Chapter 6. Requirements Engineering Processes

- Processes used to discover, analyse and validate system requirements

Objectives

- To describe the principal requirements engineering activities
- To introduce techniques for requirements elicitation and analysis
- To describe requirements validation
- To discuss the role of requirements management in support of other requirements engineering processes

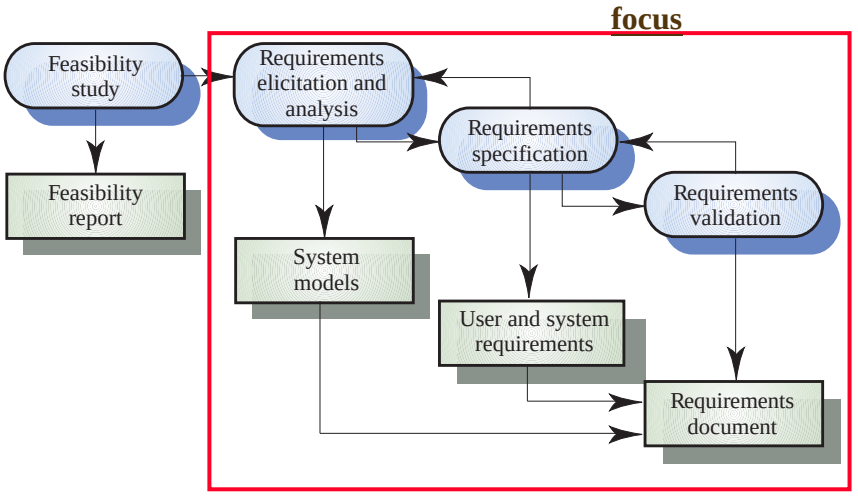
Topics covered

- Feasibility studies
- Requirements elicitation and analysis
- Requirements validation
- Requirements management

Requirements engineering processes

- Requirement engineering is a process that involves all of the activities required to create and maintain a system requirements document. There are 4 generic RE process activities: **Feasibility study, Requirement elicitation and analysis, Requirement specification, Requirement validation**
- The processes used for RE vary widely depending on the **application domain**, the **people involved** and the **organisation** developing the requirements
- However, there are 4 generic activities common to all processes
  - Requirements elicitation
  - Requirements analysis
  - Requirements validation
  - Requirements management

The 4 requirements engineering process



Feasibility studies

- A feasibility study decides whether or not the proposed system is worthwhile
- **Input** is an outline description of the system and how it will be used within an organization. **Output** is a report which re-commands whether it is worth or not
- A short focused study that checks
  - If the system contributes to organisational objectives(make money?)
  - If the system can be engineered using current technology and within budget and schedule
  - If the system can be integrated with other systems that are used (legacy system)

Feasibility study implementation

- Feasibility study involves information assessment (answer to 3 above questions), information collection and report writing
- Questions for people in the organisation
  - What if the system wasn't implemented?
  - What are problems with current process?
  - What direct contribution will the system make to the business objectives?
  - Can information be transferred to and from other organizational systems?
  - Is new technology needed? What skills?
  - What facilities must be supported by the proposed system and what is not?
- The Report of proposing changes to the scope, budget and schedule of the system and further high-level requirements for the system

Requirement elicitation and analysis

- Sometimes called requirements elicitation or requirements discovery
- Involves **technical staff** working with customers to find out about the **application domain**, the **services** that the system should provide and the system's operational **constraints**
- May involve end-users, managers, engineers involved in maintenance, domain experts, trade union representatives, etc. These are called ***stakeholders*** (some direct or indirect influence on the system requirement)

