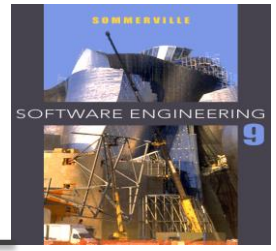
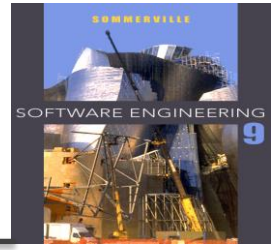


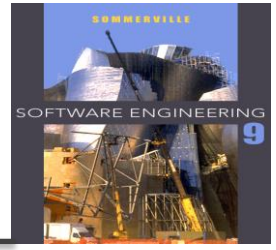
Chapter 2 – Requirements Engineering

Topics covered



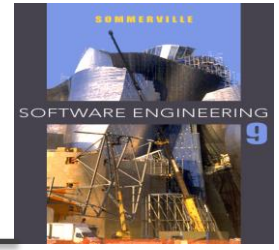
- ✧ Requirement Engineering
- ✧ Crucial Process Steps of Requirement Engineering
- ✧ Types of Requirements
- ✧ User and System Requirements
- ✧ Categories of Metrics
- ✧ Software Requirements Specification (SRS) Document
- ✧ Requirements Gathering Techniques

2.1 Requirements Engineering



- The process of establishing the services that the customer requires from a system and the constraints under which it operates and is developed.
- The requirements themselves are the descriptions of the system services and constraints that are generated during the requirements engineering process.
- Requirements engineering is one of the most crucial activity in this creation process. Without well-written requirements specifications, developers do not know what to build, customers do not know what to expect, and there is no way to validate that the built system satisfies the requirements.

2.2 Crucial Process Steps of Requirement Engineering

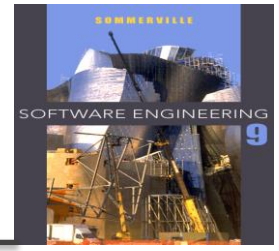


✧ The requirements engineering consists of four steps

Process Step	Description
Requirements elicitation	This is also known as gathering of requirements. Here requirements are identified with the help of customer and existing systems process if available.
Requirements analysis	The requirements are analyzed in order to identify inconsistencies, defects, omissions, and also resolve conflicts if any.
Requirements documentations	It is the foundation for the design of the software. The document is known as software requirements specification (SRS).
Requirements review	The review process is carried out to improve the quality of the SRS.

✧ The primary output of requirements engineering is requirements specifications. If it describes both hardware and software, it is a system requirements specifications. If it describes only software, it is a software requirements specifications.

