SOFTWARE ENGINEERING

1) List the 15 cost driver attributes of COCOMO and analyze why they are required to find the overall cost of a project.

Ans:- The basic COCOMO model assumes that the effort is only a function of the number of lines of code and some constants evaluated according to the different software systems. However, in reality no system's effort and schedule can be solely calculated on the basis of lines of code. For that, various other factors such as reliability, experience and capability come into consideration. These factors are known as cost drivers and the intermediate COCOMO model utilizes 15 such drivers for cost estimation

PRODUCT ATTRIBUTES [ignore abbreviations in exam]

- RELY, required reliability
- DATA, database size
- CPLX, product complexity

COMPUTER ATTRIBUTES

- TIME, execution time constraint
- STOR, main storage constraint
- VITR, virtual machine volatility
- TURN, computer turnaround time

PERSONNEL ATTRIBUTES

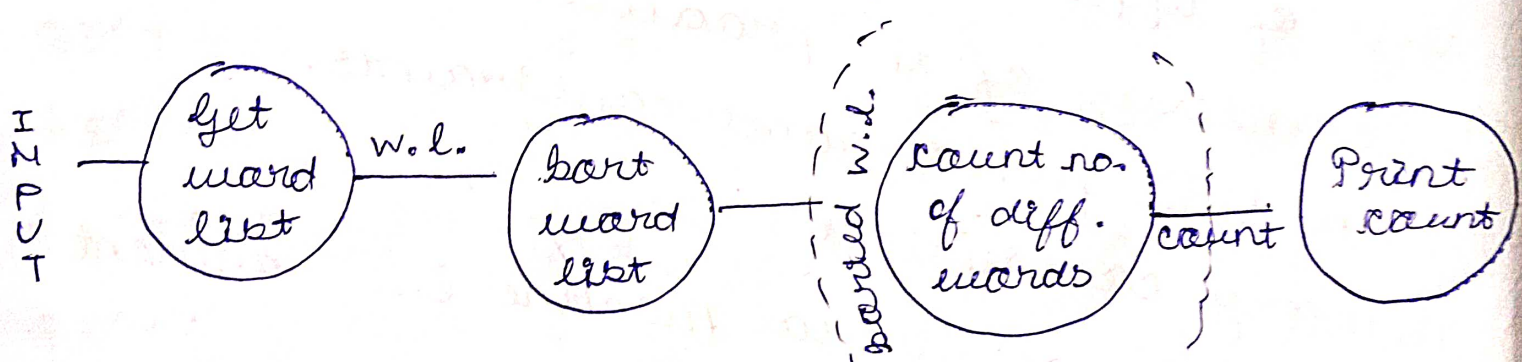
- ACAP, analyst capability.
- AEXP, applications experience
- PCAP, programmer capability
- VEXP, virtual machine experience
- LEXP, programming language experience.

PROJECT ATTRIBUTES

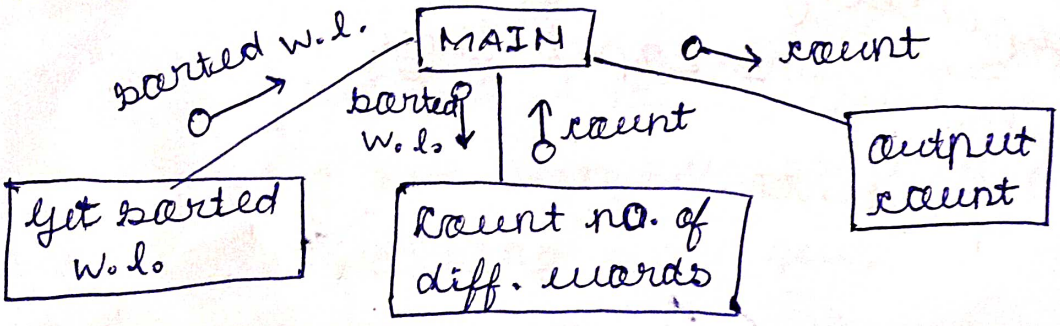
- MODP, modern programming practices.
- TOOL, use of software tools
- SCHED, development schedule.