Problems of requirements analysis

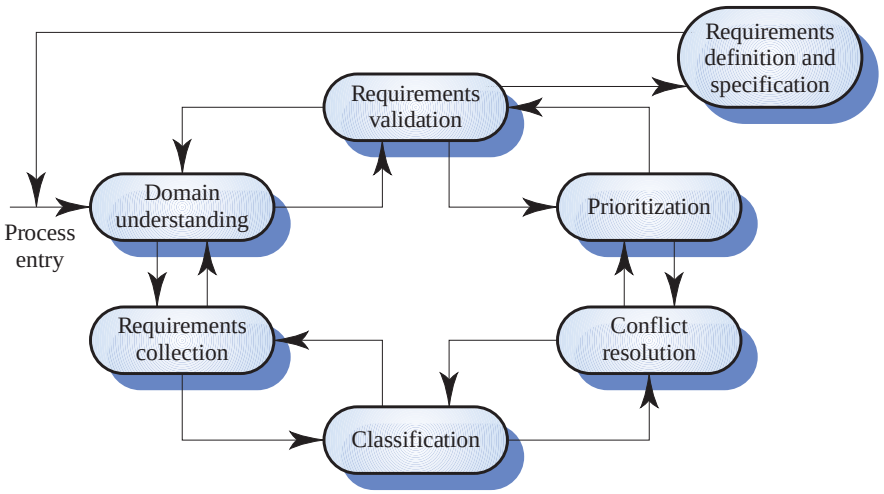
- Stakeholders don't know what they really want
- Stakeholders express requirements in their own **terms**
- Different stakeholders may have conflicting or common requirements
- Organisational and political factors may influence the system requirements
- The economic and business environment during the analysis process is dynamic. New stakeholders may emerge and new requirement emerges

Process activities(iterative process)

Generic process activities of elicitation and analysis process :

- Domain understanding
- Requirements collection – interact with stakeholders
- Classification – restructure the un-structure requirement
- Conflict resolution – because multiple stakeholders are involved
- Prioritisation – interact with stakeholders to decide
- Requirements checking – consistent and complete with stakeholders

The requirements analysis process



System models

- Different models may be produced during the requirements analysis activity
- Requirements analysis may involve three structuring activities which result in these different models
  - Partitioning. Identifies the structural (part-of) relationships between entities
  - Abstraction. Identifies generalities among entities
  - Projection. Identifies different ways of looking at a problem
- 3 techniques for requirement elicitation: Viewpoint-oriented elicitation (brain-storming perspective), scenarios (UML), and ethnography
- Structured analysis methods covered in Chapter 7
- Prototyping covered in Chapter 8
- No perfect approaches to requirement analysis

Viewpoint-oriented elicitation

- Stakeholders represent different ways of looking at a problem or problem viewpoints
- This multi-perspective analysis is important as there is no single correct way to analyse system requirements
- The stakeholder's perspectives are not completely independent and may usually overlap
- Key strength of VP-oriented analysis provides a framework for discovering conflicts in the requirements proposed by different stakeholders

Banking ATM system

- The example used here is an auto-teller system which provides some automated banking services
- I use a very simplified system which offers some services to customers of the bank who own the system and a narrower range of services to other customers
- Services include cash withdrawal, message passing (send a message to request a service), ordering a statement and transferring funds

Back

Auto-teller viewpoints

Stakeholders for ATM include:

- Current Bank customers(receive services)
- Representatives of other banks(跨行功能)
- Managers and counter staff of bank branches
- Database administrators(integrate the customer's data)
- Bank security manager(ensure the system not pose a security hazard)
- Bank's marketing department(use DB data for marketing analysis)➔**business intelligent by data mining**
- Hardware and software maintenance engineers

Types of viewpoint

A VP can be considered as:

- Data sources or sinks
  - Viewpoints are responsible for producing or consuming data. Analysis involves checking that data are produced and consumed and that assumptions about the **source and sink of data** are valid(**SADT or CORE**)
- Representation frameworks
  - Viewpoints represent particular types of system model. These may be compared to discover requirements that would be missed using a single representation. Particularly suitable for **real-time systems**(**ER or state-machine model**)
- Receivers of services
  - Viewpoints are external to the system and receive services from it. It examines the services received by different VPs, collects and resolves the conflicts. Most suited to **interactive systems**

Advantage of Receivers of services

- Natural to think of end-users as receivers of system services
- Relative easy to decide if something is a valid viewpoint
- Viewpoints are a natural way to structure requirements elicitation
- It is relatively easy to decide if a viewpoint is valid
- Viewpoints and services may be suitable to structure non-functional requirements

