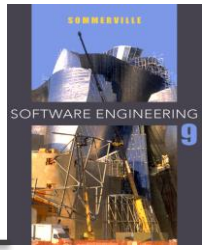
CSE 319
Software Engineering

Lecture : Dependability and Security Specification

Anisur Rahman

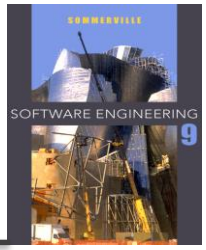
Military Institute of Science and Technology

Topics covered



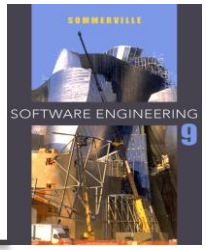
- ✧ Risk-driven specification
- ✧ Safety specification
- ✧ Security specification
- ✧ Software reliability specification

Dependability requirements



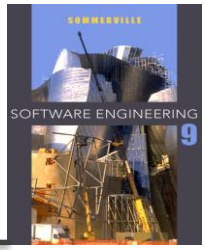
- Functional requirements to define error checking and recovery facilities and protection against system failures.
- Non-functional requirements defining the required reliability and availability of the system.
- Excluding requirements that define states and conditions that must not arise.

Risk-driven specification



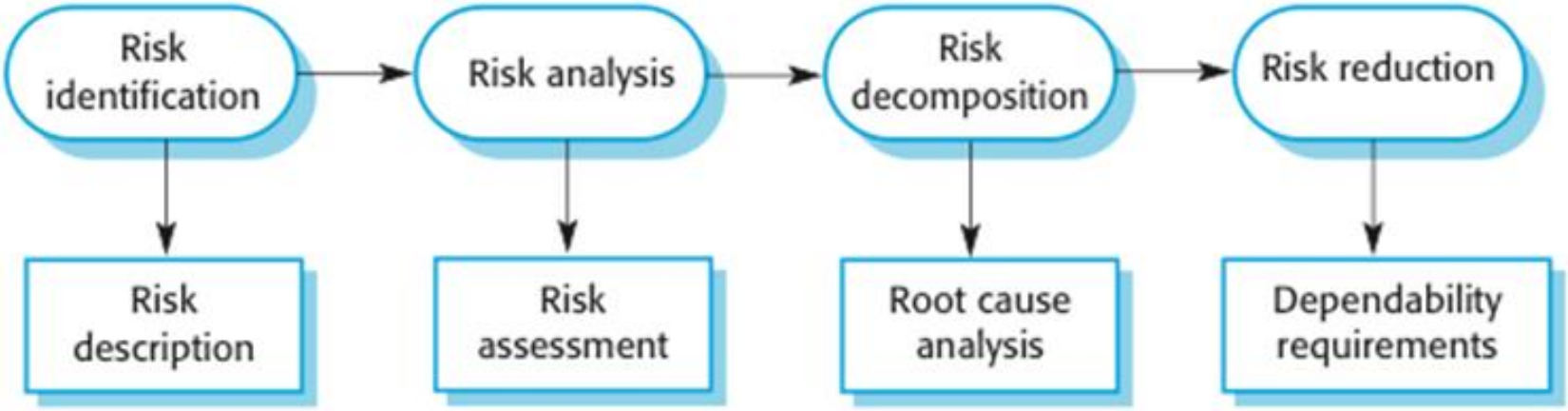
- Critical systems specification should be risk-driven.
- This approach has been widely used in safety and security-critical systems.
- The aim of the specification process should be to understand the risks (safety, security, etc.) faced by the system and to define requirements that reduce these risks.

Stages of risk-based analysis

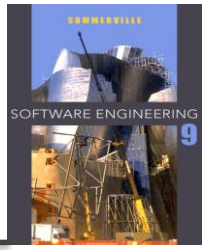


- Risk identification
  - Identify potential risks that may arise.
- Risk analysis and classification
  - Assess the seriousness of each risk.
- Risk decomposition
  - Decompose risks to discover their potential root causes.
- Risk reduction assessment
  - Define how each risk must be taken into eliminated or reduced when the system is designed.

Risk-driven specification

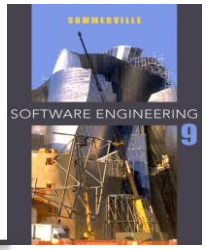


Phased risk analysis



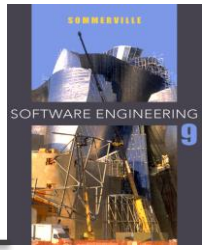
- Preliminary risk analysis
  - Identifies risks from the systems environment. Aim is to develop an initial set of system security and dependability requirements.
- Life cycle risk analysis
  - Identifies risks that emerge during design and development e.g. risks that are associated with the technologies used for system construction. Requirements are extended to protect against these risks.
- Operational risk analysis
  - Risks associated with the system user interface and operator errors. Further protection requirements may be added to cope with these.

Safety specification



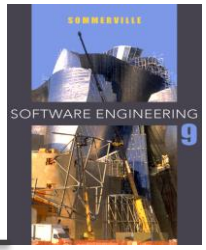
- Goal is to identify protection requirements that ensure that system failures do not cause injury or death or environmental damage.
- Risk identification = Hazard identification
- Risk analysis = Hazard assessment
- Risk decomposition = Hazard analysis
- Risk reduction = safety requirements specification

Hazard identification



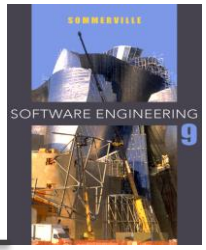
- Identify the hazards that may threaten the system.
- Hazard identification may be based on different types of hazard:
  - Physical hazards
  - Electrical hazards
  - Biological hazards
  - Service failure hazards
  - Etc.

