Engineering in One Video (EIOV)

Watch video on YouTube

B.Tech

Software Engineering

KCS-601

With Notes

UNIT-5

Software Maintenance & SPM

(in one video)

AKTU Exam

Engineering in One Video (EIOV)

Watch video on YouTube

Topics to be covered...

Software Maintenance
Types of Software Maintenance
Why Software Maintenance required?
Cost of Maintenance
Re-engineering
Reverse Engineering
CASE Tools
PERT
COCOMO
Risk Management
Happy ending!

Engineering in One Video (EIOV)

Watch video on YouTube

Software Maintenance

Software Maintenance

- Software maintenance is a part of Software Development Life Cycle.
- Its main purpose is to modify and update software applications after delivery to correct faults and to improve performance.
- The essential part of software maintenance requires preparation of an accurate plan during the development cycle.
- The cost of software maintenance is quite high.

Need for software maintenance:

1. Over a period of time, software's original requirements may change to reflects the customer's need.
2. Errors undetected during software development may be found during the use and require correction.
3. With the time, new technologies are introduced such as new hardware, operating system etc.

Engineering in One Video (EIOV)

Watch video on YouTube

Types of Software Maintenance

Types of Software Maintenance

Corrective Maintenance:

- Either discovered by user or concluded by user error reports.

Adaptive Maintenance:

- This includes modifications and updation applied to keep the software product up-to date and tuned to the ever changing world of technology and business environment.

Perfective Maintenance:

- It includes new features, new user requirements for refining the software and improve its reliability and performance.

Preventive Maintenance:

- It aims to attend problems, which are not significant at this moment but may cause serious issues in future.

Engineering in One Video (EIOV)

Watch video on YouTube

Why Software Maintenance required?

Why Software Maintenance required?

Bug Fixing:

- Searching errors in code and correcting them. This issues can be occurred in any part of the software.

Capability enhancement:

- Making improvement in features and functions to make solutions compatible with the varying market environment.

Removal of outdated functions:

- Old coding elements are removed and relaced with new coding elements.
- This helps system to cope with changing circumstances.

Performance improvement:

- To improve system performance, developers detects issues through testing and resolve them.

Engineering in One Video (EIOV)

Watch video on YouTube

Cost of Maintenance

Cost of Maintenance

- The cost of system maintenance represents a large proportion of the budget of most organizations that use software systems.
- More than 65% of software lifecycle cost is expended in the maintenance activities.

Cost of software maintenance can be controlled by postponing the development opportunity of software maintenance but this will cause the following intangible cost:

- Customer dissatisfaction when requests for repair or modification cannot be addressed in a timely manner.
- Reduction in overall software quality as a result of changes that introduce hidden errors in maintained software.

Cost of Maintenance

Key Factors affecting Cost are:

1. **Non-Technical factors**
2. **Technical factors**

Non-Technical factors:

The Non-Technical factors include:

1. Application Domain
2. Staff stability
3. Program lifetime
4. Dependence on External Environment
5. Hardware stability

Technical factors:

The Technical factors include:

1. module independence
2. Programming language
3. Programming style
4. Program validation and testing
5. Documentation
6. Configuration management techniques

Engineering in One Video (EIOV)

Watch video on YouTube

Re-engineering

Re-engineering

- Software Re- engineering is the examination and alteration of a system to reconstitute it in a new form.
- It affects positively at software cost, quality, service to the customer and speed of delivery.
- In Software Re-engineering, we are improving the software to make it more efficient and effective.
- It is a process of software development which is done to improve the maintainability of a software system.

Objectives of Re-engineering:

- To describe a cost-effective option for system evolution.
- To describe the activities involved in the software maintenance process.
- To distinguish between software and data re-engineering and to explain the problems of data re-engineering.

