The System Engineering Process

- Designing, implementing, deploying and operating systems which include **hardware, software and people**
- It *is* a mix of HR, project management, business, rational decomposition, trade studies, requirements traceability, integration, testing, verification and validation, operations, and end of life cycle disposal of systems
- Standardizes the ***flow-down and traceability*** of specifications for complex products from customer requirements through production, operation, and disposal

Systems Engineering

- Systems Engineering is an ***interdisciplinary approach*** and means to enable the realization of successful systems.
- It focuses on defining ***customer needs*** and required functionality early in the development cycle, documenting requirements, then proceeding with design synthesis and system validation
- Systems Engineering considers both the business and the technical needs of all customers with the goal of ***providing a quality product*** that meets the user needs

Systems Engineering & Component Engineering

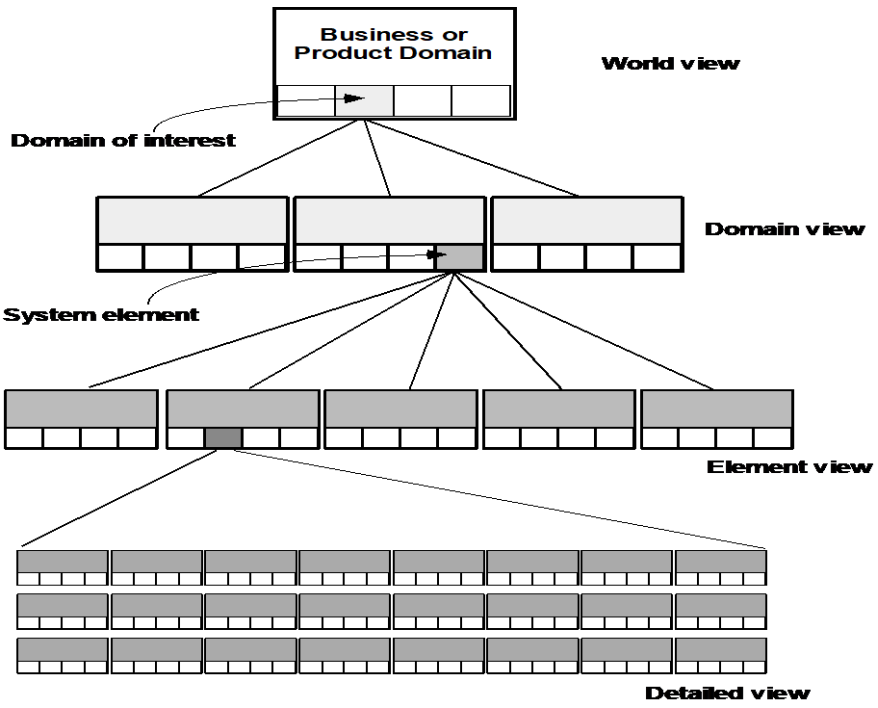
- Science
  - Determines what Is
- Component Engineering
  - Determines what Can Be
- Systems Engineering
  - Determines what Should Be

Role of Systems Engineering in Product Development

- Integrates Technical Effort across the Development Project
  - Functional Disciplines
  - Technology Domains
  - Specialty Concerns

System Engineering

- Elements of a computer-based system
  - Software
  - Hardware
  - People
  - Database
  - Documentation
  - Procedures
- Systems
  - A hierarchy of macro-elements



Problems of Systems Engineering

- Large systems are usually designed to solve **'wicked' problems**
- Systems engineering requires a great deal of **co-ordination across disciplines**
- Systems must be designed to last many years in a changing environment

Software and Systems 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
- The reliability of the system
- The usability of a system

Costs of Software Engineering

- Functional properties
- Non-functional emergent properties

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
- Software reliability
- Operator reliability