Insulin pump risks



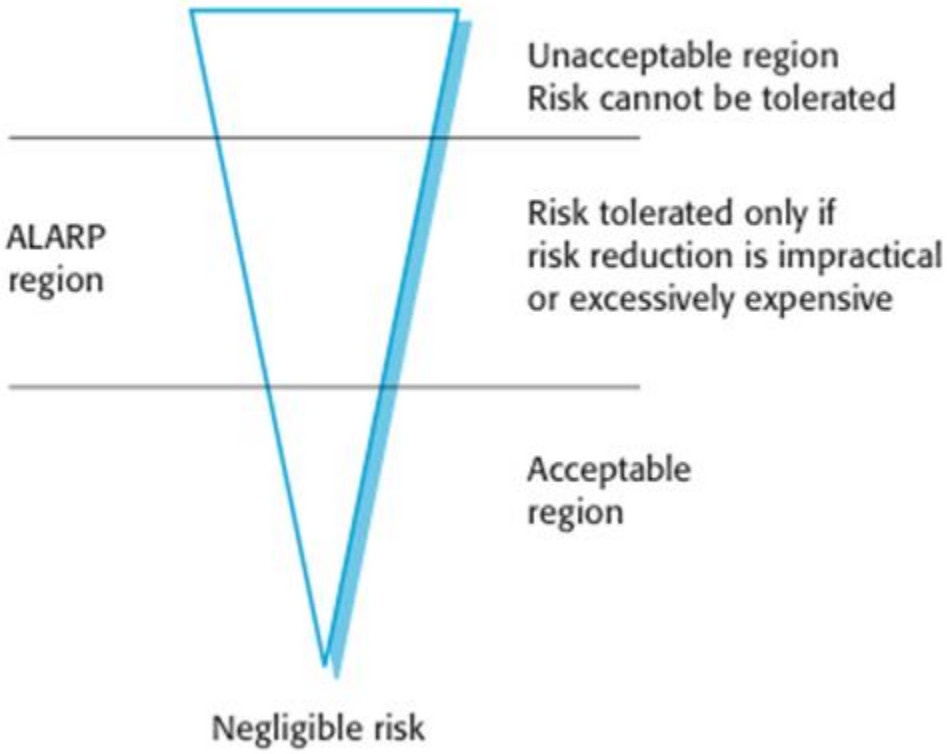
- Insulin overdose (service failure).
- Insulin underdose (service failure).
- Power failure due to exhausted battery (electrical).
- Electrical interference with other medical equipment (electrical).
- Poor sensor and actuator contact (physical).
- Parts of machine break off in body (physical).
- Infection caused by introduction of machine (biological).
- Allergic reaction to materials or insulin (biological).

Hazard assessment

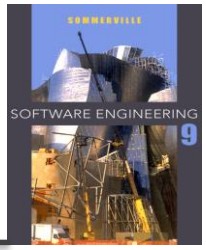


- The process is concerned with understanding the likelihood that a risk will arise and the potential consequences if an accident or incident should occur.
- Risks may be categorized as:
  - Intolerable. Must never arise or result in an accident
  - As low as reasonably practical(ALARP). Must minimise the possibility of risk given cost and schedule constraints
  - Acceptable. The consequences of the risk are acceptable and no extra costs should be incurred to reduce hazard probability

The risk triangle

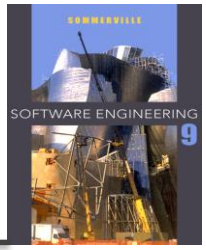


Social acceptability of risk



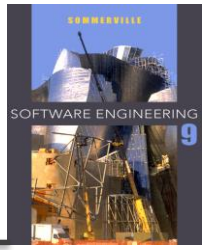
- The acceptability of a risk is determined by human, social and political considerations.
- In most societies, the boundaries between the regions are pushed upwards with time i.e. society is less willing to accept risk
  - For example, the costs of cleaning up pollution may be less than the costs of preventing it but this may not be socially acceptable.
- Risk assessment is subjective
  - Risks are identified as probable, unlikely, etc. This depends on who is making the assessment.

Hazard assessment



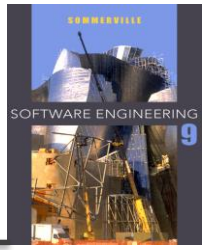
- Estimate the risk probability and the risk severity.
- It is not normally possible to do this precisely so relative values are used such as ‘unlikely’, ‘rare’, ‘very high’, etc.
- The aim must be to exclude risks that are likely to arise or that have high severity.

Risk classification for the insulin pump



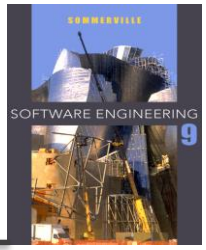
Identified hazard	Hazard probability	Accident severity	Estimated risk	Acceptability
1.Insulin overdose computation	Medium	High	High	Intolerable
2. Insulin underdose computation	Medium	Low	Low	Acceptable
3. Failure of hardware monitoring system	Medium	Medium	Low	ALARP
4. Power failure	High	Low	Low	Acceptable
5. Machine incorrectly fitted	High	High	High	Intolerable
6. Machine breaks in patient	Low	High	Medium	ALARP
7. Machine causes infection	Medium	Medium	Medium	ALARP
8. Electrical interference	Low	High	Medium	ALARP
9. Allergic reaction	Low	Low	Low	Acceptable

