Software Project Estimations

Cost Management

Project Estimation

- The project manager must set expectations about the time required to complete the software among the stakeholders, the team, and the organization's management.
- If those expectations are not realistic from the beginning of the project, the stakeholders will not trust the team or the project manager.

Elements of a Sound Estimate

- To generate a sound estimate, a project manager must have:
  - A work breakdown structure (WBS), or a list of tasks which, if completed, will produce the final product
  - An effort estimate for each task
  - A list of assumptions which were necessary for making the estimate
  - Consensus among the project team that the estimate is accurate

Software estimates- steps

- The software estimation process begins with the project's functional requirements
  - Without defining the requirements, it is impossible to determine the cost and schedule for the software project
- Once the functional requirements are known, personnel can then be assigned to develop the estimate

Software estimates- steps

- The next step in the process is to develop the size estimate
  - The traditional definition of size is Source Lines of Code (SLOC)
  - Other methods exist for calculating size, such as function or feature points
- It is advisable to use more than one method to calculate size, so that results can be compared

Source Line of Code

- Used to measure the size of a software program by counting the number of lines in the text of the program's source code.
  - Typically used to predict the amount of effort that will be required to develop a program,
  - To estimate programming productivity or effort once the software is produced.
- SLOC is defined such that:
  - Only Source lines that are DELIVERED as part of the product are included -- test drivers and other support software is excluded
  - SOURCE lines are created by the project staff -- code created by applications generators is excluded
  - One SLOC is one logical line of code
    - A single SLOC may be several physical lines.
      - For example, an "if-then-else" statement would be counted as one SLOC
  - Declarations are counted as SLOC
  - Comments are not counted as SLOC

Function Point

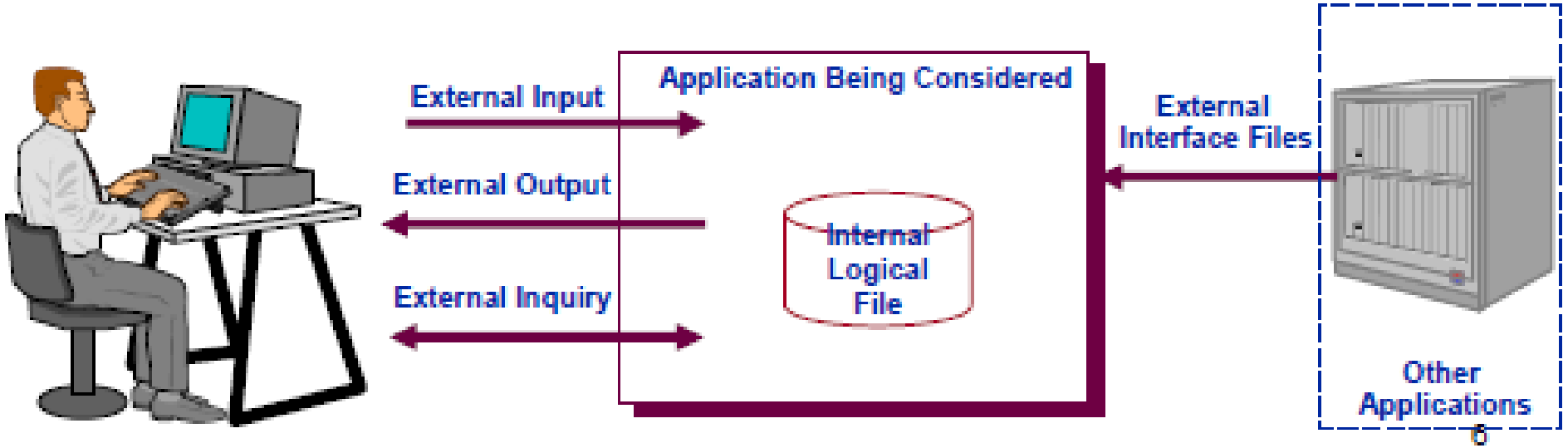
- A language independent approach to estimating software development effort
  - The technique is a measure of the size of computer applications and the projects that build them (function point measures the size)
    - The size is measured from a functional, or user, point of view
  - It is independent of
    - the computer language,
    - development methodology
  - counting lines of code measures software from the developers' point of view
  - Since function points are based on screens, reports and other external objects, this measure takes the users' view

