its development constraints
  - Development : production of the software system
  - Validation : checking that the software is what the customer wants
  - Evolution : changing the software in response to changing demands

6. What is a Software Process Model?

- A simplified representation of a software process, presented from a specific perspective.

- Examples of process perspectives are
  - Workflow perspective : sequence of activities
  - Data-flow perspective : information flow
  - Role/action perspective : who does what

- Generic process models
  - Waterfall
  - Iterative development
  - Component-based software engineering

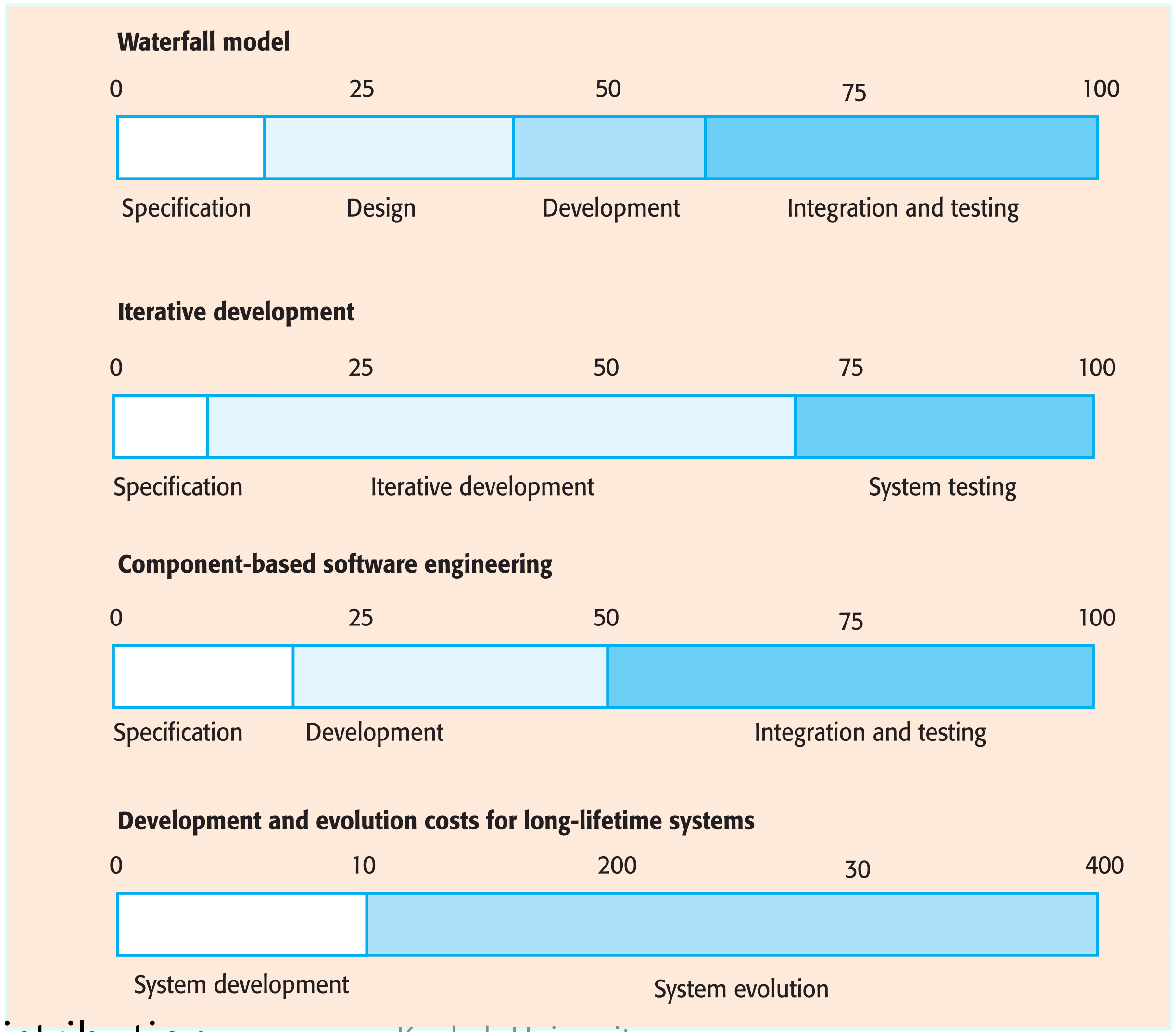
7. What are the Costs of Software Engineering?

- Roughly 60% of costs are development costs and 40% are testing costs.
- For custom software, evolution costs often exceed development costs.

- Costs vary depending on the type of system being developed and the requirements of system attributes such as performance and system reliability.

- Distribution of costs depends on the development model that is used.

7. What are the Costs of Software Engineering?



Activity-Cost Distribution

8. What are Software Engineering Methods?

- Structured approaches to software development which include system models, notations, rules, design advice and process guidance.
  - Model descriptions
    - Descriptions of graphical models which should be produced
  - Rules
    - Constraints applied to system models
  - Recommendations
    - Advice on good design practice
  - Process guidance
    - What activities to follow

9. What is CASE (Computer-Aided Software Engineering) ?

- Software systems that are intended to provide <u>automated support</u> for software process activities.

- CASE systems are often used for method support.

- Upper-CASE
  - Tools to support the early process activities of requirements and design
- Lower-CASE
  - Tools to support later activities such as programming, debugging and testing

10. What are the Attributes of Good Software?

- The software should deliver the required functionality and performance to the user and should be maintainable, dependable and acceptable.