Hazard analysis



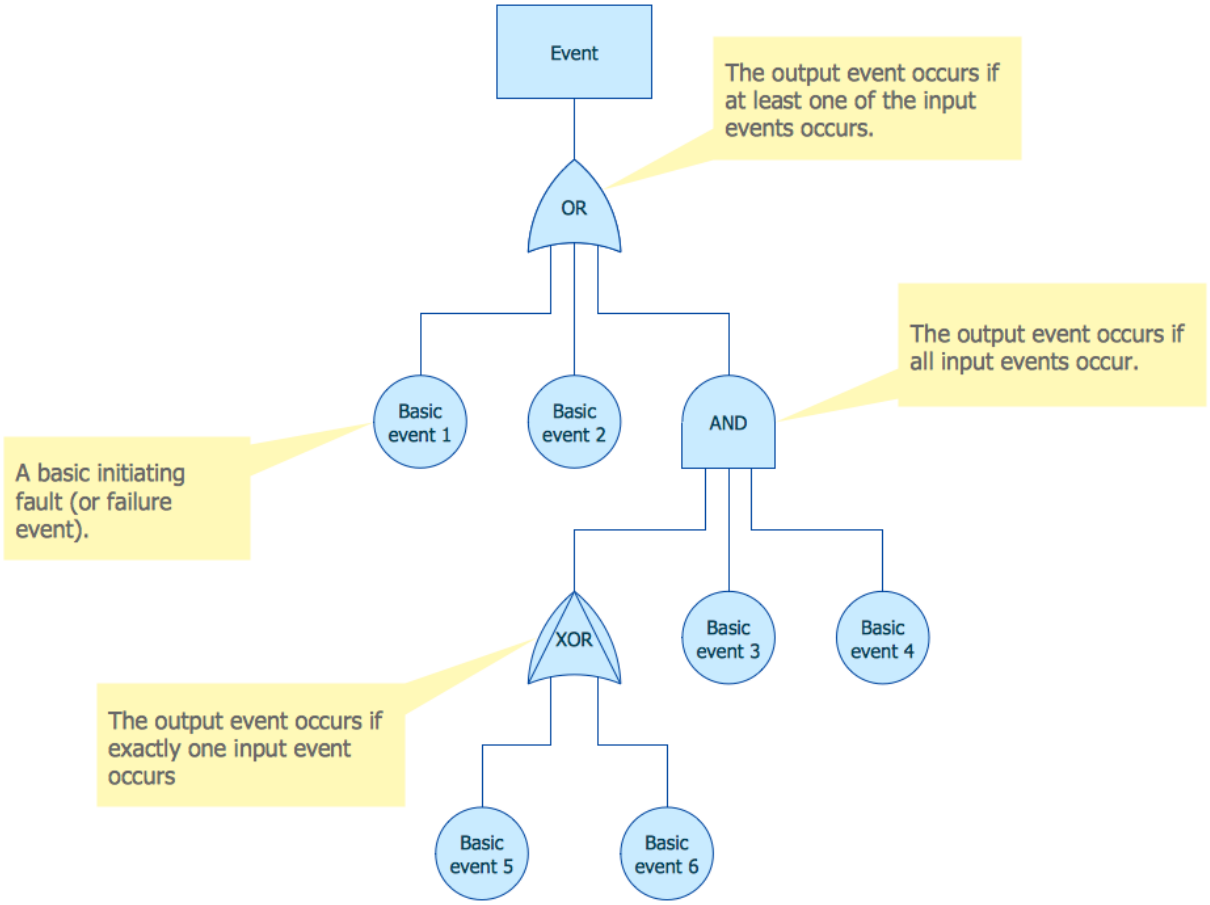
- Concerned with discovering the root causes of risks in a particular system.
- Techniques have been mostly derived from safety-critical systems and can be
  - Inductive, bottom-up techniques. Start with a proposed system failure and assess the hazards that could arise from that failure;
  - Deductive, top-down techniques. Start with a hazard and deduce what the causes of this could be.