Function Point Analysis

- Idea: Software system consists of five major components
  - External input types
    - Input transactions that update internal computer files
  - External output types
    - Transactions that output data to user such as report printing.
  - Logical internal file types
    - The standing file used by the system.
      - A group of data that is usually accessed together. It may have one or more record types.
      - Example: A PurchaseOrder may contain one or more PurchaseItems.
  - External interface file types
    - Input and output that may pass from and to other computer applications.
    - Files shared among applications would also be counted.
      - Example: the transmission of accounting data from an order processing system to the main ledger system.
  - External inquiry types
    - Transactions initiated by the user that provide information but do not update the internal files.

The Five Components of Function Points

- Data Handling Functions
  - Internal Logical Files
  - External Interface Files
- Data Transactional functions
  - External Inputs
  - External Outputs
  - External Inquiries



Software estimates- steps

- Once the size has been determined, the next step is to calculate the effort, cost, and schedule of the project using the size estimate that was calculated previously
  - Effort is usually calculated in person months, which can be translated into cost by using the applicable labor rates
- Software effort, cost and schedule are all interrelated and a change to one will affect the other two

Problems related to size estimation

- Nature of software
  - about its complexity
- Novel application of software
  - each time the software to be developed has some unique features.
- Fast changing technology and Lack of homogeneity of project experience
  - technology changes very fast, how well the personnel can manage the new technology is still not known yet.
- Political implications within the organization
  - The marketing director tends to push the product to be on the market at an early stage. Project manager may then have a tighter schedule as planned.

Assumptions Make Estimates More Accurate

- Team members make *assumptions* about the work to be done in order to deal with incomplete information
  - Any time an estimate must be based on a decision that has not yet been made, team members can assume the answer for the sake of the estimate
  - Assumptions must be written down so that if they prove to be incorrect and cause the estimate to be inaccurate, everyone understands what happened
  - Assumptions bring the team together very early on in the project so they can make progress on important decisions that will affect development

Wideband Delphi

- *Wideband Delphi* is a process that a team can use to generate an estimate
  - The project manager chooses an estimation team, and gains consensus among that team on the results
  - Wideband Delphi is a *repeatable* estimation process because it consists of a straightforward set of steps that can be performed the same way each time

Some project data- effort in work month and %age of total effort

	Design		Coding		Testing		Total	
Project	**Wm**	**%**	**Wm**	**%**	**Wm**	**%**	**Wm**	**SLOC**
A	3.9	23	5.3	32	7.4	44	16.7	6050
B	2.7	12	13.4	59	6.5	26	22.6	8363
C	3.5	11	26.8	83	1.9	6	32.2	13334
D	0.8	21	2.4	62	0.7	18	3.9	5942
E	1.8	10	7.7	44	7.8	45	17.3	3315
F	19.0	28	29.7	44	19.0	28	67.7	38988
G	2.1	21	7.4	74	0.5	5	10.1	38614
H	1.3	7	12.7	66	5.3	27	19.3	12762
I	8.5	14	22.7	38	28.2	47	59.5	26500

Productivity rate

Project	Wm	SLOC	Productivity SLOC/month
A	16.7	6050	362
B	22.6	8363	370
C	32.2	13334	414
D	3.9	5942	1524
E	17.3	3315	192
F	67.3	38988	576
G	10.1	38614	3823
H	19.3	12762	661
I	59.5	26500	445
overall	249.3	153868	617

When are estimates done?

- Estimates are carried out at various stages of a software project
  - **Strategic planning**:
    - The costs and benefits of potential applications might need to be estimated to help decide what priority to give to each project
    - Such estimates might also influence the numbers of various types of development staff to be recruited
  - **Feasibility study**
    - This ascertains that the potential system will justify the costs
    - Estimates at the design stage will also confirm that the feasibility study is still valid, taking into account all that has been learnt during detailed requirements analysis
  - **System specification**
    - The effort needed to implement different design proposals will need to be estimated

When are estimates done?