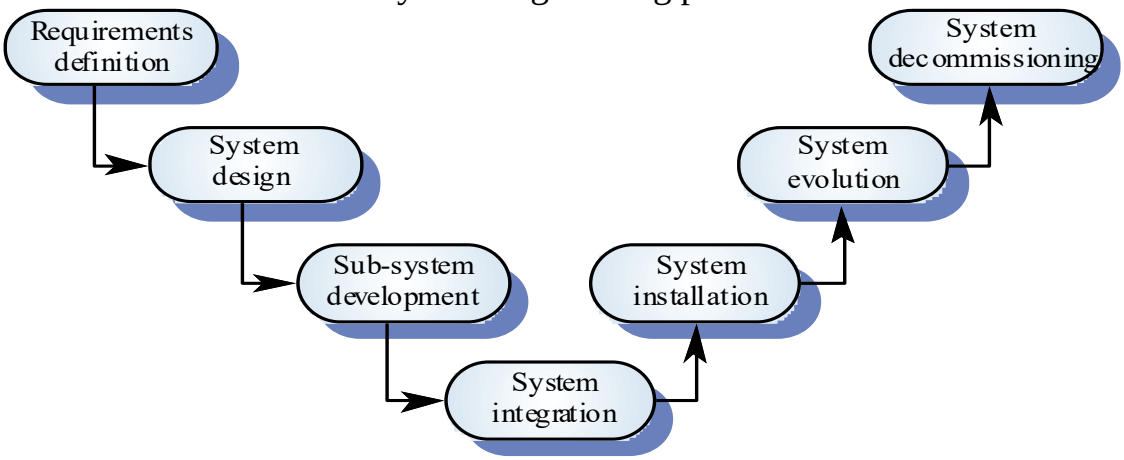
Human and organisational factors for System Engineering

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

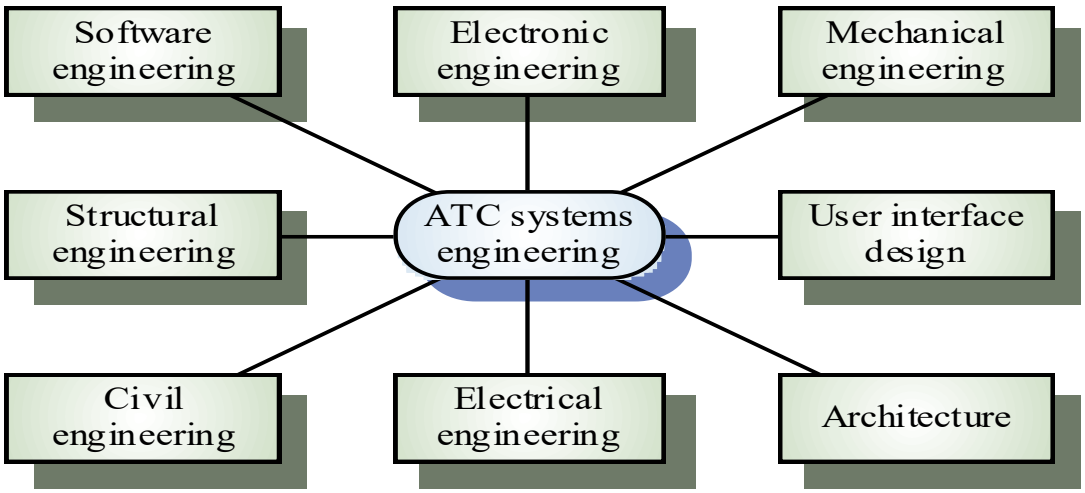
System Architecture Modelling

- 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.

The system engineering process



Inter-disciplinary involvement



System Requirements Definition

- Three types of requirement defined at this stage
  - Abstract functional requirements → System properties.
  - Non-functional requirements for the system in general are defined
  - Undesirable characteristics.
- Define overall organisational objectives for the system

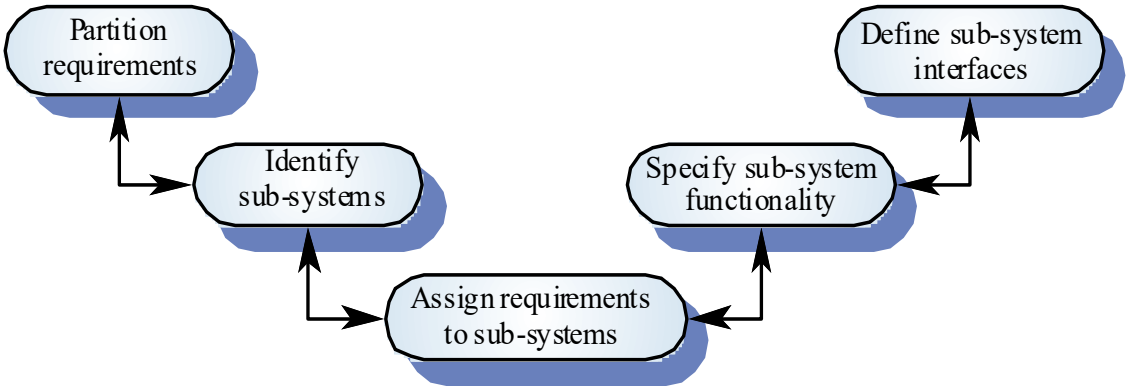
System Objectives

- Functional objectives
  - Defines the major objective
- Organisational objectives
  - Defines the system usage

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.