Fault-tree analysis

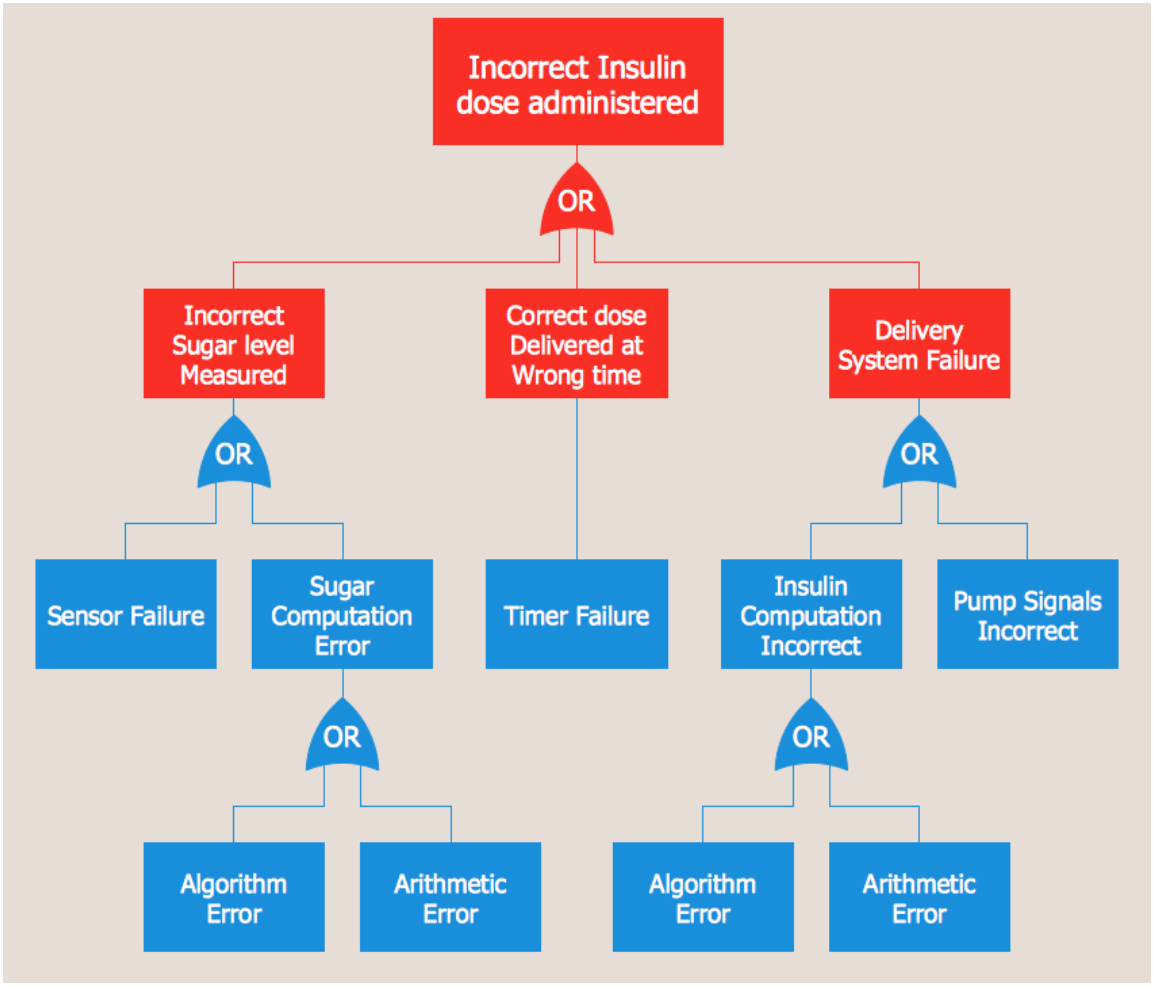


- A deductive top-down technique.
- Put the risk or hazard at the root of the tree and identify the system states that could lead to that hazard.
- Where appropriate, link these with ‘and’ or ‘or’ conditions.
- A goal should be to minimise the number of single causes of system failure.

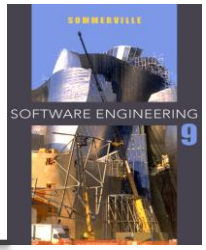
Symbols - software fault tree



An example of a software fault tree

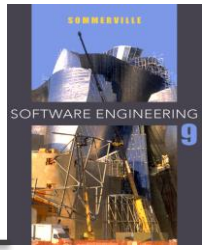


Fault tree analysis



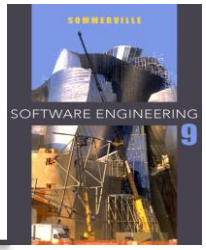
- Three possible conditions that can lead to delivery of incorrect dose of insulin
  - Incorrect measurement of blood sugar level
  - Failure of delivery system
  - Dose delivered at wrong time
- By analysis of the fault tree, root causes of these hazards related to software are:
  - Algorithm error
  - Arithmetic error

Risk reduction



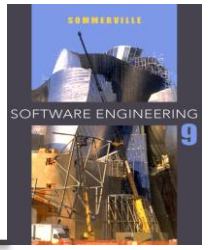
- The aim of this process is to identify dependability requirements that specify how the risks should be managed and ensure that accidents/incidents do not arise.
- Risk reduction strategies
  - Risk avoidance;
  - Risk detection and removal;
  - Damage limitation.

Strategy use



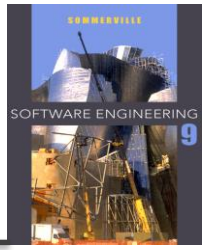
- Normally, in critical systems, a mix of risk reduction strategies are used.
- In a chemical plant control system, the system will include sensors to detect and correct excess pressure in the reactor.
- However, it will also include an independent protection system that opens a relief valve if dangerously high pressure is detected.

Insulin pump - software risks



- Arithmetic error
  - A computation causes the value of a variable to overflow or underflow;
  - Maybe include an exception handler for each type of arithmetic error.
- Algorithmic error
  - Compare dose to be delivered with previous dose or safe maximum doses. Reduce dose if too high.

Examples of safety requirements



SR1: The system shall not deliver a single dose of insulin that is greater than a specified maximum dose for a system user.

SR2: The system shall not deliver a daily cumulative dose of insulin that is greater than a specified maximum daily dose for a system user.

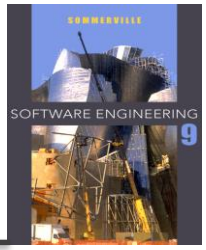
SR3: The system shall include a hardware diagnostic facility that shall be executed at least four times per hour.

SR4: The system shall include an exception handler for all of the exceptions that are identified in Table 3.

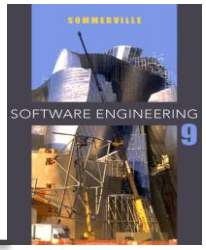
SR5: The audible alarm shall be sounded when any hardware or software anomaly is discovered and a diagnostic message, as defined in Table 4, shall be displayed.

SR6: In the event of an alarm, insulin delivery shall be suspended until the user has reset the system and cleared the alarm.