- Maintainability
  - Software must evolve to meet changing needs
- Dependability
  - Software must be trustworthy
- Efficiency
  - Software should not make wasteful use of system resources
- Acceptability
  - Software must accepted by the users for which it was designed.
  - It must be understandable, usable and compatible with other systems.

Summary

- Software engineering is an engineering discipline that is concerned with all aspects of software production.
- Software products consist of developed programs and associated documentation. Essential product attributes are maintainability, dependability, efficiency and usability.
- The software process consists of activities that are involved in developing software products. Basic activities are software specification, development, validation and evolution.
- Methods are organized ways of producing software. They include suggestions for the process to be followed, the notations to be used, rules governing the system descriptions which are produced and design guidelines.
- CASE tools are software systems which are designed to support routine activities in the software process such as editing design diagrams, checking diagram consistency and keeping track of program tests which have been run.

Chapter 2.
Socio-technical Systems

Objectives

- To explain what a socio-technical system is and the distinction between this and a computer-based system
- To introduce the concept of emergent system properties
- To explain system engineering
- To discuss legacy systems and why these are critical to many businesses

What is a System?

- A purposeful collection of inter-related components working together to achieve some common objective.
- A system may include software, mechanical, electrical and electronic hardware and be operated by people.

- <u>Technical computer-based systems</u>
  - Systems that include hardware and software, but where the operators and operational processes are not normally considered to be part of the system.
  - The system is not self-aware.

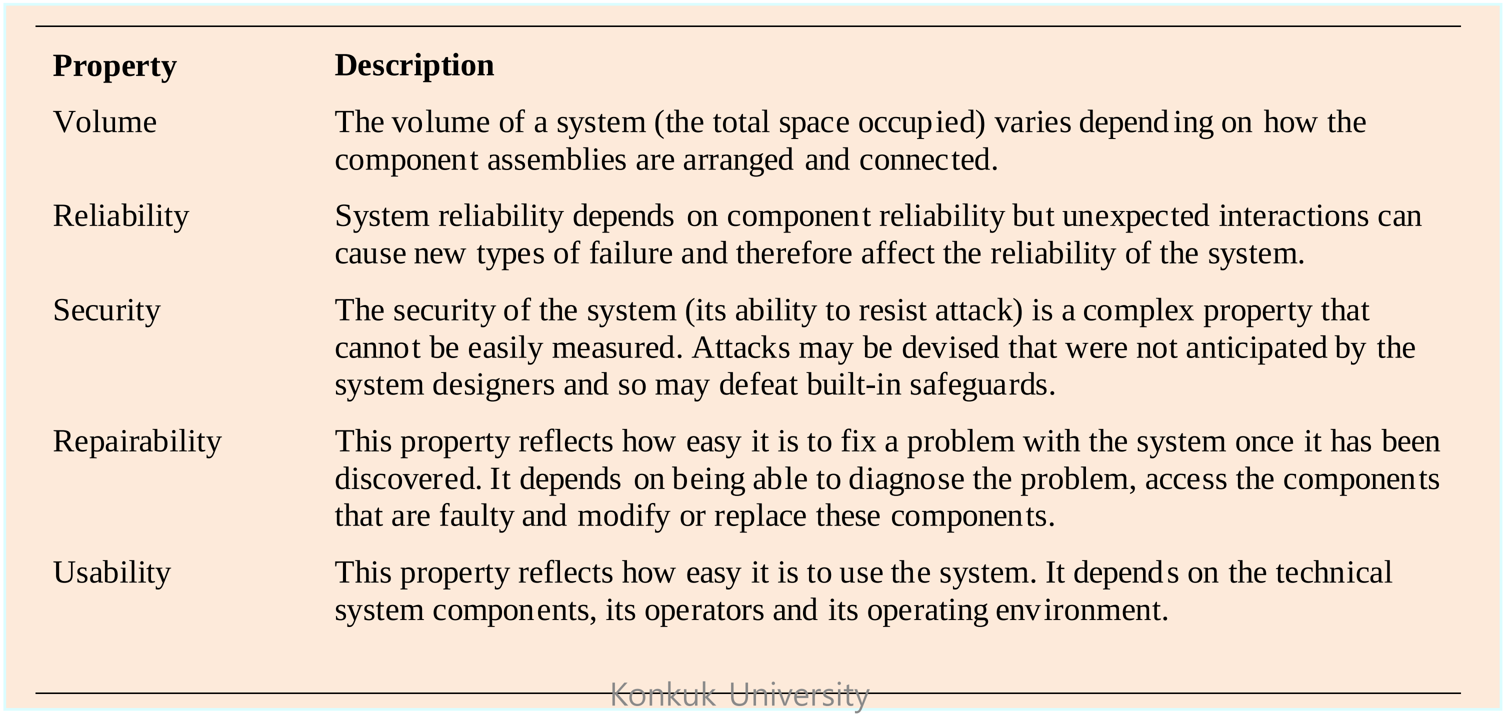
- <u>Socio-technical systems</u>
  - Systems that include technical systems but also operational processes and people who use and interact with the technical system.
  - Socio-technical systems are governed by organisational policies and rules.