Crucial Process Steps of Requirement Engineering



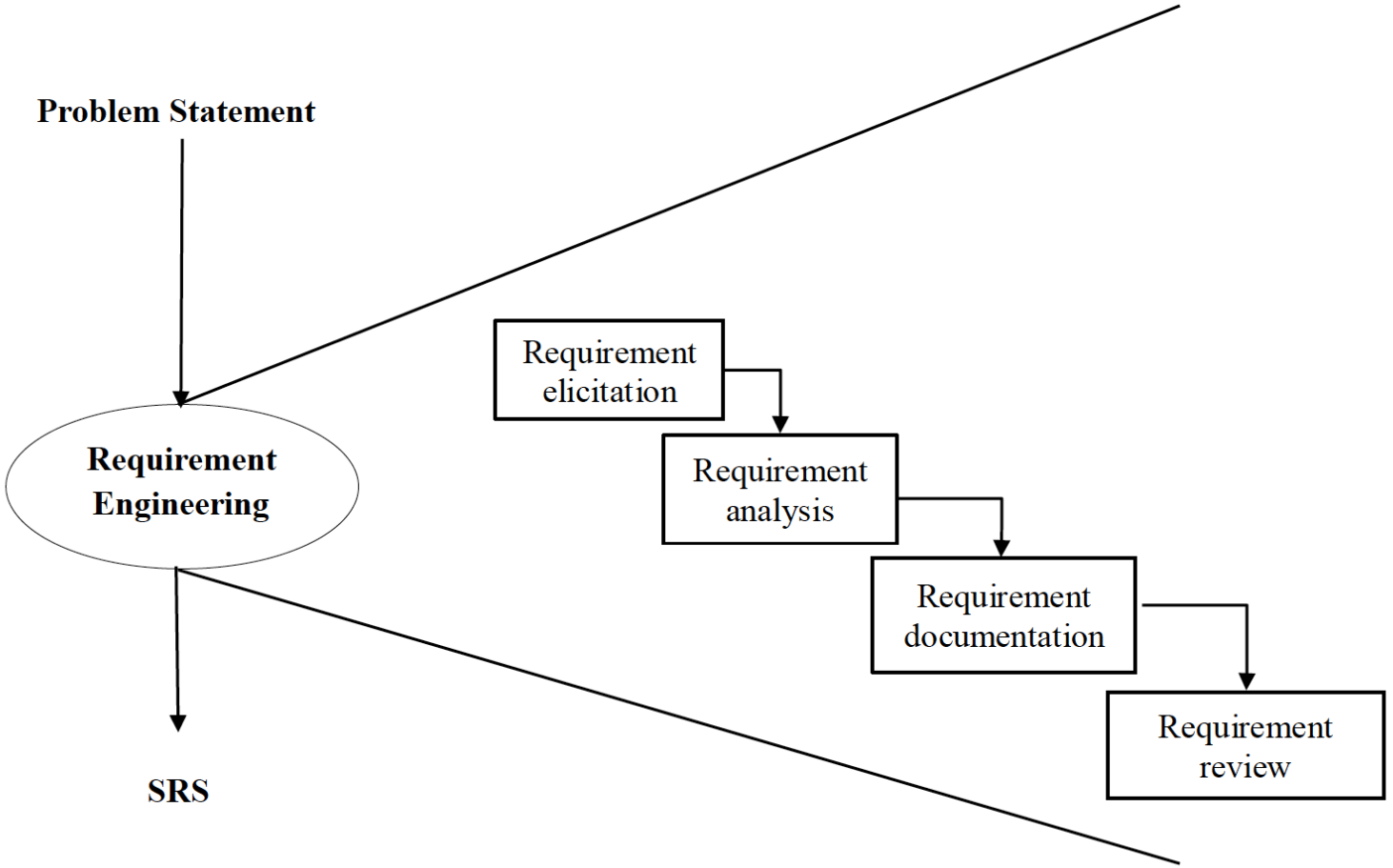
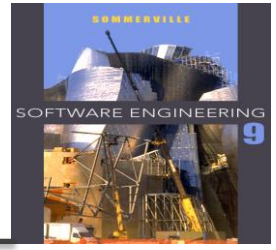


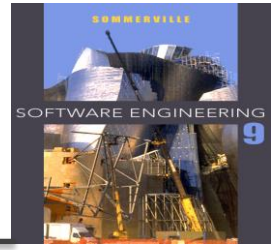
Figure 2.1: Crucial Process Steps of Requirement Engineering

2.3 Types of Requirements



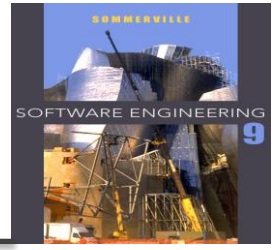
- ✧ Functional requirements
  - Statements of services the system should provide, how the system should react to particular inputs and how the system should behave in particular situations.
  - May state what the system should not do.
- ✧ Non-functional requirements
  - Constraints on the services or functions offered by the system such as timing constraints, constraints on the development process, standards, etc.
  - Often apply to the system as a whole rather than individual features or services.

Functional Requirements



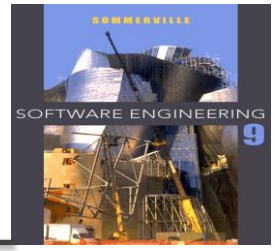
- Describe functionality or system services.
- Depend on the type of software, expected users and the type of system where the software is used.
- Functional user requirements may be high-level statements of what the system should do.
- Functional system requirements should describe the system services in detail.

Non-Functional Requirements



- These define system properties and constraints e.g. reliability, response time and storage requirements. Constraints are I/O device capability, system representations, etc.
- Process requirements may also be specified mandating a particular IDE, programming language or development method.
- Non-functional requirements may be more critical than functional requirements. If these are not met, the system may be useless.