System Design Process



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.

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
- 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.
- Human resistance to the introduction of a new system.
- System may have to coexist with alternative systems for some time.
- Physical installation problems (e.g. cabling problems).
- Operator training has to be identified.

System Operation

- 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 sometimes called *legacy systems*

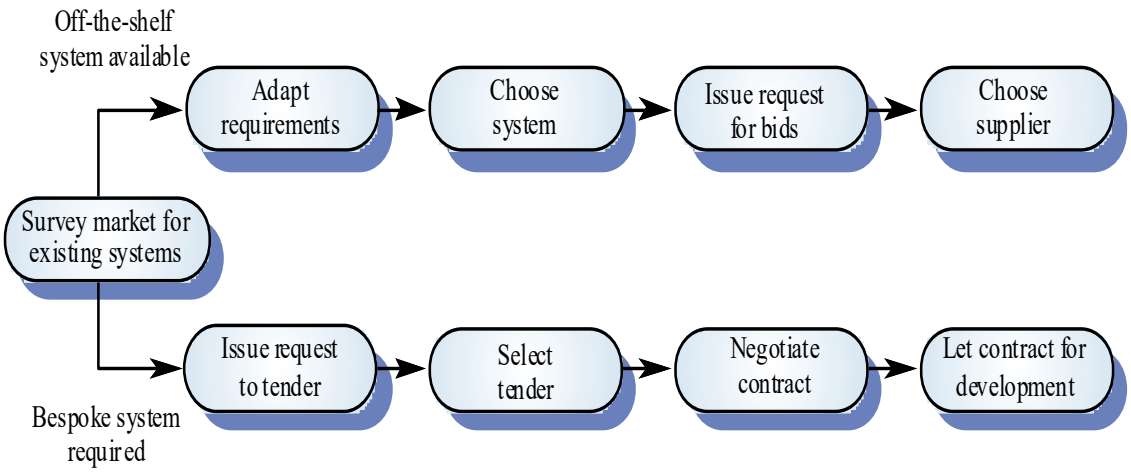
System Decommissioning

- Taking the system out of service after its useful lifetime
- May require removal of materials (e.g. dangerous chemicals) which pollute the environment
- 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
- 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

The System Procurement Process



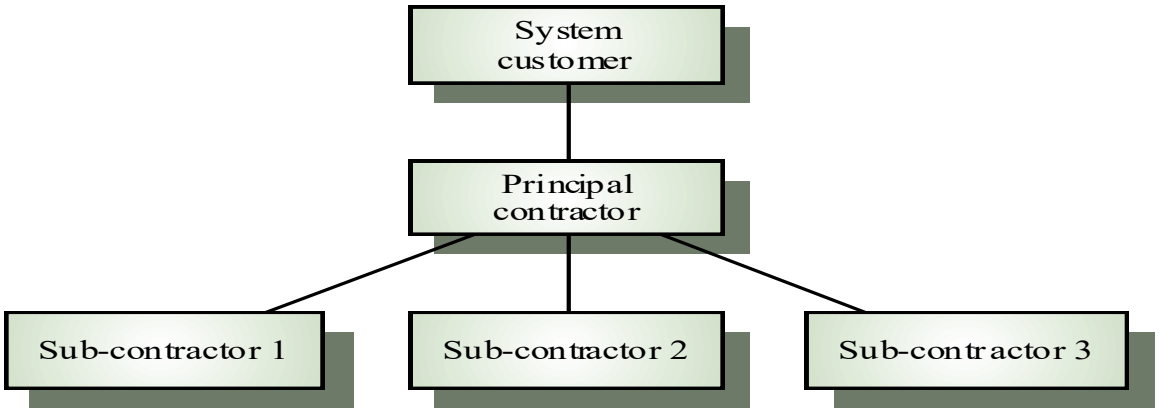
Procurement Issues

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

Contractor/Sub-Contractor Model



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 may be better at taking a systems viewpoint
- Disciplines need to recognise each others strengths and actively rather than reluctantly cooperate in the systems engineering process