Key points



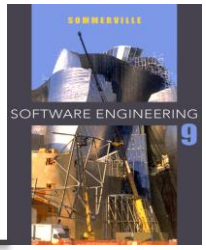
- Risk analysis is an important activity in the specification of security and dependability requirements. It involves identifying risks that can result in accidents or incidents.
- A hazard-driven approach may be used to understand the safety requirements for a system. You identify potential hazards and decompose these (using methods such as fault tree analysis) to discover their root causes.
- Safety requirements should be included to ensure that hazards and accidents do not arise or, if this is impossible, to limit the damage caused by system failure.



Chapter 12 – Dependability and Security Specification

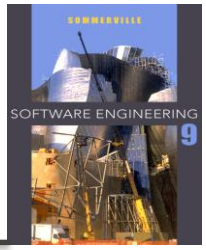
Lecture 2

System reliability specification



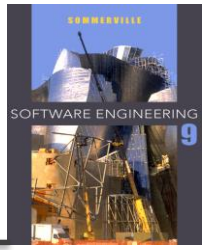
- Reliability is a measurable system attribute so non-functional reliability requirements may be specified quantitatively. These define the number of failures that are acceptable during normal use of the system or the time in which the system must be available.
- Functional reliability requirements define system and software functions that avoid, detect or tolerate faults in the software and so ensure that these faults do not lead to system failure.
- Software reliability requirements may also be included to cope with hardware failure or operator error.

Reliability specification process



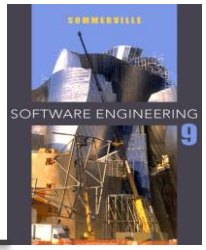
- Risk identification
  - Identify the types of system failure that may lead to economic losses.
- Risk analysis
  - Estimate the costs and consequences of the different types of software failure.
- Risk decomposition
  - Identify the root causes of system failure.
- Risk reduction
  - Generate reliability specifications, including quantitative requirements defining the acceptable levels of failure.

Types of system failure



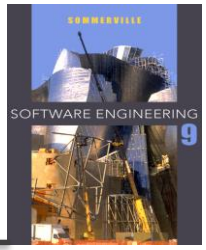
Failure type	Description
Loss of service	The system is unavailable and cannot deliver its services to users. You may separate this into loss of critical services and loss of non-critical services, where the consequences of a failure in non-critical services are less than the consequences of critical service failure.
Incorrect service delivery	The system does not deliver a service correctly to users. Again, this may be specified in terms of minor and major errors or errors in the delivery of critical and non-critical services.
System/data corruption	The failure of the system causes damage to the system itself or its data. This will usually but not necessarily be in conjunction with other types of failures.

Reliability metrics



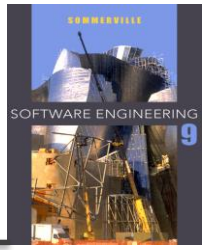
- Reliability metrics are units of measurement of system reliability.
- System reliability is measured by counting the number of operational failures and, where appropriate, relating these to the demands made on the system and the time that the system has been operational.
- A long-term measurement programme is required to assess the reliability of critical systems.
- Metrics
  - Probability of failure on demand
  - Rate of occurrence of failures/Mean time to failure
  - Availability

Probability of failure on demand (POFOD)



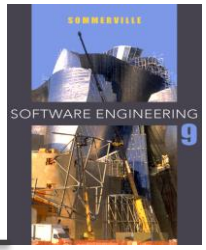
- This is the probability that the system will fail when a service request is made. Useful when demands for service are intermittent and relatively infrequent.
- Appropriate for protection systems where services are demanded occasionally and where there are serious consequence if the service is not delivered.
- Relevant for many safety-critical systems with exception management components
  - Emergency shutdown system in a chemical plant.

Rate of fault occurrence (ROCOF)



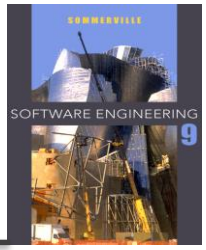
- Reflects the rate of occurrence of failure in the system.
- ROCOF of 0.002 means 2 failures are likely in each 1000 operational time units e.g. 2 failures per 1000 hours of operation.
- Relevant for systems where the system has to process a large number of similar requests in a short time
  - Credit card processing system, airline booking system.
- Reciprocal of ROCOF is Mean time to Failure (MTTF)
  - Relevant for systems with long transactions i.e. where system processing takes a long time (e.g. CAD systems). MTTF should be longer than expected transaction length.

Availability



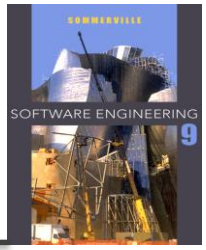
- Measure of the fraction of the time that the system is available for use.
- Takes repair and restart time into account
- Availability of 0.998 means software is available for 998 out of 1000 time units.
- Relevant for non-stop, continuously running systems
  - telephone switching systems, railway signalling systems.

Availability specification



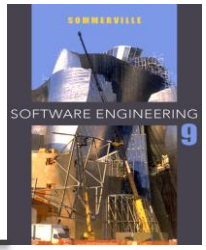
Availability	Explanation
0.9	The system is available for 90% of the time. This means that, in a 24-hour period (1,440 minutes), the system will be unavailable for 144 minutes.
0.99	In a 24-hour period, the system is unavailable for 14.4 minutes.
0.999	The system is unavailable for 84 seconds in a 24-hour period.
0.9999	The system is unavailable for 8.4 seconds in a 24-hour period. Roughly, one minute per week.

