ICS 2302 – Software Engineering

BSc. Computer Science

Third Year, First Semester

Dept of Computer Science, JKUAT

Lecturer: Joan Gichuru

Course Description

- Requirements Specification, data and functional specifications, modularity, design, verification and validation, maintenance, documentation, project management and quality assurance, CASE tools.

Software Engineering - Introduction

- **Defn:**
- S/W Engineering is an engineering discipline concerned with all aspects of s/w production from the early stages of system specification through to maintaining the system after it has gone to use

Engineering discipline –

Apply theories, methods and tools where appropriate to discover solutions to problems

Must work within organizational and financial constraints

All aspects of s/w production –

Not just technical processes of development,

Also S/w project management activities and dvpt of tools methods and theories to support s/w production

- **S/W Engineering is important since:**

i. More and more individuals and society rely on advanced software systems hence to produce reliable and trustworthy systems economically and quickly.

ii. It is cheaper in the long run to use s/w engineering methods and techniques rather than just write programs like in personal programming projects

- **Software Process**

Is a sequence of activities that leads to the production of a software product

- Common activities to all s/w processes:

1. Software specification – customers and engineers define s/w to be produced
2. Software development- s/w is designed and programmed
3. Software validation – s/w is checked to ensure it meets customers requirements
4. Software evolution – s/w is modified to reflect changing customer and market requirements

- There is no universal software engineering method or technique applicable to all types of software.

However, there are general issues that affect many types of software:

❖ Heterogeneity – flexibility to cope with diff platforms, networks, distributed sys

❖ Business and social change – s/w need to evolve as business and society changes

❖ Security and trust – s/w need be trusted esp remote applications, guard information security against malicious attacks

- **Some types of software applications**

1) Stand-alone applications – run on a local pc
2) Interactive transaction based applications – execute on a remote computer and accessed by users from their own PCs or terminals
3) Embedded control systems – control and manage hardware devices
4) Batch processing systems- process data in large batches
5) Entertainment systems- personal user for entertaining like games
6) Simulation and modeling systems – model physical processes and situations
7) Data collection systems –collect data from environment/sensors and send to other systems for processing
8) Systems of systems – composed of a number of other systems

SOFTWARE PROCESSES

- Software Process

Is a simplified representation of a software process

- Models to cover(covered in SAD)

1) Waterfall model
2) Incremental development
3) Reuse-oriented software engineering

Process Activities:

1) Software Specification

2) Software development

3) Software validation

4) Software evolution

1. Software specification

- Also called requirements Engineering

- Is the process of understanding and defining what services are required from the system and identifying the constraints on the systems operation and development.

- Aims to produce an agreed requirements document that specifies a system stakeholder requirements.

- There are four main activities in requirements engineering process

1. Feasibility study

An estimate of whether the identified user needs may be satisfied using current software and hardware technologies

Whether the proposed system will be cost effective business wise

Its results inform the decision of whether to go ahead with a more detailed analysis

2. Requirements elicitation and analysis

Process of deriving the system requirements through observation of existing systems, discussions with potential users and procurers, task analysis

May involve dvpt of a prototype to better understand the system to be specified

3. Requirements Specification

- Translating the information gathered during the analysis stage into a document that defines a set of requirements.
- Two kinds of requirements may be included:

a. User requirements – abstract statements of the system requirements for the customer or end user

b. System requirements – more detailed description of the functionality to be provided

4. Requirements validation.

Checks requirements for realism, consistency and completeness.

