Chapter 2. Computer-based Systems Engineering

- Designing, implementing, deploying and operating systems which include hardware, software and people

Objectives

- To explain why system software is affected by broader system engineering issues
- To introduce the concept of emergent system properties such as reliability,performance,safety and security
- To explain why the systems environment must be considered in the system design process
- To explain system engineering and system procurement processes

Topics covered

- Emergent system properties
- Systems and their environment
- System modelling
- The system engineering process
- System procurement

What is a system?

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

Problems of systems engineering

- Large systems are usually designed to solve 'wicked' problems (complex and so many related entities that are not defined clearly)
- 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

- The 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

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
- They can therefore only be assessed and measured once the components have been integrated into a system

Examples of emergent properties

- *The overall weight of the system*
  - This is an example of an emergent property that can be computed from individual component properties.
- *The reliability of the system*
  - This depends on the reliability of system components and the relationships between the components.
- *The usability of a system*
  - This is a complex property which is not simply dependent on the system hardware and software but also depends on the system operators and the environment where it is used. (軍規or商規pp. 23)

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 emergent properties
  - Examples are **reliability, performance, safety, and security**. These relate to the **behaviour of the system** in its operational environment. 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? (**MTBF or MTTF**)
- *Software reliability*
  - How likely is it that a software component will produce an incorrect output. Software failure is usually distinct from hardware failure in that software does not **wear out**.
- *Operator reliability*
  - How likely is it that the operator of a system will make an error?

Reliability relationships

- Hardware failure can generate spurious signals that are **outside the range of inputs** expected by the software
- Software errors can cause **alarms** to be activated which cause operator stress and lead to **operator errors**
- **Operator errors** may stress the hardware and cause more failure
- The environment in which a system is installed can affect its reliability

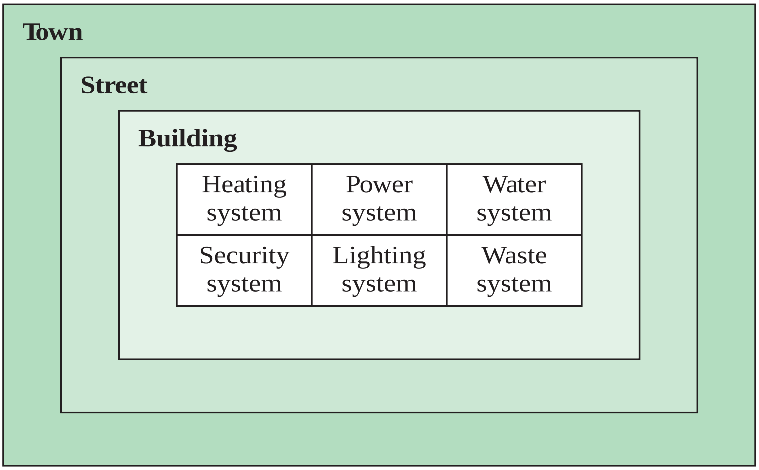
The system ‘should-not’ exhibit properties

- Properties such as **performance and reliability** can be measured after the system is operational.
- 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 and their environment

- Systems are not independent but exist in an environment
- **System’s function** may be intended to change its **environment ➔ heat to the environment**
- Environment affects the functioning of the system that is **hard to predict**. e.g. system may require electrical supply from its environment but **electrical is not enough**
- The **organizational** as well as the **physical environment** may be important

System hierarchies of building security



Human social and organisational factors

The factors that affect the system design include: (人因工程)

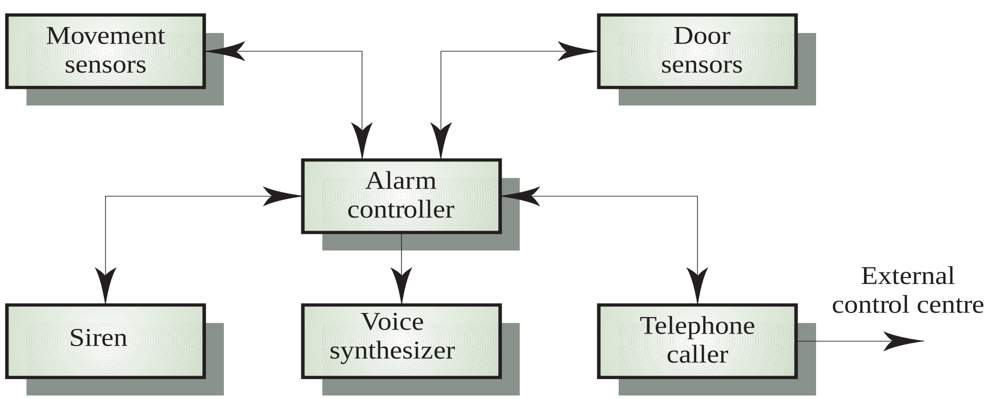
- *Process changes*
  - Does the system require changes to the work processes in the environment? → Training
- *Job changes*
  - Does the system de-skill the users in an environment or cause them to change the way they work? → resist the system into the organization
- *Organisational changes*
  - Does the system change the political power structure in an organisation?

