Name: Muhammad Johar Shah

ID:14124

BSSE

Software Project Management

Submitted to: Sir Zain Shaukat

Ans 1: **Cost-benefit Analysis:**

It is a standard way to assess the economic benefits

It consists of two steps:

1. Identify and estimate all the costs and benefits of carrying out the project
2. Express the costs and benefits in a common unit for easy comparison

Costs

1. Development costs
2. Setup costs
3. Operational costs

Benefits

1. Direct benefits
2. Assessable indirect benefits
3. Intangible benefits

Cost-benefit Evaluation Techniques Example:

Year	*Project 1*	*Project 2*	*Project 3*	*Project 4*
0	-100,000	-1,000,000	-100,000	-120,000
1	10,000	200,000	30,000	30,000
2	10,000	200,000	30,000	30,000
3	20,000	200,000	30,000	30,000
4	20,000	200,000	20,000	25,000
5	100,000	350,000	20,000	50,000
Net Profit	60,000	150,000	30,000	45,000
Payback	5	5	4	4
ROI	12%	4%	6%	7.5%

Ans 2: Function Point Analysis (FPA)

FPA was Developed by A. Albrecht in IBM. The main aim is to estimate the LOC of a system (line of code). LOC of system= FP of system × LOC-per-FP of the language

FPA is a top-down approach.

Developed by Albrecht (1979) and later refined by Albrecht and Gaffney (1983)

Used in development with 3GL (3rd Generation Language)

LOC means Line of Code (programming statement)

For COBOL, the LOC per FP is 91.

For C, the LOC per FP is 128.

Idea: Software system consists of five major components (or, *external user types*)

- External input types
- External output types
- Logical internal file types
- External interface file types
- External inquiry types

Identify each instance of each external user type in the proposed system

Classify each instance as having high, medium or low complexity

Assign the FP of each instance

FP of the system = sum of FP of individual components

Function Point Analysis

Number of FPs	Complexity		
External user type	Low	Average	High
External input type	3	4	6
External output type	4	5	7
Logical internal file type	7	10	15
External interface file type	5	7	10
External inquiry type	3	4	6

Function Point Analysis – Example

A component of an inventory system consisting of ‘Add a record’, ‘Delete a record’, ‘Display a record’, ‘Edit a record’, and ‘Print a record’ will have

- 3 external input types
- 1 external output type
- 1 external inquiry type

Then, assign FPs based on the complexity of each type

Object Point Analysis

Similar to function point analysis

Used on 4GL development projects

Takes account of features that may be more readily identifiable if the system is built on high-level application building tools

Number of sections contained	Number and source of data tables		
	Total < 4 (<2 server, <2 client)	Total < 8 (2-3 server, 3-5 client)	Total 8+ (>3 server, >5 client)
< 2	Simple	Simple	Medium
2 or 3	Simple	Medium	Difficult
> 3	Medium	Difficult	Difficult

Steps:

Identify the number of screens, reports and 3GL components

Classify each object as Simple, Medium and Difficult

Assign the weight accordingly

Calculate the total object points

Total OP = sum of individual OP × weighting

Deduct the reused objects (r% reused)

NOP = OP × (1 – r%)

Identify the productivity rate of both developer and CASE

Number of views contained	Number and source of data tables		
	Total < 4 (<2 server, <2 client)	Total < 8 (2-3 server, 3-5 client)	Total 8+ (>3 server, >5 client)
< 3	Simple	Simple	Medium
3 – 7	Simple	Medium	Difficult
8+	Medium	Difficult	Difficult

Productivity rate = average of the two PRs

Calculate the effort

Effort = NOP / Productivity Rate

Object Point Analysis – Screens

Object Point Analysis – Reports

Object Point Analysis – Complexity Weightings

	Complexity		
Type of object	Simple	Medium	Difficult
Screen	1	2	3
Report	2	5	8
3GL component	N/A	N/A	10

Object Point Analysis – Productivity Rate

	Very low	Low	Nominal	High	Very High
Developer's experience and capability	4	7	13	25	50
CASE maturity and capability	4	7	13	25	50

Ans3:

Cocomo (Constructive cost model):

It is a parametric cost model in which important aspects of software projects are characterized by variables (or parameters). Once the value of the parameters are determined, the cost can be computed from an equation.

It recognizes different approaches to software development such as

Prototyping, Incremental development etc.

History:

COCOMO originally proposed by Boehm in 1981, now called COCOMO 81

Later evolved to *Ada COCOMO* in 1989

In 1995, Boehm proposed COCOMO II

The basic model equation

$$\text{Effort} = \text{Constant} \times (\text{Size})^{\text{scale factor}} \times \text{Effort Multiplier}$$

- Effort in terms of person-months
- Constant: 2.45 in 1998
- Size: Estimated Size in KSLOC
- Scale Factor: combined process factors
- Effort Multiplier (EM): combined effort factors

Constructive Cost Model (COCOMO) Example:

A new project with estimated 400 KLOC embedded system has to be developed. Project manager has a choice of hiring from two pools of developers with very high application experience and very little experience in the programming language being used or developers of very low application experience but a lot of experience with the programming language. What is the impact of hiring all developers from one or the other pool.

Solution:

This is the case of embedded mode:

Hence $E = a_i (KLOC)^{bi} * EAF$ $D = c_i(E)^{di}$

Case 1: Developers are with very high application experience and very little experience

in the programming language being used:

$$EAF = 0.82*1.14=09348$$

$$E = 2.8(400)^{1.20} * 09348 = 3470 \text{ PM}$$

$$D = 2.5(3470)^{0.32} = 33.9 \text{ M}$$

Case 2: Developers of very low application experience but a lot of experience with the programming language.

$$EAF = 1.29 * 0.95 = 1.22$$

$$E = 2.8 (400)^{1.20}*1.22 = 4528\text{PM}$$

$$D = 2.5(4528)^{0.32} = 36.9\text{M}$$

Case 2 requires more effort and time. Hence, low quality application experience but a lot of programming language experience could not match with the very high application experience and very little programming language experience