Socio-technical System Characteristics

- Emergent properties
  - Properties of the system as a whole that depend on the system components and their relationships
  - Non-deterministic
    - They do not always produce the same output when presented with the same input because the system's behaviour is partially dependent on human operators.
  - Complex relationships with organisational objectives
    - The extent to which the system supports organisational objectives does not just depend on the system itself.

Emergent Properties

- Properties of the system as a whole rather than properties that can be derived from the properties of components of a system
- Emergent properties are a consequence of the relationships between system components. Therefore, they can only be assessed and measured once the components have been integrated into a system.



Property	**Description**
Volume	The volume of a system (the total space occupied) varies depending on how the component assemblies are arranged and connected.
Reliability	System reliability depends on component reliability but unexpected interactions can cause new types of failure and therefore affect the reliability of the system.
Security	The security of the system (its ability to resist attack) is a complex property that cannot be easily measured. Attacks may be devised that were not anticipated by the system designers and so may defeat built-in safeguards.
Repairability	This property reflects how easy it is to fix a problem with the system once it has been discovered. It depends on being able to diagnose the problem, access the components that are faulty and modify or replace these components.
Usability	This property reflects how easy it is to use the system. It depends on the technical system components, its operators and its operating environment.

Types of Emergent Property

- Functional properties
  - These appear when all the parts of a system work together to achieve some objective.
  - For example, a bicycle has the functional property of being a transportation device once it has been assembled from its components.

- Non-functional properties
  - These relate to the behaviour of the system in its operational environment.
  - Examples are reliability, performance, safety, and security.
  - They are often critical for computer-based systems as failure to achieve some minimal defined level in these properties may make the system unusable.

System Reliability Engineering

- Because of component inter-dependencies, faults can be propagated through the system. System failures often occur because of unforeseen inter-relationships between components.
- It is probably impossible to anticipate all possible component relationships.
- Software reliability measures may give a false picture of the system reliability.

- Influences on reliability
  - Hardware reliability
    - What is the probability of a hardware component failing and how long does it take to repair that component?
  - Software reliability
    - How likely is it that a software component will produce an incorrect output.
  - Operator reliability
    - How likely is it that the operator of a system will make an error?

The 'Shall-Not' Properties

- Properties such as performance and reliability can be measured.
- However, some properties are properties that the system should not exhibit
  - Safety : the system should not behave in an unsafe way;
  - Security : the system should not permit unauthorised use.
- Measuring or assessing these properties is very hard.