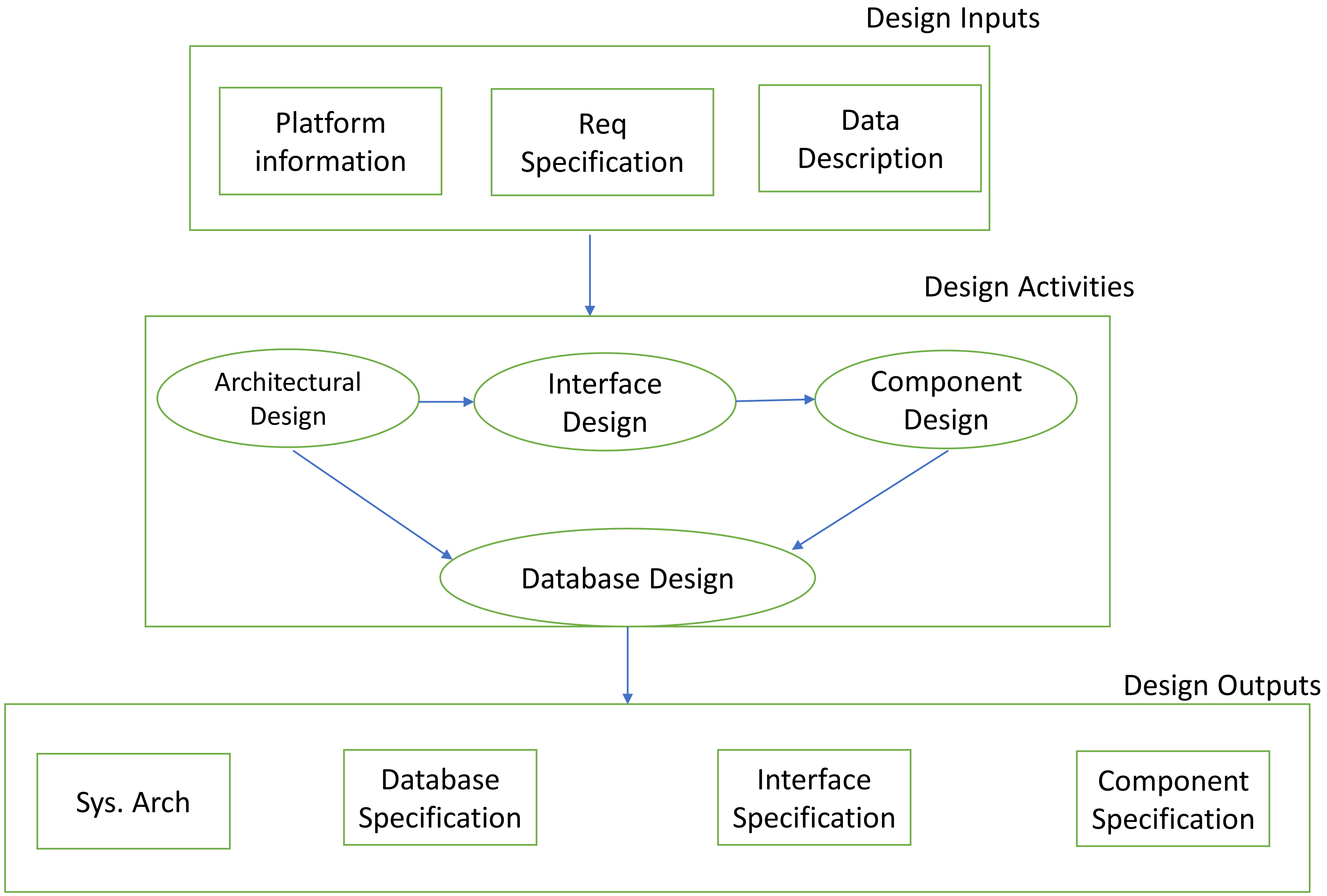
Errors in the req document are identified and rectified

2. Software Design and Implementation

- Process of converting a system specification into an executable system.

- A software design is a description of the structure of the software to be implemented, the data models and structures used by the system, the interface between system components and algorithms used.

- It is developed iteratively.


Design Inputs
Platform information
Req Specification
Data Description
Design Activities
Architectural Design
Interface Design
Component Design
Database Design
Design Outputs
Sys. Arch
Database Specification
Interface Specification
Component Specification

Architectural design

We identify the overall structure of the system, the principal components(modules or subsystems), their relationships and how they are distributed

Interface design

Define interfaces between system components

Should be unambiguous

Once interface specification are agreed, components can be designed and developed concurrently

