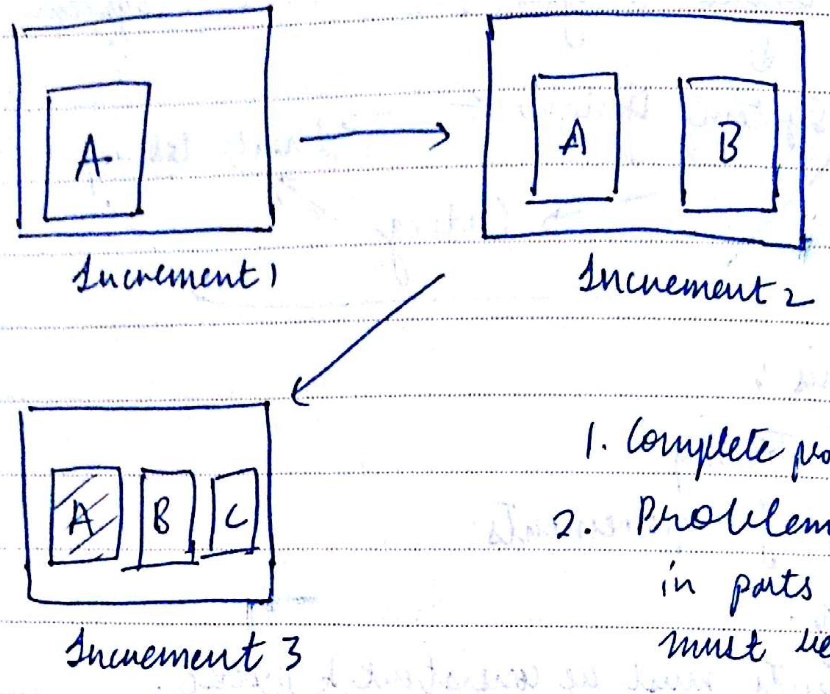
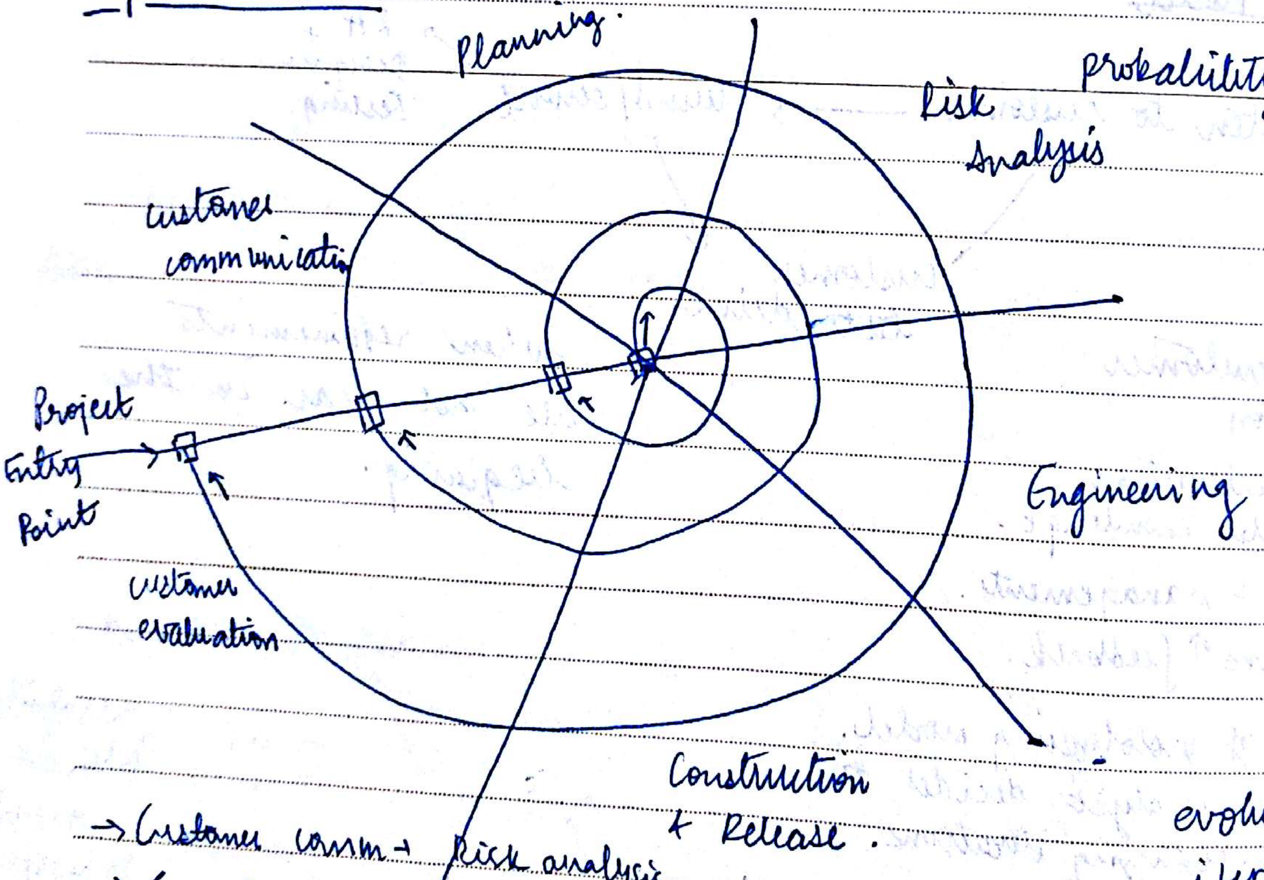


linear seq pro
+
Prototyping

Working sets

1. Complete product not delivered
2. Problem must be divided in parts i.e. all requireme must be known
3. Interfacing b/w ~~each~~ two parts

Spiral Model



probability of failure.

