

# Systems Engineering

---

- ⊗ Designing, implementing and installing systems which include hardware, software and people

# Objectives

---

- ⊗ To introduce concepts of system engineering to software engineers
- ⊗ To discuss system engineering difficulties
- ⊗ To describe the system procurement and system engineering processes
- ⊗ To discuss reliability in a system context

## Topics covered

---

- ⊗ Systems and their environment
- ⊗ System procurement
- ⊗ The system engineering process
- ⊗ System architecture modelling
- ⊗ Human factors
- ⊗ System reliability engineering

## What is a system?

---

- ⊗ A set of inter-related components working together towards some common objective. The system may include software, mechanical, electrical and electronic hardware and be operated by people.
- ⊗ System components are dependent on other system components
- ⊗ The properties and behaviour of system components are inextricably inter-mingled

## Problems of systems engineering

- ⊗ Large systems are usually designed to solve 'wicked' problems
- ⊗ Systems engineering requires a great deal of co-ordination across disciplines
  - Almost infinite possibilities for design trade-offs across components
  - Mutual distrust and lack of understanding across engineering disciplines
- ⊗ Systems must be designed to last many years in a changing environment

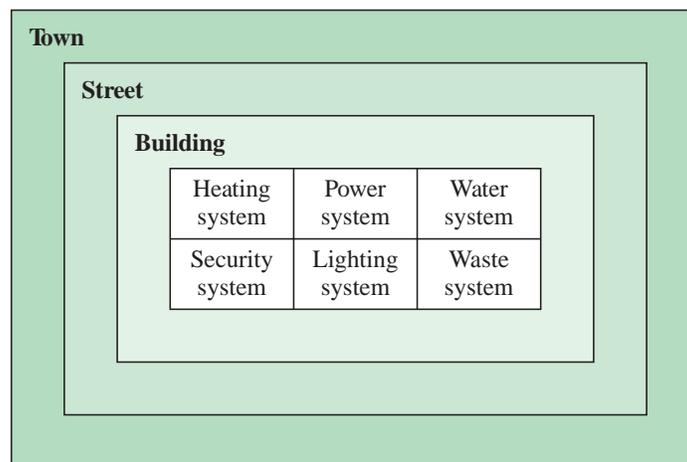
## Software and systems engineering

- ⊗ Proportion of software in systems is increasing. Software-driven general purpose electronics is replacing special-purpose systems
- ⊗ Problems of systems engineering are similar to problems of software engineering
- ⊗ Software is (unfortunately) seen as a problem in systems engineering. Many large system projects have been delayed because of software problems

# Systems and their environment

- ⊗ Systems are not independent but exist in an environment
- ⊗ System's function may be to change its environment
- ⊗ Environment affects the functioning of the system e.g. system may require electrical supply from its environment
- ⊗ Organizational as well as physical environment may be important

# System hierarchies



## System procurement

---

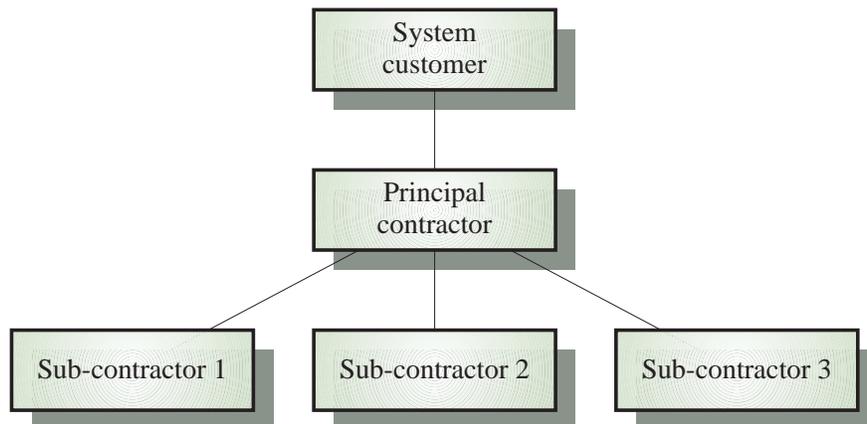
- ⊗ Acquiring a system for an organization to meet some need
- ⊗ Some system specification and architectural design is usually necessary before procurement
  - You need a specification to let a contract for system development
  - The specification may allow you to buy a commercial off-the-shelf (COTS) system. Almost always cheaper than developing a system from scratch

