Parul®
University

Software Engineering (203105303)

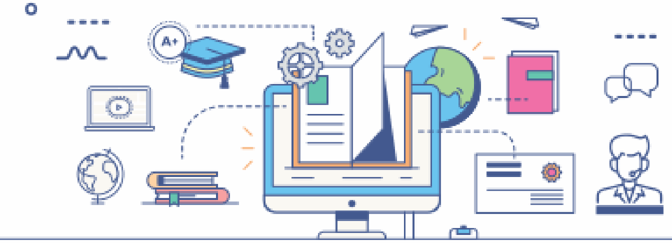
Mridul Mishra, Assistant Professor
Computer Science & Engineering Department

UNIT-7

CASE Tools and Advance Practices of System Dependability and Security

CASE Tools

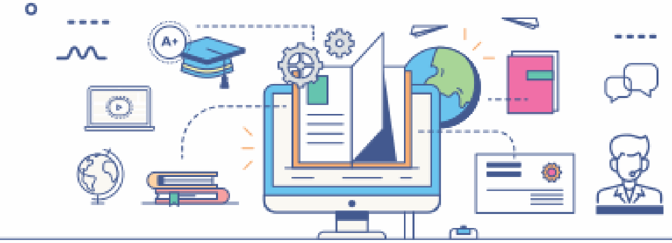
- To make the software system building faster , a new concept of designing software was introduced, known as Computer Aided Software Engineering (CASE)..
- It deals with development and maintenance of software projects by using various software tools.
- CASE includes software systems that are intended to provide Automated Support for software process activities. Automated Support is something that we use some software to develop another software



Why CASE Tools?

- CASE makes use of computer-assisted method to organize and control the development of software, **especially on large, complex projects that involve many software components and people**.

- Use of CASE allows **designers, code writers, testers, planners and managers to share a common understanding** of project progress at each stage of development.
- CASE tools are mainly used to decrease the development time & cost and improve software quality



- CASE tools are developed for the following purposes:
  - Quick Installation
  - Reduce Coding & Testing time.
  - Improve graphical techniques and data flow.
  - Enhanced analysis and design development.
  - Easily work with documentation
  - Increase rate of system development .

Components of CASE Tools

- There are two types of CASE tools:
  - **Classic CASE tools**: Interactive debuggers, compilers, project progress control systems
  - **Real Case tools**: support several phases of development
    1. Upper CASE tools support analysis and design.
    2. Lower CASE tools refer to the location in these phases in the Waterfall Model.
    3. Integrated CASE tools support analysis, design, and coding.

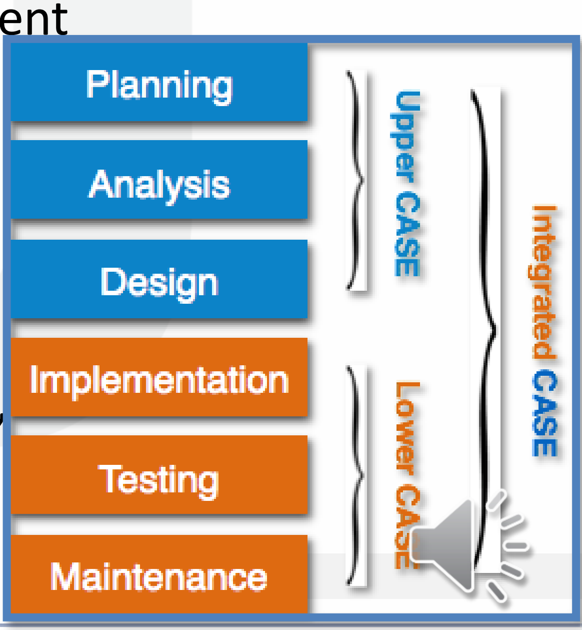


Image source : Google

Types of CASE Tools

- **Diagram tool :**These tools represent system components, data and control flow in a large set of software components and system structure in a graphical manner.
- **Process Modelling Tools :** Process modelling is used to create software process model, which is used to develop the software.
- **Project Management Tools :** These tools will be used for project planning, cost and effort estimation, project scheduling and resource planning.
- **Documentation Tools :** Documentation in a software project starts prior to the software process, goes throughout all phases of SDLC and after the completion of the project.

