**Unit – IV**

**System Design**

1. **System Planning and Initial investigation**

**Initial Investigation**

• This is the first phase of SDLC and is known as identification of need.

• This is a user’s request to change, improve or enhance an existing system.

• The objective is to determine whether the request is valid or feasible

• The user request identifies the need for change and authorizes the initial investigation.

**System Planning**

Once a system is found to be feasible, software project managers undertake system planning. System planning is undertaken and completed even before any development activity starts. System planning consists of the following essential activities:

• Estimating the following attributes of the system:

**System size**: What will be problem complexity in terms of the effort and time required to develop the system?

**Cost**: How much is it going to cost to develop the system?

**Duration**: How long is it going to take to complete development?

**Effort**: How much effort would be required?

The effectiveness of the subsequent planning activities is based on the accuracy of these estimations.

• Scheduling manpower and other resources

• Staff organization and staffing plans

• Risk identification, analysis, and abatement planning

• Miscellaneous plans such as quality assurance plan

1. **Project scheduling**

Project-task scheduling is an important project planning activity. It involves deciding which tasks would be taken up when. In order to schedule the project activities, a software project manager needs to do the following:

* Identify all the tasks needed to complete the project.
* Break down large tasks into small activities.
* Determine the dependency among different activities.
* Establish the most likely estimates for the time durations necessary to complete the activities.
* Allocate resources to activities.
* Plan the starting and ending dates for various activities.
* Determine the critical path. A critical path is the chain of activities that determines the duration of the project.

The first step in scheduling a software project involves identifying all the tasks necessary to complete the project. A good knowledge of the intricacies of the project and the development process helps the managers to effectively identify the important tasks of the project. Next, the large tasks are broken down into a logical set of small activities which would be assigned to different engineers. The work breakdown structure formalism helps the manager to breakdown the tasks systematically.

After the project manager has broken down the tasks and created the work breakdown structure, he has to find the dependency among the activities.

Dependency among the different activities determines the order in which the different activities would be carried out. If an activity A requires the results of another activity B, then activity A must be scheduled after activity B. In general, the task dependencies define a partial ordering among tasks, i.e. each tasks may precede a subset of other tasks, but some tasks might not have any precedence ordering defined between them (called concurrent task). The dependencies among the activities are represented in the form of an activity network.

Once the activity network representation has been worked out, resources are allocated to each activity. Resource allocation is typically done using a Gantt chart. After resource allocation is done, a PERT chart representation is developed. The PERT chart representation is suitable for program monitoring and control. For task scheduling, the project manager needs to decompose the project tasks into a set of activities. The time frame when each activity is to be performed is to be determined. The end of each activity is called milestone. The project manager tracks the progress of a project by monitoring the timely completion of the milestones. If he observes that the milestones start getting delayed, then he has to carefully control the activities, so that the overall deadline can still be met.

**Work breakdown structure**

Work Breakdown Structure (WBS) is used to decompose a given task set recursively into small activities. WBS provides a notation for representing the major tasks need to be carried out in order to solve a problem. The root of the tree is labeled by the problem name. Each node of the tree is broken down into smaller activities that are made the children of the node. Each activity is recursively decomposed into smaller sub-activities until at the leaf level, the activities requires approximately two weeks to develop. Fig. represents the WBS of an MIS (Management Information System) software.

While breaking down a task into smaller tasks, the manager has to make some hard decisions. If a task is broken down into large number of very small activities, these can be carried out independently. Thus, it becomes possible to develop the product faster (with the help of additional manpower).

Therefore, to be able to complete a project in the least amount of time, the manager needs to break large tasks into smaller ones, expecting to find more parallelism. However, it is not useful to subdivide tasks into units which take less than a week or two to execute. Very fine subdivision means that a disproportionate amount of time must be spent on preparing and revising various charts.

MIS Application

Requirements Specification

Design

Code

Test

Document

Graphical User Interface Part

Database Part

Graphical User Interface Part

Database Part

Fig. Work breakdown structure of an MIS problem

**Activity Networks**

WBS representation of a project is transformed into an activity network by representing activities identified in WBS along with their interdependencies. An activity network shows the different activities making up a project, their estimated durations, and interdependencies (as shown in fig. ). Each activity is represented by a rectangular node and the duration of the activity is shown

along side each task.

Code Database Part

105

Design Database Part

45

15

Integrate and Test 120

Specification

15

Finish

0

Code GUI Part

30

15

Design GUI Part

30

15

Write User Manual

60

Fig. Activity network representation of the MIS problem

Managers can estimate the time durations for the different tasks in several ways. One possibility is that they can empirically assign durations to different tasks. This however is not a good idea, because software engineers often resent such unilateral decisions. A possible alternative is to let engineer himself estimate the time for an activity he can assigned to. However, some managers prefer to estimate the time for various activities themselves. Many managers believe that an aggressive schedule motivates the engineers to do a better and faster job. However, careful experiments have shown that unrealistically aggressive schedules not only cause engineers to compromise on intangible quality aspects, but also are a cause for schedule delays. A good way to achieve accurately in estimation of the task durations without creating undue schedule pressures is to have people set their own schedules.