Non-Functional Classifications



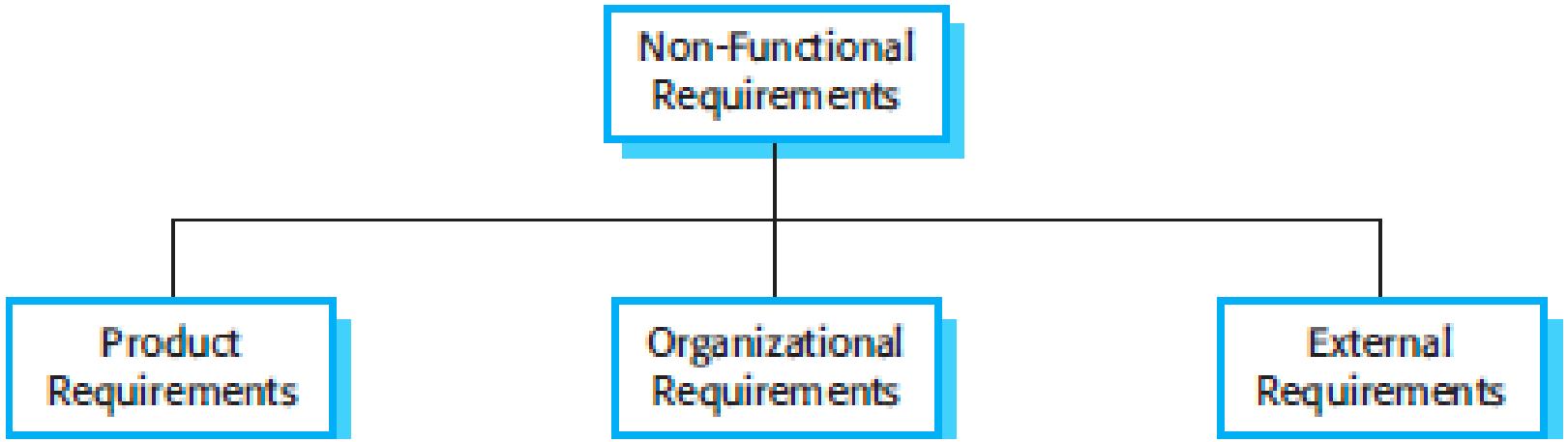
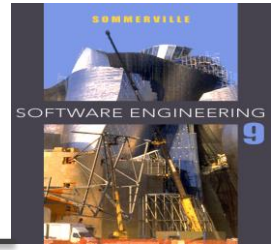


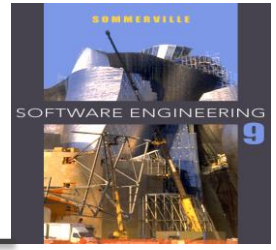
Figure 2.2: Types of Non-Functional Requirement

Non-Functional Classifications



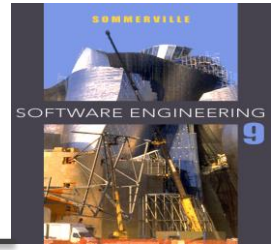
- ✧ Product requirements
  - Requirements which specify that the delivered product must behave in a particular way e.g. execution speed, reliability, etc.
- ✧ Organisational requirements
  - Requirements which are a consequence of organisational policies and procedures e.g. process standards used, implementation requirements, etc.
- ✧ External requirements
  - Requirements which arise from factors which are external to the system and its development process e.g. interoperability requirements, legislative requirements, etc.

2.4 User and System Requirements



- ***User requirement*** are written for the users and include functional and non-functional requirement. User requirements should specify the external behavior of the system with some constraints and quality parameters.
- ***System requirement*** are derived from user requirement. They are expanded form of user requirements.
  - ***A measure*** provides a quantitative indication of the extent, amount, dimension, capacity, or size of some attribute of the product or process".
  - ***Measurement*** is the act of determine a measure.
  - The ***metric*** is a quantitative measure of the degree to which a system, component, or process possesses a given attribute.

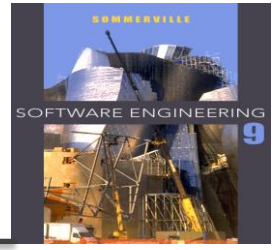
2.5 Categories of Metrics



✧ **Product metrics**: describe the characteristics of the product such as size, complexity, design features, performance, efficiency, reliability, portability, etc.