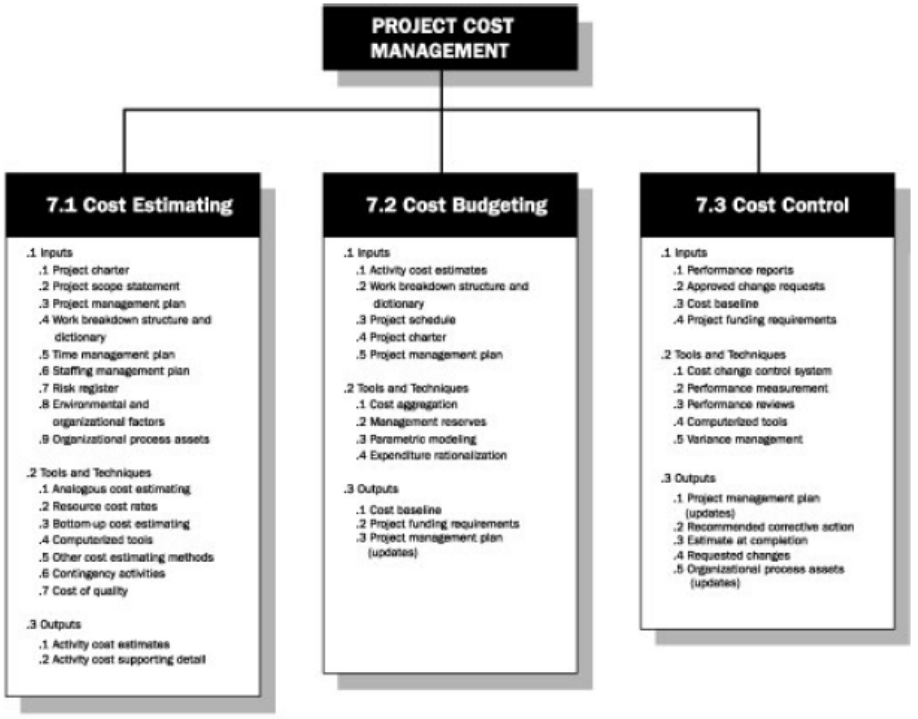
- Evaluation of suppliers' proposals
  - A bid need to scrutinize the system specification and produce estimates on which to base proposals
  - question a proposal that seems too low
    - whether the proposer had properly understood the requirements
  - If, the bids seem too high, reconsider
- Project planning
  - As the planning and implementation of the project progresses to greater levels of detail, more detailed estimates of smaller work components will be made.

over- and under-estimates

- A project leader will need to be aware that the estimate, if known to the development team, will influence the time required to implement the application
  - An over-estimate might cause the project to take longer than it would otherwise
  - This can be explained by the application of two 'laws'.
- Parkinson's Law '*Work expands to fill the time available*', which implies that given an easy target staff will work less hard.
- Brooks' Law '*putting more people on a late job makes it later'*
  - As the project team grows in size so will the effort that has to go into management, coordination and communication
- an over-estimate of the effort might lead to more staff being allocated than are needed and managerial overheads will be increased
- The under-estimate effect on quality

Project Cost Management Processes

- Cost is a resource sacrificed to achieve a specific objective or something given up in exchange
- There are three project cost management processes:
  - **Cost estimating**: developing an approximation or estimate of the costs of the resources needed to complete a project
  - **Cost budgeting**: allocating the overall cost estimate to individual work items to establish a baseline for measuring performance
  - **Cost control**: controlling changes to the project budget



Software cost components

- Estimates are made to discover the cost, to the developer, of producing a software system
- The major Cost Components:
  - Hardware and software costs
  - Travel and training costs
  - Effort costs (the dominant factor in most projects)
    - Salaries of engineers involved in the project
    - Insurance costs etc.
  - Effort costs must take overheads into account
    - Costs of building, heating, lighting
    - Costs of networking and communications
    - Costs of shared facilities (e.g library, staff restaurant, etc.)