Types Of CASE Tools:

- Analysis Tools
- Design Tools
- Configuration Management Tools
- Change Control Tools
- Programming Tools
- Prototyping Tools
- Web Development Tools
- Quality Assurance Tools
- Maintenance Tools

Advantages

- Improve quality and productivity of software
- Produces software that meets user demands & requirements
- Produce system with good documentation
- Effective for large & complex systems
- Reduce error correction & maintenance time
- More Flexible systems

Disadvantages

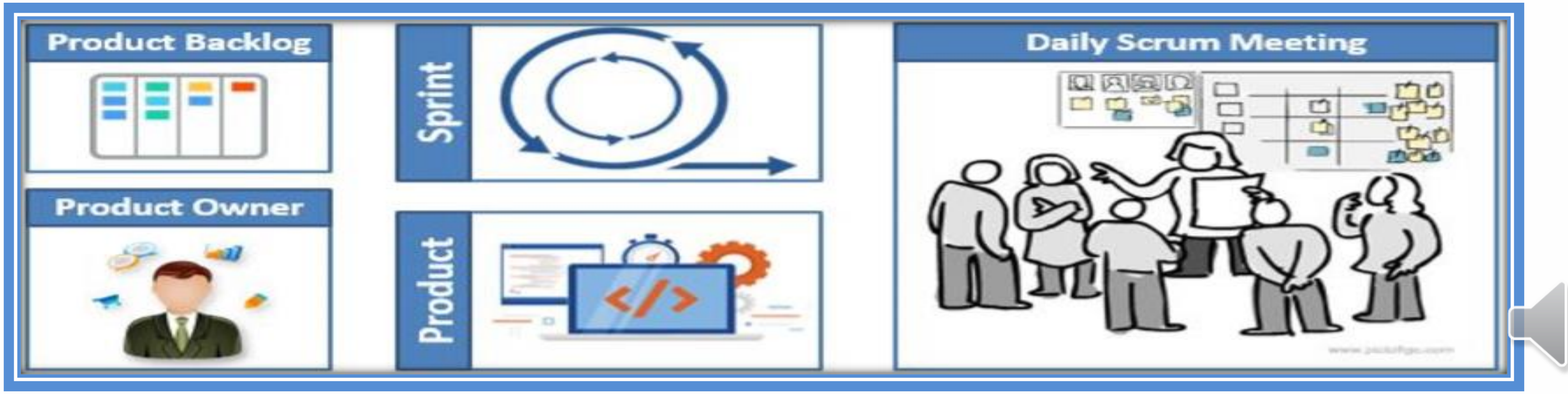
- CASE tools are complex to use
- Not easy to maintain
- Good quality CASE tools are very expensive
- Require training of maintenance staff
- It might become difficult to integrate with existing systems

What Is Scrum?

A **scrum** is a method of restarting play in rugby that involves players packing closely together with their heads down and attempting to gain possession of the ball

SCRUM Developments

- Scrum is a strategy for product development that organizes **software developers as a team** to achieve a common goal i.e. to create a product that is ready for market. It is a subset of agile, that is widely used.
- It is an agile process model which is used for developing the complex software systems.
- It is a lightweight process framework. (Lightweight refers to minimizing process overhead to maximize the productivity.)



Roles in Scrum Team

Product Owner

Voice & inputs of Customer

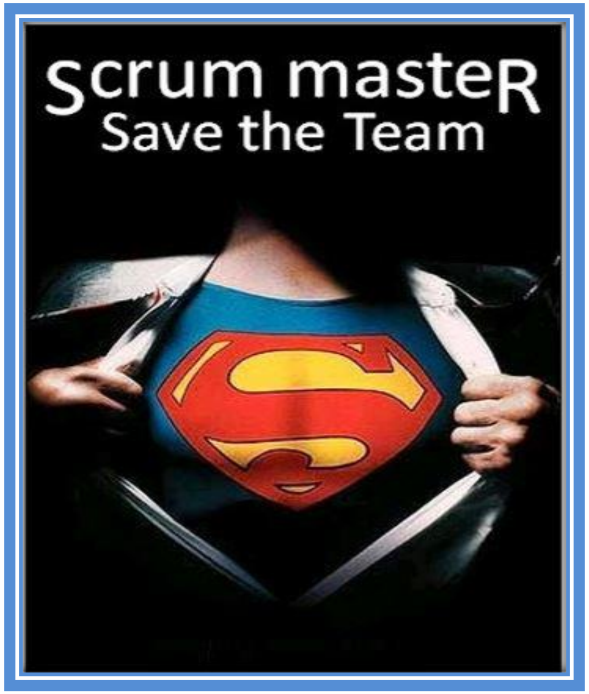
Voice of the Business

He takes the scenarios that have the most business value

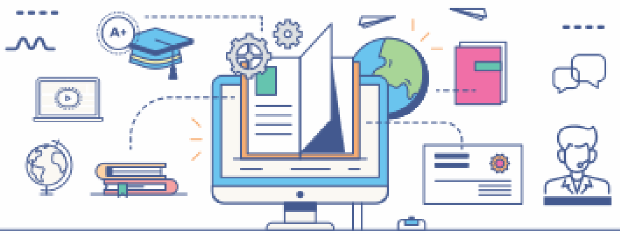
What Does He / She Do?