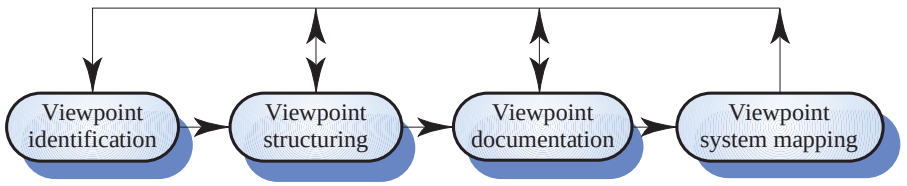
Method-based analysis

- Widely used approach to requirements analysis. Depends on the application of a structured method to understand the system
- Methods have different **emphases**. Some are designed for **requirements elicitation**, others are close to design methods
- A Viewpoint-Oriented Requirement Definition(VORD) has been designed as a service-oriented framework for requirement elicitation and analysis. It also illustrates the use of viewpoints

VORD process model

- Viewpoint identification
  - Discover viewpoints which receive system services and identify the services provided to each viewpoint
- Viewpoint structuring (generalization v.s. specialization)
  - Group related viewpoints into a hierarchy. Common services are provided at higher-levels in the hierarchy and are inherited by low-levels VPs (OO)
- Viewpoint documentation
  - Refine the description of the identified viewpoints and services
- Viewpoint-system mapping
  - Transform the analysis to an object-oriented design. Identify objects in an OOD way using service information provided by VPs (encapsulated in VP)

The VORD method



Object-Oriented approach

- Information hiding or Encapsulation (Data Abstraction)
  - Object, Attributes, Methods
- Inheritance (Single or Multiple)
  - Class hierarchy, reuse
- Polymorphism
  - Operator overloading, operator overwriting

VORD uses diagrammatic notations including:

- **Viewpoint hierarchy diagram**
- **Event scenario**

VORD standard forms

Viewpoint template

Reference:	The viewpoint name.
Attributes:	Attributes providing viewpoint information.
Events:	A reference to a set of event scenarios describing how the system reacts to viewpoint events.
Services	A reference to a set of service descriptions.
Sub-VPs:	The names of sub-viewpoints.

Service template

Reference:	The service name.
Rationale:	Reason why the service is provided.
Specification:	Reference to a list of service specifications. These may be expressed in different notations.
Viewpoints:	List of viewpoint names receiving the service.
Non-functional requirements:	Reference to a set of non-functional requirements which constrain the service.
Provider:	Reference to a list of system objects which provide the service.

ATM examples

Functions

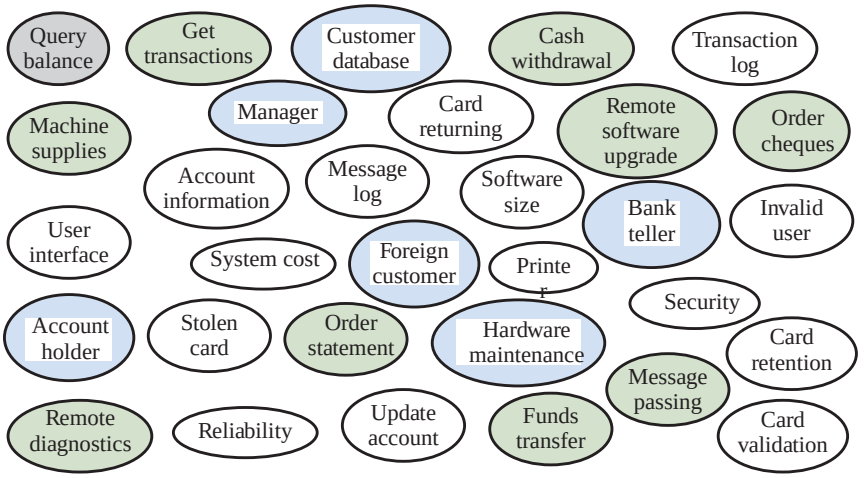
- Accept customer requests

  - Withdraw cash
  - Check their balance
  - Transfer funds from one account to another
  - Cheque book service
- Account information DB update