2) Design the structure charts (levels of structure chart are obtained upon factoring). for the problem of determining the number of different words in an input file. Explain the structure chart at each level.

Ans:- First we construct the data flow diagram for this problem. We have only one input data stream, the input file, and the desired output is the count of the different words in the file. To transform this, we first make a list of all the words in the file, sort it and count the number of different words.

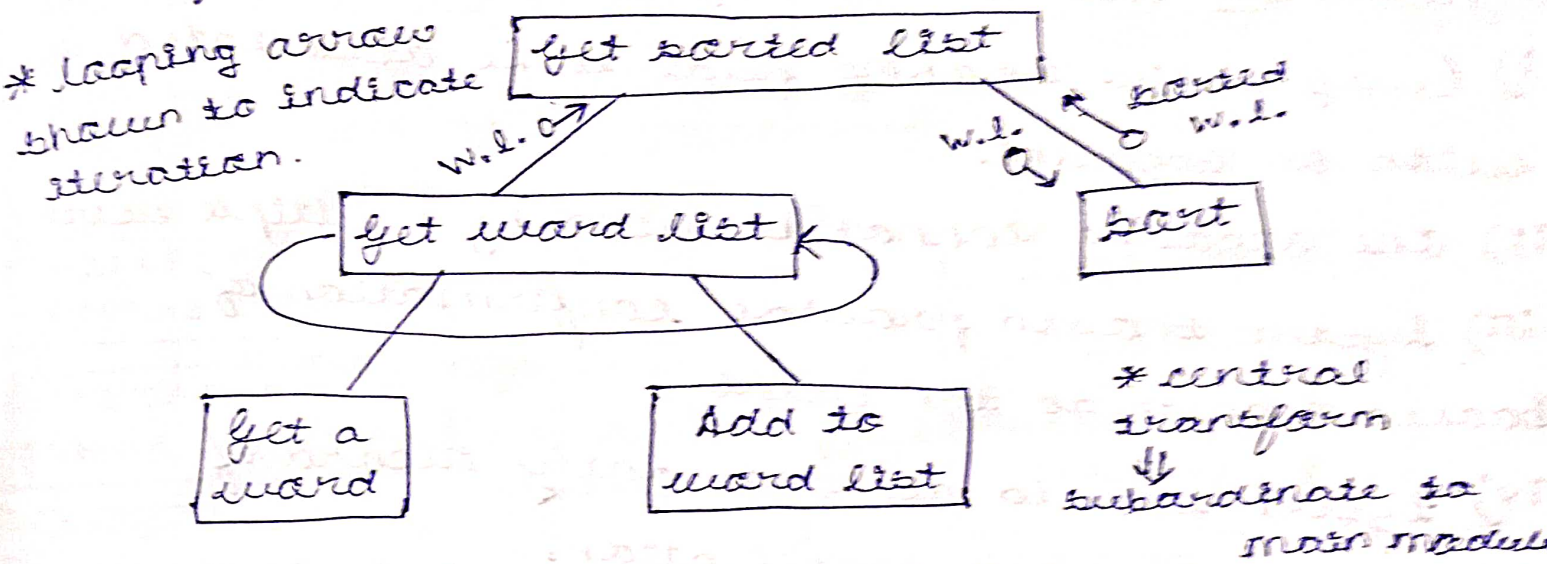


The structure after the first-level factoring of the word-count problem is:

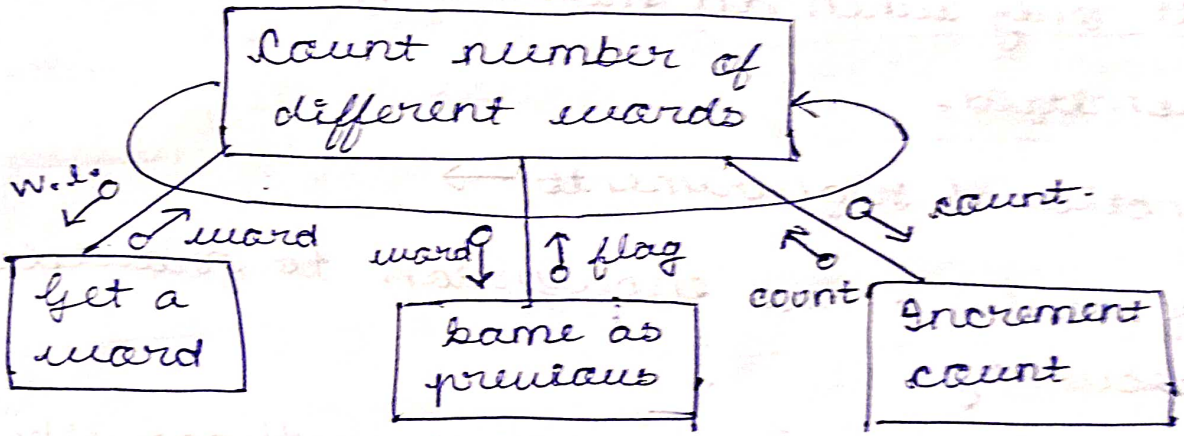


There is one input module, which returns the sorted wordlist to the main module. The output module takes from the main module the value of the count.

Further factoring of input module get sorted list in the first level structure is:



The factoring of the central transform, count number of different words, gives:



To determine the number of words, we have to get a word repeatedly, determine if it is same as previous (for a sorted list, this checking is sufficient to determine unique word), and then count the word if it is different. For each of the 3 functions, we have a subordinate module, and get the above structure

3) List 5 functional and non-functional requirements of a Railway Reservation system. Also draw a level DFD set for the same system.

Ans:- The functional requirements for the SRS of a Railway Reservation system are:

i) Every online booking needs to be associated with an account.

ii) One account cannot be used by multiple users.

iii) System should provide confirmation of booked tickets to the users.

iv) System should generate daily record of ticket booking and cancellation.

v) System should only allow users to move to payments only when all mandatory details have been submitted.

Non-functional requirements →

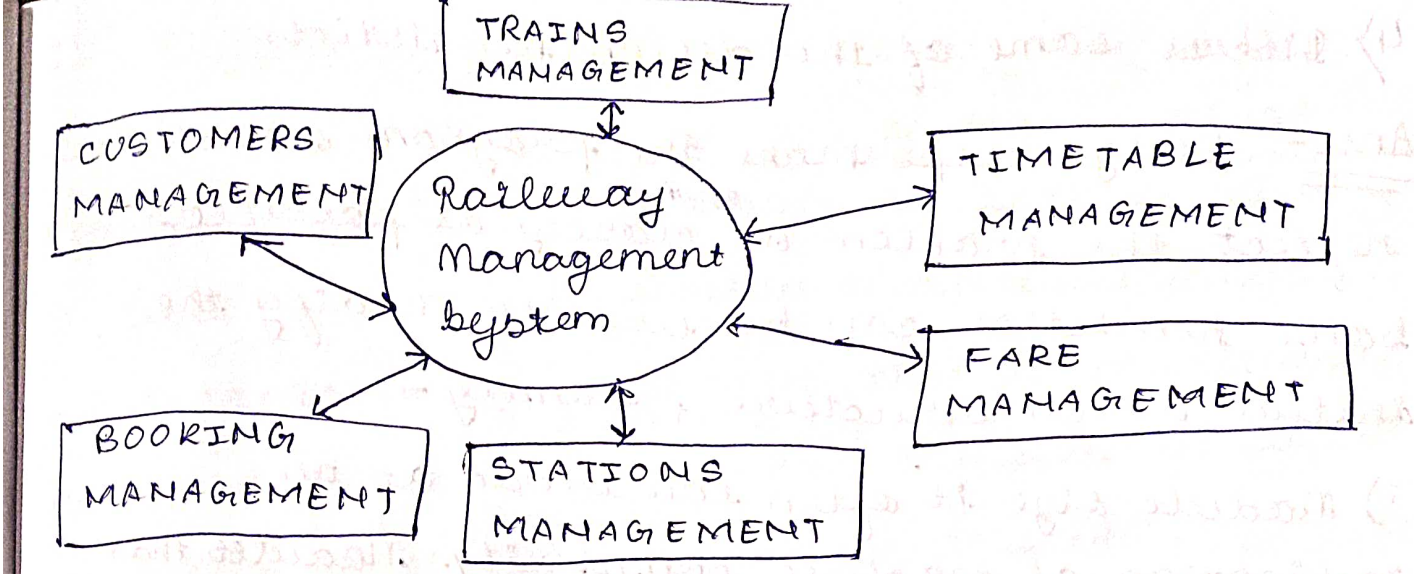
i) Use of captcha and encryption to provide more security.

