M1 - OBJECT ORIENTED ANALSYS AND DESIGN

Wednesday, August 4, 2021 6:37 PM

Module

1. SDLC Models
2. Building use case model.
3. Creating system level artefacts
4. Getting into object oriented design
5. Visibility between objects, class diagrams and package diagram
6. Design patterns : GRASP, Additional, solid design principles.
7. Some gang of four patterns.
8. Further gang of four patterns.

Introduction from text book :

The Goal is to create good object oriented designs.
We are going to study OOA/D, while applying UML and patterns. And to iterative development using an agile approach to unified process.

- How to assign Responsibilities to the object. ???????
- How should objects collaborate ??????
- What classes should do what ?????
- A critical ability in OO development is to skillfully assign responsibilities to software objects.

1. Analysis: Investigation of the problem and requirements rather than a solution. How ???
   a. Finding and describing objects in the problem domain.
2. Design: Emphasizes a conceptual solution, that fulfills the requirements. (Design is not implementation)
   a. Ex: Description of schemas in data base design.
   b. Emphasis on designing software objects and how they collaborate.

Iterative , Evolutionary and agile.

1. Iterative and evolutionary development - contrasted with a sequential or waterfall lifecycle involves early programming and testing of partial system in repeating cycles.

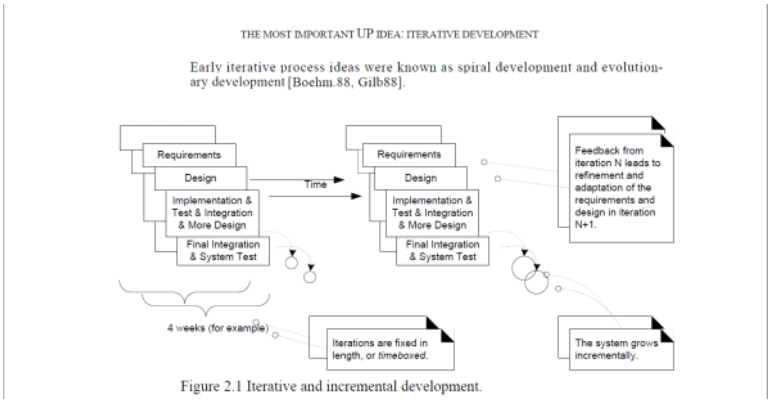


Figure 2.1 Iterative and incremental development.

SDLC MODELS

Programming Paradigms

1. Procedural programming
2. Object oriented programming.

Basics: software is a collection of programs , where program is sequence of instructions.

1. **Procedural :** means functional
   a. Lot of importance to procedures. / functions / logic.
   b. Functions are building blocks.
   c. Data is global.
   Ex: C Language programming.

2. **Modular programming**: Data organized into modules and functions which operate on them.
3. **Object oriented programming**: modelling of objects, also support of inheritance and polymorphism.
   a. Primary importance is given to data.
   b. Secondary importance to logic.
   c. Different objects connects to each other passes data .
   d. Objects are nothing but real world entities : mouse, keyboard.
   e. Maintainability of software is easy.

SDLC: (SYSTEM DEVELOPMENT LIFE CYCLE)

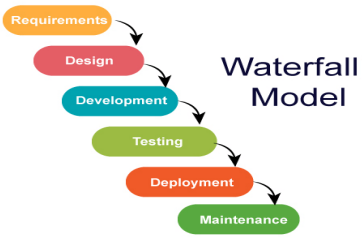
- *Analyze*
  - Gather requirement from the customer.
  - Analyst should be the domain expert.
  - Generates requirement documents

- *Design*
  - People here are called designers. / Architects.
  - May not be a domain expert.
  - Awareness about domain is sufficient
  - Create the blue print of the software using tools.
  - Uses UML like tools if required.
- *Implement*
  - People here are called coders.
  - Coders should understand the design and implement it.
- *Test*
  - Testers
  - Tests for correctness, performance and usability.
- *Deployment*
  - People : support engineers
  - Deployed to customer environment
  - Releases
- *Maintenance*
  - Fixes
  - Maintenance.

There are different SDLC Models.

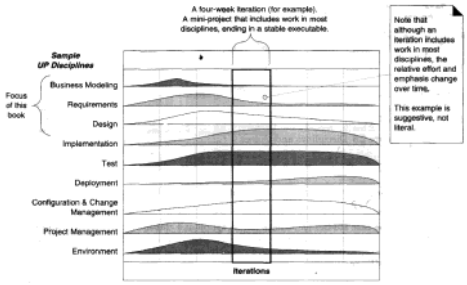
1. Waterfall
   a. Requirements, analysis, design, coding, testing, acceptance.
   b. Used for simple problems.

c. Firstly gather all the requirements, then get it into analysis
d. No more requirements addition in the middle of phases.
e. Everything happen in sequential model.
f. Vendor centric model. (no inputs from the customer in between)
g. Small communication gap can cascade down and leads to disaster at the end.
h. Disadvantages:
    i. Sequential
    ii. Not change friendly
    iii. Waterfall problem
    iv. Poor resource allocation
    v. utilization



Unhealthy waterfall thinking:
a. Let's write all requirements before implementing them.
b. Let's draw all uml diagrams.