Software Cost & efforts Estimation Tools and Techniques

- The Basic tools and techniques for cost estimates & deriving estimates of software development effort are:
  - **Analogous** or **top-down estimates:** use the actual cost of a previous, similar project as the basis for estimating the cost of the current project
  - **Bottom-up estimates:** involve estimating individual work items or activities and summing them to get a project total
  - **Parametric modeling** uses project characteristics (parameters) in a mathematical model to estimate project costs
  - **Computerized tools:** tools such as spreadsheets and project management software

- **Analogous**
  - Less costly but less accurate also
  - Depends on expertise of people involved (if new s/w used then not suitable)
- **Bottom-up**
  - More accurate as in WBS(work involved calculated by ppl who are doing the work)
  - Time consuming and difficult to develop

Compute FP

FP= UFP * CAF

- UFP– Unadjusted functional point
- CAF-- Complexity adjustment factor

Albrecht Complexity Multipliers

- After trial and error, empirical weighting factors were developed for the five items

Table 5.2 *Albrecht complexity multipliers*

External user type	*Multiplier*		
	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

Measurement Parameter	Counts		Weighting Factor Low	Average	High		
External Inputs (EI)		X	3	4	6	=	
External Outputs (EO)		X	4	5	7	=	+
External Enquired (EQ)		X	3	4	6	=	+
Internal Logic Files (ILF)		X	7	10	15	=	+
External Interface Files (EIF)		X	5	7	10	=	+

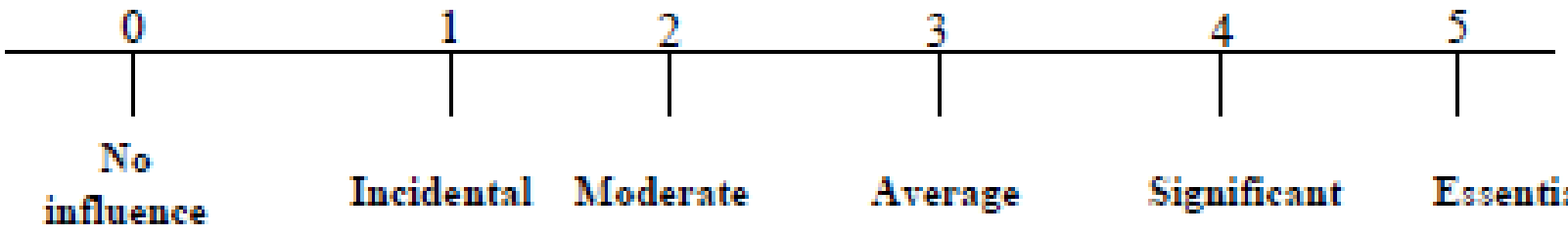
Unadjusted Function Points (UFP) = Total Count =

