SOFTWARE PROJECT MANAGEMENT

> main goal: to enable a group of developers to work efficiently towards successfull completion of project.

> complexities → invisibility
changeability
complexity
uniqueness
exactness of sol".
Team-oriented coork.

> Responsibilities of S.P. Manager:

- project planning: undertaken immediately after feasibility study phase
- project monitoring & control: starts when dev act start.

> Project planning:

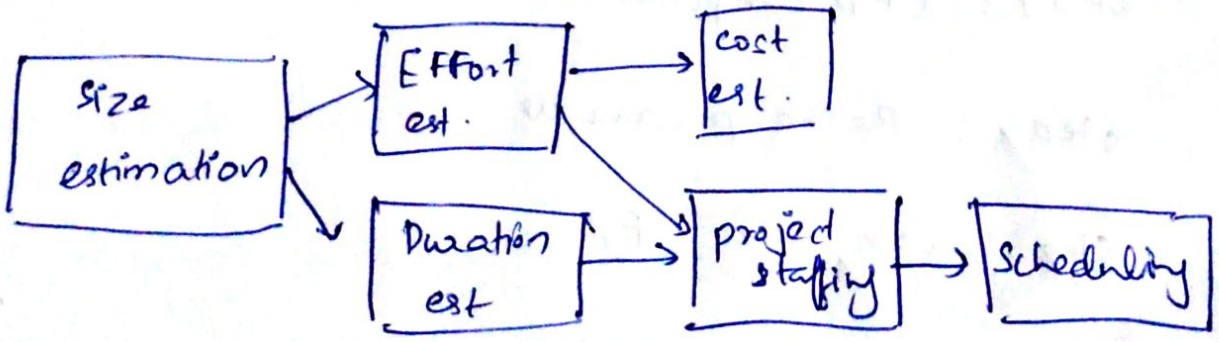
Estimation: cost, duration, effort

Scheduling

Staffing

Risk Management

Miscellaneous plans.



Sliding window planning:

Start with an initial plan, the project is planned more accurately over a number of stages.

METRICS FOR PROJECT SIZE ESTIMATION:

project size: measure of problem complexity in terms of effort & time.

- LOC [Lines of code]
  - no. of source instructions, comment lines, & header lines
  - LOC of leaf-level modules are small enough to be predicted
  - overall product dev. effort is determined from coding effort alone.

- Function point [FP]
  - based on idea that a SW product supporting many features would certainly be of larger size than a product with less leaves.

Step 1: UFP computation

Step 2: Refine parameters

Step 3: Compute FP.

PROJECT ESTIMATION TECH:-

- empirical estimati

COCOMO

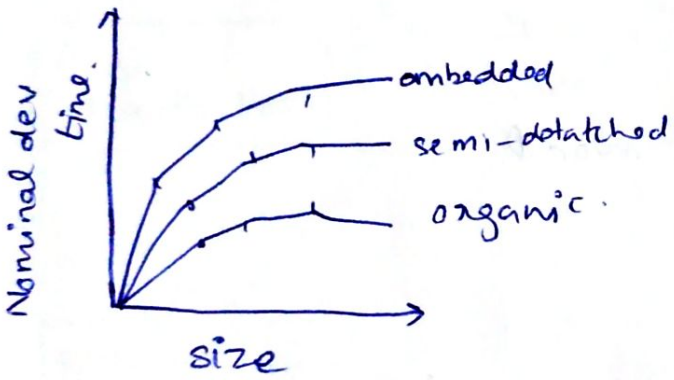
It is a single variable heuristic model that gives approx. estimate of project parameters.

$$\text{Effort} = a_1 \times (KLOC)_{a_2} \text{ PM}$$

$$T_{dev} = b_1 \times (\text{Effort})_{b_2} \text{ months;}$$

- estimation of dev effort
- observation from the effort-size point.

Cost estimation: project cost can be obtained by multiplying estimated effort by manpower cost.



- effort and duration values computed by COCOMO are the values for completing work in shortest time without unduly increasing manpower cost.

Intermediate COCOMO:

Refines initial estimate obtained using basic cocomo expressions by scaling the estimate up or down based on the evaluation of a set of attr. of s/w dev.

COMPLETE COCOMO

shortcoming of intermediate : considers s/w product as single homogeneous entity.

estimates effort and dev time as sum of estimates of individual sub-systems

COCOMO 2 :

provides 3 models to arrive at increasingly accurate cost estimation used to estimate project costs at different phase of s/w product.