2. Incremental
3. Spiral.
4. Unified process:
    a. For complex and sophisticated systems, linear approach is not realistic.
    b. UP accommodates changes at run time.
    c. Is a process of building object oriented systems.
    d. Iteration process has concept of mini project and iterations.
    e. The outcome of each project is tested, integrated and executable system.
    f. Each iteration includes its own Requirement analysis, design and implementation and testing activities.
    g. UML is compulsory here.
    h. If there are 100 requirements in waterfall model all 100 req needs to be given at a time, but in UP the main project is divid ed into mini projects. So let's say there the requirements are broken down to 10. all phases are applied on mini project.
    i. Will show this mini project requirement to customer, if ok will proceed with other requirements, if not there will be iterati ons in the 1st mini project and a new version will be given. Finally we can **integrate** all mini projects together (m1 + m2 + . Mn --> M)
    j. Every mini project is time boxed. Short duration is allotted to each project. Recommended 2 -6 weeks is given to complete M1.
    k. If deadline is difficult, remove tasks or requirements from iteration.
    l. Iterative & Evolutionary development:
        i. Multiple iterations with feedback and adaption. (customers can give 2 more requirements in the m2 or m3)
        ii. The system grows incrementally.
        iii. Feedback from iteration n leads to refinement and adaptation in N+1.
        iv. Advantages:
            1) Change friendly.
            2) Choose small subset of work.
    m. Phases and disciplines:
        i. Inception and its disciplines
            1) Approximate vision, business case, scope and vague estimates
            2) Define scope of project
            3) In initial stage customer gives req for diff vendors (company) , and vendor need to submit proposal to customer to how we are going to fulfill the requirements, give quotations.
            4) Preliminary analysis happens here.
            5) Feasibility study. (whatever customer is requiring is our cup of team or not)
            6) Estimation : Effort estimation, Time estimation, cost estimation.
            7) Creates a proposal and submitted to customer.
        ii. Elaboration
            1) Refined vision, Iterative implementation of core architecture, resolution of high risks, identification of most requirements, more realistic estimates.
            2) Plan project, specify features, baseline architecture
            3) Detailed analysis (multiple interactions with customer, creates UML documents / artefacts)
            4) Design: Higher level design and Lower level design (Blue print of software)
        iii. Construction
            1) Iterative implementation of remaining low level risk, prepare for deployment.
            2) Build the product
            3) Implementation
            4) Coding
            5) Testing
        iv. Transition
            1) Beta tests, deployment.
            2) Deploy whole product in customer premises (Production environment)
            3) User manuals creation
            4) Training of users.



v.

How to do iterative, Evolutionary analysis and Design ?.

- The development starts at inception phase itself , look into 25th page
- Timeboxed workshops - 2 days (BA and development people)
- Identify high level use cases and features.
- Pick 10% of the whole requirements.
- Do intensive detailed analysis of functional requirements.
- Choose design, build and test within a specified time.

Agile:

Agile development usually apply timeboxed iterative and evolutionary development.
Employ adaptive planning.
Promote incremental delivery.
Include other values and practices that encourage agility.

Agile Manifesto:

Individuals and Interactions (over process and tools)
Working software (over comprehensive documentation)
Customer collaboration (Over contract negotiation)
Responding to change. (over following a plan)

Principles:

1. Satisfy customer (Highest priority)
2. Welcome changing requirements
3. Deliver working software frequently.
4. Business people and developers must work together
5. Build projects around motivated individuals.
6. Face to Face conversation
7. Working software
8. Sustainable development
9. Sponsors, Developers and users should be able to maintain a constant pace
10. Continuous attention to technical excellence
11. Simplicity
12. Self-organizing teams

Agile modelling:

- Purpose of modelling (sketching UML) is primarily to understand not to document.
- Modelling should provide a way to better understand the problem
- Purpose of modelling is to quickly explore alternatives and path to good OO design.
- Support understanding and communication, not documentation.

Inception

1. Inception is the initial short step to establish common vision and basic scope for the project.
2. Analysis of 10% of use cases,
3. analysis of critical nonfunctional req,
4. creation of business case.
5. Prep of development env
6. Initial exploration… (Not deeper exploration)
7. Most requirements analysis occurs during the **elaboration** phase
8. ** Envision the product scope, vision and business case **

What is inception ?

1. What is the vision and business case for this project ?
2. Feasible
3. Buy or build
4. Cost
5. Proceed or stop ?

Requirements: FURPS+ (Page 57)

1. Functional
2. Usability
3. Reliability
4. Performances
5. Supportability.

Inception artifacts: (Page : 50)

1. Vision and business case
2. Use case model.
3. Supplementary specification
4. Glossary
5. Risk list and Risk management plan
6. Prototypes and proof of concepts
7. Iteration plan
8. Phase plan and SDP
9. Development case

Requirements:

1. Use case model (functional)
2. Supplementary (non functional)
3. Glossary (note worthy terms)
4. Vision
5. Business rules.

Key points:

1. Requirement analysis completes around first 4 iterations . (almost 90%) where 10% code is already build for 10% requirements .
2. Inception phase can start requirements but not complete requirements are done in inception
3. Most requirement analysis work occurs during the elaboration phase.
4. Only UC and UCD comes under inception phase.

Midterm important.
Object oriented concepts,
Unified process -> artefacts, phases, activities, (which should be done in waterfall, up etc)
Principles of guiding agile manifesto.
Unified process
Use case
Ssd
Operation contract
InteraSequence diagrams
Collaboration diagrams / communication diagrams
State diagram
Activity diagram

Shades of gray between
Use case diagram
Use case text
System sequence diagram
Sequence diagram
Collaboration diagram
Activity diagram
Swim lane diagram
Relationships: Association, dependency, generalization and realization.
Aggregation
Composition
Implementation (Interface)
Inheritance
Various type of dependencies

Abstraction, encapsulation
Iterative VS incremental.
Agilemanifesto.org (importance for exam)