→ Customer comm + Risk analysis
→ Everything.
× Complex - Doc diff.
× When to stop.

evolutionary.
iterative
controlled.
systematic
linear seq.
model.

RAD - Rapid Action Development.

→ very quickly made, short span.

→ quality compromise however if problem is small → suitable.

AGILE software development

→ no strict rigid rule

→ ADHOC → as per requirement of customer + resource availability.

XP - extreme programing | no documentation

SCRUM | small documentation.

FDD

— — —

Find 2 domains where spiral model will be aptly used.

Requirements Analysis

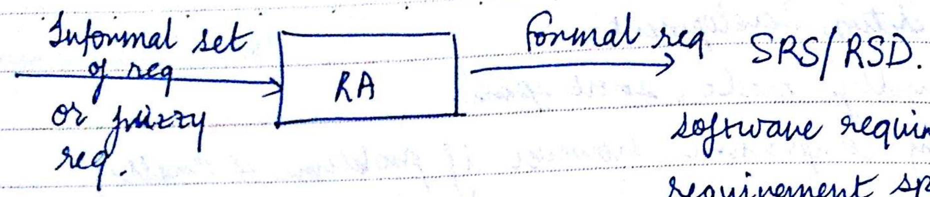
{ Demands / desire of customer

Understanding of requirements is called requirement analysis.

Good quality end product.

Requirements analysis results in specification of software operational characteristics such as function, data & behaviour, indicate software interface where other system elements and establish constraints that software must meet. R.A. provides software designer with a representation of information function, and behaviour that can be translated to data architectural interface and component level designs.

1. Problem Recognition
2. Evaluation & synthesis : feasibility of req.
3. Modeling - data & control flow.
4. Specification
5. Review.



software requirement specification
requirement specification document

Requirements

A req. is defined as ~~a condition~~

1.) a condition/capability needed by a user to solve a problem or to achieve an objective.

2.) a condition/capability that must be met/possessed by a system/a system component to satisfy a contract standard specification or other formally imposed document.

3.) a documented representation of condition/capability as ① + ②.

Types of Requirements

↳ functional : paying fees

↳ Non-functional : constraints imposed on system.

↳ Domain. : requirement of particular domain. under which s/w is built.

Requirement Process

↳ Req. Elicitation : gathering of info.

↳ Req. Specification : representation of req. in charts/diagrams.

↳ Req. validation : validation of diagrams.

fewer errors → propogation

Req. Elicitation —— 1. Interview

2. Brainstorming.

3. Task Analysis

4. Form Analysis

5. Domain analysis

6. Ethenography.

7. User Scenario & Use case req. elicitation.

8. Delphi technique.

9. Facilitated App Specificatn Tech (FAST)

Ethnography.

gather information, sitting & observe.

Interview.

structure, non-structured, open-ended

already formed questionnaire — discussion.

Brainstorming

Group discussion from developer + client side.

Task Analysis.

Splitting of bigger tasks into smaller tasks: hierarchical structure

Form Analysis

Filling of form for account : data existing in system.

Domain Analysis

Legacy systems ; then build your software on top.

User scenario & Use case based Req. elicitation

narrate story, right sequence of events to get requirements + alternate steps if something goes wrong.

Challenges in Requirements Elicitation :

⑦ Validating and tracing requirements traceability from one module to another.

⑧ Identification of critical requirements

⑨ Proper documentation of requirements

⑩ Meeting all the constraints of customers

① understanding large & complex requirements

② Defining the boundaries of the system

③ user/customer may not be clear about their needs.

④ Conflicting requirements

⑤ lot of requirement are kept under the category "to be done/ determined"

⑥ partitioning the system suitably to reduce complexity.

Organization of SRS - software requirements specification.

Other SRS formats are also available IEEE 830

1) Introduction

1.1 Purpose

1.2 Scope.

1.3. Definitions, Acronyms, Abbreviation.

1.4 References. (Legacy systems, pre-existing systems).

1.5 Overview.

2.) Overall description

2.1 Product perspective.

2.2. Product functions.

2.3 User characteristics

2.4. Constraints

2.5. Assumption & Dependencies.

2.6 Apportioning requirements ← (inclusion of 3rd party) requirements

3) Specific Requirements

3.1 External Interface.

3.2 Funtional Req.

3.3. Performance Req.

3.4 Design constraints.