- Application composition model
- early design model
- post-architecture model

Early design model :

$$effort = K\ SLOC \times \prod_i \text{cost drivers}_i$$

post-arch model:

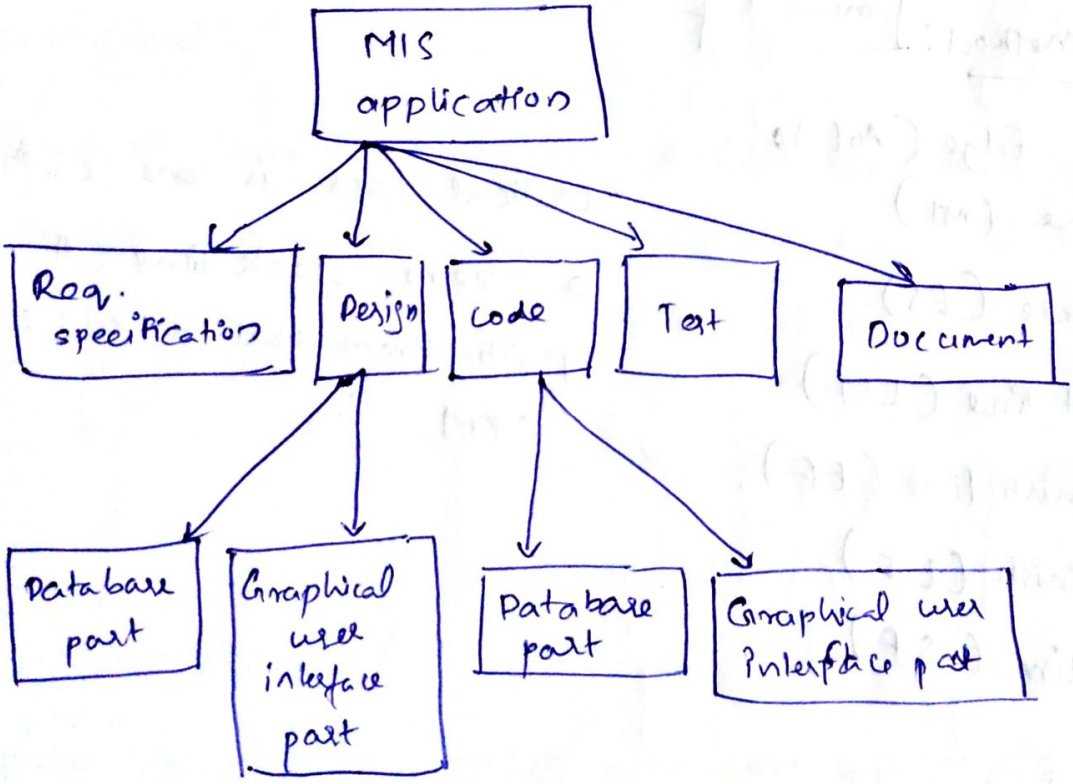
$$effort = a \times K\ SLOC^b \times \prod_i \text{cost drivers}_i$$

> SCHEDULING.

consists of deciding which tasks would be taken up when and by whom.