ii) Accept payment via various methods like credit cards, netbanking.

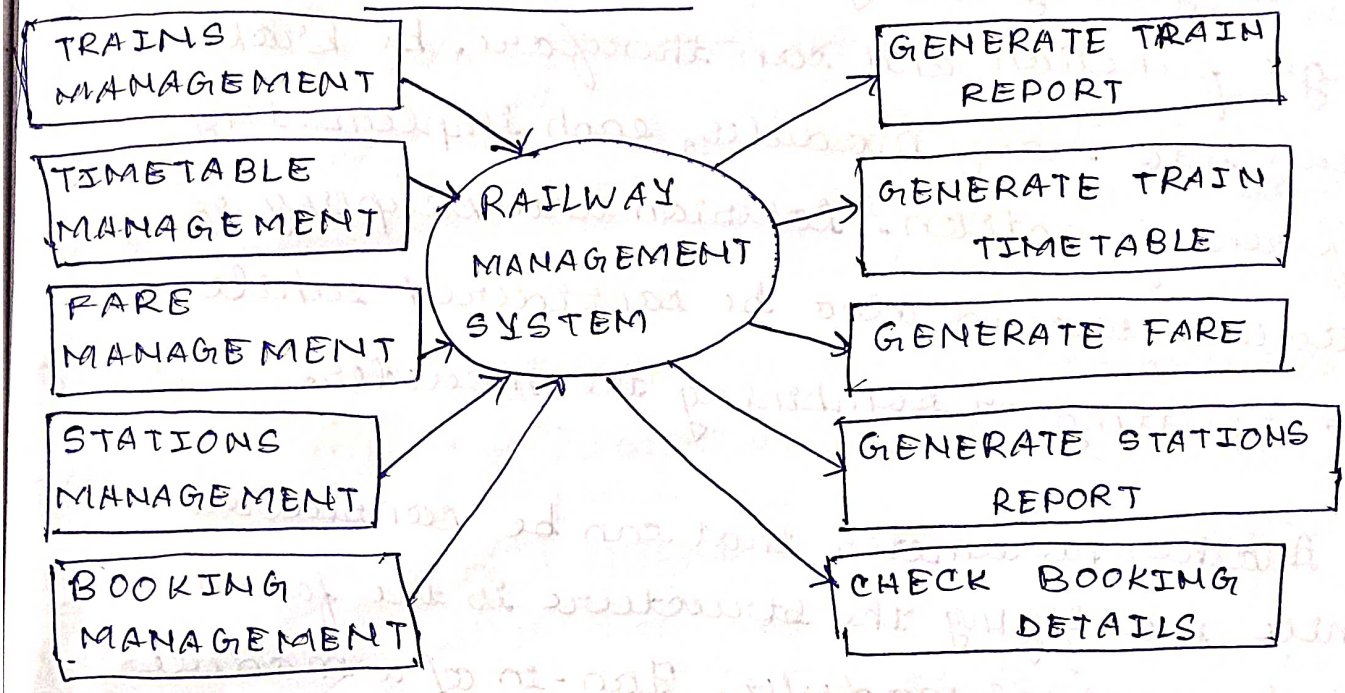
iii) Refund fare of tickets in case of cancellation.

iv) Check availability of seats and status of booking.