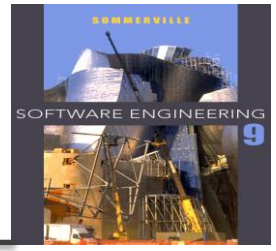
Property	Measure
Speed	Processed transactions/second User/event response time Screen refresh time
Size	Mbytes Number of ROM chips
Ease of use	Training time Number of help frames
Reliability	Mean time to failure Probability of unavailability Rate of failure occurrence Availability
Robustness	Time to restart after failure Percentage of events causing failure Probability of data corruption on failure
Portability	Percentage of target dependent statements Number of target systems

Categories of Metrics



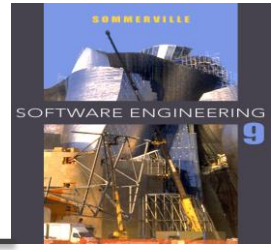
- **Process metrics**: describe the effectiveness and quality of the processes that produce the Software product. Examples are:
  - Effort required in the process
  - Time to produce the product
  - Effectiveness of defect removal during development
  - Number of defects found during testing
  - Maturity of the process
- **Project metrics:** describe the project characteristics and execution. Examples are:
  - Number of software developers
  - Staffing pattern over the life cycle of the software
  - Cost and schedule
  - Productivity

2.6 Software Requirements Specification (SRS) Document



- The SRS is a specification for a particular s/w product, program, or set of programs that performs certain functions in a specific environment.
- The SRS serve as contract document between customer and developer. SRS reduces the probability of the customer being disappointed with the final product.
- **Nature of the SRS**: The basic issues of that SRS writer(s) shall address the following:
    1. Functionality
    2. External interface
    3. Performance
    4. Attributes
    5. Design constraints imposed on an implementation

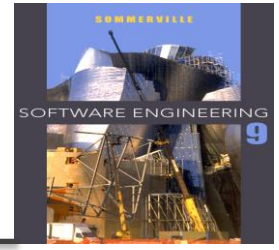
Characteristics of a Good SRS



✧ The SRS should be:

- Correct
- Unambiguous
- Complete
- Consistent
- Rank for importance and/ stability
- Verifiable
- Modifiable
- Traceable

Organization of the SRS



- The (IEEE) has published guidelines and standards to organize an SRS document.
- Different projects may require their requirements to be organized differently but still first two sections of the SRS are the same in all of them.

1. Introduction
- 1.1 Purpose
- 1.2 Scope
- 1.3 Definition, Acronyms and Abbreviations
- 1.4 References
- 1.5 Overview

2. The Overall Description
- 2.1 Product Perspective
  - 2.1.1 System Interfaces
  - 2.1.2 Hardware Interfaces
  - 2.1.3 Software Interfaces
  - 2.1.4 Communication Interfaces
  - 2.1.5 Memory Constraints
  - 2.1.6 Operations
  - 2.1.7 Site Adaptation Requirements
- 2.2 Product Functions
- 2.3 User Characteristics
- 2.4 Constraints
- 2.5 Assumptions for Dependencies
- 2.6 Apportioning of Requirements

3. Specific Requirements
- 3.1 External Interfaces
- 3.2 Functions
- 3.3 Performance Requirements
- 3.4 Logical Database Requirements
- 3.5 Design Constraints
- 3.6 Software System Attributes
  - 3.6.1 Reliability
  - 3.6.2 Availability
  - 3.6.3 Security
  - 3.6.4 Maintainability
  - 3.6.5 Portability
- 3.7 Organization of Specific Requirements
  - 3.7.1 System Mode
  - 3.7.2 User Class
  - 3.7.3 Objects
  - 3.7.4 Feature
  - 3.7.5 Stimulus
  - 3.7.6 Response
  - 3.7.7 Functional Hierarchy
- 3.8 Additional Comments.

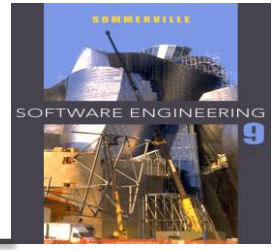
4. Change Management Process

5. Document Approvals

6. Supporting Information

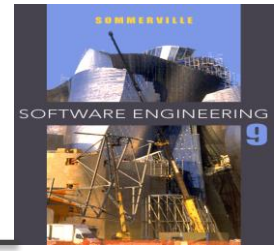
The template to organize and draft the SRS for any project.

2.7 Requirements Gathering Techniques



- ✧ Some requirements gathering techniques may be beneficial in one project but may not be in other.
- ✧ The usefulness of a technique is determined by its need and the kind of advantages.
- ✧ Following are some popular requirements gathering techniques:

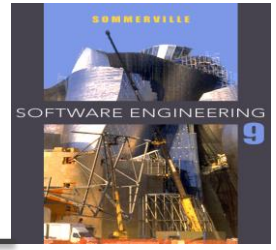
Brainstorming



Description

Brainstorming is human nature to solve any problem as an early thought process. It can be utilized to gather a good number of ideas from a group of people by sharing ideas to identify all possible solutions.