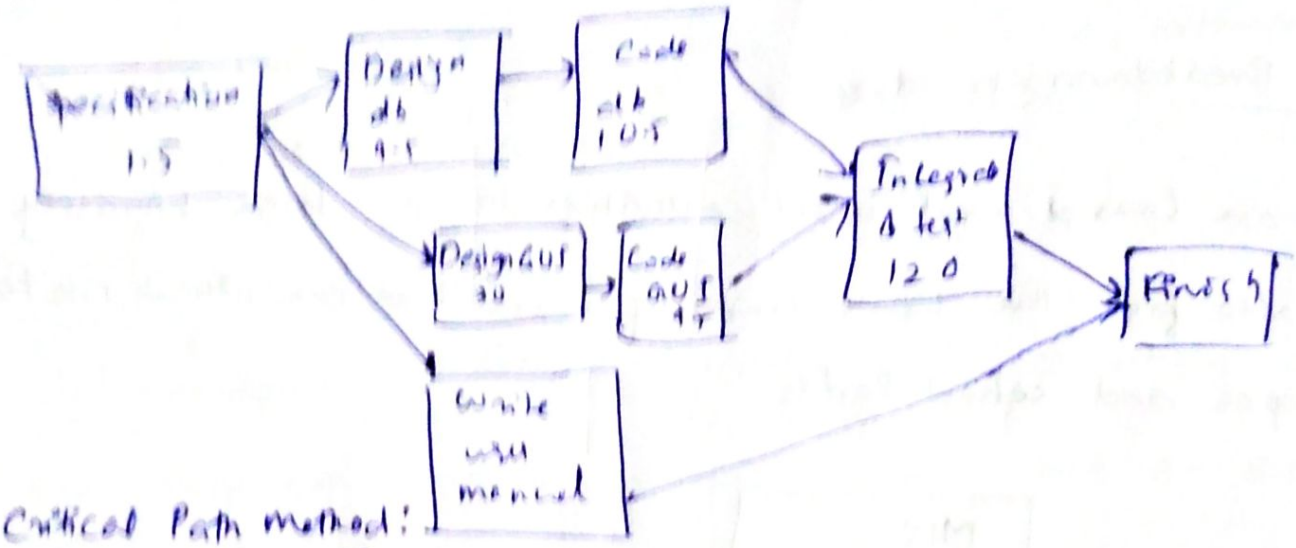
> Work Breakdown structure :

Tasks are lowest level work activities in a WBS hierarchy They also form the basic unit of work that are allocated to the developer and scheduled.



Activity Networking:

AoN [Activity on Networking]: each activity rep as rectangle node and duration noted alongside.



> Critical Path method:

- Activity on Edge (AoE).
- Minimum time (MT)
- Earliest start time (EST)
- Latest start time (LST)
- Earliest Finish time (EF).
- Latest Finish (LF)
- Slack time (SF)