Component design

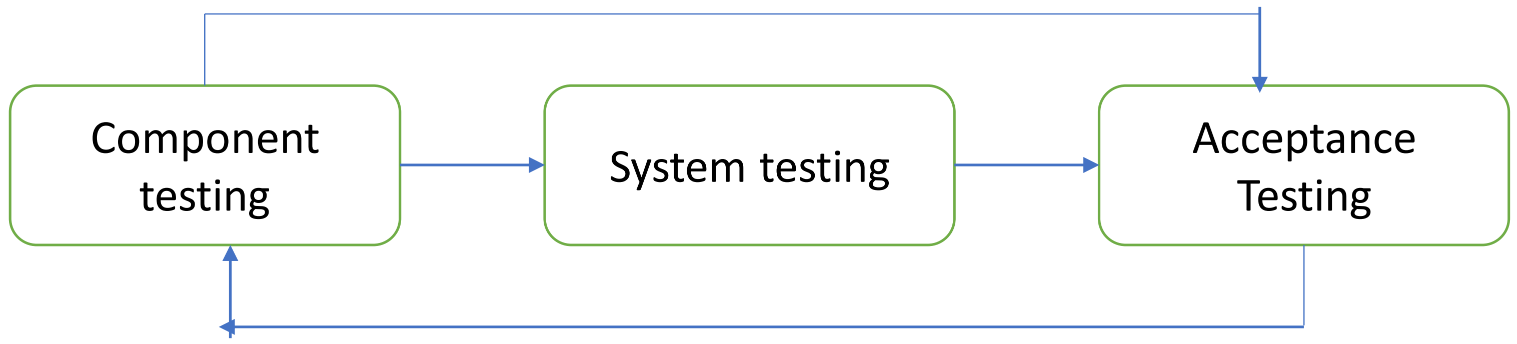
Each system component is designed how it will operate, or how expected functionality will be implemented. Mainly for the programmer

Database design

Design system data structures and how to be represented in a database

3. Software Validation

- Also called Verification and validation
- To show that a system conforms to its specifications and that it meets the expectations of the system customer.
- Program testing is a key activity in validation.
- May also involve checking processes like inspections and reviews at each stage of program dvpt.



Stages of testing

- Systems are not tested as a single unit.
- Testing process is an iterative one

<u>Stages of testing</u>

Development testing –

- Components of the system are tested by those making them. (functions, objects etc)
- Each component is tested independently

System testing –

System components are integrated to create a complete system

Targets finding errors that result from not anticipated interactions btn components and component interface problems.

Also concerned with showing that the system meets its functional and non-functional requirements

- **Acceptance testing**
- The system is tested with data supplied by the customer rather than simulated test data
- May reveal errors and omissions in the system requirements definition
- May also reveal requirements problems where the system facilities do not really meet the users needs or the sys performance is unacceptable.
- NB:
- For a single client or custom systems, sometimes acceptance testing is called alpha testing – it continues until developer and customer are in agreement of acceptable implementation.
- For system to be marketed as a software product, acceptance testing is called Beta testing –delivering the system to a number of potential customers who agree to use that system, they report problems to the developers. The feedback is used to modify before release.

4. Software evolution

- Handles maintenance after a product has already been deployed to evolve with business needs or market needs

REQUIREMENTS ENGINEERING

Definitions:

- **Requirements** of a system are the descriptions of what the system should do – the services that it provides and the constraints on its operation.
- They reflect the needs of the customers for a system, serving a particular purpose.

- **Requirements engineering**
- The process of finding out, analyzing, documenting and checking these services and the constraints therein.

Functional requirements

- These are statements of services the system should provide, how the system should react to particular inputs and how the system should behave in particular situations.
- They describe what the system should do.
- They depend on the type of software being developed, the expected users of the s/w, and the approach taken by the org to write requirements.

- Imprecision in requirements specification.

Non – functional requirements

- These are constraints on the services or functions offered by the system.
- They include timing constraints, development process constraints, and constraints imposed by standards.

- They arise through user needs, budget constraints, organizational policies, and the need for interoperability with other software or hardware.

Classification of non-functional requirements

- Product req – specify the behavior of the software e.g. performance, reliability, security, usability.
- Organizational req – derived from policies and procedures in the customers or developers organization e.g. operational process req of how the system will be used, development process req that specify the programming language or process standards to be used, environmental req that specify the operating environment of the system
- External requirements – derived from factors external to the system and its development process e.g. regulatory req, legislative req, ethical req

SOFTWARE REQUIREMENTS DOCUMENT

- Is an official statement of what the system developers should implement

- It should include both the user requirements for a system and a detailed specification of the system requirements. (Sometimes the two are integrated into one)

- Sometimes the user req are defined as an introduction to the system req specification.