Engineering in One Video (EIOV)

Watch video on YouTube

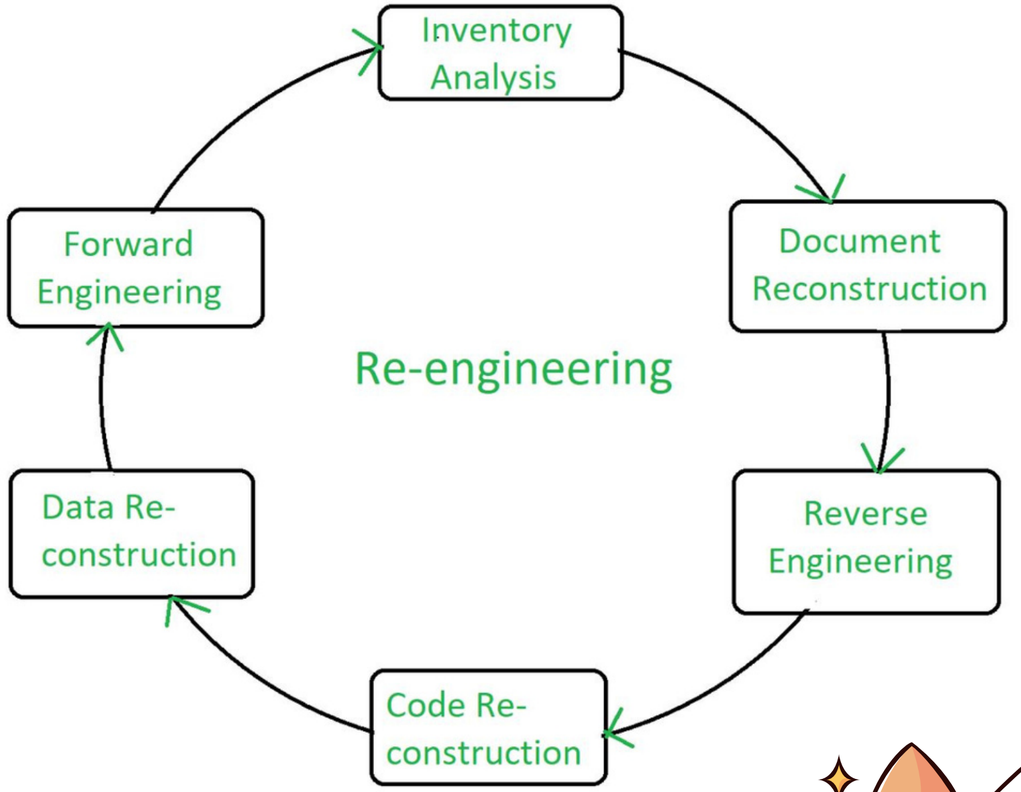
Re-engineering

Steps involved in Re-engineering:

- Inventory Analysis
- Document Reconstruction
- Reverse Engineering
- Code Reconstruction
- Data Reconstruction
- Forward Engineering

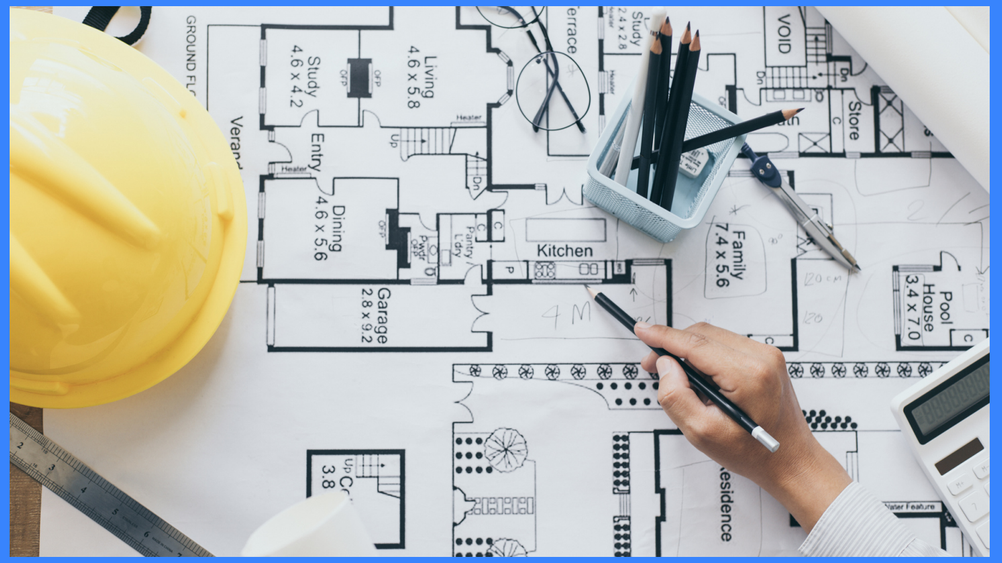
Advantages of Re-engineering:

- Reduced Risk
- Reduced Cost
- Revelation of Business Rules
- Better use of Existing Staff



Engineering in One Video (EIOV)

Watch video on YouTube



Reverse Engineering

EIOV

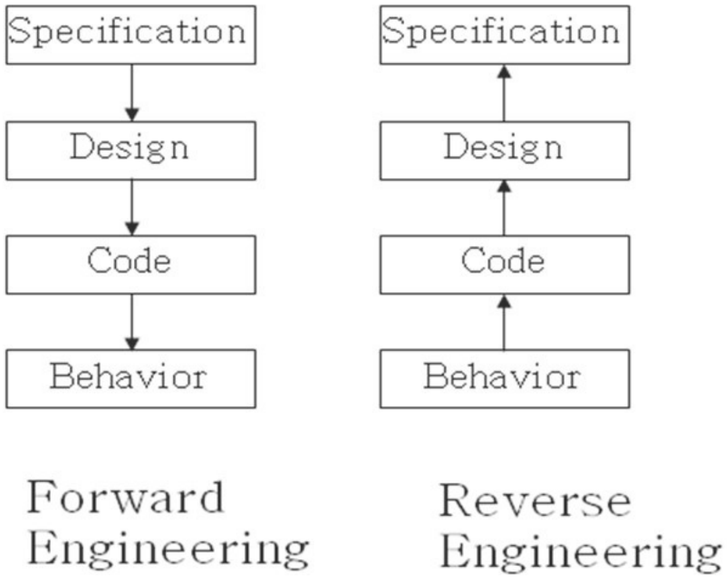
Engineering

IN ONE VIDEO

CONNECT, LEARN AND GROW

Reverse Engineering

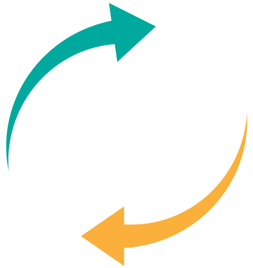
- Reverse Engineering tools extract data, architectural and procedural design information from an existing program.
- Software Reverse Engineering is the process of recovering the design and the requirements specification of a product from an analysis of it's code.
- Reverse Engineering is becoming important, since several existing software products lack proper documentation, are highly unstructured.
- The aim of reverse engineering is to improve the understandability of the system by helping the maintenance work.



Reverse Engineering

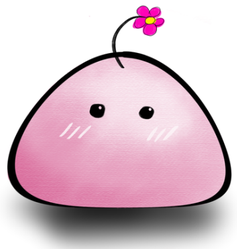
Goals:

- Complexity Co-Operation.
- Recovering the Lost Information.
- Determining the Side Effects.
- Higher Abstraction Synthesis.
- Providing the Facility for Reuse.



Applications:

Software Reverse Engineering is used in software design, reverse engineering enables the developer or programmer to add new features to the existing software with or without knowing the source code.