## Contractors and sub-contractors

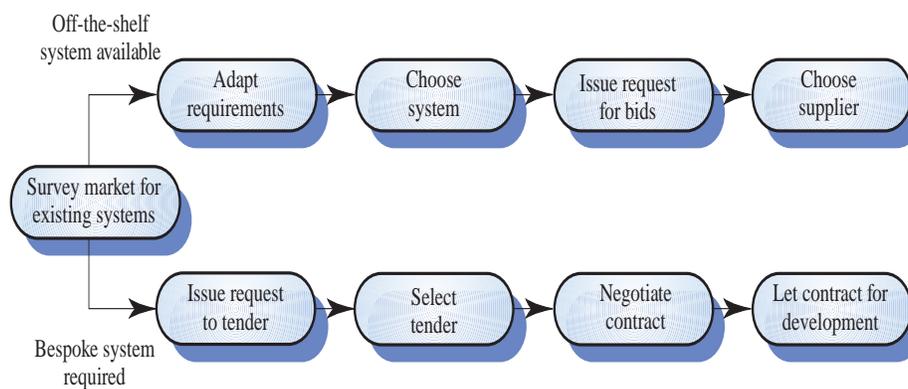
---

- ⊗ The procurement of large hardware/software systems is usually based around some principal contractor
- ⊗ Sub-contracts are issued to other suppliers to supply parts of the system
- ⊗ Customer communicates with the principal contractor and does not deal directly with sub-contractors

## Contractor/Sub-contractor model



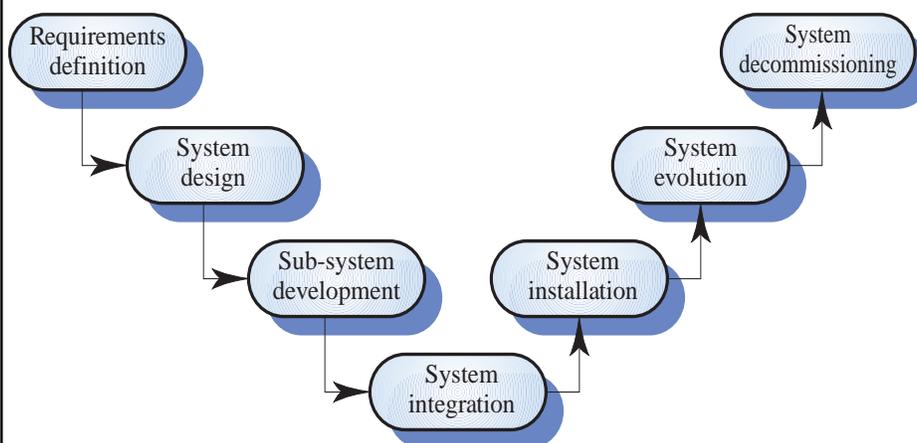
## The system procurement process



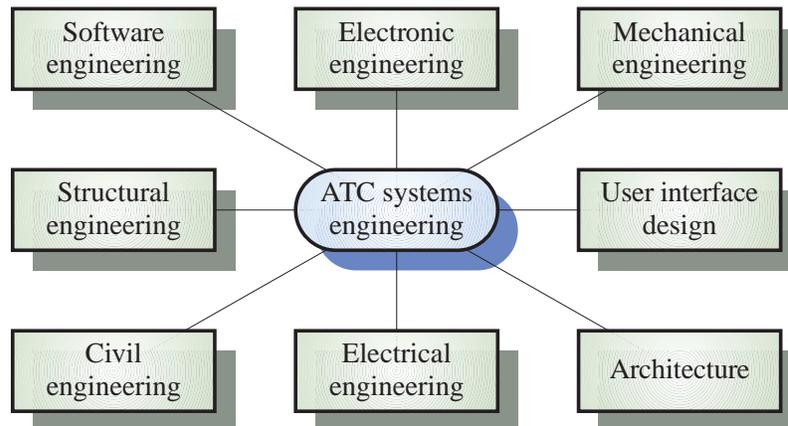
## The system engineering process

- ⊗ Inevitably involves engineers from different disciplines who must work together
  - Much scope for misunderstanding here. Different disciplines use a different vocabulary and much negotiation is required. Engineers may have personal agendas to fulfil
- ⊗ Usually follows a ‘waterfall’ model because of the need for parallel development of different parts of the system
  - Little scope for iteration between phases because hardware changes are very expensive. Software may have to compensate for hardware problems

