Date / /

Page 1

Module - 2

Project Planning :-

Once a project is found to be feasible then software project managers undertake project planning. This process is completed even before any development activity starts.

=) Project planning consists of the following essential activities.

(i) Estimating four basic attributes of the project.
   a) project size.
   b) cost.
   c). effort.
   d). duration.

(ii) Scheduling men power & other resources.

(iii) Staff organization & staffing plans.

(iv) Risk identification, analysis and planning.

(v) Miscellaneous plans such as quality assurance plan, configuration management plan etc.

Matrices of for software project size estimation.

In order to accurately estimate the project size some important matrix should be defined in terms of which the project size can be expressed.

LOC (Line of Code) :→

=) It is the simplest among all matrics available to estimate project size. In this the project size is estimated by counting the no. of source instructions in the developed program.

- Lines used for commenting & the header lines should be ignored.
- Determining the LOC at the end of project is very simple However accurate estimation of LOC at the beignning is very difficult.
- In order to estimate LOC count at the beginning of a project Project managers usually divide the problem into Modules and each Modules into sub modules & so on until the size of the different leaf levels can be predicted approximately.

Function Point :- (Numerical)

? This matric overcome many of the shortcomings of LOC

- One of its advantage is that it can easily estimate the size of software product directly from the problem specification.

UFP = # no. of inputs x 4 + no. of outputs x 5 + no. of enquiries x 4 + no. of files x 10 + no. of interfaces x 10

UFP => Unadjusted function point.

Function point matric can be used to compute size of a software product using no. of input values and output data values along with no. of enquiries, files and interfaces.

No. of inputs :- each data item input by the user is counted. Data inputs should be distinguished from user enquiries. (Enquiries are user commands such as Print account balance).
Individual data items input by the user are not considered in the calculation in the input but a group of related inputs are considered as single input.

No. of outputs :- It refers to reports printed, screen outputs, error messages produced. In this also individual data items within a report are not considered but a set of related data items are counted as one.

No. of enquiries :- It is no. of distinct interactive queries which can be made by the users.

No. of files :- Each logical file is counted where a logical file means group of logically related data.

No. of interfaces :- These are the interfaces used to exchange information with other systems.
For ex:- data files on tapes, disks, communication links with other system etc.

3/08/17

Cocomo Model :- (Constructive cost model)
There are 3 categories of software projects in cocomo model
(i) Organic
(ii) Semi-detached
(iii) Embedded

Organic :→
If the project deals with developing a well understood application program where the size of development team is reasonably small & the team members are experienced in developing similar type of projects.

Semi-detached :→
If the development consists of mixture of experience and inexperienced staff. Team members may have limited experience on the related system but may be unfamiliar with some aspects of the system being developed.

Embedded :→
If the software being developed is strongly coupled to complex hardware.

Numerical (100%) Basic cocomo model :→

1. Effort :
$$A_1 \times (KLOC)^{A_2} \quad \text{per month.}$$

2. Time : T_{dev} (development time)
$$= B_1 \times (Effort)^{B_2}$$ → per month. / person month

KLOC is the estimated size of the software product expressed in KILO lines of code.
A_1, A_2, B_1, B_2 → are constants for each category of software products.

T_{dev} is the estimated time to develop the software

expressed in months.
Effort is the effort required to develop a software product. It is expressed in per month. - (person month)

	A_1	A_2	B_1	B_2
Organic	2.4	1.05	2.5	0.38
Semi-detatched	3.0	1.152	2.5	0.35
Embedded	3.86	1.20	2.5	0.32

Q. Project → Organic
KLOC → 1000

Effort ⇒ $2.4(1000)^{1.05}$

time ⇒ $2.5 \times [(2.4)(1000)^{1.05}]^{0.38}$

08/08/17

Organic → 2-50 KLOC (small, stable), little innovation
Semi-detatched → 50-300 KLOC (medium size, average abilities, medium time constraints)
Embedded → More than 300 KLOC (large project team, complex, innovative)

Q. 200 KLOC
organic, Semi-detached, Embedded compare Staff month.

↳ Effort.

Organic ⇒ $2.4(200)^{1.05} = 625.594 \approx 626$

Semi-detatched ⇒ $3.0(200)^{1.12} = 1133$

Embedded ⇒ $3.6(200)^{1.2} = \cancel{664}\ 2077.487$

development time

D ⇒ $2.5[(2.4)(200)^{1.05}]^{0.38} = 28.88 \approx 29$

S.D. = $2.5[(3.0)(200)^{1.12}]^{0.35} = \approx 29$

E. = $2.5[(3.6)(200)^{1.2}]^{0.32} \approx 29$

Staff size :→ Effort / time
(Avg. Staff size)

productivity :⇒ $\frac{\text{size}}{\text{Effort}}$ → (KLOC)

Q% A project has 7.5 KLOC, find effort, development time, staff size & productivity?