System architecture modelling

- An architectural model presents an **abstract graphical view** of the sub-systems making up a system➔**overall view**
- May include major **information flows** between sub-systems
- Usually presented **sub-system** as a **block diagram**

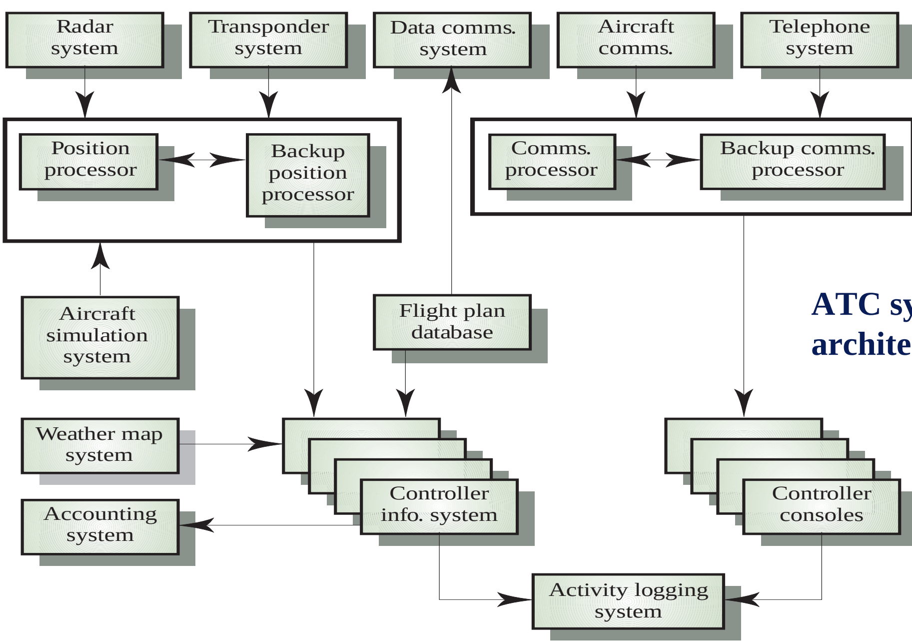
  ex. Network linking machine consist of physical cables + repeater + gateway
- May identify **different types of functional component** in the model ➔hw/sw trade-offs

Intruder alarm system



Subsystem functions in alarm system

- Movement sensor, Door sensor
  - Detect movement in a protected space, door open
- Alarm controller
  - Controls the operation of the system
- Siren
  - Emit an audible warning when an intruder is suspected
- Voice synthesizer
  - Synthesis message giving the location of the intruder
- Telephone caller
  - call to external control



ATC system architecture

Functional system components

Without consider whether SW/HW

- Sensor components➔**collect environment data**
- Actuator components➔**valve open/close control**
- Computation components➔**processor ability**
- Communication components➔**communicate with other component**
- Co-ordination components➔**coordinate the operation of other component**
- Interface components➔ **convert representation of each other components**

Component types in alarm system

- Sensor(Detect movement in a protected space, door open)
  - Movement sensor, door sensor
- Actuator(Audible warning of intrusion)
  - Siren
- Communication(call to external control centre)
  - Telephone caller
- Co-ordination(Coordinate all system components)
  - Alarm controller
- Interface(Synthesis message giving location of intrusion)
  - Voice synthesizer

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

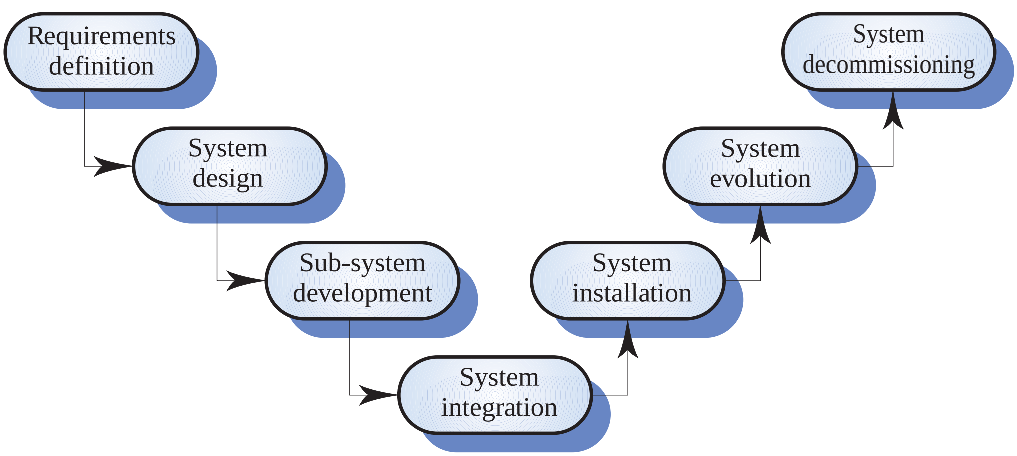
- Communication components
  - Allow system components to communicate with each other e.g. network linking distributed computers
- Co-ordination components
  - Co-ordinate the interactions of other system components e.g. scheduler in a real-time system
- Interface components
  - Facilitate the interactions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