Engineering in One Video (EIOV)

Watch video on YouTube

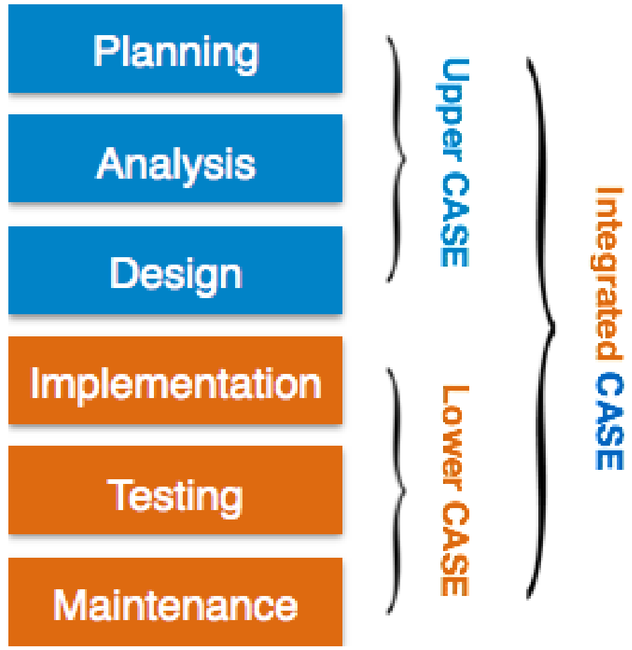
CASE Tools

CASE Tools

- It stands for **Computer Aided Software Engineering**.
- It means, development and maintenance of software projects with help of various automated software tools.
- CASE tools are set of software application programs, which are used to automate SDLC activities.
- CASE tools are used by software project managers, analysts and engineers to develop software system.
- There are a number of CASE tools available to simplify various stages of Software Development Life Cycle such as Analysis tools, Design tools, Project management tools, Database Management tools, Documentation tools etc.

CASE Tools

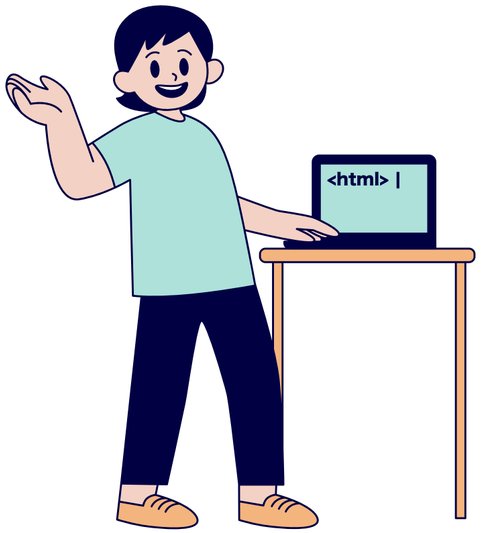
- **Upper Case Tools:** Upper CASE tools are used in planning, analysis and design stages of SDLC.
- **Lower Case Tools:** Lower CASE tools are used in implementation, testing and maintenance.
- **Integrated Case Tools:** Integrated CASE tools are helpful in all the stages of SDLC, from Requirement gathering to Testing and documentation.



CASE Tools

CASE Tools:

1. **Configuration Management Tools**
    a. Change Control Tools
    b. Programming Tools
    c. Web Development Tools
    d. Quality Assurance Tools
    e. Maintenance Tools
2. **Case Tool Types**
    a. Diagram Tools
    b. Process Modelling Tools
    c. Project Management Tools
    d. Documentation Tools
    e. Analysis Tools
    f. Design Tools