## The system engineering process



## Inter-disciplinary involvement



## System requirements definition

- ⊗ Three types of requirement defined at this stage
  - Coarse-grain functional requirements. System functions are defined in an abstract way
  - System properties. Non-functional requirements for the system in general are defined
  - Undesirable characteristics. Unacceptable system behaviour is specified
- ⊗ Should also define overall organisational objectives for the system

## System objectives

---

- ⊗ Functional objectives
  - To provide a fire and intruder alarm system for the building which will provide internal and external warning of fire or unauthorized intrusion
- ⊗ Organisational objectives
  - To ensure that the normal functioning of work carried out in the building is not seriously disrupted by events such as fire and unauthorized intrusion

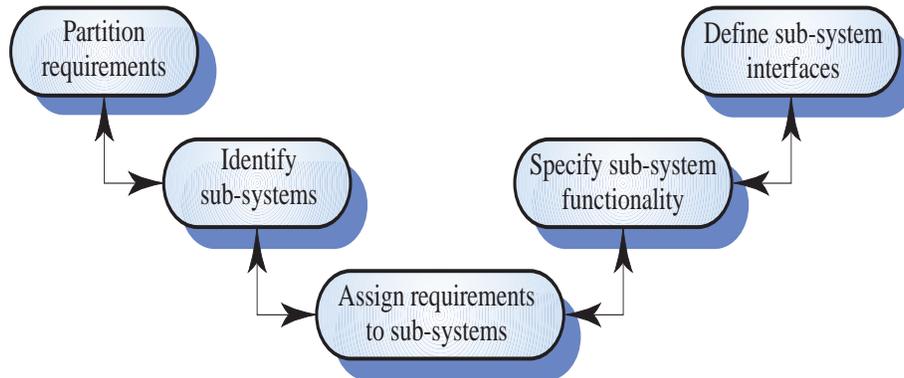
## System requirements problems

---

- ⊗ Changing as the system is being specified
- ⊗ Must anticipate hardware/communications developments over the lifetime of the system
- ⊗ Hard to define non-functional requirements (particularly) without an impression of component structure of the system.

## The system design process

---



## The system design process

---

- ⊗ Partition requirements
  - Organise requirements into related groups
- ⊗ Identify sub-systems
  - Identify a set of sub-systems which collectively can meet the system requirements
- ⊗ Assign requirements to sub-systems
  - Causes particular problems when COTS are integrated
- ⊗ Specify sub-system functionality
- ⊗ Define sub-system interfaces
  - Critical activity for parallel sub-system development

## System design problems

---

- ⊗ Requirements partitioning to hardware, software and human components may involve a lot of negotiation
- ⊗ Difficult design problems are often assumed to be readily solved using software
- ⊗ Hardware platforms may be inappropriate for software requirements so software must compensate for this

## Sub-system development

---

- ⊗ Typically parallel projects developing the hardware, software and communications
- ⊗ May involve some COTS procurement
- ⊗ Lack of communication across implementation teams
- ⊗ Bureaucratic and slow mechanism for proposing system changes means that the development schedule may be extended because of the need for rework

## System integration

---

- ⊗ The process of putting hardware, software and people together to make a system
- ⊗ Should be tackled incrementally so that sub-systems are integrated one at a time
- ⊗ Interface problems between sub-systems are usually found at this stage
- ⊗ May be problems with uncoordinated deliveries of system components

## System installation

---

- ⊗ Environmental assumptions may be incorrect
- ⊗ May be human resistance to the introduction of a new system
- ⊗ System may have to coexist with alternative systems for some time
- ⊗ May be physical installation problems (e.g. cabling problems)
- ⊗ Operator training has to be identified