- Critical task is one with a zero slack time. A path from start to end is CP

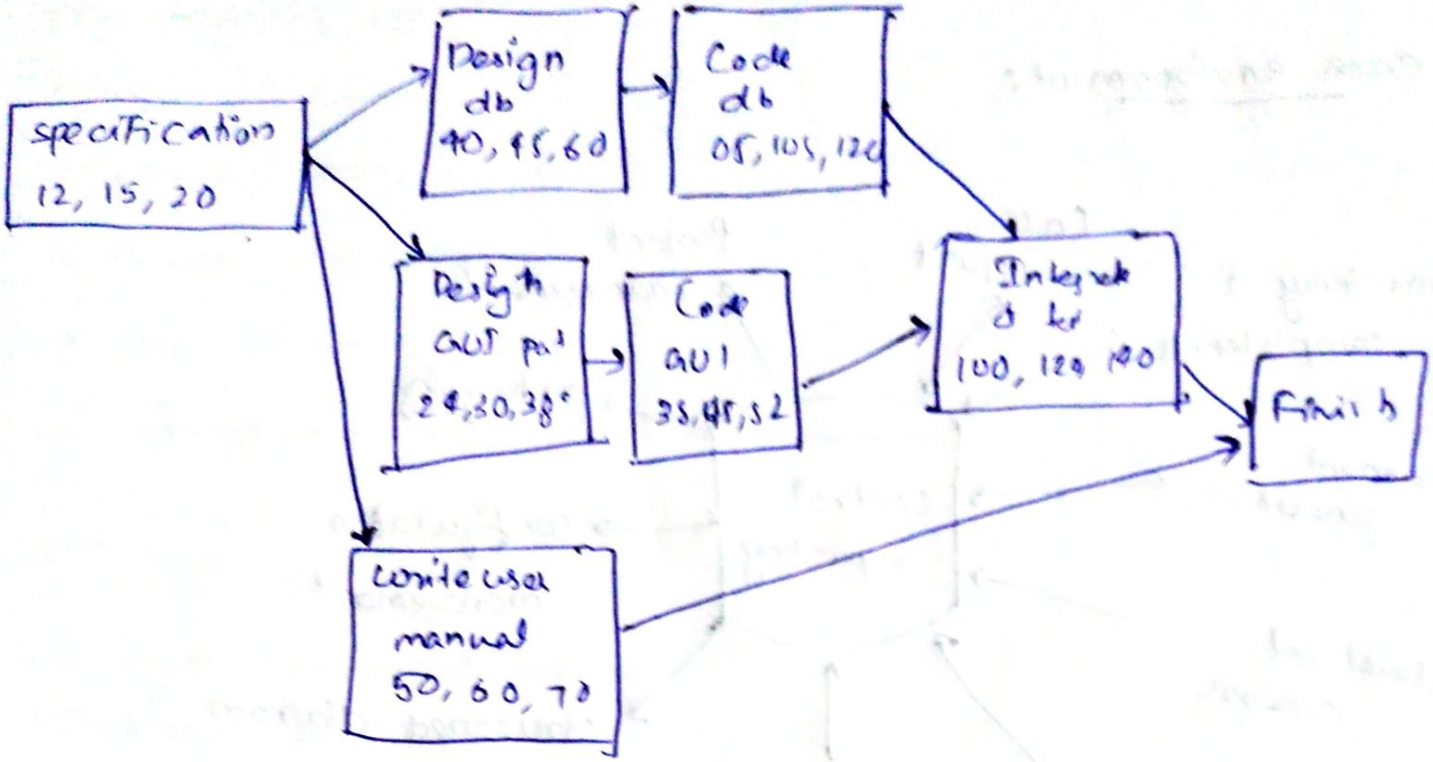
CPM

PERT charts:

Project evaluation & review chart → sophisticated form of activity chart.

can be used to determine the probability distribution. that represents statistical variations

- Optimistic (o):
- Most likely estimate (m):
- Worst (w):



Gantt Chart:

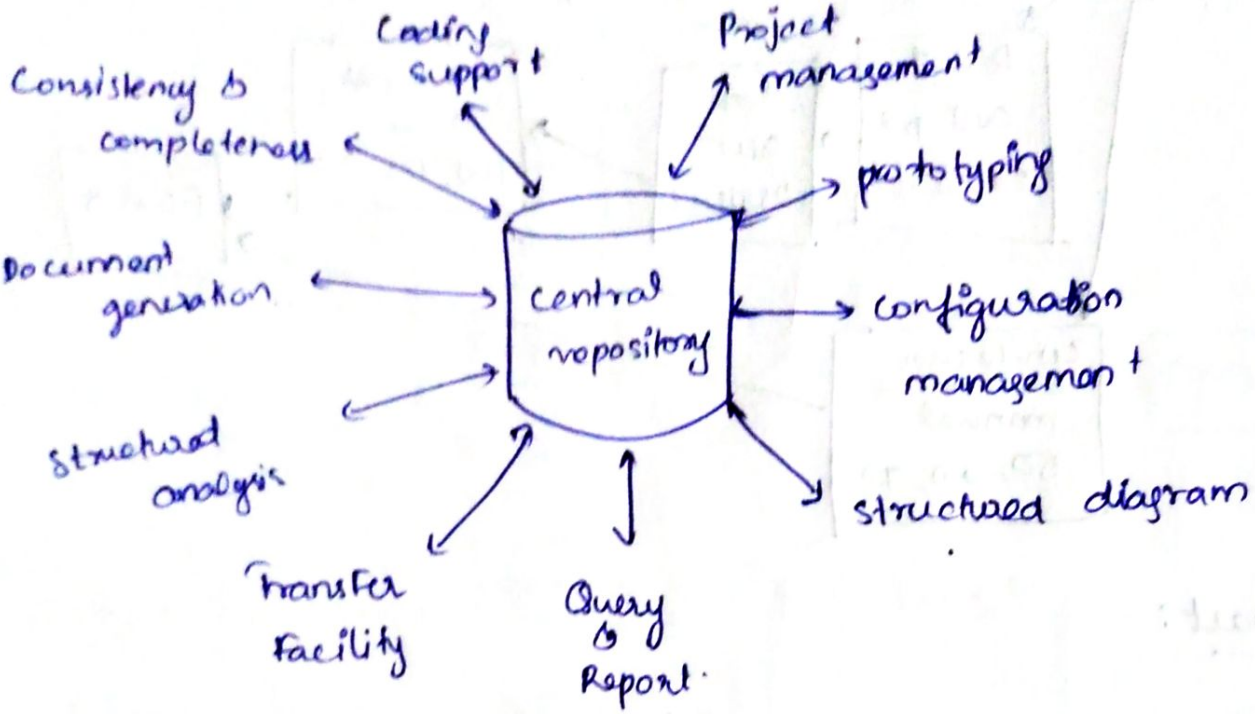
Form of chart. vertical lists all tasks, bar drawn on y-axis shaded part of bar shows length of time.

Project monitoring & control – important activity as milestone.
If any delay occurs, entire project delay

COMPUTER AIDED SE.