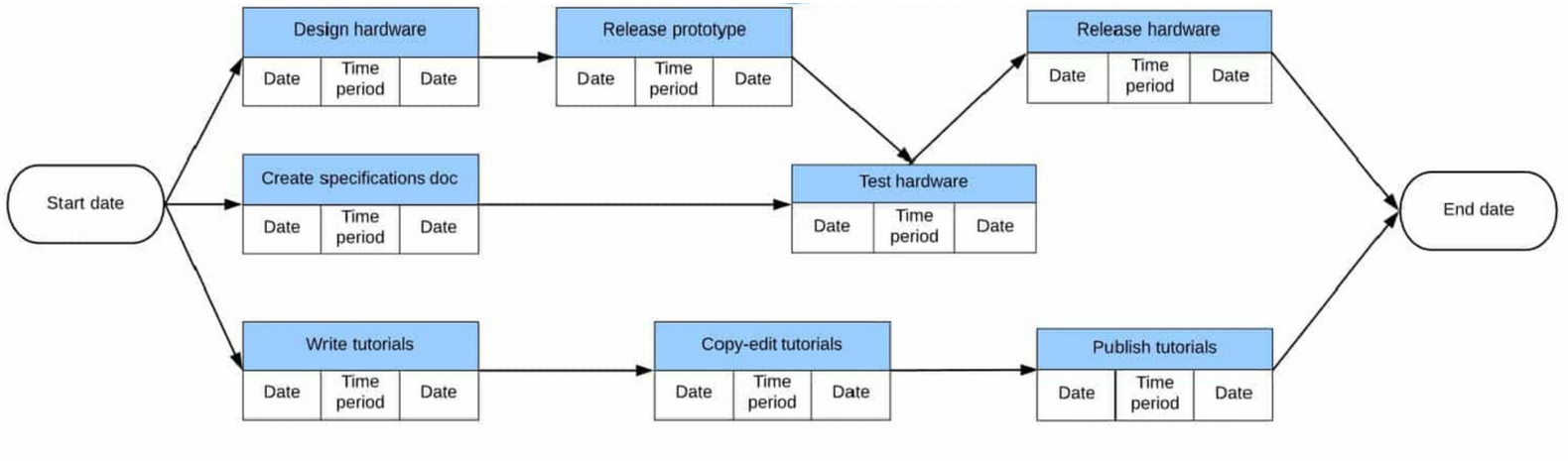
Engineering in One Video (EIOV)

Watch video on YouTube

PERT

PERT

- It stands for **Program Evaluation Review Technique**.
- PERT is a project management planning tool used to calculate the amount of time it will take to realistically finish a project.
- PERT charts are used to plan tasks within a project making it easier to schedule and coordinate team members.
- PERT can be both a cost and a time management system.
- it does not deal very well with task overlap.



Engineering in One Video (EIOV)

Watch video on YouTube

COCOMO

COCOMO

- It stands for **Constructive Cost Models**.
- COCOMO is a regression model based on number of Lines of Code(LOC).
- It is a procedural cost estimate model for software projects and is often used as a process of reliably predicting the various parameters associated with making a project such as size, effort, cost, time, and quality.
- It was proposed by Barry Boehm in 1981 and is based on the study of 63 projects, which makes it one of the best-documented models.

The key parameters:

- **Effort:** The number of labor required to complete the work. It is measured in person-months units.
- **Schedule:** The amount of time required to complete the work is directly proportional to the effort. It is measured in the unit of time, for example, months, and weeks.

COCOMO

Software projects are classified into three categories:

Organic: In the organic type, the project deals with developing a well-understood application program; the team size is generally small. This category is for the small to medium size software product. In this type, team members have good experience and knowledge.

Semi-detached: In the semi-detached type, the essential elements are team-size, experience, and knowledge of the multiple programming languages. The projects that come under the semi-detached are less familiar and hard to develop. It also requires better guidance, more experienced developers.

Embedded: In the embedded type, a software project requires the highest level of complexity, creativity, and experience. In this category, the larger team size is needed as compared to the previous models.

Engineering in One Video (EIOV)

Watch video on YouTube

Risk Management

Risk Management

- "Tomorrow problems are today's risk." Hence, a clear definition of a "risk" is a problem that could cause some loss or threaten the progress of the project, but which has not happened yet.
- These potential issues might harm cost, schedule or technical success of the project and the quality of our software device, or project team morale.
- Risk Management is the system of identifying addressing and eliminating these problems before they can damage the project.

There square measure 3 main classes of risks:

1. Project risks
2. Technical risks
3. Business risks

Risk Management

Project risks: Project risks concern differ forms of budgetary, schedule, personnel, resource, and customer-related problems.

Technical risks: Technical risks concern potential method, implementation, interfacing, testing, and maintenance issue.

Business risks: This type of risks contain risks of building an excellent product that no one need, losing budgetary or personnel commitments, etc.

Happy Ending!

Congratulations!