## System operation

---

- ⊗ Will bring unforeseen requirements to light
- ⊗ Users may use the system in a way which is not anticipated by system designers
- ⊗ May reveal problems in the interaction with other systems
  - Physical problems of incompatibility
  - Data conversion problems
  - Increased operator error rate because of inconsistent interfaces

## System evolution

---

- ⊗ Large systems have a long lifetime. They must evolve to meet changing requirements
- ⊗ Evolution is inherently costly
  - Changes must be analysed from a technical and business perspective
  - Sub-systems interact so unanticipated problems can arise
  - There is rarely a rationale for original design decisions
  - System structure is corrupted as changes are made to it
- ⊗ Existing systems which must be maintained are called legacy systems

## System decommissioning

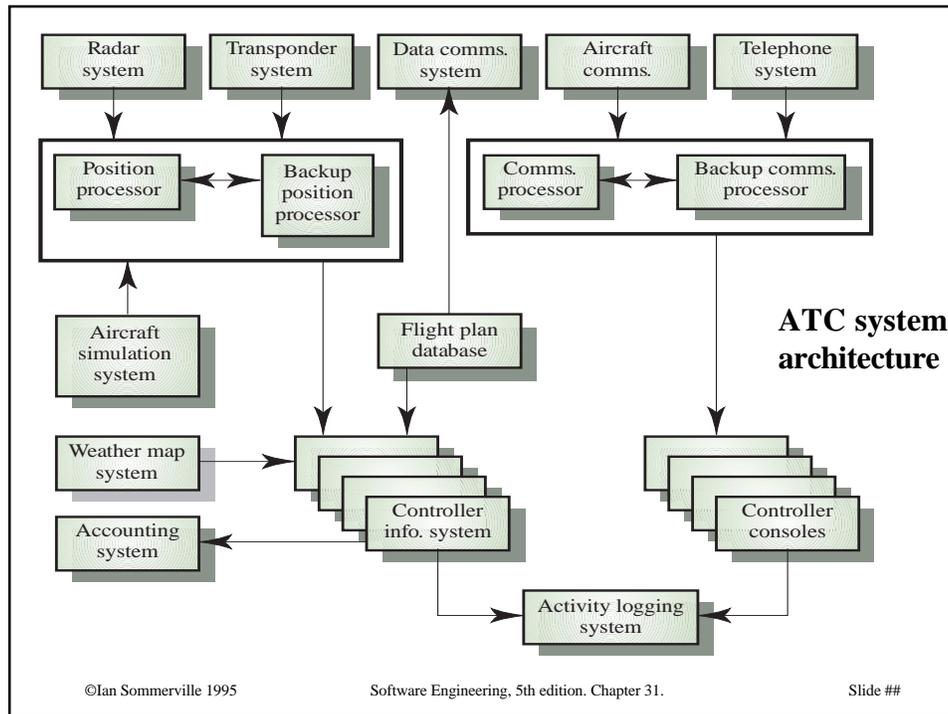
---

- ⊗ Taking the system out of service after its useful lifetime
- ⊗ May require removal of materials (e.g. dangerous chemicals) which pollute the environment
  - Should be planned for in the system design by encapsulation
- ⊗ May require data to be restructured and converted to be used in some other system

## System architecture modelling

---

- ⊗ An architectural model presents an abstract view of the sub-systems making up a system
- ⊗ May include major information flows between sub-systems
- ⊗ Usually presented as a block diagram
- ⊗ May identify different types of functional component in the model



## System functional components

- ⊗ Sensor components
- ⊗ Actuator components
- ⊗ Computation components
- ⊗ Communication components
- ⊗ Co-ordination components
- ⊗ Interface components