The system engineering process

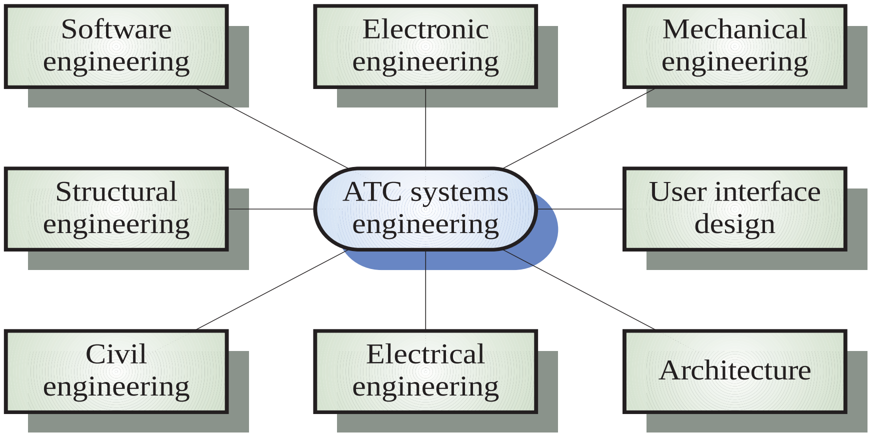
- Reduced scope for rework during system development Usually follows a 'waterfall' model because of the need for parallel development of different parts of the system
  - Little scope for iteration between phases because hardware changes are very expensive. Reworking the system design to solve problems is rarely possible. **Software may have to compensate for hardware problems**
- Interdisciplinary involvement

  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

The system engineering process



Inter-disciplinary involvement



System requirements definition

- Three types of requirement defined at this stage
  - **Abstract functional requirements**. System basic functions are defined in an abstract way
  - **System properties**. Non-functional emergent requirements for the system in general are defined
  - **Characteristics which the system must not exhibit**. What the system should do and not do is specified ➔ **constraints**
- Should also define overall organisational objectives for the system

System objectives for an office building

- 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

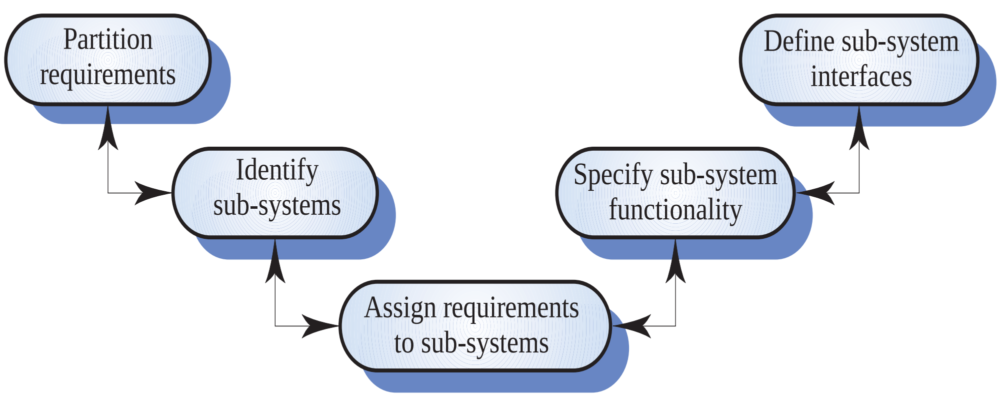
- **Changing** as the system is being specified
- Must **anticipate hardware/communications developments** over the lifetime of the system
- Hard to define **non-functional requirements** particularly without an impression of component structure of the system.ex. **Earthquake, typhoon**…

➔ How to solve wicked problem(complex and so many related entities that are not defined clearly)

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** → modification
- Specify sub-system functionality
- Define sub-system interfaces
  - **Parallel sub-system development** when interfaces have been agreed