Effort = $2.4(7.5)^{1.05}$ ⇒ $19.9 \approx 20$

T_{dev} ⇒ $2.5[(2.4)(7.5)^{1.05}]^{0.38}$ ⇒ $7.8 \approx 8$

Staff size ⇒ $\frac{19.9}{7.8}$ ⇒ 2.55

productivity ⇒ $\frac{2.55}{20} = 0.12751$

LOC ← $\frac{7500}{20} = 375$

Q4. Suppose an embedded project has 50 KLOC, find development time, Effort, Staff size, productivity.

Effort $\Rightarrow$ $3.0(50)^{1.12} \simeq 393.6 \simeq \simeq 394$

T_{dev} : $(2.5)\left[(3.0)(50)^{1.12}\right]^{0.35} \simeq 17$

Staff size : $\frac{\text{Effort}}{\text{time}} = \frac{394}{17} = 23.17 \simeq 23$

productivity : $\frac{50\,000}{394} \simeq 126.5$

Putnam Model :-

It is an empirical software effort estimation model. It describes time and effort required to finish a software project of a specified size. It is closely related to constructive cost model (cocomo).

$$\text{Effort} = \left[\frac{\text{size}}{\text{Productivity} \times (\text{time})^{4/3}}\right]^3 \times B$$

Size → it is the product size in ESLOC (Effective source line of code)

B → it is a scaling vector & is a function of project size.

Productivity → It is the ability of a particular software organization to produce software of a given size at a particular rate.

Effort → It is the total effort applied to the project in person years.

Time → It is the total schedule of the project in years.

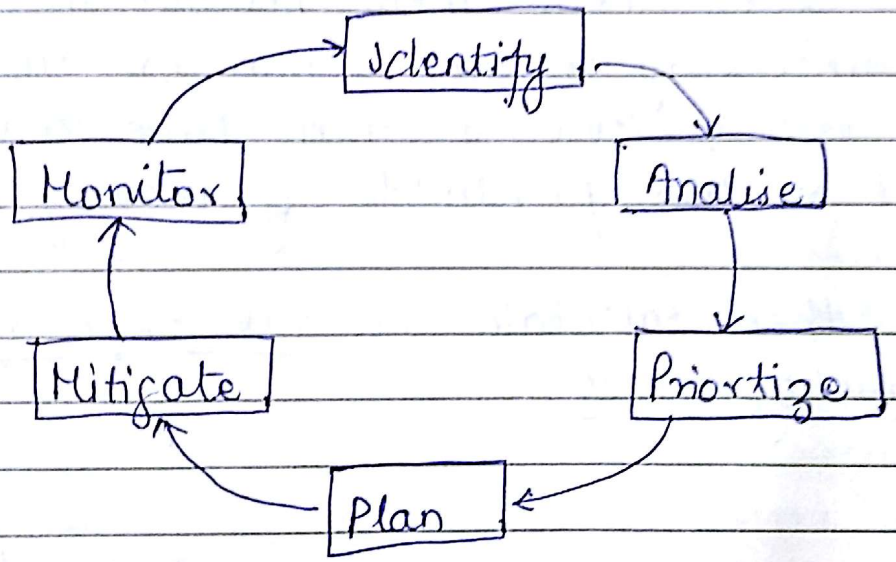
Productivity - Size

10/08/17

Risk Management :-

A risk is a potential future harm that may arise from present actions. Risk management is a series of step whose objectives are to identify, address and eliminate software risk items before they become either threats to successful software operation or a major source of expensive rework.

Steps in risk management cycle :-



Risk Identification : In this step team systematically enumerates as many project risk as possible to make them explicit before they becomes problem.

Categories of risks

i) Generic

(ii) Product specific.
(iii) Project risks.
(iv) Product risks.
(v) Buisness risks.

2) Analize :- After risk have been identified and enumerated next step is risk analysis. In this step we transform the risk into desision making information.
For each risk team must do the following :-
1). Assess the probability of a loss occuring.
2). Assess the impact of the loss if it occured.
3). Pr

3). Prioritize :- After step 2 the team prioritize the risk after ranking them because it is too costly & unnecessary to take action on every identified risk as some of them have a very low impact or low probability of occuring (or both)

The team then calculate the risk exposure (probability x cost)

Ex :-

A	0.01	1,00,000	₹ 1000 loss
B	0.1	100	₹ 10 loss (still better)

4). Plans :- Risk management plans should be developed for each of the "above the line" prioritize the risk so that pro active actions can take place.

Some of the examples of risk planning actions are information buying, (ii) contingency plan (iii) risk reduction (iv) Risk acceptance.

5). Mitigate :→ Through risk mitigation team develops strategies to reduce the possibility or the loss/impact of a risk.
Examples of mitigation strategies :→ Risk avoidance, risk protection.

Risk leverage :→ $\frac{(\text{Risk exposure before reduction} - \text{risk exposure after reduction})}{\text{cost of risk reduction}}$

If risk leverage ≤ 1, then the benefit of applying risk reduction is not worthy of its cost. But if Risk reduction is ≥ 1, then the benefits are questionable and should (may) be applied.

6). Monitor :→ After risks are identified, analyze & prioritize & actions are established, it is essential that the team regularly monitors the process of the product & the resolution of the risk items, taking corrective actions when necessary.

27/8/17