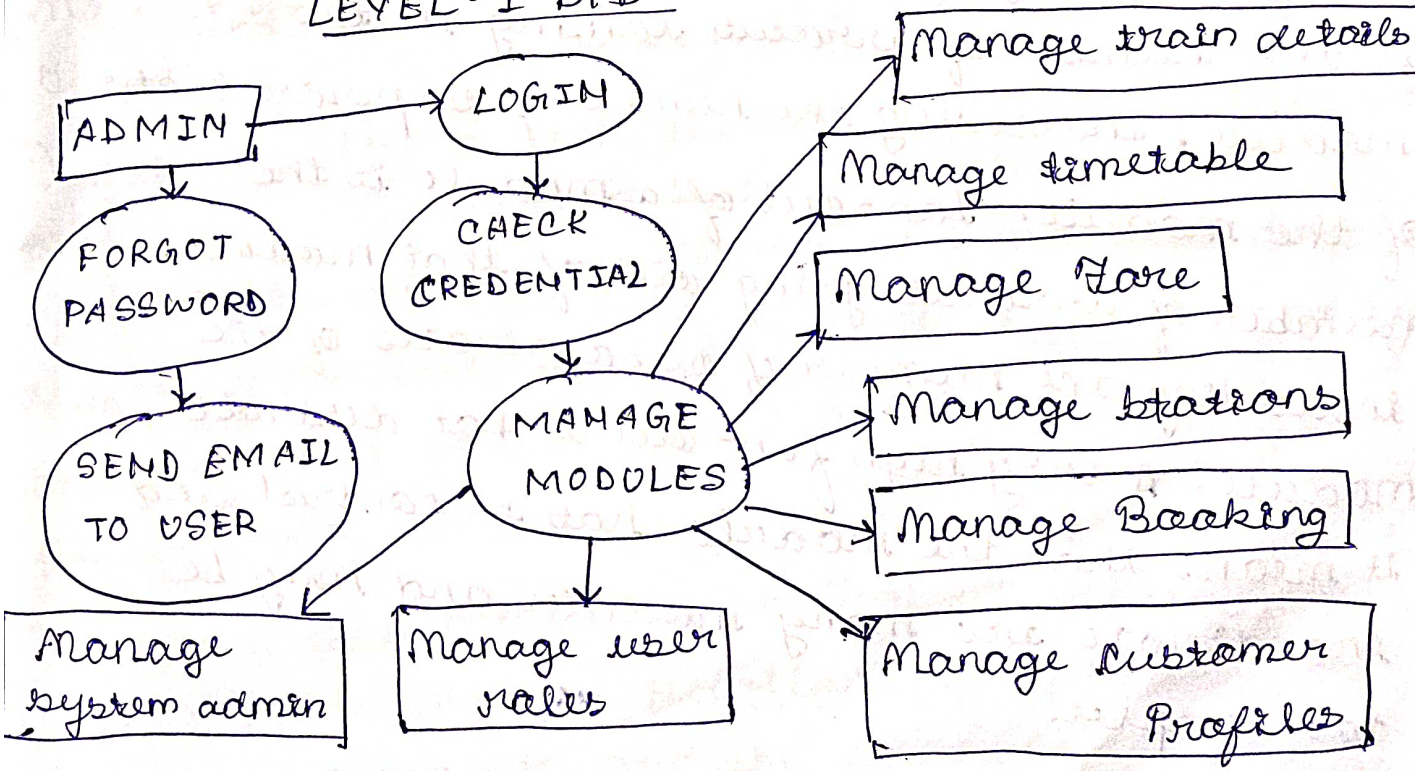
v) See the arrival and departure time of each train.



LEVEL-0 DFD



LEVEL-1 DFD



LEVEL-2 DFD

4) Discuss some of the design heuristics.