Users of Requirements document

1) System customers – they specify the req and read then to check if they meet their needs. They also specify changes to the req.
2) Managers – Use the req document to plan a bid for the system and plan the sys development process.
3) System Engineers – Use the req to understand what the system is to be developed
4) System test engineers –use the req to develop validation tests for the system
5) System maintenance Engineers – use the req to understand the system and relationships between its part.

Structure sample (IEEE)

Introduction

- describes the need for the system.
- Briefly describe the system’s functions and how it will work with others
- Describe how the system fits into the overall business objectives of the org.

Glossary

- Define the technical terms used in the document

User requirements definition

- Describe the services provided for the user
- Non functional requirements too
- Product and process standards to be followed should be specified.

System architecture

- Present a high-level overview of the anticipated system architecture showing the distribution of functions across modules

System requirements specification

- Describe the functional and non-functional req in more detail
- Interfaces to other systems may also be defined

System models

Graphical models showing relationships between the system components, the system, and its environment e.g. data flow models, object models

System evolution

- Fundamental assumptions on which the system is based
- Anticipated changes due to hardware evolution, changing user needs, etc

Appendices – detailed information about the system being developed e.g. hardware description

Index – index of diagrams, functions etc

Requirements specification

- Process of writing down the user and system requirements in a req document.
- Should be clear, unambiguous, easy to understand, complete and consistent

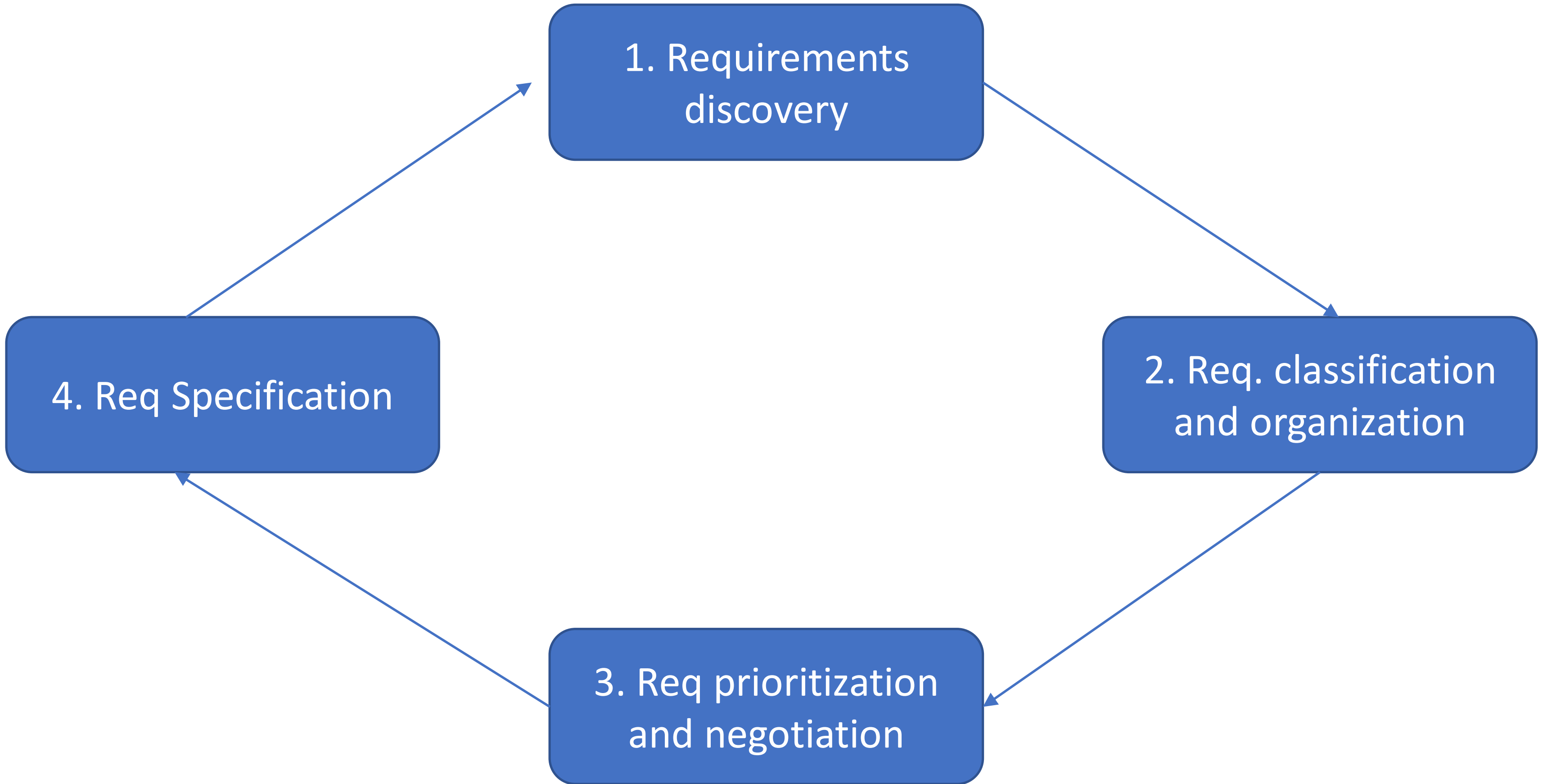
Ways of writing system req:

1. Natural language sentences – numbered sentences in natural language, each representing one requirement
2. Structured natural language – written in natural language on a standard template or form
3. Graphical notations – graphical models with text annotations
4. Mathematical specifications – based on mathematical concepts like finite state machines or sets

Requirements engineering processes

1. Feasibility study

2. Requirements elicitation and analysis


1. Requirements discovery
2. Req. classification and organization
3. Req prioritization and negotiation
4. Req Specification

Requirements discovery

- **Requirements discovery:** Interacting with system stakeholders to discover their requirements. A system stakeholder is anyone who will have some direct or indirect influence on the system requirement.
- **Requirements classification and organization** – takes the unstructured collection of requirements, groups related requirements and organizes them into coherent clusters. involves identifying sub-systems and the req associated with them.
- **Requirements prioritization and negotiation** – Req conflict when multiple stakeholders are involved. This step prioritizes req and resolves req conflicts thro negotiation. They meet and agree on compromise req.
- **Requirements specification** – req are documented and input into the next spiral

Eliciting and understanding req. from stakeholders is a difficult process for the following reasons:

1) Stakeholders often don’t know what they want from a computer system except in general terms.
2) They express req. in their terms; the domain may be difficult to understand for system engineers
3) Different stakeholders have different requirements
4) Political factors may influence a system’s requirements
5) Dynamic business and economic environment

Ways of gathering requirements:

- **Interviewing** – closed and open interviews
- Not suitable to elicit domain knowledge because
- Application specialists use jargon and terminology specific to a domain
- Some domain knowledge is so familiar to stakeholders that they find it too basic and not worth mentioning while it's not obvious to the interviewer
- **Scenarios** – real-life examples and how they might interact with a software system. Scenarios may be written as text, diagrams, screenshots etc
- **Use-cases** –they identify the actors involved in an interaction and name the type of interaction
- **Ethnography** – analyst immerses himself in the working environment where the system will be used, observation technique of day to day activities, note taking

3. Requirements validation

- Process of checking that requirements actually define the system that the customer really wants. It overlaps with analysis since its about finding problems with req.
- Different types of checks are carried out on the requirements in the req document:

1) Validity checks – additional or different functions may be obtained other than those initially thought about
2) Consistency checks – Req should not conflict. There should not be contradictory descriptions of the same system function.
3) Completeness checks – document should define all functions and constraints intended by the system user
4) Realism checks – can the req actually be implemented? – cost , budget, technology
5) Verifiability – write a set of tests that can demonstrate that the delivered system meets each req.

