Date: / / Page no:

Unit - 2

Measures, Metrices and Indicators

Software metrices -

A software metric is a measure of software characteristics which are measurable or countable. Software metrices are valuable for many reasons including measuring software performance, planning work items, measuring productivity, and many other uses.

Classification of software metrices -

Software metrices can be classified into two types -

1) Product metrics - These are the measures of various characteristics of the software product. The two important software characteristics are -
   1) Size and complexity of software
   2) Quality and reliability of software

2) Process metrics - These are the measures of various characteristics of the software development process. For example, the efficiency of fault detection. They are used to measure the characteristics of methods, techniques, and tools that are used for developing software.

3) Project metrices -
   It describes the project characteristics and execution.

Date: / / Page no:

Software metrics

Product metrics	Process metrics	Project metrics
1) Size	1) Effort required in the process.	1) No of software developers,
2) Complexity	2) Time to produce the product	2) Staffing pattern over the life cycle of software.
3) Design feature	3) Effect of development techniques and tools.	3) Cost
4) Performance	4) No of defects found during testing, etc.	4) Schedule
5) Quality level	5) Productivity	5) Productivity
6) Reliability	6) Quality	
7) Functionality		

Types of metrics -

1) Internal metrics - Internal metrics are the metrics used for measuring properties that are viewed to be of greater importance to a software developer. For example, Lines of Code (LOC) measure.
2) External metrics - External metrics are the metrics used for measuring properties that are viewed to be of greater importance to the user, eg. portability, reliability, functionality, etc.
3) Hybrid metrics - Hybrid metrics are the metrics that combine product, process and resource metrics. For example - cost per FP where FP = Function point metric.
4) Project metrics - Project metrics are the metrics used by the project manager to check the project's progress. Data from past projects are used to collect various metrics, like time and cost.

Date: / / Page no:

LOC metrics (Lines of code) -
It is one of the earliest and simplest metrics for calculating the size of the computer program. It is generally used in calculating and comparing the productivity of programmers. These metrices are derived by normalizing the quality and productivity measures by considering the size of the product as a metric.

Advantages	Disadvantages
1) Size oriented metric	LOC has no counting standards.
2) Language dependent	Not useful to compare hand written and auto-generated code.
3) Easy to compute.	Differs from person to person

~~Function~~ Based on LOC/KLOC count of software, many other metrics can be computed -

1) Errors / KLOC.
2) $ / KLOC
3) Defect / KLOC
4) Pages of documentation / KLOC
5) Errors / PM
6) Productivity = KLOC / PM

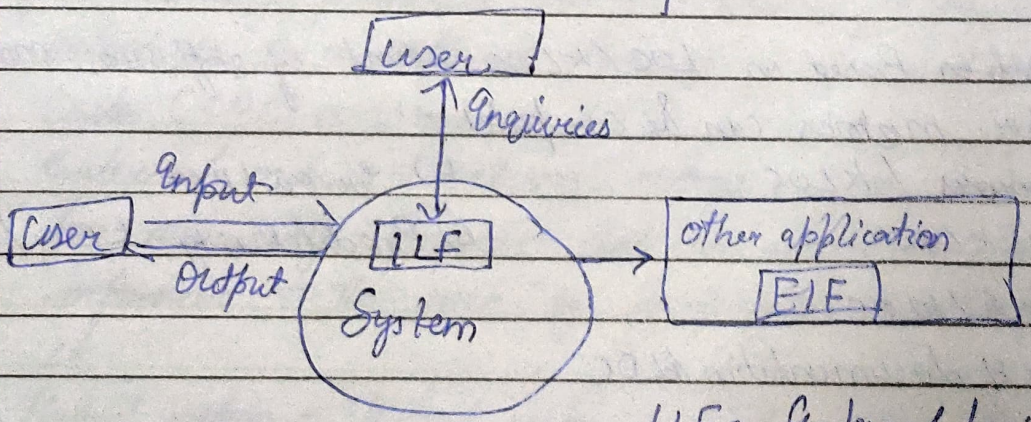
Functional point (FP) analysis -
FPA is used to make estimation of the software project, including its testing in terms of functionality or function size of the software product. The functional size of the product is measured in terms of the functional point, which is a standard of measurement to measure the software application.

Date / / Page no:

The basic and primary purpose of the functional point analysis is to measure and provide the software application functional size to the client, customer and the stakeholder on their request. Further, it is used to measure the software project development along with maintenance, consistency throughout the project irrespective of the tools and the technologies.

Types of FP attributes -

Measurement parameter	Example
1) Number of external inputs (EI)	Input screen and tables.
2) Number of external output (EO)	Output screens and reports
3) Number of external inquiries (EQ)	Prompts and interrupts
4) Number of internal files (ILF)	Databases and directories
5) Number of external interfaces (ELF)	Shared databases and shared routines.



ILF = Internal logical files
EIF = External interface

Date: / / Page no:

Advantages	Disadvantages
1) Size oriented metric	Manually counting process
2) Language dependent	Difficult to understand
3) Understood by the non-technical user.	Requires experience
4) to estimate cost and resources required for software development.	

Software quality metrics -
Software quality metrics are a subset of software metrics that focus on the quality aspects of the product, process and project. These are more closely associated with process and ~~project~~ product metrics than with project metrics.
Software quality metrics ~~include~~ can be further divided into -
1) Product quality metrics
2) In-process quality metrics
3) Maintenance quality metrics

Software reliability -
Software reliability means operational reliability. Software reliability is also defined as the probability that a software system fulfills its assigned task in a given environment for a predefined number of input cases, assuming that the hardware and the input are free of error.