Ans:- Design steps make the program structure reflect the problem as closely as possible. Some heuristics can be used to modify the initial design structure, namely -

i) Module size is often considered as an indication of module complexity. Modules that are very large may not be implementing a single function and can therefore, be broken down into many modules, each implementing a different function. Cohesion and coupling of modules should also be considered while ~~split~~ splitting or combining the modules.

ii) Another parameter that can be considered while modifying the structure is the fan-in and fan-out of modules. Fan-in of a module is the number of arrows coming in the module, indicating the number of superordinates of the module. Fan-out of a module is the number of arrows going out of that module, indicating the number of subordinates of the module. A very high fan-out is not desirable, as it means that the module has to control and coordinate too many modules and may be too complex.

iii) Another important factor that should be considered is the correlation of the scope of effect and scope of control. The scope of effect of a decision is the collection of all the modules that contain any processing that is conditional on that decision or whose invocation is dependent on the outcome of the decision. Scope of control of a module is the module itself and all its subordinates. System is simpler when scope of effect is a subset of the scope of control of the module in which the decision is located.

—X—

5) Discuss the different levels of cohesion in function oriented design methodology.

Ans:- Cohesion of a module gives the designer an idea about whether the different elements of a module belong together in the same module. The different levels of cohesion are:

i) Coincidental — It is the lowest level of cohesion and occurs when there is no meaningful relationship among the elements of a module. It can occur if an existing program is modularized by chopping it into pieces and making different pieces modules.

ii) Logical — a module has logical cohesion if there is some logical relationship between the elements of a module, and the elements perform

functions that fall in the same logical class.
eg. a module that performs all i/o operation.

iii) Procedural — A procedurally cohesive module contains elements that belong to a common procedural unit. eg. a loop or a sequence of decision statements in a module ~~comp~~ combined to form a separate module.

iv) Temporal — Temporal cohesion is the same as logical cohesion, except that the elements are also related in time and are executed together. eg. modules that perform activities like initialization, clean-up and termination.

v) Communicational — A module with communicational cohesion has elements that are related by a reference to the same input or output data, i.e. the elements are together because they operate on the same input or output data.

vi) Sequential — When the elements are together in a module because the output of one forms the input to another, we get sequential cohesion.

vii) Functional — It is the strongest cohesion. In a functionally bound module, all the elements of the module are related to performing a single function. Functions like "compute square root" and "sort the array" are examples of functionally cohesive modules.

Q. Discuss the function-point calculation in detail.

Ans:- Function Point (FP) is an element of software development which helps to approximate the cost of development early in the process. It may measure functionality from user's point of view.

Calculation of Function Point →

Step 1:

$$F = 14 * scale$$

Scale varies from 0 to 5 according to character of complexity Adjustment Factor (CAF).

The values of scale are defined as:

0 - no influence

1 - Incidental

2 - Moderate

3 - Average

4 - Significant

5 - Essential.

Step - 2: Calculate complexity Adjustment Factor,

$$CAF = 0.65 + (0.01 * F).$$

Step - 3: Calculate Unadjusted Function Point (UFP) from the table given. Multiply each individual function point to corresponding values in the table. $UFP = \sum_{i=1}^{5} \sum_{j=1}^{3} w_{ij} c_{ij}$

Step - 4: Calculate Function Point, $FP = UFP * CAF$.

Example:

Given the following values, compute function point when all complexity adjustment factor (CAF) and weighting factors are average.

User Input = 50

User Output = 40

User Enquiries = 35

User Files = 6

External Interface = 4.

FUNCTIONAL UNITS	WEIGHTING FACTORS		
	LOW	AVG	HIGH
External i/p (EI)	3	4	6
External o/p (EO)	4	5	7
External enquiries (EQ)	3	4	6
Internal Logic Files (ILF)	7	10	15
External Interface Files (EIF)	5	7	10

Ans:- Step-1: As CAF is average, so

scale = 3,

F = 14 * 3 = 42

Step-2: CAF = 0.65 + (0.01 * 42) = 1.07

Step-3: As weighting factors are also average, so we will multiply each individual function point to corresponding values in the table.

UFP = (50 * 4) + (40 * 5) + (35 * 4) + (6 * 10) + (4 * 7) = 628

Step-4: Function Point = 628 * 1.07 = 671.96 (Ans.)