Requirements validation techniques:

1) Requirements reviews – the requirements are analyzed systematically by a team of reviewers who check for errors and inconsistencies
2) Prototyping – an executable model of the system is demonstrated to the end-users/customers, they experiment with it to see if it meets their real needs.
3) Test-case generation – tests are designed as part of the validation process….esp with extreme programming

4. Requirements management

- Is the process of understanding and controlling changes to system requirements.
- Involves keeping track of individual requirements and maintaining links between dependent requirements.
- Also involves establishing a formal process for making change proposals and linking them to system requirements.
- Req for large s/w systems are always changing.
- Sometimes because these systems are developed to solve '**wicked' problems** – problems that can not be defined, thus req are bound to be incomplete.

Change is inevitable in systems for the following reasons:

1) The business and technical environment of the system always changes after installation – new hardware maybe introduced, system may interface with other systems, business priorities may change, new legislation and regulations may be introduced etc

2) The people who pay for the system and the end-users of the system are rarely the same people – system customers impose req bue to organizational and budgetary constraints. These may conflict with end-user req and after delivery, new features have to be added for user support if the system is to meet its goals.

3) Large systems have a diverse user community, with many users having different req and priorities that may be conflicting or contradictory

Requirements management planning

- It’s the first stage of the req management process
- It establishes the level of req management detail that is required.
- The following have to be decided on:
- Requirements identification – each req must be uniquely identified so as to be cross-referenced with other req and used in the traceability assessments
- A change management process – the set of activities that assess the impact and cost of changes
- Traceability policies – they define the relationships between each requirement and between requirements
- Tool support-since large amounts of information about requirements' is involved, tools used range from spreadsheets, simple databases and specialist req mangt systems

- Req management needs automated support and the software tools involved should be chosen during planning phase.

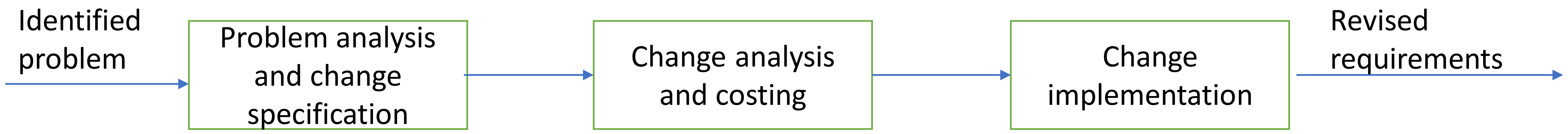
- Tools support is necessary for;

1) Requirements storage – should be maintained in a secure managed data store that is accessible to everyone involved in the req engineering process

2) Change management – to simplify and keep track of the process

3) Traceability management – to discover and show related req

Requirements change management


Identified problem
Problem analysis and change specification
Change analysis and costing
Change implementation
Revised requirements

- The process (in above fig) should be applied to all proposed changes to a system’s req after the req document has been approved.

- Necessary to decide if the benefits of implementing new requirements are justified by the costs of implementation

Key stages in the change management process:

1) Problem analysis and change specification – starts with an identified problem or change proposal, its analyzed for validity. The analysis is fed back to the change requestor who may withdraw the proposal or respond with more specific details to the change proposal

2) Change analysis and costing – the effect of the proposed change is assessed using traceability information, cost is estimated and decision to change or not is made

3) Change implementation – the req document , system design and implementation are modified

SYSTEM MODELING

- **System modeling** is the process of developing abstract models of a system where each model represents a different view or perspective of the system.
- Involves representing the system using some kind of graphical notation, UML is most commonly used notation.

- Different perspectives may be represented:
- Context models – environment of the system
- Interaction models – interaction between a system and its environment or between the components of a system
- Structural models – the organization of a system or structure of data processed by the system
- Behavioral models – the dynamic behavior of the system and how it responds to events

Context modeling:

- Involves specification of system boundaries
- A simple architectural model is the first step
- Context models normally show that the environment includes several other systems but do not show the relationships between the systems in the environment.
- External systems might produce data for or consume data from the system. Tey might share data with the system or might be connected directly with the system thro networks.
- **Activity diagrams** are commonly used to show context models.
- Activity diag are intended to show the activities that make up the system process and the flow of control from one activity to another.
- The start of a process is indicated by a filled circle, the end by a filled circle inside another circle. Rectangles with rounded corners represent activities.

Interaction models

- All systems involve interaction of some kind – user interaction, interaction btn system with other systems or interaction between system components.
- Two approaches commonly used;

1) Use case modeling – models interaction between a system and external actors(users and other systems)
2) Sequence diagrams – models interaction between system components, external agents may also be included.