Viewpoint identification

- Identify all possible VPs is the most difficult stage
- Brainstorming approach to propose the potential services, data inputs, non-functional requirements, control events, exceptions, and VPs
- VPs are shown as dark blue bubbles and services are shown as shaded bubbles

Viewpoint identification



Viewpoint service information

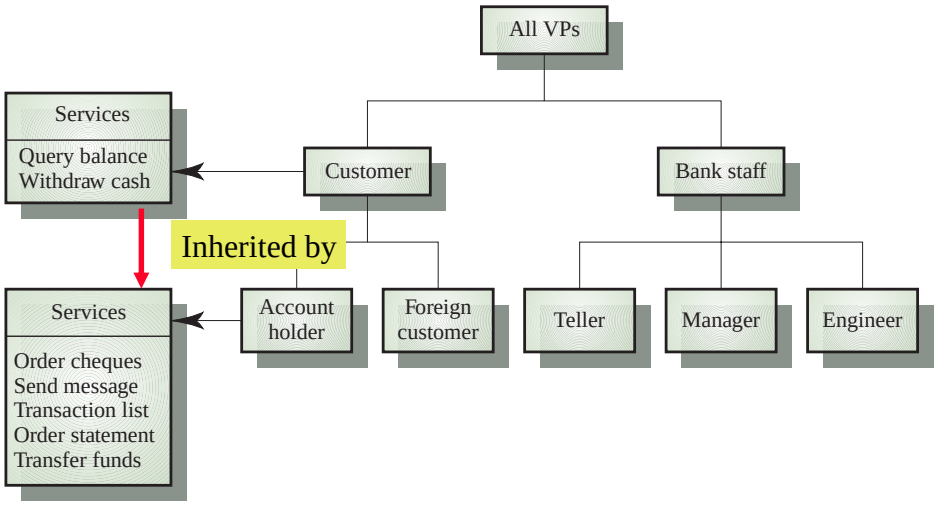
A ACCOUNT HOLDER	B FOREIGN CUSTOMER	C BANK TELLER
Service list	Service list	Service list
Withdraw cash	Withdraw cash	Run diagnostics
Query balance	Query balance	Add cash
Order cheques		Add paper
Send message		Send message
Transaction list		
Order statement		
Transfer funds		

Viewpoint data/control

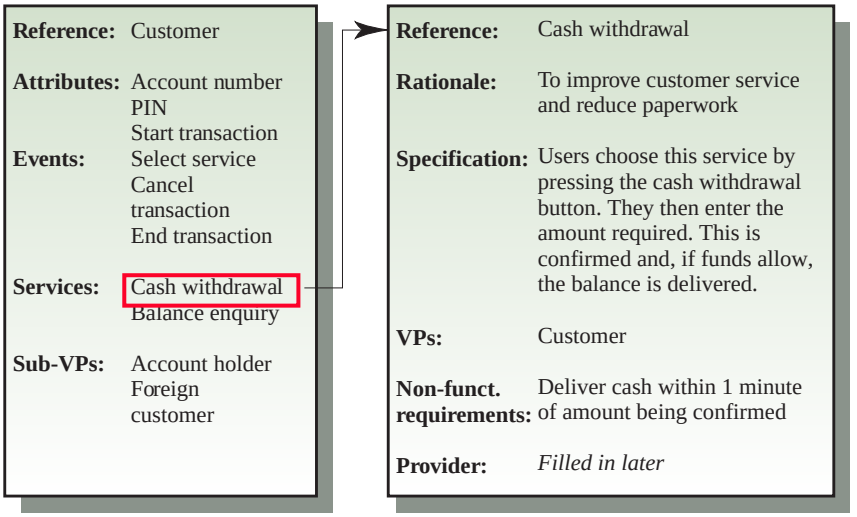
ACCOUNT HOLDER

Control input	Data input
Start transaction Cancel transaction End transaction Select service	Card details PIN Amount required Message

Viewpoint hierarchy



Customer/cash withdrawal templates



Scenarios

- Scenarios are descriptions of how a system is used in practice
- They are helpful in requirements elicitation as people can relate to these more readily than abstract statement of what they require from a system
- Scenarios are particularly useful for adding detail to an outline requirements description
- The scenarios starts with an outline of the interaction during elicitation, details are added to create a complete description of that interaction