Failure consequences



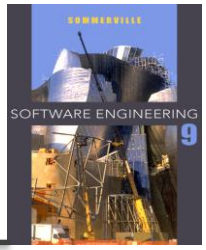
- When specifying reliability, it is not just the number of system failures that matter but the consequences of these failures.
- Failures that have serious consequences are clearly more damaging than those where repair and recovery is straightforward.
- In some cases, therefore, different reliability specifications for different types of failure may be defined.

Over-specification of reliability



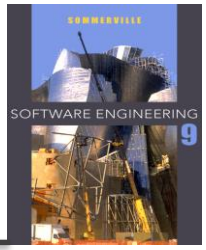
- Over-specification of reliability is a situation where a high-level of reliability is specified but it is not cost-effective to achieve this.
- In many cases, it is cheaper to accept and deal with failures rather than avoid them occurring.
- To avoid over-specification
  - Specify reliability requirements for different types of failure. Minor failures may be acceptable.
  - Specify requirements for different services separately. Critical services should have the highest reliability requirements.
  - Decide whether or not high reliability is really required or if dependability goals can be achieved in some other way.

Steps to a reliability specification



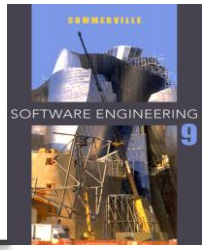
- For each sub-system, analyse the consequences of possible system failures.
- From the system failure analysis, partition failures into appropriate classes.
- For each failure class identified, set out the reliability using an appropriate metric. Different metrics may be used for different reliability requirements.
- Identify functional reliability requirements to reduce the chances of critical failures.

Insulin pump specification



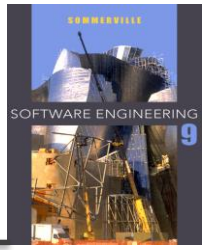
- Probability of failure (POFOD) is the most appropriate metric.
- Transient failures that can be repaired by user actions such as recalibration of the machine. A relatively low value of POFOD is acceptable (say 0.002) – one failure may occur in every 500 demands.
- Permanent failures require the software to be re-installed by the manufacturer. This should occur no more than once per year. POFOD for this situation should be less than 0.00002.

Functional reliability requirements



- Checking requirements that identify checks to ensure that incorrect data is detected before it leads to a failure.
- Recovery requirements that are geared to help the system recover after a failure has occurred.
- Redundancy requirements that specify redundant features of the system to be included.
- Process requirements for reliability which specify the development process to be used may also be included.

Examples of functional reliability requirements for MHC-PMS



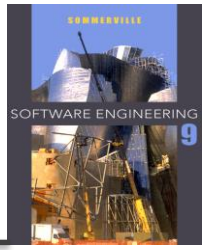
RR1: A pre-defined range for all operator inputs shall be defined and the system shall check that all operator inputs fall within this pre-defined range. (Checking)

RR2: Copies of the patient database shall be maintained on two separate servers that are not housed in the same building. (Recovery, redundancy)

RR3: N-version programming shall be used to implement the braking control system. (Redundancy)

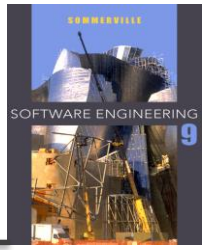
RR4: The system must be implemented in a safe subset of Ada and checked using static analysis. (Process)

Security specification



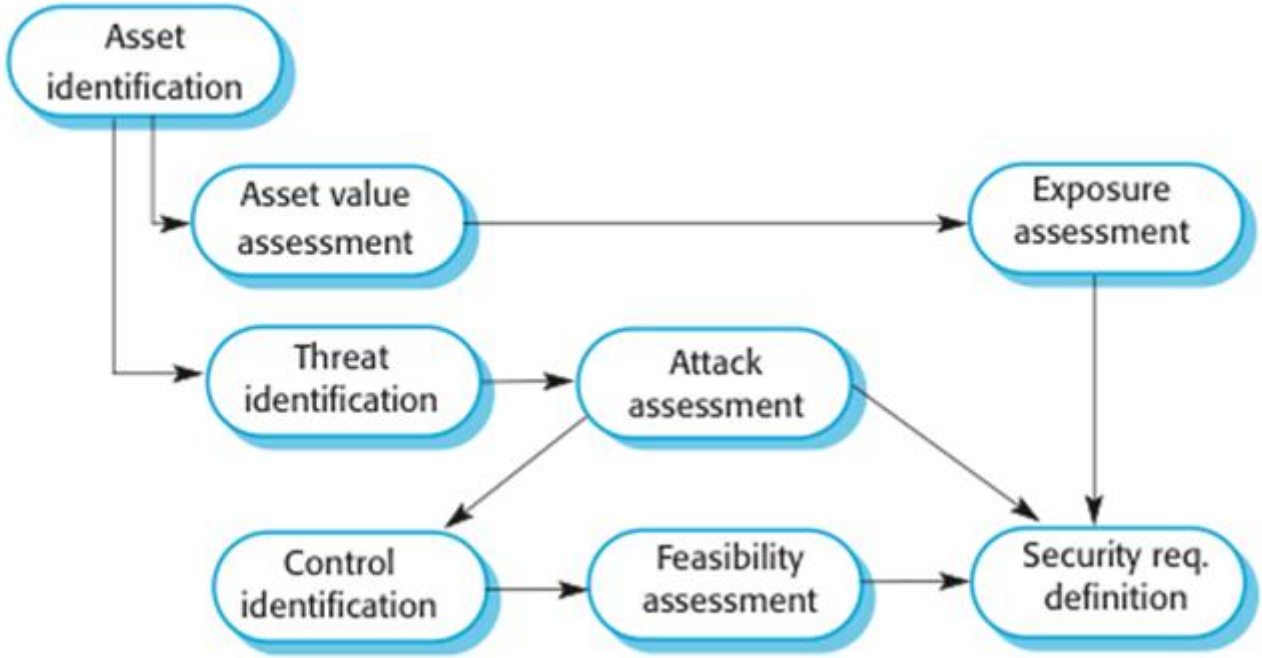
- Security specification has something in common with safety requirements specification – in both cases, your concern is to avoid something bad happening.
- Four major differences
  - Safety problems are accidental – the software is not operating in a hostile environment. In security, you must assume that attackers have knowledge of system weaknesses
  - When safety failures occur, you can look for the root cause or weakness that led to the failure. When failure results from a deliberate attack, the attacker may conceal the cause of the failure.
  - Shutting down a system can avoid a safety-related failure. Causing a shut down may be the aim of an attack.
  - Safety-related events are not generated from an intelligent adversary. An attacker can probe defenses over time to discover weaknesses.