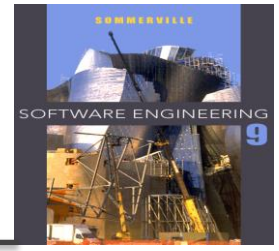
Document Analysis



Description

Usually followed where a system is already in place, so evaluating the documentation of a present system can assist to gather requirements for updating or replacing existing system.

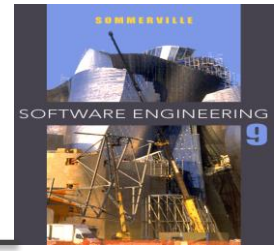
Focus Group



Description

A focus group is a gathering of people who are customers or user representatives for a product to gain its feedback. The feedback can be collected about opportunities, needs, and problems to determine requirements or it can be collected to refine and validate the already elicited requirements.

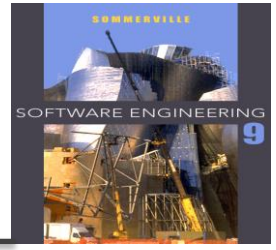
Interface Analysis



Description

Interface for any software product are either be human or machine. Integration with external devices and systems is another interface. The user-centric design approaches are quite effective to collect and develop usable software.

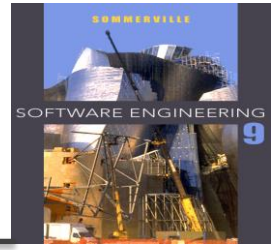
Interview



Description

Interviews of users and stakeholders are important in creating wonderful software. Without knowing the expectations and goal of the stakeholders and users it is impossible to satisfy them. To understand the perspective of every interviewee it is important to properly collect their inputs. Like a good reporter, listening is a quality that assists an excellent analyst to gain better value through an interview.

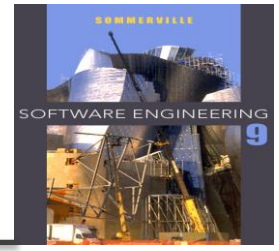
Observation



Description

The observation covers the study of users in their workplace. By watching users, a process flow, pain points, awkward steps and opportunities can be determined by an analyst for improvement. Observation can either be passive or active. Passive observation provides better feedback to refine requirements on the same hand active observation works best for obtaining an understanding over an existing business process.