Technical Complexity

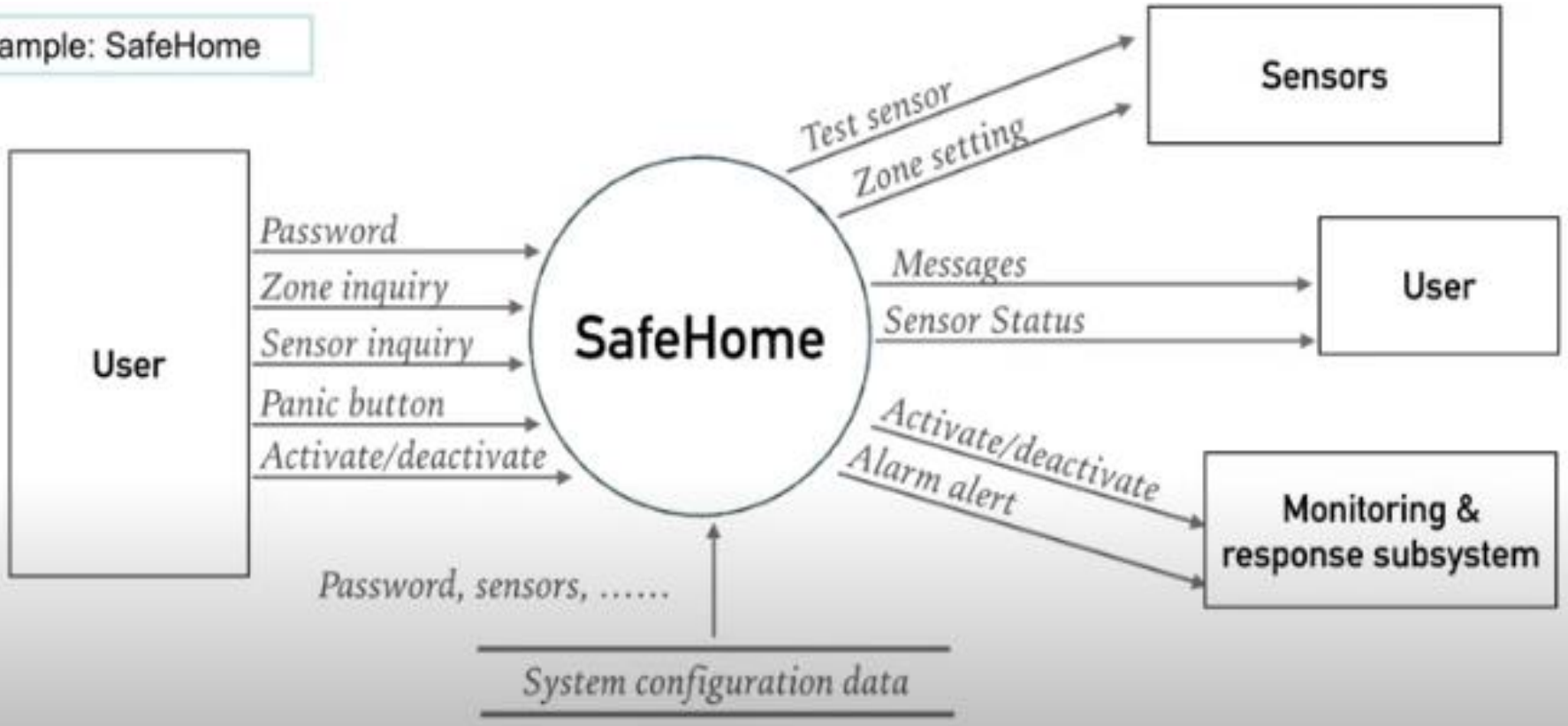
- The Unadjusted Function Point (**UFP**) count is multiplied by the Complexity Adjustment Factor (**CAF**)
- This factor considers the system's **technical** and **operational** characteristics (14 in all)