Scenario descriptions

A scenario may include:

- A system state at the beginning of the scenario
- Normal flow of events in the scenario
- What can go wrong and how this is handled
- Other activities which might be going on at the same time
- System state after completion of the scenario

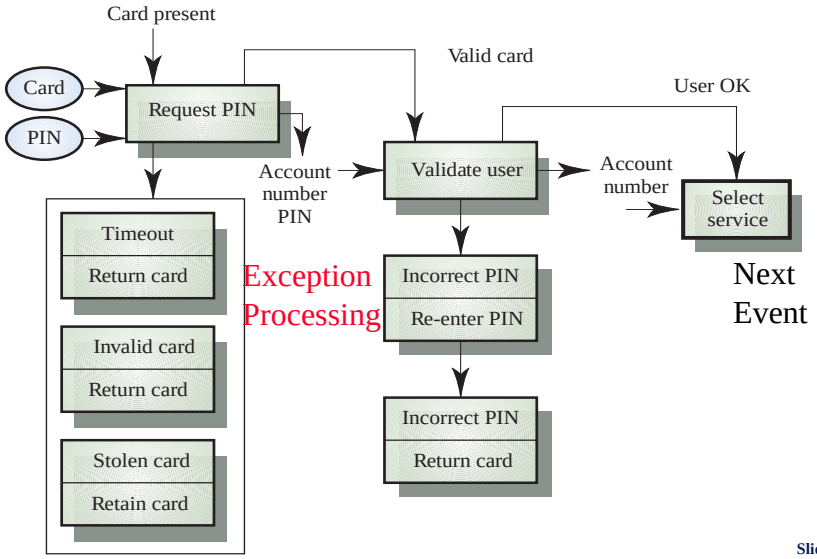
Event scenarios

- Event scenarios include a description of data flows and the actions of the system and document the exceptions which can arise
- Event scenarios may be used to describe how a system responds to the occurrence of some particular event
- A diagrammatic conventions used in event scenarios:
  - Data provided from a VP and delivered to a VP
  - Control information
  - Exception processing
  - The next expected event

Notation for data and control analysis

- **Ellipses**. data provided from or delivered to a viewpoint(Card, PIN)
- **Control information** enters and leaves at the top of each box(Card present, Valid card)
- **Data** leaves from the right of each box(Account number PIN, Account number)
- **Exceptions** are shown at the bottom of each box
- **Name of next event** is in **a shaded box**

Event scenario - start transaction



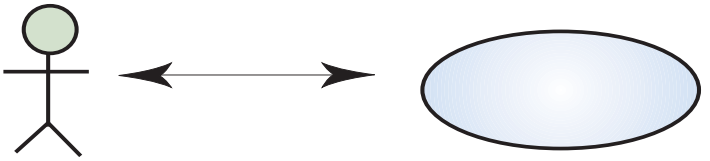
Exception description

- Most methods do not include facilities for describing exceptions
- In this example, at request PIN stages, exceptions are
  - Timeout. Customer fails to enter a PIN within the allowed time limit➔ The card is returned
  - Invalid card. The card is not recognised and is returned➔ The card is returned
  - Stolen card. The card has been registered as a stolen card➔ The card is retained by the machine
- At validate user stage, exceptions are Incorrect PIN checking(return card or re-enter PIN)

Use cases

- Use-cases are a scenario based technique in the UML which identify the actors in an interaction and names the type of interaction
- A set of use cases should describe all possible interactions with the system
- Sequence diagrams may be used to add detail to use-cases by showing the sequence of event processing in the system