Structural models

- Structural models of software display the organization of a system in terms of the components that make up the system and their relationships.

- Class diagrams

Behavioral models

- Data driven- activity
- Event driven – state diagram

SOFTWARE TESTING

- Testing -intended to show that a program does what it is intended to do and to discover program defects before its put to use.
- Testing process has two distinct goals:

1. To demonstrate to the developer and the customer that the software meets its requirements -(validation testing)
2. To discover situations in which behaviour of the software is incorrect or undesirable (defect testing)

- Testing is part of a broader process of software verification and validation:
- Validation - are we building the right product?
- verification - are we building the product right?

- THe ultimate goal of V&V is to establish confidence that the software is fit for purpose.
- The level of confidence required depends on:

1. Software purpose - The more critical the software, the more important that it is reliable. Confidence in a safety critical system is higher than a new product prototype.

2. User expectations - when a new system is installed users may tolerate failures because benefits outweigh the costs of failure . however as it matures, its expected to be more reliable.

3. Marketing environment - has to do with competing products, prices etc. low priced products may have lower level of reliability

- V&V process may also involve <u>software inspections and reviews</u>.
- These analyze and check the system requirements, design models, program code and proposed system tests.
- They are static V&V techniques where the code need not be executed.
- knowledge of the system , its application domain and the programming language are used to discover errors.
- **Advantages of software inspection over testing:**

1. Its a static process hence errors can not mask other errors like in testing
2. incomplete versions of the system can ne inspected without additional costs
3. inspections can consider a broader aspect like compliance with standards, maintainability, portability

Stages of testing:

- A commercial software has to gp through three stages of testing
- Development testing - system is tested during dvpt to discover bugs and defects
- Release testing - a separate testing team tests a complete sytsem before its released to users
- User testing -users or potential users of a system test the system in their own environment

Development testing

- Includes all testing activities that are carried out by the team developing the system.
- <u>During dvpt, testing may be carried out at three levels of granularity:</u>

1. Unit testing - individual program units or object classes are tested. Focuses on testing the functionality of methods or objects
2. Component testing - several individual units are integrated to create composite components. focus is on testing component interfaces.
3. System testing - all components in a system are integrated and the system is tested as a whole.

- Release testing
- Its the process of testing a particular release of a system that is intended for use outside of the development team.
- Its primary goal is to convince the supplier of the system that its good enough for use. If so then it can be released as asystem or as a product to the customer.
- It has to therefore show that the system delivers its specified functionality, performance and dependability and does not fail during normal use.
- Its a black-box testing process where tests are derived from the system specification or functionality testing - tester is concerned with functionality not the implementation.

- Approaches:

1. Requirements-based testing - a systematic approach to test case design where each req is considered and a set of tests derived for it.
2. Scenario testing - tester devises several scenarios of use and uses them to develop test cases for the system. A scenario is a story that describes one way in which the system might be used, should re realistic and real system users shud be able to relate to them.
3. Performance testing -done once the system has been integrated whereby emergent propoerties are tested - performance, reliability

User Testing

- Users or customers provide input and advice on system testing.
- It is important since influences from the users working environment have a major effect on reliability, performance, usability and robustness of the system.
- There are three different types of system testing:
- Alpha testing - users work with the devpt team to test the system at developers site.
- Beta testing - a release of the software is made available to users to allow them to experiment and raise problems that they discover with the system developers.
- Acceptance testing - customers test a system to decide whether or not it is ready to be accepted from the system developers and deployed in the customer environment.

- There are six stages in the acceptance testing process:

- Define acceptance criteria - defined early before system contract is signed.
- Plan acceptance testing - decide on resources, time and budget for acceptance testing and establishing atesting schedule
- Derive acceptance tests - design tests to check if the system is acceptable, should test both functional and non-functional xtics.
- Run acceptance tests - The agreed acceptance tests are executed on the system
- Negotiate test results - its usually unlikely that all defined acceptance tests will pass. The developer and the customer therefore have to negotiate and decide on whether or not the system is good enough for use
- Reject/accept system - a meeting between the developer and the customer to decide whether or not the system should be accepted.