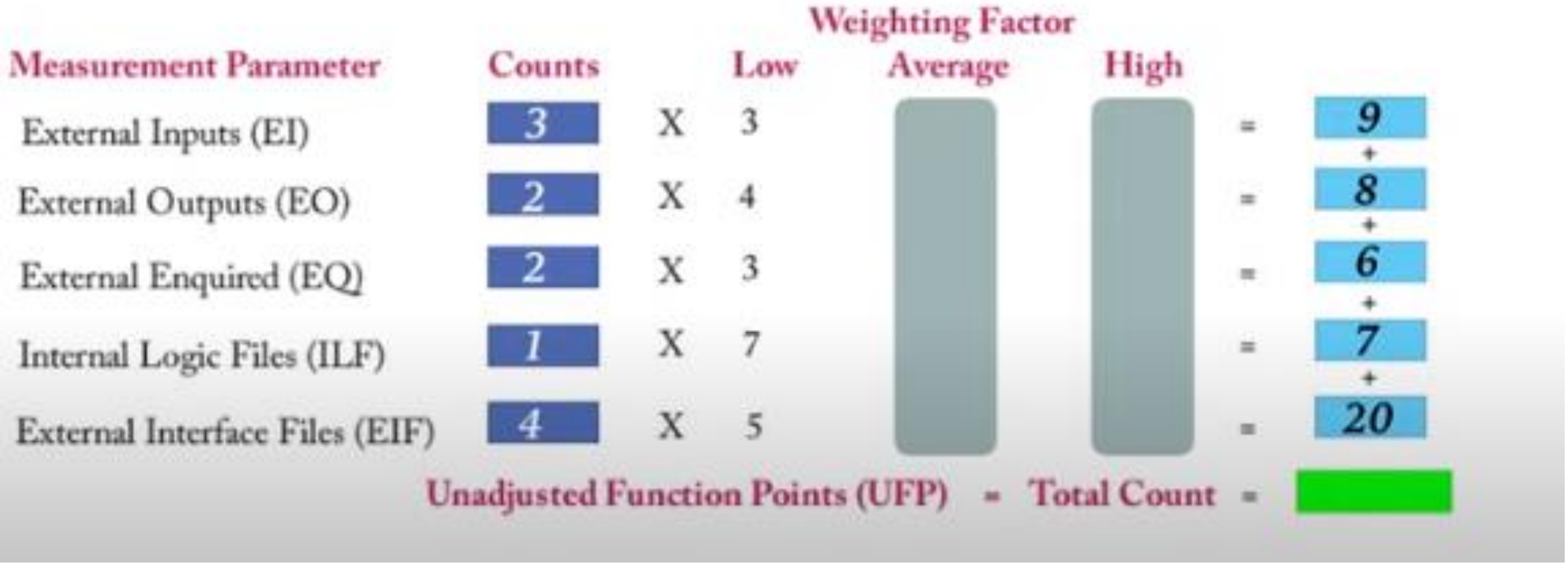
1) Data Communications
2) Distributed Data Processing
3) Performance
4) Heavily Used Configuration
5) On-line Data Entry
6) Transaction Rate
7) End -User Efficiency
8) On-line Update
9) Complex Processing
10) Reusability
11) Installation Ease
12) Operational Ease
13) Multiple Sites
14) Facilitate Change

Rate each technical complexity factor (TCF) on a scale of 0 to 5:



Example: SafeHome




Weighting Factor
Measurement Parameter
Counts
Low
Average
High
External Inputs (EI)
3
X
3
=
9
External Outputs (EO)
2
X
4
=
8
External Enquired (EQ)
2
X
3
=
6
Internal Logic Files (ILF)
1
X
7
=
7
External Interface Files (EIF)
4
X
5
=
20
Unadjusted Function Points (UFP)
=
Total Count
=

$$F.P = UFP \ X \ CAF$$

$$CAF = 0.65 + (0.01 \times \Sigma Fi)$$ Moderately complex product

Then,

$$\Sigma Fi = 14 \times 2 = 28$$ 2 - Moderate

$$CAF = 0.65 + (0.01 \times 28)$$

$$CAF = 0.93$$

$$F.P = UFP \ X \ CAF$$

$$F.P = 50 \ X \ 0.93 = 46.5$$

Example

Given the Following Values, calculate the Functional Point when complexity adjustment factors are significantly complex product and weighting factors are high.

User input = 55

User Output = 35

User Enquires = 40

User Files = 8

External Interfaces = 5

$$F.P = UFP \ X \ CAF$$

Where,

UFP = Sum of all the Complexities of all the EI's, EO's, EQ's, ILF's, and EIF's

$CAF = 0.65 + (0.01 \times \Sigma Fi)$

Solution

$$F.P = UFP \times CAF$$

$$CAF = 0.65 + (0.01 \times \Sigma Fi)$$

Significantly complex

Hence,

$$\Sigma Fi = 14 \times 4 = 56$$

$$CAF = 0.65 + (0.01 \times 56)$$

$$CAF = 1.21$$

$$F.P = UFP \times CAF$$

$$F.P = 985 \times 1.21 = 1191.85$$

Complexity factors

5 questions = Average
5 questions = Moderate
4 questions = No influence

$\Sigma Fi = (5 \times 3) + (5 \times 2) + (4 \times 0) = 25$

- **Based on the FP measure of software many other metrics can be computed:**
  1. Errors/FP
  2. $/FP.
  3. Defects/FP
  4. Pages of documentation/FP
  5. Errors/PM.
  6. Productivity = FP/PM (effort is measured in person-months).
  7. $/Page of Documentation.
- LOCs of an application can be estimated from FPs.