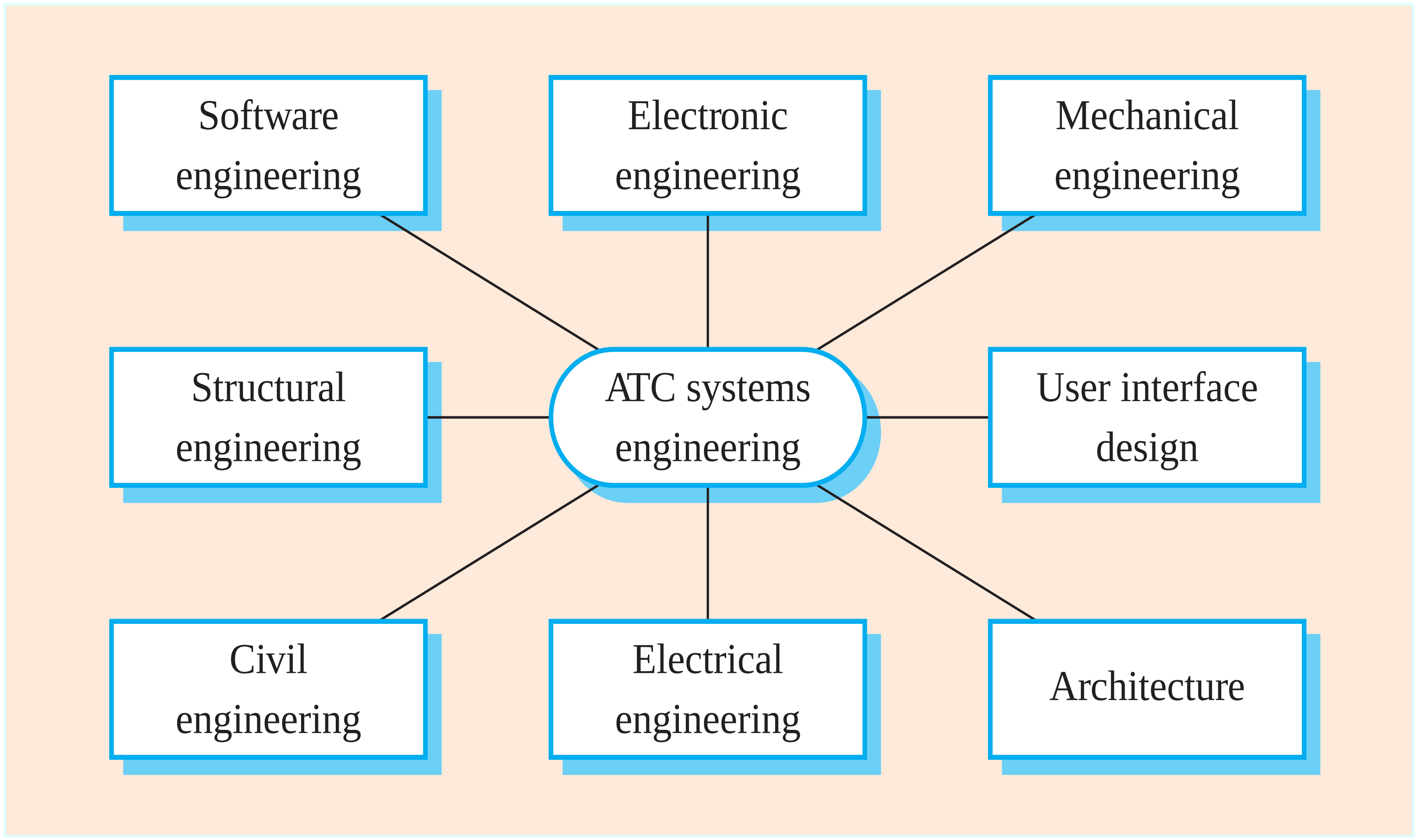
Systems Engineering

- Specifying, designing, implementing, validating, deploying and maintainin g socio-technical systems

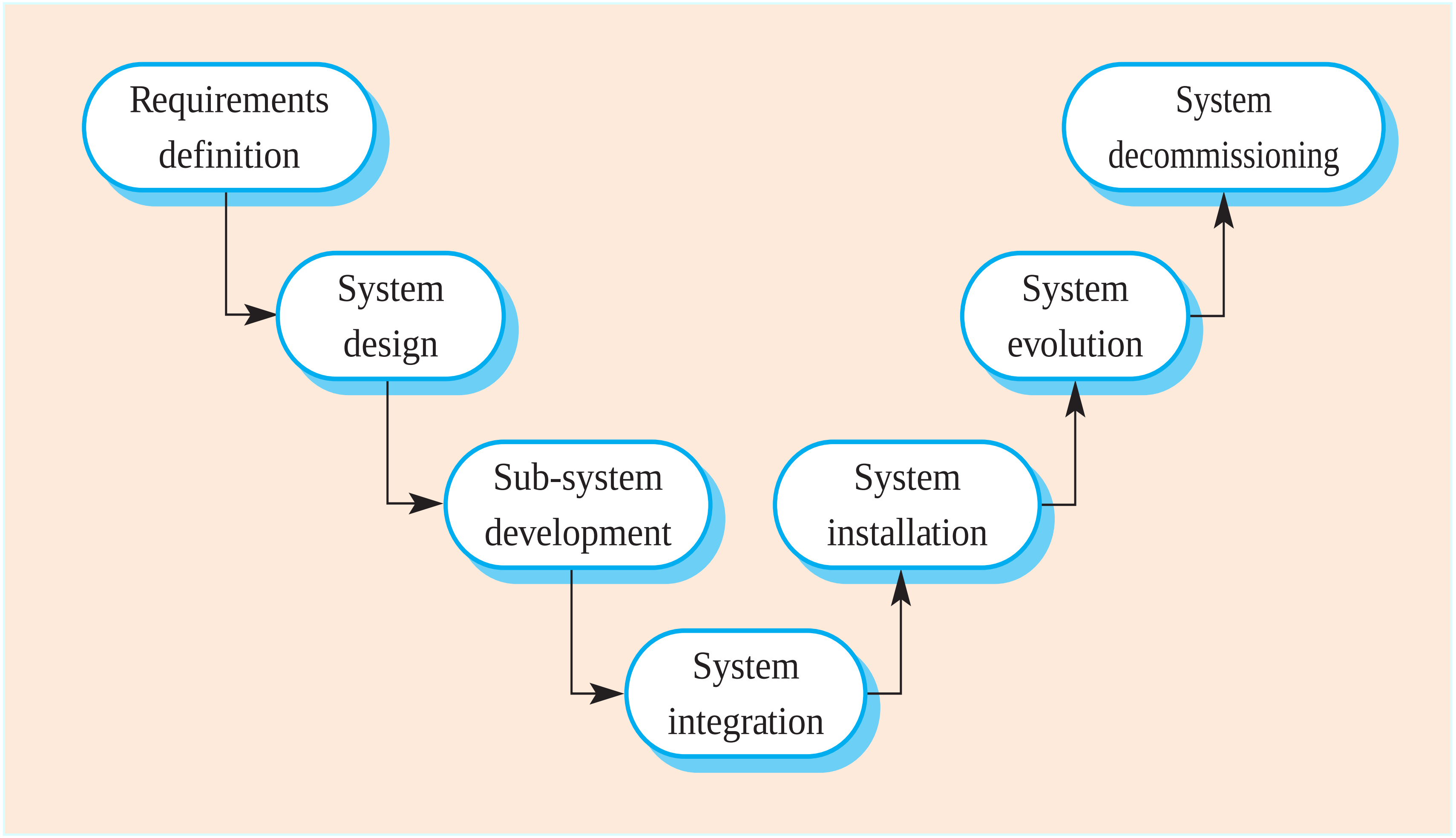
- System engineering is concerned with
  - services provided by the system,
  - constraints on its construction, and
  - operation and the ways in which it is used

- <u>System engineering process</u>
  - Usually follows a "Waterfall" model because of the need for parallel development of different parts of the system.
  - Inevitably involves engineers from different disciplines who must work together, and misunderstanding occurs here.