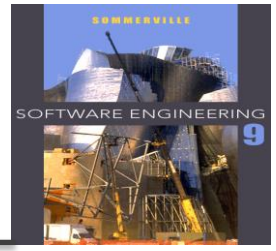
Prototyping



Description

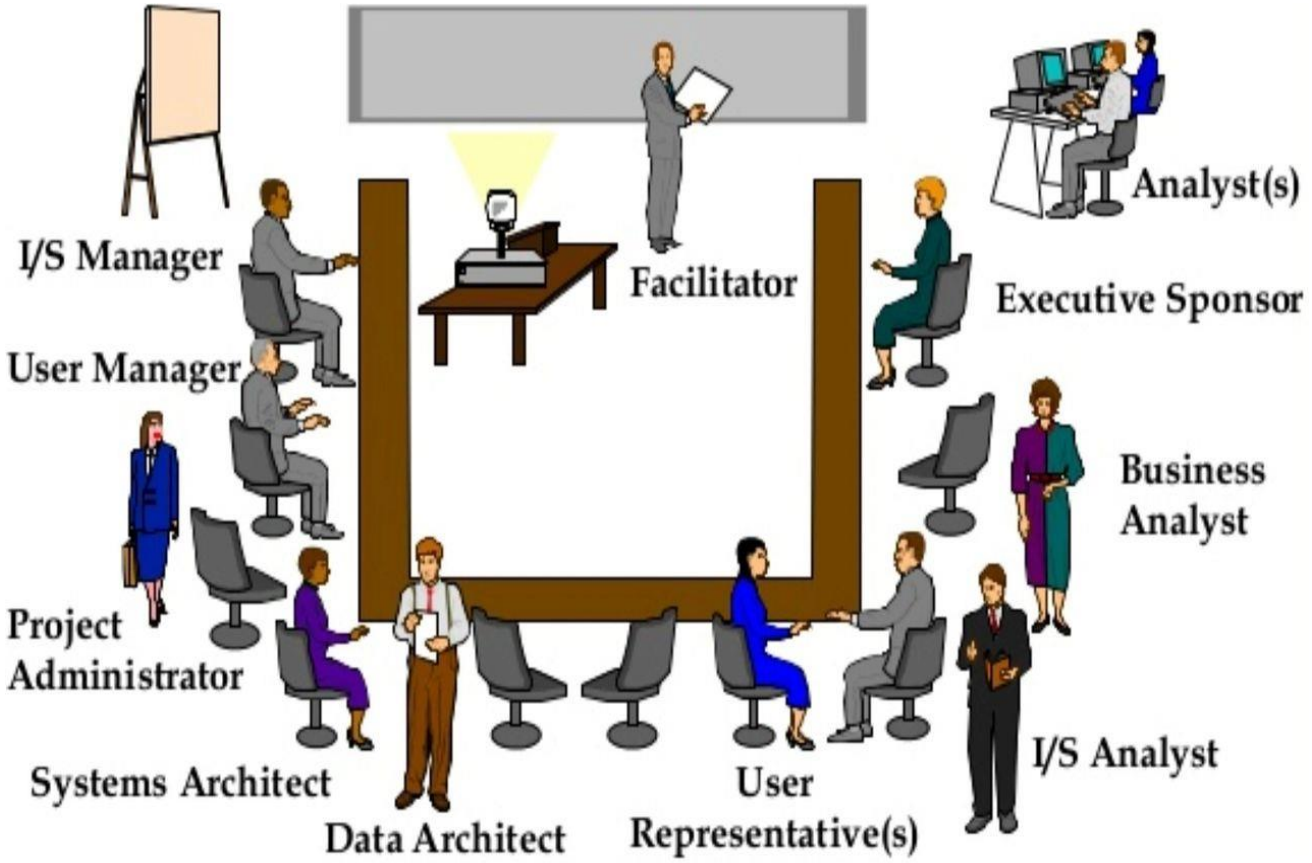
Prototyping can be very helpful at gathering feedback. Low fidelity prototypes make a good listening tool. Many a times, people are not able to articulate a specific need in the abstract. They can swiftly review whether a design approach would satisfy their need. Prototypes are very effectively done with fast sketches of storyboards and interfaces.

Requirements Workshop

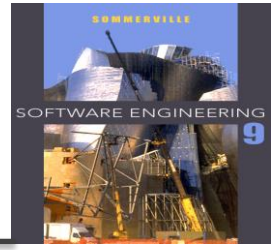


Description

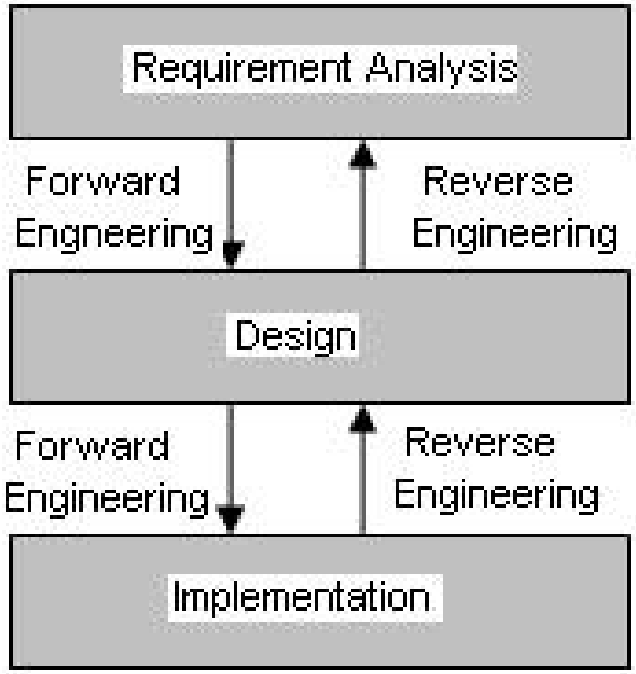
Popularly known as JAD or joint application design, these workshops can be efficient for gathering requirements. The requirements workshops are more organized and structured than a brainstorming session where the involved parties get together to document requirements.