Example

Compute the function point, productivity, documentation, cost per function for the following data:

1. Number of user inputs = 24
2. Number of user outputs = 46
3. Number of inquiries = 8
4. Number of files = 4
5. Number of external interfaces = 2
6. Effort = 36.9 p-m
7. Technical documents = 265 pages
8. User documents = 122 pages
9. Cost = $7744/ month

- **Various processing complexity factors are: 4, 1, 0, 3, 3, 5, 4, 4, 3, 3, 2, 2, 4, 5.**

Measurement Parameter	Count		Weighing factor
1. Number of external inputs (EI)	24	*	4 = 96
2. Number of external outputs (EO)	46	*	4 = 184
3. Number of external inquiries (EQ)	8	*	6 = 48
4. Number of internal files (ILF)	4	*	10 = 40
5. Number of external interfaces (EIF) Count-total →	2	*	5 = 10 378

So sum of all f_i (i ← 1 to 14) = 4 + 1 + 0 + 3 + 5 + 4 + 4 + 3 + 3 + 2 + 2 + 4 + 5 = 43

FP = Count-total * [0.65 + 0.01 *$\sum(f_i)$]

= 378 * [0.65 + 0.01 * 43]

= 378 * [0.65 + 0.43]

= 378 * 1.08 = 408

$$\text{Productivity} = \frac{\text{FP}}{\text{Effort}} = \frac{408}{36.9} = 11.1$$

Total pages of documentation = technical document + user document

= 265 + 122 = 387pages

Documentation = Pages of documentation/FP

= 387/408 = 0.94

$$\text{Cost per function} = \frac{\text{cost}}{\text{productivity}} = \frac{7744}{11.1} = \$700$$

Object Points

- Object metrics involve estimating the size of the application in terms of the number of objects required to deliver the desired functionality
- Objects only – 3.G.L. components, reports, screens

Object Point Analysis – 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

Object Point Analysis – Steps (cont'd)

- Deduct the reused objects (r% reused)

$$NOP = OP \times (1 - r\%)$$

- Identify the productivity rate of both developer and CASE
- Productivity rate = average of the two PRs
- Calculate the effort

$$Effort = NOP / Productivity\ Rate$$

Object Point Analysis – Screens

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

Object Point Analysis – Reports

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

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

Example

Example:

Consider a database application project with

1. The application has four screens with four views each and seven data tables for three servers and four clients.
2. Application may generate two reports of six section each from seven data tables for two servers and three clients.

10% reuse of object points.

Developer's experience and capability in similar environment is low. Calculate the object point count, New object point and effort to develop such project.

Step-1:

Number of screens = 4

Number of records = 2

Step-2:

For screens,

Number of views = 4

Number of data tables = 7

Number of servers = 3

Number of clients = 4

by using above given information and table (For Screens),

Complexity level for each screen = medium

For reports,

Number of sections = 6

Number of data tables = 7

Number of servers = 2

Number of clients = 3

by using above given information and table (For Reports),

Complexity level for each report = difficult

Step-3:

By using complexity weight table we can assign complexity weight to each object instance depending upon their complexity level.

Complexity weight for each screen = 2

Complexity weight for each report = 8

Step-4:

```
Object point count
= sigma (Number of object instances) * (its Complexity weight)
= 4 * 2 + 2 * 8 = 24
```

Step-5:

```
%reuse of object points = 10% (given)
NOP = [object points * (100 - %reuse)]/100
= [24 * (100 -10)]/100 = 21.6
```

Step-6:

Developer's experience and capability is low (given)

Using information given about developer and productivity rate table

Productivity rate (PROD) of given project = 7

Step-7:

```
Effort
= NOP/PROD
= 21.6/7
= 3.086 person-month
```

Therefore, effort to develop the given project = 3.086 person-month.