Case: denote any form of automated support for s.e.
- to ↑ productivity
→ to help produce better quality s/w at low cost.

Case environment:



Benefits of case: cost saving
improvement in quality.
high quality & consistent docs
effortless
easy orientation.

> case support in s/w life cycle

- Prototyping support: GUI using a graphic editor
  data dictionary
  sequences of state
  support mockup run of system.
- structured analysis and design.
- Test case generator : support design and req. testing.
- Code Design : module skeleton, records, structures, UI codes etc.

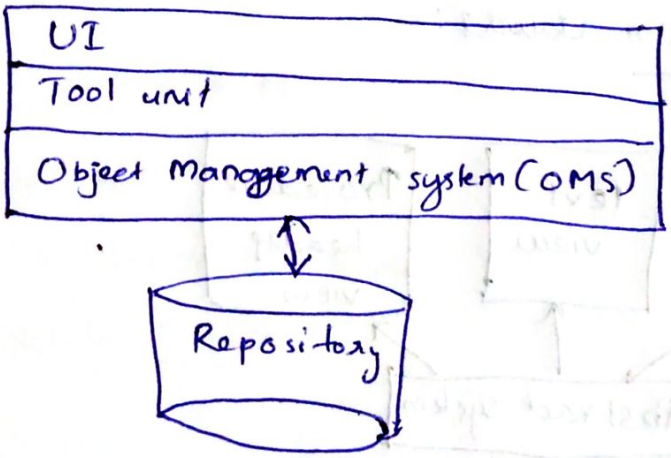
Other chara:
- Documentation support :
- H/w and environmental req:

- Documentation support
- Project Management
- External Interface.
- Reverse Engineering
- Data Dictionary Interface
- Tutorial and Help

2nd gen case tool:

→ Intelligent diagramming support:
→ Integration with implementation environment
→ Data dictionary standards
→ customisation support.

Arch. of a case environment:



WORKBENCH:

A set of tools representing particular phase in s/w process

Types

1. programming
2. Design
3. Testing
4. GO dev.
5. Config management.
6. Documentation.

Close workbench:

- control & data integration are peoperty.
- Impossible to integrate 3rd party tools

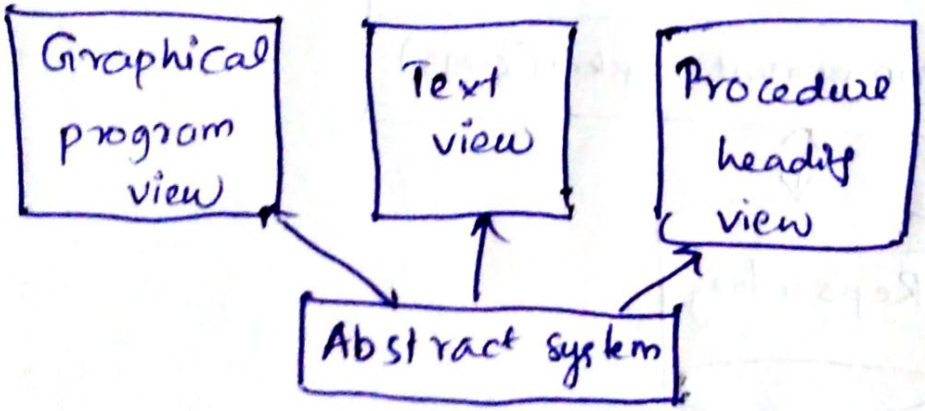
Open workbench:

- 3rd party tool integrated.

Programming workbench:

- Tools to support program development
- include linker, loader.
- Integration around stored source code.

Multiple program viewers:



Language derived workbenches

- Multiple program analysis
- Multiple views of program

4 GL workbenches:

- Provides facilities for developing 4 GL prgrms.
- Integration around a theme.

Components:

DBL.

Form design system

Spreadsheet

Repeat generator.

> Design of analysis workbenches:

- Generator system moderator during design & analysis.
- Provides editors plus shared repo.
- Code generator (optional).

> workbench advantages.

- cheap economality
- Results in standardized format
- Highly efficient (40%)

> Drawbacks

- Import/export Facilitates are limited.
- preparatory
- Not adaptable to org needs.

> Testing workbenches

- expensive phase
- Generally open systems

Meta-Case.

Tools which aunt the process of creating workbench.

Multi-notation design editor.