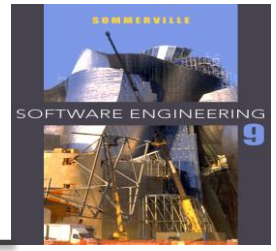
Reverse Engineering



Description
When a legacy project does not have enough documentation, reverse engineering can determine what system does? It do not determine what features went wrong with the system and what a system must do.

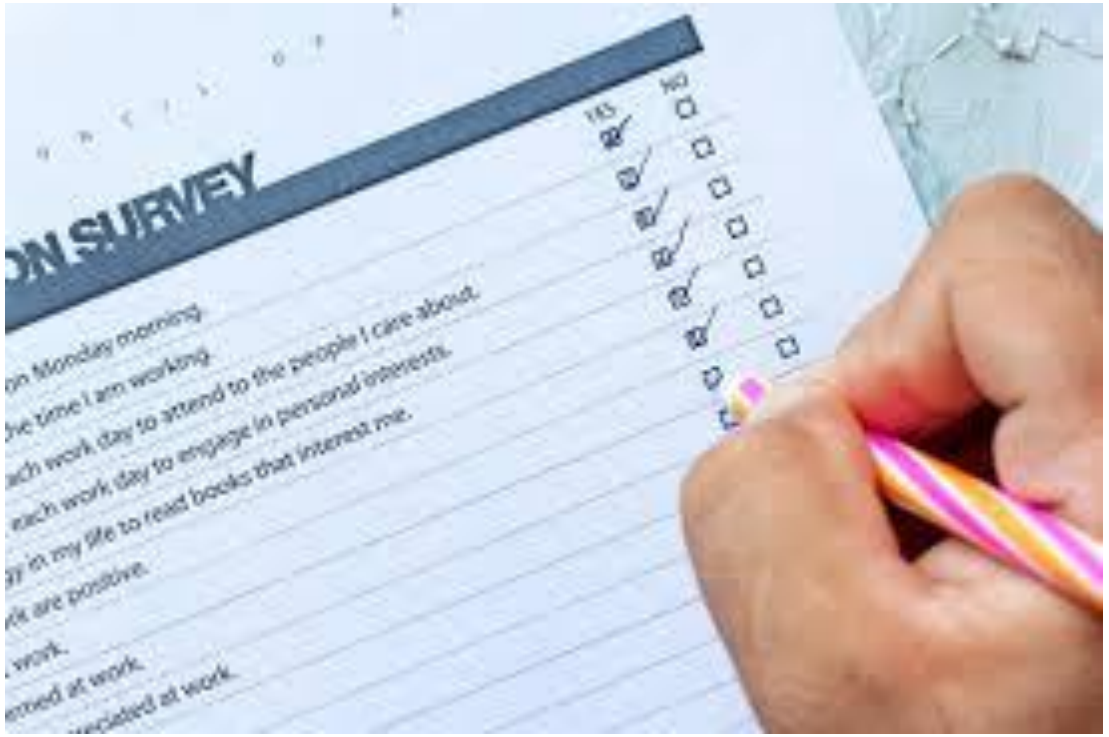


Survey

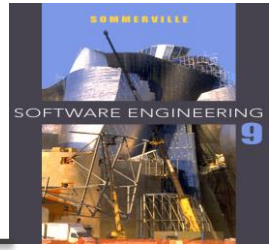


Description

When gathering information from many people: too many to interview with time constraints and less budget: a questionnaire survey can be used. The survey insists the users to choose from the given options agree / disagree or rate something.

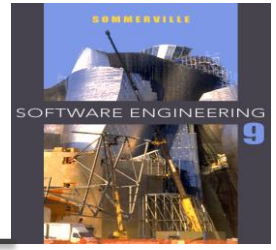


Key points



- Requirements for a software system set out what the system should do and define constraints on its operation and implementation.
- Functional requirements are statements of the services that the system must provide or are descriptions of how some computations must be carried out.
- Non-functional requirements often constrain the system being developed and the development process being used.
- They often relate to the emergent properties of the system and therefore apply to the system as a whole.

Review Questions



1. Define: requirements engineering, measure, measurement, metrics, SRS
2. Illustrate crucial process steps of requirement engineering with neat diagram
3. Differentiate Functional and Non-functional Requirements
4. Explain different classifications of Non-functional requirements
5. What are the different categories of metrics?
6. What are the basic issues addressed by SRS writers
7. List down the characteristics of good SRS
8. Explain various requirement gathering techniques.