Example: Inter-Disciplinary Involvement



Systems Engineering Process



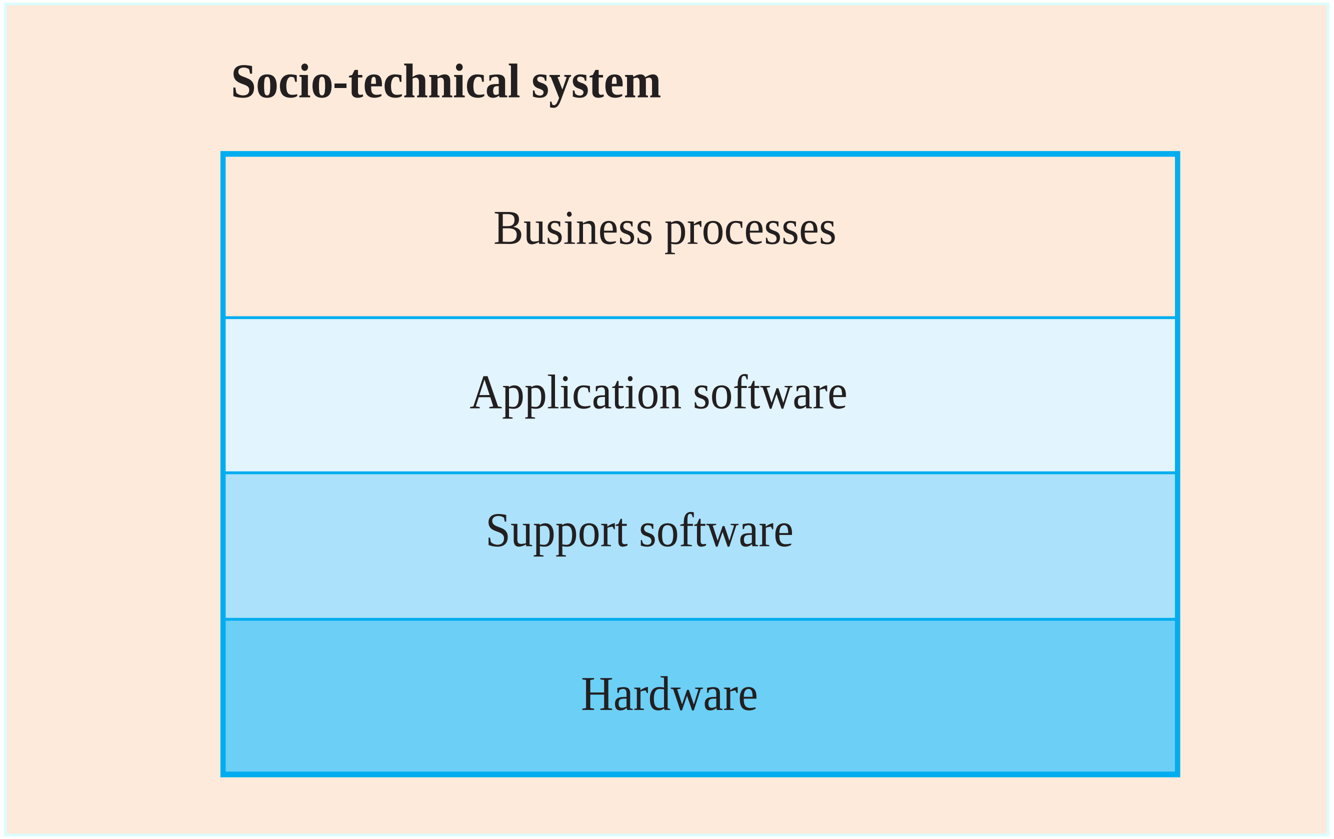
Legacy Systems

- Socio-technical systems that have been developed using old or obsolete technology.

- Crucial to the operation of a business and it is often too risky to discard these systems
  - Bank customer accounting system
  - Aircraft maintenance system

- Legacy systems constrain new business processes and consume a high proportion of company budgets.