- Vision for product
- Represent the interests of the business
- Represents the customers
- Owns the product backlog
- Creates acceptance criteria for the backlog items
- Prioritizes tasks

Scrum Master



“Servant leader”



What Does He/She Do ?

- Removes obstacles to the ability of team to deliver sprint goals
- Ensures that scrum process is executed as per plans
- Facilitates changes
- He/She is not the team boss. They prioritize a high-performing, self-organized team.

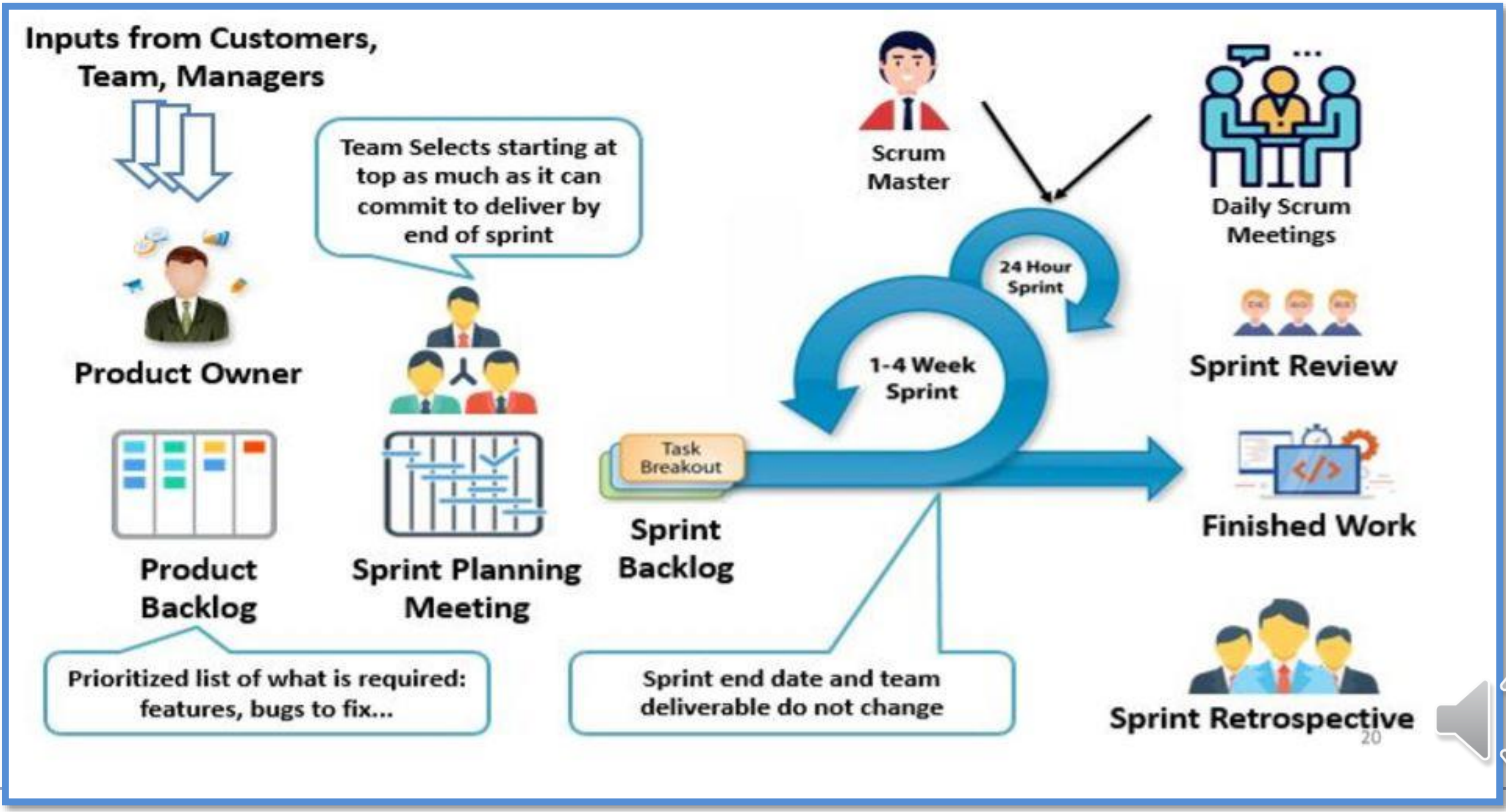
Development Team

One for one and one for all

What Does He/She Do ?