Lending use-case

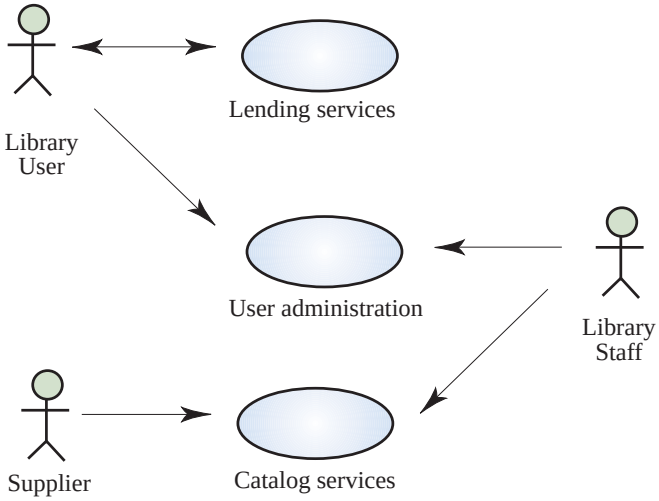


Actor

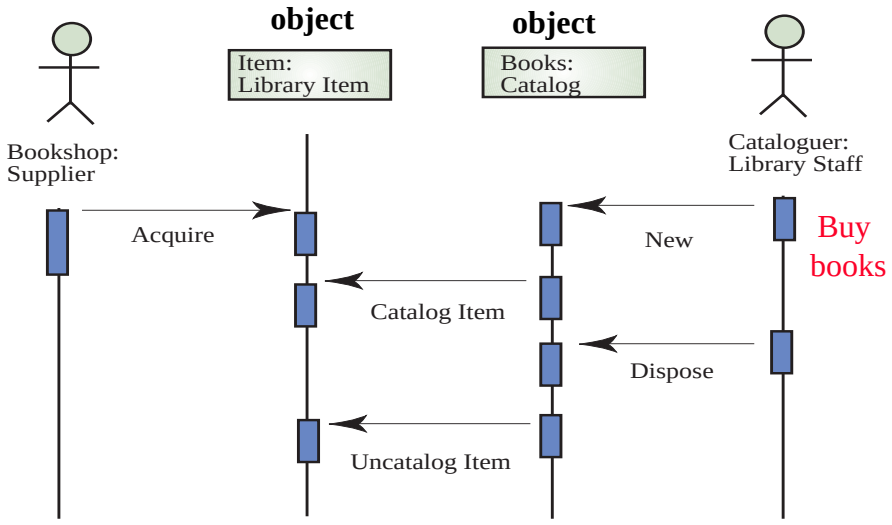
Lending services

Class of interaction

Library use-cases



Catalogue management(sequence diagram)



Social and organisational factors

- Software systems are used in a social and organisational context. This can influence or even dominate the system requirements
- Social and organisational factors are not a single viewpoint but are influences on all viewpoints
- Good analysts must be sensitive to these factors but currently no systematic way to tackle their analysis

Ethnography

- A social scientists spends a considerable time observing and analysing how people actually work
- People do not have to explain or express their work clearly
- Social and organisational factors of importance may be observed
- Ethnography helps to discover implicit system requirement which reflect the actual people work
- Ethnographic studies have shown that work is usually richer and more complex than suggested by simple system models

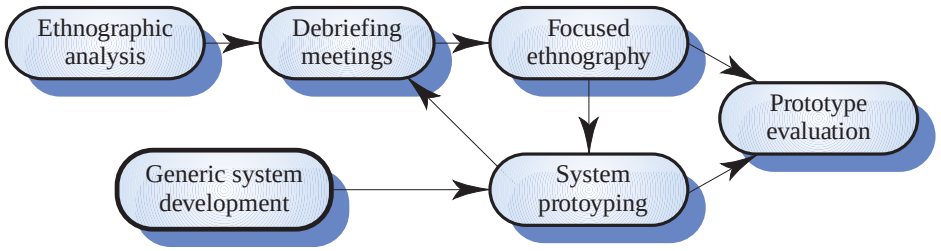
Example

- Consider a system which allows senior management to access information without going through middle managers in an OA system
  - Managerial status. Senior managers may feel that they are too important to use a keyboard. This may limit the type of system interface used
  - Managerial responsibilities. Managers may have no uninterrupted time where they can learn to use the system
  - Organisational resistance. Middle managers who will be made redundant may deliberately provide misleading or incomplete information so that the system will fail