The system design process



System design problems

- Requirements partitioning to hardware, software and human components may involve a lot of **negotiation** and **trade-off**
- **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(Commercial Off-the-Shelf)** systems procurement
- Lack of communication across implementation teams

→ Cut across subsystem boundaries →system modification required

- **Slow mechanism for proposing system changes** means that the development schedule may be extended because of the need for **re-work**

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→version control**

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. network cabling, air-conditioning 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 **recorded** for original design decisions
  - **System structure is corrupted** as changes are made to it
- Existing systems which must be maintained are sometimes called legacy systems

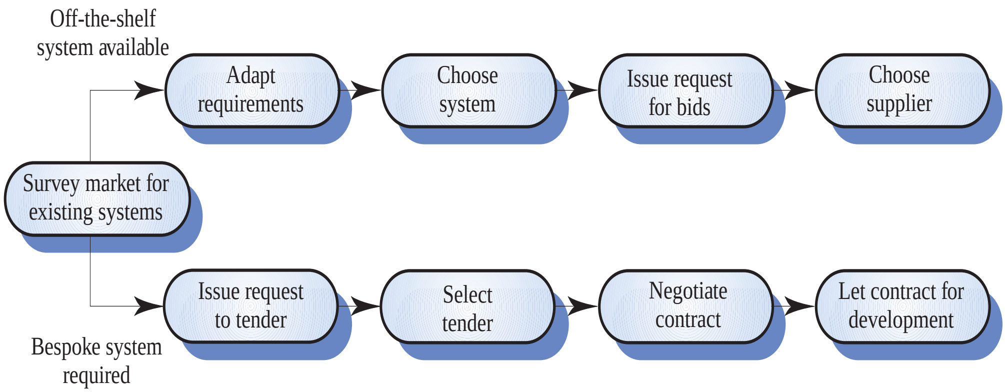
System decommissioning

- 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 procurement

- Acquiring a system for an organization to meet some need(to buy or contract design to build a system)
- 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 COTS is always cheaper than developing a system from scratch

The system procurement process



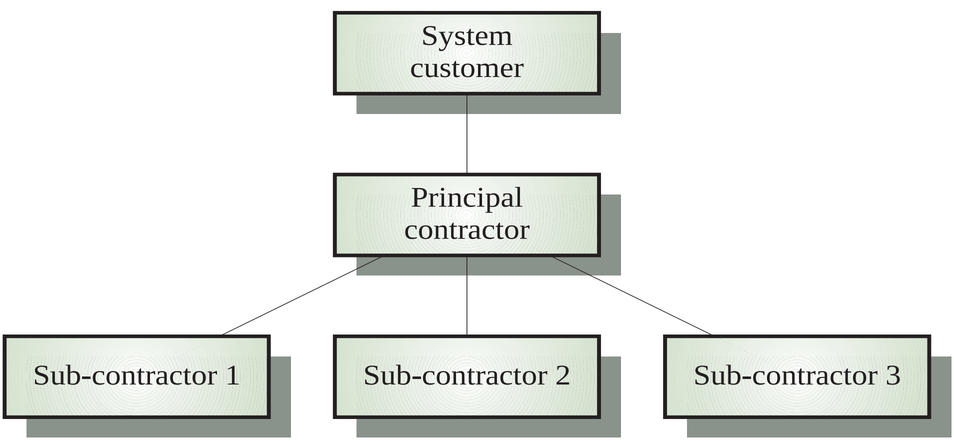
Procurement issues

- Requirements may have to be **modified** to match the capabilities of off-the-shelf components
- The requirements specification may be **part of the contract** for the development of the system
- There is usually a **contract negotiation period** to agree changes after the contractor to build a system has been selected

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 contacts with the principal contractor** and does not deal directly with sub-contractors

Contractor/Sub-contractor model



Key points

- System engineering involves input from a range of disciplines(Inter-discipline)
- Emergent properties are properties that are characteristic of the system as a whole and not its component parts
- System architectural models show major sub-systems and inter-connections. They are usually described using block diagrams

Key points

- System component types are sensor, actuator, computation, co-ordination, communication and interface
- The systems engineering process is usually a waterfall model and includes specification, design, development and integration.
- System procurement is concerned with deciding which system to buy and who to buy it from

Conclusion

- Systems engineering is hard! There will **never be an easy answer** to the ‘wicked’ problems of complex and interrelated subsystem development
- Software engineers do not have all the answers but may be better at taking a **systems viewpoint**
- **Disciplines** need to recognize each others strengths rather than reluctantly cooperate in the systems engineering process

HomeWork#2

- Prepare your project name and team members
- Prepare to analyze your project into subsystems(Fig. 2.2)
- 2.7
- 2.10