- Complete user stories to incrementally increase the value of the product
- Self-organizes to get all of the necessary work done
- Creates the estimates
- Make the “how to accomplish goal” decisions
- Avoid “not my job” thinking

Scrum Framework At A Glance


Inputs from Customers, Team, Managers
Team Selects starting at top as much as it can commit to deliver by end of sprint
Scrum Master
Daily Scrum Meetings
24 Hour Sprint
1-4 Week Sprint
Product Owner
Sprint Review
Task Breakout
Finished Work
Sprint Backlog
Product Backlog
Sprint Planning Meeting
Prioritized list of what is required: features, bugs to fix...
Sprint end date and team deliverable do not change
Sprint Retrospective

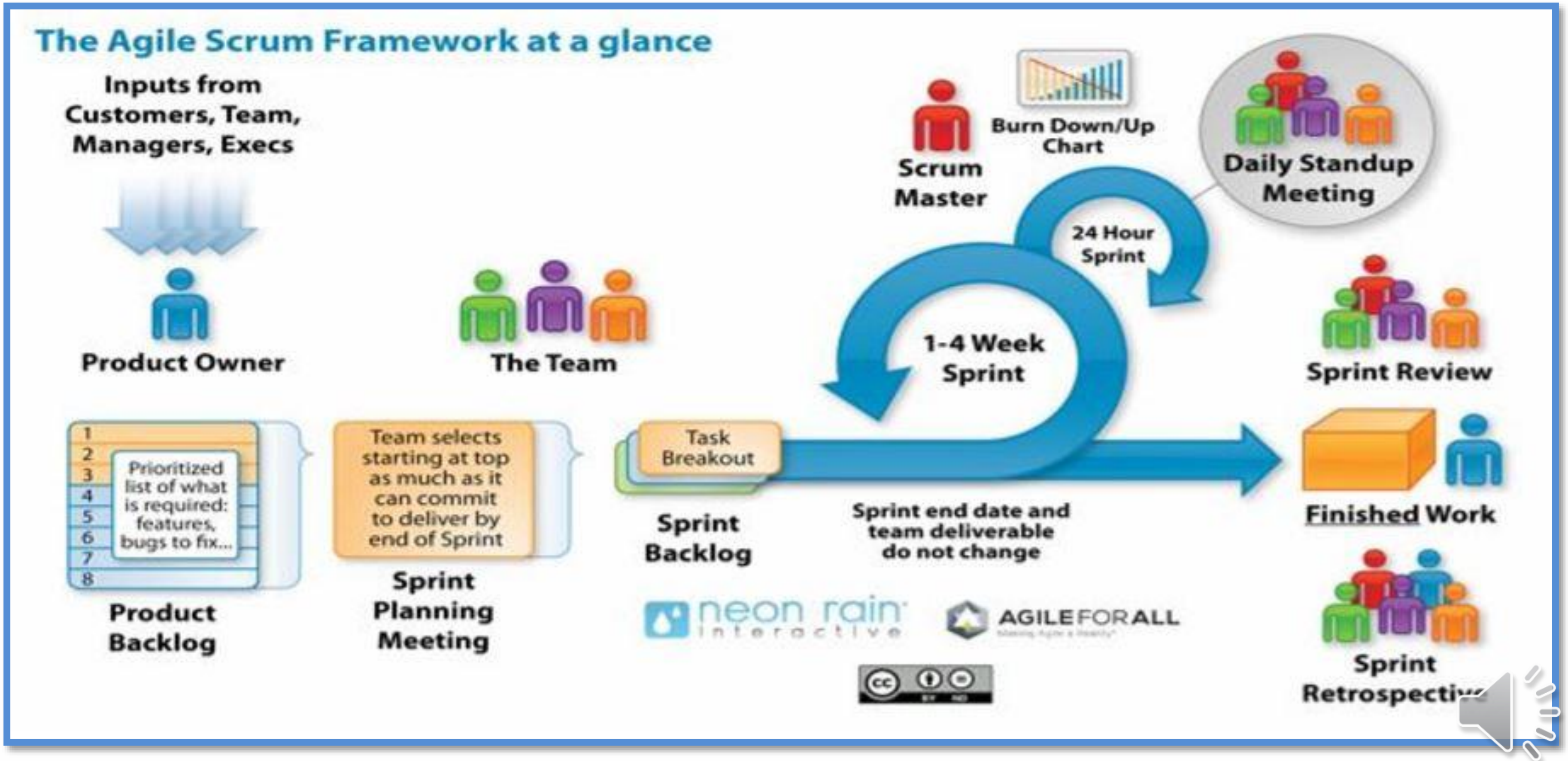
Sprint

- fixed length, normally 1–4 weeks
- The starting point for sprint planning is the product backlog
- During the assessment phase of the sprint, this is reviewed after which the priorities and risks are assigned
- selection phase involves all of the project team who work with the customer to finalize the features and functionality to be developed during the sprint.
- Short daily meetings
- Scrum master protects the development team from external distractions

Stand-up Meetings

- share information
- describe their progress
- problems that have arisen
- what is planned for the following day.
- everyone on the team is well informed of what is going on
- short-term planning

Continued..