# System components

---

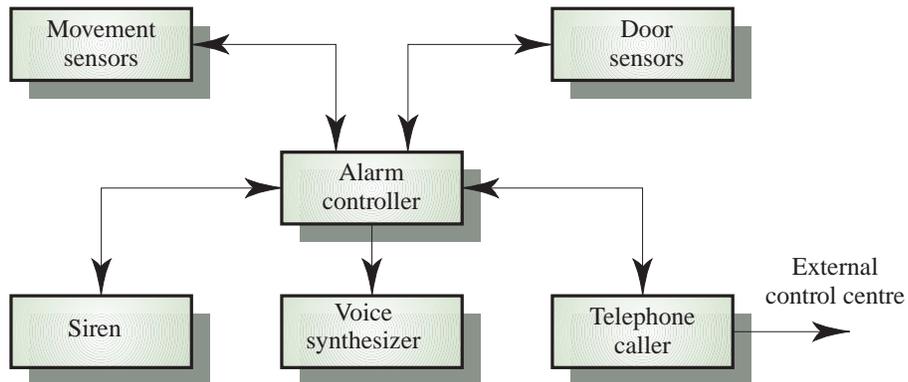
- ⊗ **Sensor components**
  - Collect information from the system's environment e.g. radars in an air traffic control system
- ⊗ **Actuator components**
  - Cause some change in the system's environment e.g. valves in a process control system which increase or decrease material flow in a pipe
- ⊗ **Computation components**
  - Carry out some computations on an input to produce an output e.g. a floating point processor in a computer system

# System components

---

- ⊗ **Communication components**
  - Allow system components to communicate with each other e.g. network linking distributed computers
- ⊗ **Co-ordination components**
  - Co-ordinate the interactions of other system components e.g. scheduler in a real-time system
- ⊗ **Interface components**
  - Facilitate the interactions of other system components e.g. operator interface
- ⊗ **All components are now usually software controlled**

## Intruder alarm system



©Ian Sommerville 1995

Software Engineering, 5th edition. Chapter 2

Slide 33

## Component types in alarm system

- ⊗ Sensor
  - Movement sensor, door sensor
- ⊗ Actuator
  - Siren
- ⊗ Communication
  - Telephone caller
- ⊗ Co-ordination
  - Alarm controller
- ⊗ Interface
  - Voice synthesizer

©Ian Sommerville 1995

Software Engineering, 5th edition. Chapter 2

Slide 34

## Human factors

---

- ⊗ All systems have human users and are used in a social and organisational context
- ⊗ An appropriate user interface is essential for effective system operation
- ⊗ Human factors are often the most important factor in determining the success or otherwise of a system

## Other human factors

---

- ⊗ Changes to work processes in the system's environment
  - May be resisted by users if jobs are lost
- ⊗ De-skilling of users
  - May be resented by professionals
- ⊗ Changes to organisation power structure
  - Managers don't like to lose control
- ⊗ Work changes
  - Some changes to work practice may be unacceptable

## 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

## Reliability assessment

---

- ⊗ Has to be formulated at a systems level and not just at a software level
- ⊗ Hardware engineers have good (but limited) reliability models and can't understand the problems of software reliability
- ⊗ Operational profile depends on the way in which the system is used.

## System resilience

---

- ⊗ What degree of resilience should be built into the system to allow for component availability?
- ⊗ What components need be duplicated to ensure adequate service
- ⊗ What alternative ways of providing a service can be devised

## Interface engineering

---

- ⊗ User interface determines what system facilities are used. If the interface to some facilities is better than others, they will be more heavily used
- ⊗ Some reliability problems are actually user interface problems
- ⊗ User interfaces should be designed to minimise operator error

## User interface engineering problems

---

- ⊗ What is a mistake?
- ⊗ How can these mistakes be eliminated?
  - Error avoidance - don't allow the operator to do something that is incorrect
  - Error detection - detect an incorrect action and report it to the operator
- ⊗ What degree of operator over-riding should be allowed?

## Conclusion

---

- ⊗ Systems engineering is hard! There will never be an easy answer to the problems of complex system development
- ⊗ Software engineers do not have all the answers but are often better at taking a systems viewpoint
- ⊗ Disciplines need to recognise each others strengths and actively rather than reluctantly cooperate in the systems engineering process

## Key points

---

- ⊗ System engineering involves input from a range of disciplines
- ⊗ COTS systems are cheapest. However, most large systems require some tailored sub-systems
- ⊗ Software may act as 'glue' between COTS systems from different suppliers
- ⊗ Systems engineering process is usually a waterfall model

## Key points

---

- ⊗ System architectural models should show major sub-systems and inter-connections. They are usually described using block diagrams
- ⊗ System component types are sensor, actuator, computation, co-ordination, communication and interface
- ⊗ System reliability depends on hardware, software and operator reliability