Project estimation techniques -
Estimation of various project parameters is an important project planning activity. The different parameters of project that need to be estimated include -

Date: / / Page no:

1) Project size
2) Effort required to complete the project
3) Project duration and cost

COCOMO model-

Boehm proposed COCOMO model. It stands for constructive cost estimation model. COCOMO is one of the most used software estimation models in the world. COCOMO predicts the efforts and schedules of a software product based on the size of the software.

The baseline of COCOMO originally underlines a waterfall model lifecycle. The COCOMO model has basically two parameters like effort calculation and development time to define the quality of any software product.

Effort calculation - Efforts can be calculated by the number of persons required to complete the task successfully. It is calculated in unit person/month.

Development time - The time that is required to complete the task. It is calculated in the unit of time like months, weeks and days.

Software projects under COCOMO model strategies are classified into 3 categories -

1) Organic - A software project is said to be an organic type if -
   - Project is small and simple
   - Project team is small with prior experience
   - The problem is well understood and has been solved in the past.

Date: / / Page no:

- Requirements of projects are not rigid, such a mode example is payroll processing system.

2) Semi-detached mode -
- Project has complexity.
- Project team requires more experience, better guidance and creativity.
- The project has an intermediate size and has mixed rigid requirements such as mode example is a transaction processing system which has fixed requirements.
- It also includes the elements of organic mode and embedded mode.
- Few such projects are DBMS, new unknown OS.

3) Embedded mode -
- A software project has fixed requirements of resources.
- Product is developed within very tight constraints.
- A software project requiring the highest level of complexity, creativity, and experience requirement fall under this category.
- Such mode software requires a large team size than the other two modes.

Types of COCOMO models -

1) Basic COCOMO model -

The first level, basic COCOMO can be used for quick and slightly rough calculations of software costs. The basic COCOMO estimation model is given by the following expressions -

$E = a \times (KLOC)^b$

$D = c \times (\text{Effort}(E))^d$, $P = \text{effort} / \text{time}$

E = Effort applied in person-months.
D = developement time in months
P = total no. of persons required to accomplish the project.

The constant values a, b, c and d for the Basic model for the different categories of the system are -

Software project	A	B	C	D
Organic	2.4	1.05	2.5	0.38
Semi-detached	3	1.12	2.5	0.35
Embedded	3.6	1.20	2.5	0.32

2) Intermediate COCOMO model -
It is an extension of the Basic COCOMO model which includes a set of cost drivers into account in order to enhance more accuracy to the cost estimation model as a result.

Here, Effort (E) = $a \times (KLOC)^b \times EAF$

$D = c \times (Effort)^d$

E = Total effort required,
EAF = Effort adjustment factor, for ideal the value is 1.

Classification of cost drivers and their attributes -

1) Product attributes -
- Required software reliability extent
- Size of the application database
- The complexity of the product.

Date: / / Page no:

2) Hardware attributes -
- Run time performance constraints
- Memory constraints
- The volatility of the virtual machine environment
- Required turnabout time

3) Personal attributes -
- Analyst capability
- Software engineering capability
- Applications experience
- Virtual machine experience
- Programming language experience
-

4) Project attributes -
- Use of software tools
- Application of software engineering methods
- Required development schedule.

3) Detailed / Advanced COCOMO model -
The model incorporates all qualities of both Basic COCOMO and intermediate COCOMO strategies on each software engineering process. It uses multiplier for each phase of the project. It is a complex mode. It includes more factors that influence the projects and give more accurate estimations.
The six phases of detailed COCOMO model are -
1) Planning and requirements
2) System design
3) Detailed design
4) Module code and test
5) Integration and test
6) Cost constructive model

Date: / / Page no:

Advantages	Disadvantages
1) Cocomo is realistic and easy to interpret.	Cocomo model ignores requirement and all documentation.
2) Works on historical data and hence is more predictable and accurate.	It ignores customer skills, cooperation, knowledge and other parameters.
3) The drivers are very helpful to understand the impact on the different factors that affect the project cost.	It ignores hardware issues.
	It is dependent on the amount of time spent in each phase.

Project scheduling and tracking -
Scheduling in project management means to list out activities, deliverables and milestones within a project that are delivered. Effective project scheduling leads to success of project, reduced cost, and increased customer satisfaction.

Reverse Engineering -
Reverse engineering is also known as backward engineering, it is a process of forward engineering in reverse. In this, the information is collected from the given or existing application. It takes less time than forward engineering to develop an application. In reverse engineering, the application is broken to extract knowledge on its architecture.

Date: / / Page no:

Difference between forward and reverse engineering -

Forward Engineering	Reverse engineering
1) In forward engineering, the application are developed with the given requirements.	In reverse engineering, the information are collected from the given application.
2) Forward engineering is a high proficiency skill.	Reverse engineering is a low proficiency skill.
3) Forward engineering takes more time to develop an application.	Reverse engineering takes less time to develop an application.
4) The nature of forward engineering is perspective.	The nature of reverse engineering is adaptive.
5) In this, production is started with given requirements.	In this, production is started by taking the existing products.
6) The example of forward engineering is electronic kit, DC motors etc.	An example of backward engineering is research on instruments etc.