The Agile Scrum Framework at a glance

Dependable Systems

- A **dependable system** is one that can be trusted by users. It requires the system to be **highly available** (to authorized users) and at the same time ensure a **high degree of service integrity**.
- A system is dependable when it can produce the results for which it was designed, and creates no adverse effects.

System Dependability

- For many computer-based systems, the most important system property is the dependability of the system.
- The dependability of a system reflects the extent of user’s trust in that system. It reflects how much user trusts that the system will operate as users expect and that it will not ‘fail’ in normal use.
- Dependability covers the related systems properties of reliability, availability and security. These are all inter dependent.

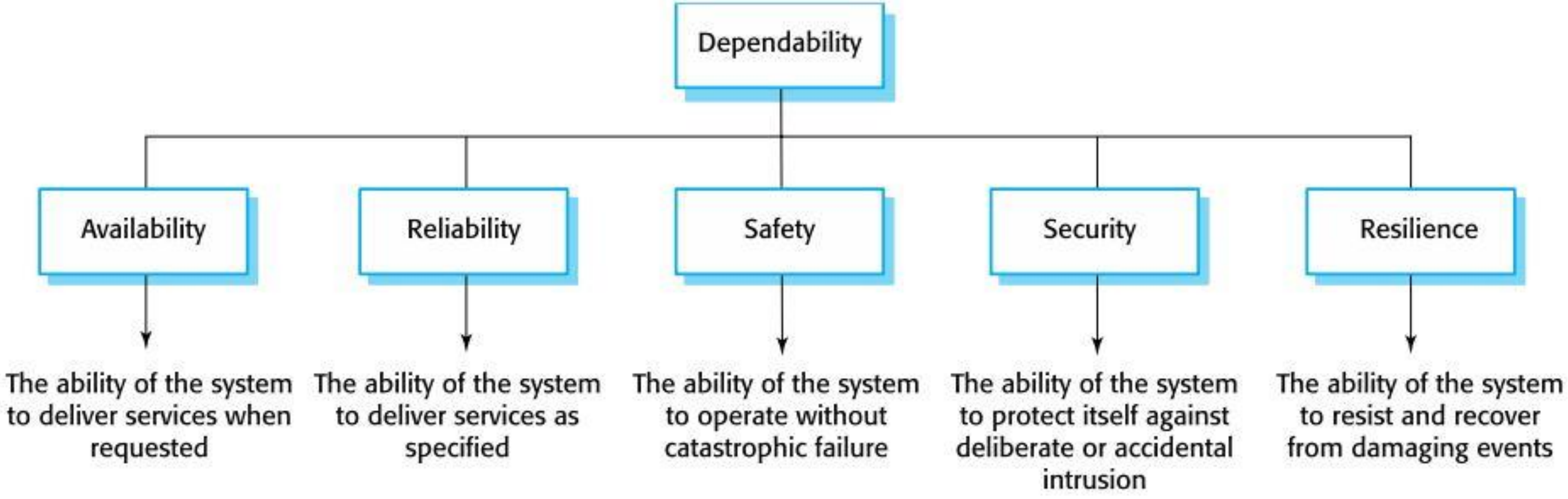
Importance of dependability

- System failures may have huge effects with huge number of users affected by the failure.
- Systems that are not dependable and are unreliable, unsafe or insecure are prone to user rejection.
- The system failure may be very costly if the failure leads to monetory or physical loss.
- Undependable systems may lead to loss of information causing a high consequent recovery cost.

Causes of failure

- Hardware failure
  - Hardware fails because of errors in design and manufacturing or due to components wear out.
- Software failure
  - Software can fail because of specification, implementation or design errors.
- Operational failure
  - Human operators make mistakes. Now perhaps the largest single cause of system failures in socio-technical systems.