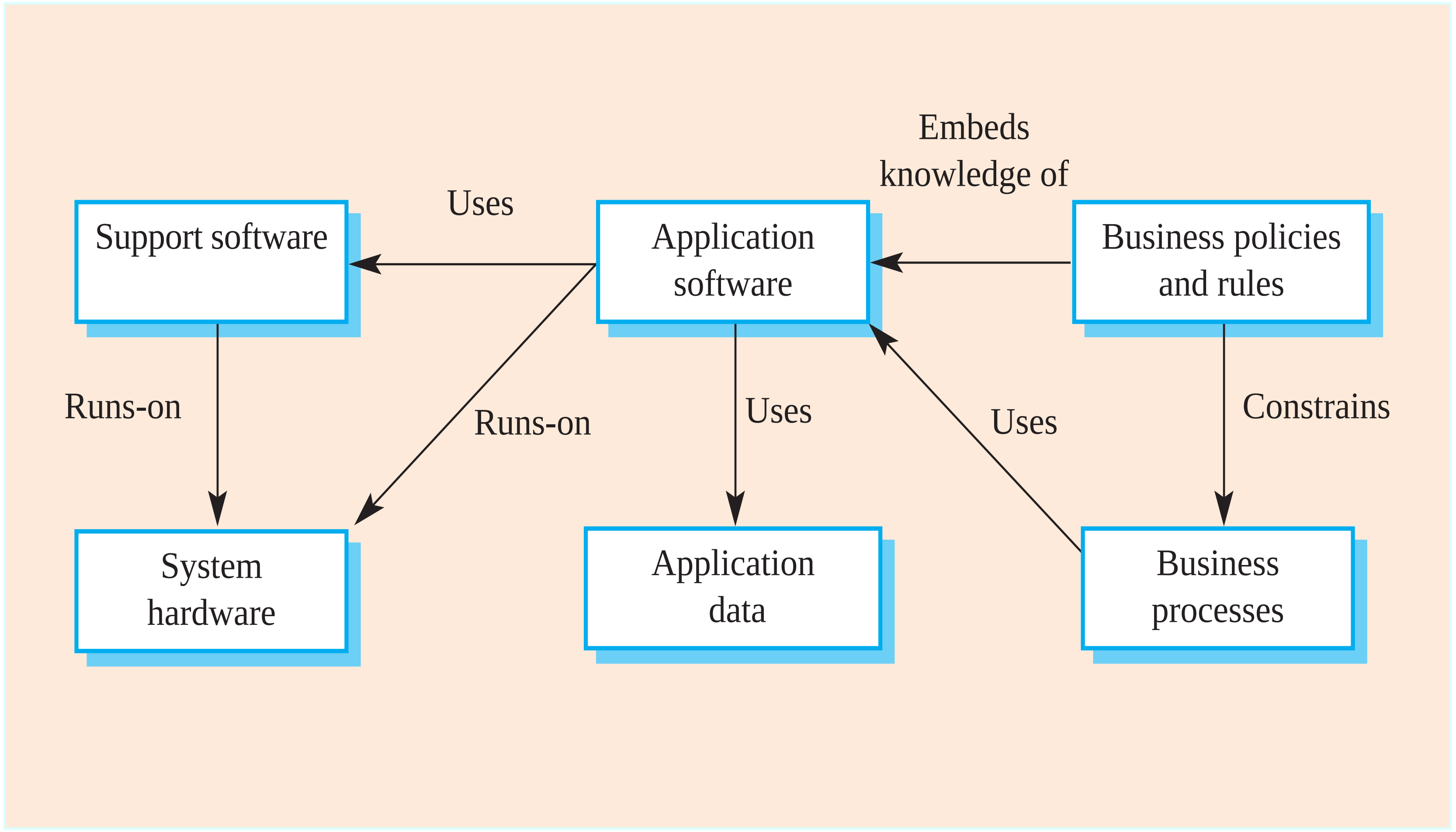
Socio-technical Legacy System



Legacy System Components

- Hardware
  - may be obsolete mainframe hardware.
- Support software
  - may rely on support software from suppliers who are no longer in business.
- Application software
  - may be written in obsolete programming languages.
- Application data
  - often incomplete and inconsistent.
- Business processes
  - may be constrained by software structure and functionality.
- Business policies and rules
  - may be implicit and embedded in the system software.

Relationship between Legacy System Components



Summary

- Socio-technical systems include computer hardware, software and people and are designed to meet some business goal.
- Emergent properties are properties that are characteristic of the system as a whole and not its component parts.
- The systems engineering process includes specification, design, development, integration and testing. System integration is particularly critical.
- Human and organisational factors have a significant effect on the operation of socio-technical systems.
- A legacy system is an old system that continues to provide essential services.
- Legacy systems include business processes, application software, support software and system hardware.

Chapter 3.
Critical Systems

Objectives

- To explain what is a critical system
- To explain four dimensions of dependability - availability, reliability, safety and security.
- To explain that, to achieve dependability, you need to avoid mistakes, detect and remove errors and limit damage caused by failure.

Critical Systems

- Safety-critical systems
  - Failure results in loss of life, injury or damage to environment
  - Chemical plant protection system
- Mission-critical systems
  - Failure results in failure of some goal-directed activity
  - Spacecraft navigation system
- Business-critical systems
  - Failure results in high economic losses
  - Customer accounting system in bank

System Dependability

- For critical systems, it is usually the case that the most important system property is the dependability of the system.
- The dependability of a system reflects the user's degree of trust in that system.
- It reflects the extent of the user's confidence that it will operate as users expect and that it will not 'fail' in normal use.

- 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.
- Undependable systems may cause information loss with a high consequent recovery cost.