Focused ethnography

- Developed in a project studying the air traffic control process
- Combines ethnography with prototyping
- Prototype development results in unanswered questions which focus the ethnographic analysis
- **Problem** with ethnography is that it **studies existing practices** which may have some historical basis which may be no longer relevant

Ethnography and prototyping



Scope of ethnography

- Ethnography is particularly effective at discovering 2 types of requirements:
  - Requirements that are derived from the way that **people actually work** rather than the way which process definitions suggest that they ought to work
  - Requirements that are derived from cooperation and awareness of other people's activities

Requirements validation

- Concerned with demonstrating that the requirements define the system that the customer really wants
- Requirements error costs are high so validation is very important
  - Fixing a requirements error after delivery may cost up to 100 times the cost of fixing an implementation error

Requirements checking

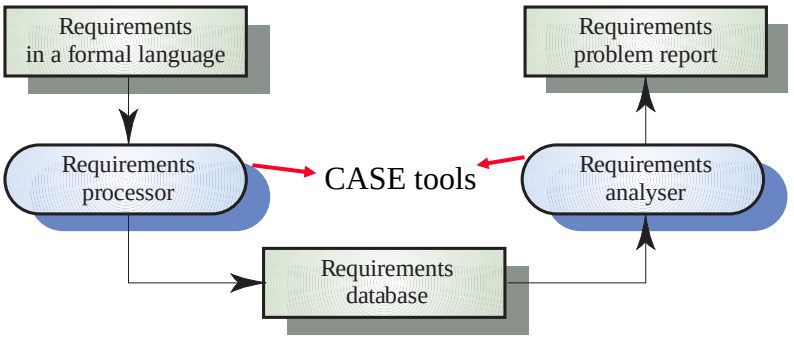
During requirement validation process, different types of checks should be carried out on the requirement:

- **Validity**. Does the system provide the functions which best support the customer's needs?
- **Consistency**. Are there any requirements conflicts?
- **Completeness**. Are all functions required by the customer included?
- **Realism**. Can the requirements be implemented in given available budget and technology
- **Verifiability**. Can the requirements be checked?

Requirements validation techniques

- Requirements reviews
  - The requirements are analysed by a team of reviewer
- Prototyping (evolutionary or throw-away)
  - Using an executable model of the system to check requirements is demonstrated to end-user and customer. Covered in Chapter 8
- Test-case generation
  - Developing tests for requirements to check testability
- Automated consistency analysis
  - Checking the consistency of a structured requirements description by CASE tools

Automated consistency checking



Requirements reviews

- Regular reviews should be held while the requirements definition is being formulated
- Both client and contractor staff should be involved in reviews to check for omission and anomalies
- Reviews may be formal (with completed documents) or informal. Good communications between developers, customers and users can resolve problems at an early stage

Review checks

Reviewers check for :

- **Verifiability**. Is the requirement realistically testable?
- **Comprehensibility**. Is the requirement properly understood by the end-user?
- **Traceability**. Is the origin of the requirement clearly stated?
- **Adaptability**. Can the requirement be changed without a large impact on other requirements?

Conflicts, contradiction, errors and omissions in the requirement should be pointed out during the review and formally recorded

Requirements management

- Requirements management is the process of **managing changing requirements** during the requirements engineering process and system development
- Developer's understands the problem is constantly changed and fed back to the requirement
- Requirements are inevitably incomplete and inconsistent
  - New requirements emerge during the process as business needs change and a better understanding of the system is developed
  - Different viewpoints have different requirements and these are often contradictory

Requirements change

Reasons for end-users to emerge new requirements:

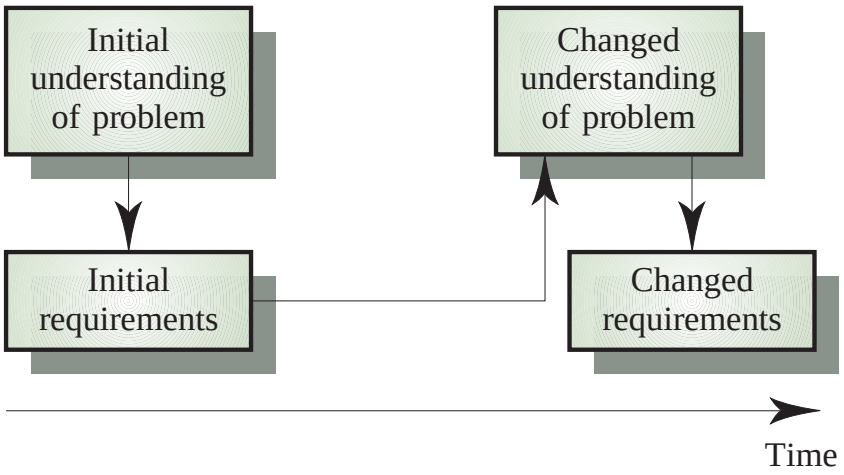
- The priority of requirements from different viewpoints changes during the development process
- System customers may specify requirements from a business perspective(under the constraints of budget) that conflict with end-user requirements
- The business and technical environment of the system changes during its development(new HW and business environment)

Enduring and volatile requirements

From an evolution perspective, requirement fall into 2 classes:

- **Enduring requirements**. Stable requirements derived from the core activity of the customer organisation. E.x. a hospital will always have doctors, nurses, etc. May be derived from domain models
- **Volatile requirements**. Requirements which change during development or when the system is in use. Ex. A hospital, volatile requirements derived from health-care policy of government

Requirements evolution



Classification of volatile requirements

- Mutable requirements
  - Requirements that change due to the system's environment
- Emergent requirements
  - Requirements that emerge as the customer's understanding of the system develops
- Consequential requirements
  - Requirements that result from the introduction of the computer system
- Compatibility requirements
  - Requirements that depend on particular systems or organisational processes

Requirements management planning

- During the requirements management stage, you have to plan:
  - Requirements identification
    - » How requirements are individually identified(for cross-reference with other requirements)
  - A change management process
    - » The activities which assess the impact and cost changes
  - Traceability policies
    - » The records of relationships between requirements and how these records should be maintained
  - CASE tool support
    - » The tool support to help manage requirements change

Traceability

Traceability is concerned with the relationships between requirements, their sources and the system design. 3 types of traceability information may be maintain:

- Source traceability
  - Links from requirements to stakeholders who proposed these requirements
- Requirements traceability
  - Links between dependent requirements
- Design traceability
  - Links from the requirements to the design module

Traceability information is represented using **traceability matrices** which relate requirement to stakeholders, each other or design module

A traceability matrix

Req. id	1.1	1.2	1.3	2.1	2.2	2.3	3.1	3.2
1.1		U	R					
1.2			U			R		U
1.3	R			R				
2.1			R		U			U
2.2								U
2.3		R		U				
3.1								R
3.2							R	

U: the row uses the facilities specified in the column name
R: there is some other weaker relationship between the requirements

CASE tool support

CASE tools support for requirement managements:

- Requirements storage ➔ repository
  - Requirements should be managed in a secure, managed data store
- Change management ➔CM tool
  - The process of change management is simplified if active tool support is available
- Traceability management
  - Automated retrieval of the links between requirements

Requirements change management

- Should apply to all proposed changes to the requirements
- 3 Principal stages to a change management process:
  - Problem analysis and change spec. Discuss requirements problem and the change proposal is analysed to check it valid or not
  - Change analysis and costing. Assess effects of change using traceability information and general knowledge of the system requirements
  - Change implementation. Modify requirements document and other documents to reflect change

Requirements change management

Key points

- The requirements engineering process includes a feasibility study, requirements elicitation and analysis, requirements specification and requirements management
- Requirements analysis is iterative involving domain understanding, requirements collection, classification, structuring, prioritisation and validation
- Systems have multiple stakeholders with different requirements

Key points

- Social and organisation factors influence system requirements
- Requirements validation is concerned with checks for validity, consistency, completeness, realism and verifiability
- Business changes inevitably lead to changing requirements
- Requirements management includes planning and change management

HomeWork

- 6.2
- 6.3
- 6.5
- 6.8
- Propose your project's VP by brainstorming to identify the Viewpoints and corresponding services!
- Propose your project's Use case and interactions analysis! (use-case and sequence diagrams)