The principal dependability properties



The ability of the system to deliver services when requested

The ability of the system to deliver services as specified

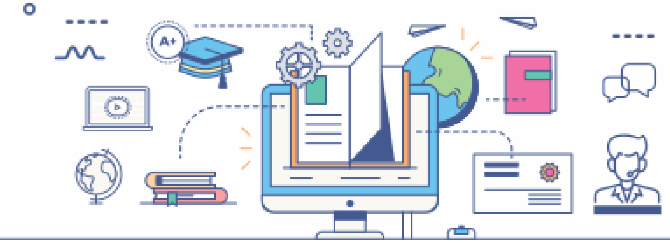
The ability of the system to operate without catastrophic failure

The ability of the system to protect itself against deliberate or accidental intrusion

The ability of the system to resist and recover from damaging events

Reliability Engineering

- In general, software customers expect all software to be dependable. However, for non-critical applications, they may be willing to accept some system failures.
- Some applications (critical systems) have very high reliability requirements and special software engineering techniques may be used to achieve this.
  - Medical systems
  - Telecommunications and power systems
  - Aerospace systems



- Reliability can be defined formally only with respect to specification of system i.e. a failure is a deviation of results or operations from a specification.
- But, large number of specifications are incomplete or incorrect – so, a system that adheres to its specification may still ‘fail’ from the users point of view.
- Additionally, users don’t read specifications, hence are unaware about ideal system behaviour.
- Therefore perceived reliability is more important in practice.
- The formal definition of reliability does not always reflect the user’s perception of a system’s reliability The assumptions that are made about the environment where a system will be used may be incorrect
- Usage of a system in an office environment is likely to be quite different from usage of the same system in a university environment

- The consequences of system failures affects the perception of reliability
- Unreliable windscreen wipers in a car may be irrelevant in a dry climate
- Failures that have serious consequences (like an engine failure in a car) are given greater weight by users than failures that are inconvenient

Fault, Errors & Failures

Term	Description
Human error or mistake	Human behaviour that results in the introduction of faults into a system. For example, in the wilderness weather system, a programmer might decide that the way to compute the time for the next transmission is to add 1 hour to the current time. This works except when the transmission time is between 23.00 and midnight (midnight is 00.00 in the 24-hour clock).
System fault	A characteristic of a software system that can lead to a system error. The fault is the inclusion of the code to add 1 hour to the time of the last transmission, without a check if the time is greater than or equal to 23.00.
System error	An erroneous system state that can lead to system behavior that is unexpected by system users. The value of transmission time is set incorrectly (to 24.XX rather than 00.XX) when the faulty code is executed.
System failure	An event that occurs at some point in time when the system does not deliver a service as expected by its users. No weather data is transmitted because the time is invalid.

Reliability achievement

- Fault avoidance
  - Development technique are used that will either nullify the probability of mistakes or catch them before they lead to system faults.
- Fault detection and removal
  - Verification and validation techniques are used that will improve the chances of detecting and correcting errors before the system goes into service.
- Fault tolerance
  - Run-time techniques are so that system faults do not lead to system errors and/or that system errors don’t cause system failure.

Safety Engineering

- Safety is a characterstic of a 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.
- It is important to consider software safety as most devices whose failure is critical now incorporate software-based control systems.

Software in safety-critical systems

- Some systems may be software-controlled so the decisions taken by the software and subsequent actions are safety-critical.
- Therefore, the software behaviour is directly related to the overall safety of the system.
- Software is extensively used for checking and monitoring other safety-critical components in a system.
  - Eg.,all aircraft engine components are monitored by software looking for early indications of component failure. This software is safety-critical because, if it fails, other components may fail and cause an accident.

Safety and reliability

- Safety and reliability are distinct yet related
  - In general, reliability and availability are necessary but not sufficient to ensure system safety.
- Reliability is concerned with conformance to a given specification and delivery of service
- Safety is concerned with ensuring system cannot cause damage irrespective of whether or not it conforms to its specification.
  - System reliability is essential for safety but is not enough
  - Reliable systems can be unsafe

Unsafe reliable systems

- There might be hidden faults in a system that are not detected for long time and rarely cause trouble.
- Specification errors
  - If specification of a system is wrong, system might still behave in specified manner but lead to an an.
- Hardware failures generating spurious inputs
  - Hard to anticipate in the specification.
- Context-sensitive commands i.e. issuing the right command at the wrong time
  - Often the result of operator error.

Safety criticality

- Primary safety-critical systems
  - The failure of Embedded software can cause the related hardware to fail and directly affect people. Example is the insulin pump control system.
- Secondary safety-critical systems
  - Systems whose failure results in faults in other (socio-technical) systems, which can then have safety consequences.
    - For example, a hospital system is safety-critical as failure may lead to inappropriate treatment being prescribed.
    - Infrastructure control systems are also secondary safety-critical systems.

Hazards

- Events or circumstances that can cause an accident
  - Faulty valve in reactor control system
  - Wrong computation by software in navigation system
  - Failure to detect possible allergy in medication prescribing system
- Hazards do not inevitably result in accidents
  - accident prevention actions can be taken.