Types of security requirement

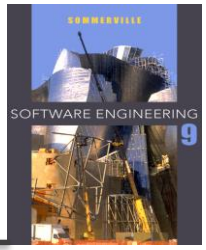


- ✧ Identification requirements.
- ✧ Authentication requirements.
- ✧ Authorisation requirements.
- ✧ Immunity requirements.
- ✧ Integrity requirements.
- ✧ Intrusion detection requirements.
- ✧ Non-repudiation requirements.
- ✧ Privacy requirements.
- ✧ Security auditing requirements.
- ✧ System maintenance security requirements.

The preliminary risk assessment process for security requirements

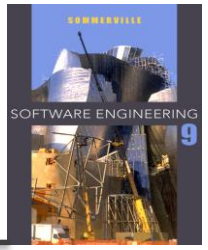


Security risk assessment



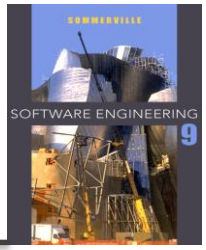
- ✧ Asset identification
  - Identify the key system assets (or services) that have to be protected.
- ✧ Asset value assessment
  - Estimate the value of the identified assets.
- ✧ Exposure assessment
  - Assess the potential losses associated with each asset.
- ✧ Threat identification
  - Identify the most probable threats to the system assets

Security risk assessment



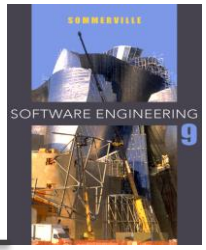
- ✧ Attack assessment
  - Decompose threats into possible attacks on the system and the ways that these may occur.
- ✧ Control identification
  - Propose the controls that may be put in place to protect an asset.
- ✧ Feasibility assessment
  - Assess the technical feasibility and cost of the controls.
- ✧ Security requirements definition
  - Define system security requirements. These can be infrastructure or application system requirements.

Asset analysis in a preliminary risk assessment report for the MHC-PMS



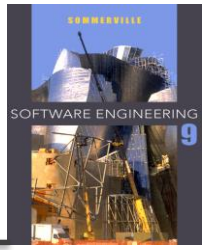
Asset	Value	Exposure
The information system	High. Required to support all clinical consultations. Potentially safety-critical.	High. Financial loss as clinics may have to be canceled. Costs of restoring system. Possible patient harm if treatment cannot be prescribed.
The patient database	High. Required to support all clinical consultations. Potentially safety-critical.	High. Financial loss as clinics may have to be canceled. Costs of restoring system. Possible patient harm if treatment cannot be prescribed.
An individual patient record	Normally low although may be high for specific high-profile patients.	Low direct losses but possible loss of reputation.

Threat and control analysis in a preliminary risk assessment report



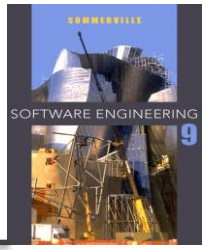
Threat	Probability	Control	Feasibility
Unauthorized user gains access as system manager and makes system unavailable	Low	Only allow system management from specific locations that are physically secure.	Low cost of implementation but care must be taken with key distribution and to ensure that keys are available in the event of an emergency.
Unauthorized user gains access as system user and accesses confidential information	High	Require all users to authenticate themselves using a biometric mechanism. Log all changes to patient information to track system usage.	Technically feasible but high-cost solution. Possible user resistance. Simple and transparent to implement and also supports recovery.

Security policy



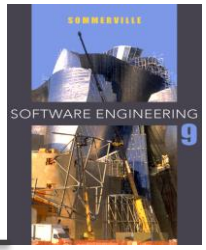
- An organizational security policy applies to all systems and sets out what should and should not be allowed.
- For example, a military policy might be:
  - Readers may only examine documents whose classification is the same as or below the readers vetting level.
- A security policy sets out the conditions that must be maintained by a security system and so helps identify system security requirements.

Security requirements for the MHC-PMS



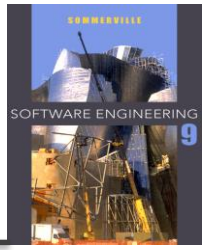
- Patient information shall be downloaded at the start of a clinic session to a secure area on the system client that is used by clinical staff.
- All patient information on the system client shall be encrypted.
- Patient information shall be uploaded to the database after a clinic session has finished and deleted from the client computer.
- A log on a separate computer from the database server must be maintained of all changes made to the system database.