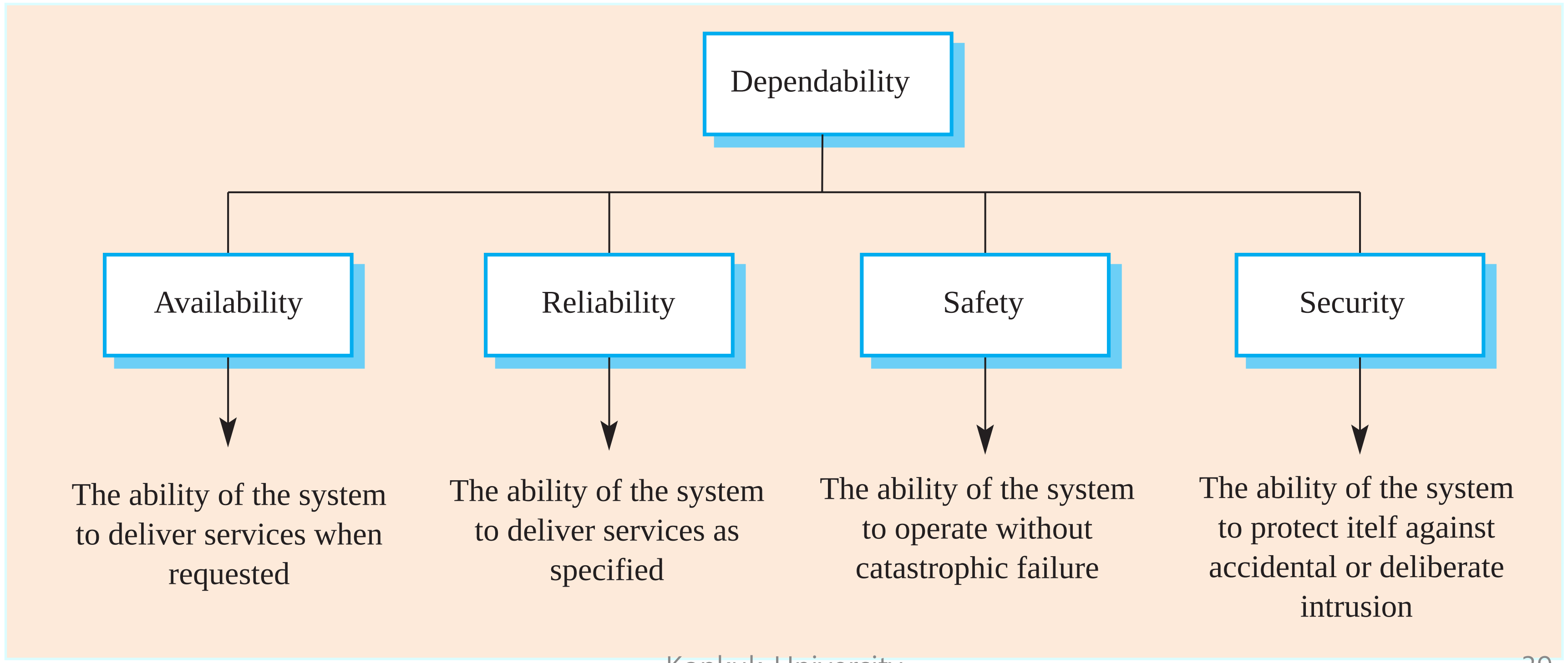
Development Methods for Critical Systems

- The costs of critical system failure are so high.
- Development methods for critical systems are not cost-effective for other types of system.

- Examples of development methods
  - Formal methods of software development
  - Static analysis
  - External quality assurance

Dependability

- Dependability of system equates to its trustworthiness.
- Dependable system is a system that is trusted by its users.
- Principal dimensions of dependability are:
  - Availability, Reliability, Safety, Security

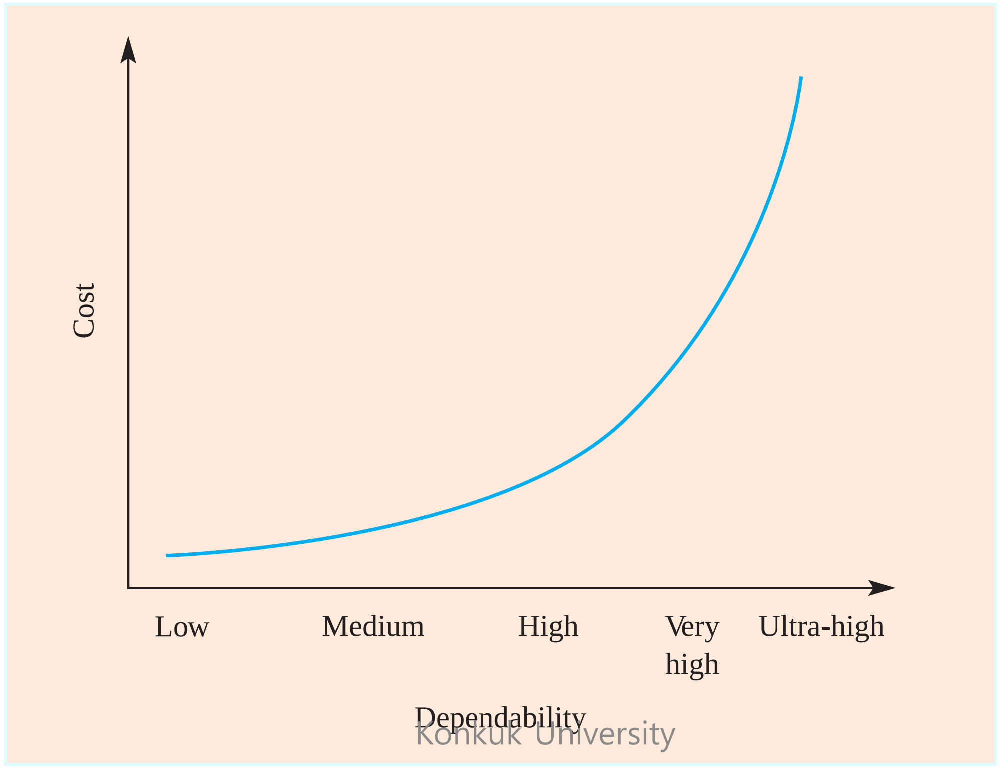

Dependability
Availability
Reliability
Safety
Security
The ability of the system to deliver services when requested
The ability of the system to deliver services as specified
The ability of the system to operate without catastrophic failure
The ability of the system to protect itelf against accidental or deliberate intrusion

Other Dependability Properties

- Repairability
  - To which extent the system can be repaired in the event of a failure
- Maintainability
  - To which extent the system can be adapted to new requirements
- Survivability
  - To which extent the system can deliver services whilst under hostile attack
- Error tolerance
  - To which extent user input errors can be avoided and tolerated

Dependability Costs

- Dependability costs tend to increase exponentially as required levels of dependability increase.
  - More expensive development techniques and hardware are required.
  - Increased testing and system validation are also required.