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. An overdose of insulin 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 measures blood glucose is an example of a hazard.
Damage	A measure of the loss resulting from a mishap. Damage can range from many people being killed as a result of an accident to minor injury or property damage. Damage resulting from an overdose of insulin could be serious injury or the death of the user of the insulin pump.
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. When an individual death is a possibility, a reasonable assessment of hazard severity is 'very high'.
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 in which the hazard could occur). The probability of a sensor failure in the insulin pump that results in an overdose is probably low.
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 the hazard will lead to an accident. The risk of an insulin overdose is probably medium to low.

Safety Specification

- The aim of safety engineering is to find protection requirements that can assure that system failures do not cause damage to any human or environment.
- Safety requirements can very well be ‘shall not’ requirements i.e. they explain situations and events which must never occur.
- Functional safety requirements define:
  - Checking and recovery features which should be part of a system
  - Features that provide security from external attacks &system failures

Safety achievement

- Hazard avoidance
  - The system is built in such a way 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 Engineering

- Security Engineering is the use of methods, techniques and tools that support the maintenance and development of systems that can withstand malicious attacks that are aimed at damaging a computer-based system and/or data in system.
- It is a sub-field of computer security.

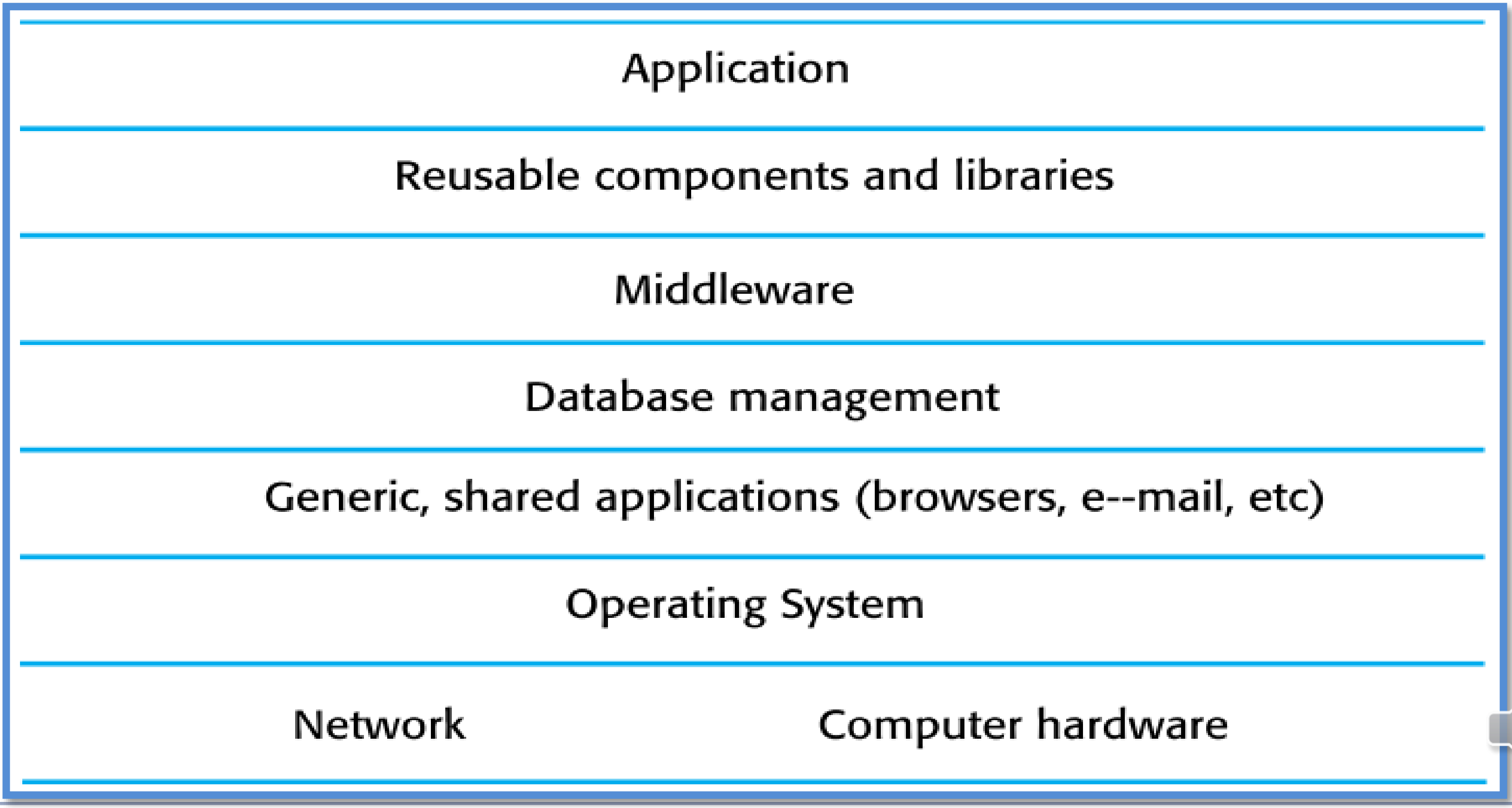
Security dimensions

- Confidentiality
  - Information in the system is disclosed only to the intended audience.
- Integrity
  - The system should ensure that there is no corruption or malicious modification in data.
- Availability
  - Access to a system or its data that is normally available may not be possible.

Security levels

- Infrastructure security
- Application security
- Operational security

System layers where security may be compromised



Application

Reusable components and libraries

Middleware

Database management

Generic, shared applications (browsers, e--mail, etc)

Operating System

Network

Computer hardware

Security terminology

Term	Definition
Asset	Something of value which has to be protected. The asset may be the software system itself or data used by that system.
Attack	An exploitation of a system’s vulnerability. Generally, this is from outside the system and is a deliberate attempt to cause some damage.
Control	A protective measure that reduces a system’s vulnerability. Encryption is an example of a control that reduces a vulnerability of a weak access control system
Exposure	Possible loss or harm to 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.
Threat	Circumstances that have potential to cause loss or harm. You can think of these as a system vulnerability that is subjected to an attack.
Vulnerability	A weakness in a computer-based system that may be exploited to cause loss or harm.

Threat Types

- Interception threats that allow an attacker to gain access to an asset.
    - A possible threat to the Medical care system might be a situation where an attacker gains access to the records of an individual patient.
- Interruption threats that allow an attacker to make part of the system unavailable.