Formal specification



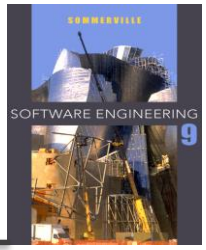
- Formal specification is part of a more general collection of techniques that are known as ‘formal methods’.
- These are all based on mathematical representation and analysis of software.
- Formal methods include
  - Formal specification;
  - Specification analysis and proof;
  - Transformational development;
  - Program verification.

Use of formal methods



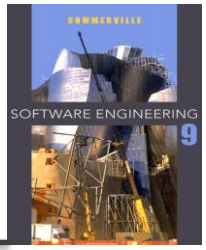
- The principal benefits of formal methods are in reducing the number of faults in systems.
- Consequently, their main area of applicability is in critical systems engineering. There have been several successful projects where formal methods have been used in this area.
- In this area, the use of formal methods is most likely to be cost-effective because high system failure costs must be avoided.

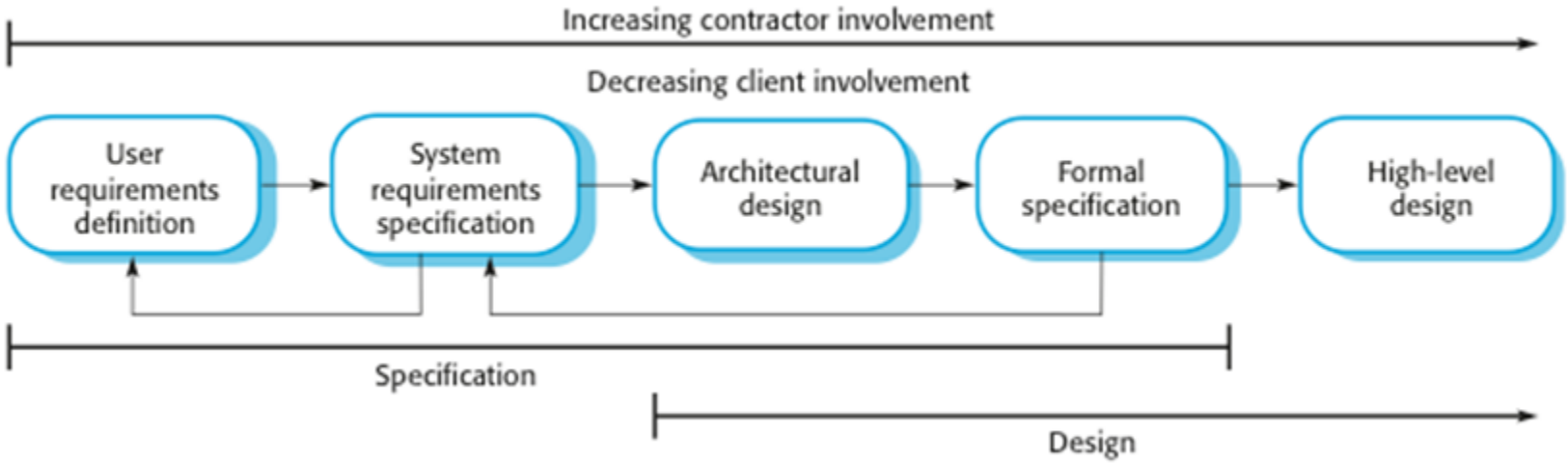
Specification in the software process



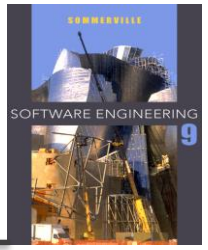
- Specification and design are inextricably intermingled.
- Architectural design is essential to structure a specification and the specification process.
- Formal specifications are expressed in a mathematical notation with precisely defined vocabulary, syntax and semantics.

Formal specification in a plan-based software process



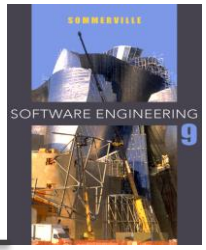


Benefits of formal specification



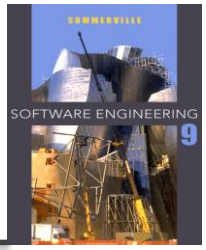
- Developing a formal specification requires the system requirements to be analyzed in detail. This helps to detect problems, inconsistencies and incompleteness in the requirements.
- As the specification is expressed in a formal language, it can be automatically analyzed to discover inconsistencies and incompleteness.
- If you use a formal method such as the B method, you can transform the formal specification into a ‘correct’ program.
- Program testing costs may be reduced if the program is formally verified against its specification.

Acceptance of formal methods



- Formal methods have had limited impact on practical software development:
  - Problem owners cannot understand a formal specification and so cannot assess if it is an accurate representation of their requirements.
  - It is easy to assess the costs of developing a formal specification but harder to assess the benefits. Managers may therefore be unwilling to invest in formal methods.
  - Software engineers are unfamiliar with this approach and are therefore reluctant to propose the use of FM.
  - Formal methods are still hard to scale up to large systems.
  - Formal specification is not really compatible with agile development methods.

Key points



- Reliability requirements can be defined quantitatively. They include probability of failure on demand (POFOD), rate of occurrence of failure (ROCOF) and availability (AVAIL).
- Security requirements are more difficult to identify than safety requirements because a system attacker can use knowledge of system vulnerabilities to plan a system attack, and can learn about vulnerabilities from unsuccessful attacks.
- To specify security requirements, you should identify the assets that are to be protected and define how security techniques and technology should be used to protect these assets.
- Formal methods of software development rely on a system specification that is expressed as a mathematical model. The use of formal methods avoids ambiguity in a critical systems specification.