Dependability Economics

- Because of very high costs of dependability achievement
- It may be more <u>cost effective</u> to accept untrustworthy systems and pay for failure costs.

- However, it depends on social and political factors. Poor reputation for products may lose future business.

- It also depends on system type
  - For business systems, modest levels of dependability may be adequate.

Availability and Reliability

- <u>Availability</u>
  - The probability that a system will be operational and able to deliver the requested services, at a point in time

- <u>Reliability</u>
  - The probability of failure-free system operation over a specified time in a given environment for a given purpose

- Both of these attributes can be expressed quantitatively.

Reliability Terminology

Term	Description
System failure	An event that occurs at some point in time when the system does not deliver a service as expected by its users
System error	An erroneous system state that can lead to system behaviour that is unexpected by system users.
System fault	A characteristic of a software system that can lead to a system error. For example, failure to initialise a variable could lead to that variable having the wrong value when it is used.
Human error or mistake	Human behaviour that results in the introduction of faults into a system.

Reliability Achievement

- Fault avoidance
  - Development technique which either minimise the possibility of mistakes or trap mistakes before they result in the introduction of system faults are used.
- Fault detection and removal
  - Verification and validation techniques which increase probability of detecting and correcting errors before system goes into service are used.
- Fault tolerance
  - Run-time techniques are used to ensure that system faults do not result in system errors and/or to ensure that system errors do not lead to system failures.

Safety

- Safety is a property of system that reflects the system's ability to operate, normally or abnormally, without danger of causing human injury or death and without damage to the system's environment.
- Safety requirements are exclusive requirements
  - Exclude undesirable situations rather than specify required system services.

- Primary safety-critical systems
  - Failure can cause associated hardware to fail and directly threaten people.
- Secondary safety-critical systems
  - Failure results in faults in other systems which can threaten people.

Safety Terminology

Term	Definition
Accident (or mishap)	An unplanned event or sequence of events which results in human death or injury, damage to property or to the environment. A computer-controlled machine injuring its operator is an example of an accident.
Hazard	A condition with the potential for causing or contributing to an accident. A failure of the sensor that detects an obstacle in front of a machine is an example of a hazard.
Damage	A measure of the loss resulting from a mishap. Damage can range from many people killed as a result of an accident to minor injury or property damage.
Hazard severity	An assessment of the worst possible damage that could result from a particular hazard. Hazard severity can range from catastrophic where many people are killed to minor where only minor damage results.
Hazard probability	The probability of the events occurring which create a hazard. Probability values tend to be arbitrary but range from *probable* (say 1/100 chance of a hazard occurring) to implausible (no conceivable situations are likely where the hazard could occur).
Risk	This is a measure of the probability that the system will cause an accident. The risk is assessed by considering the hazard probability, the hazard severity and the probability that a hazard will result in an accident.

Safety Achievement

- Hazard avoidance
  - The system is designed so that some classes of hazard simply cannot arise.
- Hazard detection and removal
  - The system is designed so that hazards are detected and removed before they result in an accident
- Damage limitation
  - The system includes protection features that minimise the damage that may result from an accident

Security

- Security is a system property that reflects the system’s ability to protect itself from accidental or deliberate external attack.
- Security is becoming increasingly important as systems are networked so that external access to the system through the Internet is possible.
- Security is an essential pre-requisite for availability, reliability and safety.

Security Terminology

Term	Definition
Exposure	Possible loss or harm in a computing system. This can be loss or damage to data or can be a loss of time and effort if recovery is necessary after a security breach.
Vulnerability	A weakness in a computer-based system that may be exploited to cause loss or harm.
Attack	An exploitation of a system vulnerability. Generally, this is from outside the system and is a deliberate attempt to cause some damage.
Threats	Circumstances that have potential to cause loss or harm. You can think of these as a system vulnerability that is subjected to an attack.
Control	A protective measure that reduces a system vulnerability. Encryption would be an example of a control that reduced a vulnerability of a weak access control system.

Security Assurance

- <u>Vulnerability avoidance</u>
  - The system is designed so that vulnerabilities do not occur.
  - For example, if there is no external network connection, any external attack is impossible.
- <u>Attack detection and elimination</u>
  - The system is designed so that attacks on vulnerabilities are detected and neutralised before they result in an exposure.
  - For example, virus checkers find and remove viruses before they infect a system.
- <u>Exposure limitation</u>
  - The system is designed so that the adverse consequences of a successful attack are minimised.
  - For example, a backup policy allows damaged information to be restored.

Summary

- A critical system is a system where failure can lead to high economic loss, physical damage or threats to life.
- Dependability of a system reflects user's trust in that system.
- Availability is the probability that it will be available to deliver services when requested.
- Reliability is the probability that system services will be delivered as specified.
- Safety is a system attribute that reflects the system's ability to operate without threatening people or the environment.
- Security is a system attribute that reflects the system's ability to protect itself from external attack.
- Reliability and availability are generally seen as necessary but not sufficient conditions for safety and security.