    - A possible threat might be a denial of service attack on a system database server so that database connections become impossible.

Continued..

- Modification threats that allow an attacker to tamper with a system asset.
  - In the hospital management system, a modification threat would be where an attacker alters or destroys a patient record.
- Fabrication threats that allow an attacker to insert false information into a system.
  - In a bank system, false transactions might be added to the system that transfer money to the perpetrator’s bank account.

Security and dependability

- **Security and reliability**
  - If a system is attacked and the system or its data are corrupted as a consequence of that attack, then this may induce system failures that compromise the reliability of the system.
- **Security and availability**
  - A common attack on a web-based system is a denial of service attack, where a web server is flooded with service requests from a range of different sources. The aim of this attack is to make the system unavailable.

Continued…

- **Security and safety**
  - An attack that corrupts the system or its data means that assumptions about safety may not hold. Safety checks rely on analysing the source code of safety critical software and assume the executing code is a completely accurate translation of that source code. If this is not the case, safety-related failures may be induced and the safety case made for the software is invalid.
- **Security and resilience**
  - Resilience is a system characteristic that reflects its ability to resist and recover from damaging events. The most probable damaging event on networked software systems is a cyberattack of some kind, so most of the work now done in resilience is aimed at preventing cyber attacks

Resilience Engineering

- The resilience of a system is a judgment of how well that system can maintain the continuity of its critical services in the presence of disruptive events, such as equipment failure and cyberattacks.
- Cyber attacks by malicious outsiders are perhaps the most serious threat faced by networked systems but resilience is also intended to cope with system failures and other disruptive events.

Essential resilience ideas

- some of the services offered by a system are critical services whose failure could have serious human, social or economic effects.
- some events are disruptive and can affect the ability of a system to deliver its critical services.
- The idea that resilience is a judgment – there are no resilience metrics and resilience cannot be measured. The resilience of a system can only be assessed by experts, who can examine the system and its operational processes.

Resilience engineering assumptions

- **Recognition :** The system or its operators should recognise early indications of system failure.
- **Resistance :** If the symptoms of a problem or cyberattack are detected early, then resistance strategies may be used to reduce the probability that the system will fail.
- **Recovery :** If a failure occurs, the recovery activity ensures that critical system services are restored quickly so that system users are not badly affected by failure.
- **Reinstatement :** In this final activity, all of the system services are restored and normal system operation can continue.

References

- Pressman, Roger S. *Software engineering: a practitioner's approach*. Palgrave Macmillan, 2005.
- Sommerville, Ian. *Software Engineering: Pearson New International Edition*. Pearson Education Limited, 2013.
- Jalote, Pankaj. *An integrated approach to software engineering*. Springer Science & Business Media, 2012.
- www.google.com
- www.wikipedia.com
- www.pixabay.com

DIGITAL LEARNING CONTENT

Parul® University

www.paruluniversity.ac.in