**Critical Path Method (CPM)**

From the activity network representation following analysis can be made. The minimum time (MT) to complete the project is the maximum of all paths from start to finish. The earliest start (ES) time of a task is the maximum of all paths from the start to the task. The latest start time is the difference between MT and the maximum of all paths from this task to the finish. The earliest finish time (EF) of a task is the sum of the earliest start time of the task and the duration of the task.

The latest finish (LF) time of a task can be obtained by subtracting maximum of all paths from this task to finish from MT. The slack time (ST) is LS – EF and equivalently can be written as LF – EF. The slack time (or float time) is the total time that a task may be delayed before it will affect the end time of the project.

The slack time indicates the “flexibility” in starting and completion of tasks. A critical task is one with a zero slack time. A path from the start node to the finish node containing only critical tasks is called a critical path. These parameters for different tasks for the MIS problem are shown in the following table.

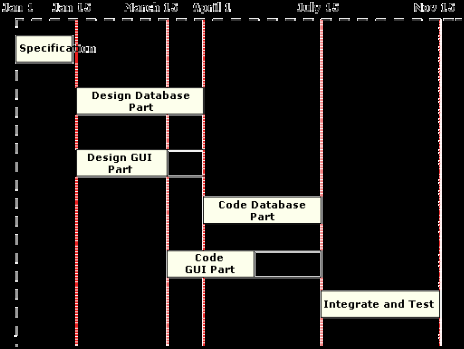
|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Task** | **ES** | **EF** | **LS** | **LF** | **ST** |
| Specification | 0 | 15 | 0 | 15 | 0 |
| Design Database | 15 | 60 | 15 | 60 | 0 |
| Design GUI Part | 15 | 45 | 90 | 120 | 75 |
| Code Database | 60 | 165 | 60 | 165 | 0 |
| Code GUI Part | 45 | 90 | 120 | 165 | 75 |
| Integrate and Test | 165 | 285 | 165 | 285 | 0 |
| Write User Manual | 15 | 75 | 225 | 285 | 210 |

**Gantt chart**

Gantt charts are mainly used to allocate resources to activities. The resources allocated to activities include staff, hardware, and software. Gantt charts (named after its developer Henry Gantt) are useful for resource planning. A Gantt chart is a special type of bar chart where each bar represents an activity. The bars are drawn along a time line. The length of each bar is proportional to the duration of time planned for the corresponding activity.

Gantt charts are used in software project management are actually an enhanced version of the standard Gantt charts. In the Gantt charts used for software project management, each bar consists of a white part and a shaded part. The shaded part of the bar shows the length of time each task is estimated to take. The white part shows the slack time, that is, the latest time by which a task must be finished.

A Gantt chart representation for the MIS problem is shown in the fig.



Write User Manual

Fig. Gantt chart representation of the MIS problem

1. **Requirement Analysis:**

Requirements analysis is the first stage in the systems engineering process and software development process.

Requirements analysis in systems engineering and software engineering, encompasses those tasks that go into determining the needs or conditions to meet for a new or altered product, taking account of the possibly conflicting requirements of the various stakeholders, such as beneficiaries or users.

Requirements analysis is critical to the success of a development project. Requirements must be actionable, measurable, testable, related to identified business needs or opportunities, and defined to a level of detail sufficient for system design. Requirements can be functional and non-functional.

**Overview**

Conceptually, requirements analysis includes three types of activity:

* Eliciting requirements: the task of communicating with customers and users to determine what their requirements are. This is sometimes also called requirements gathering.
* Analyzing requirements: determining whether the stated requirements are unclear, incomplete, ambiguous, or contradictory, and then resolving these issues.
* Recording requirements: Requirements might be documented in various forms, such as natural-language documents, use cases, user stories, or process specifications.

Requirements analysis can be a long and arduous process during which many delicate psychological skills are involved. New systems change the environment and relationships between people, so it is important to identify all the stakeholders, take into account all their needs and ensure they understand the implications of the new systems. Analysts can employ several techniques to elicit the requirements from the customer. Historically, this has included such things as holding interviews, or holding focus groups (more aptly named in this context as requirements workshops) and creating requirements lists. More modern techniques Software Development Process – activities and steps include prototyping, and use cases. Where necessary, the analyst will employ a combination of these methods to establish the exact requirements of the stakeholders, so that a system that meets the business needs is produced.

**Types of Requirements**

Requirements are categorized in several ways. The following are common categorizations of requirements that relate to technical management:

Customer Requirements

Statements of fact and assumptions that define the expectations of the system in terms of mission objectives, environment, constraints, and measures of effectiveness and suitability (MOE/MOS). The customers are those that perform the eight primary functions of systems engineering, with special emphasis on the operator as the key customer. Operational requirements will define the basic need and, at a minimum, answer the questions posed in the following listing:

* Operational distribution or deployment: Where will the system be used?
* Mission profile or scenario: How will the system accomplish its mission objective?
* Performance and related parameters: What are the critical system parameters to accomplish the mission?
* Utilization environments: How are the various system components to be used?
* Effectiveness requirements: How effective or efficient must the system be in performing its mission? Software Development Process – activities and steps
* Operational life cycle: How long will the system be in use by the user?
* Environment: What environments will the system be expected to operate in an effective manner?

Functional Requirements

Functional requirements explain what has to be done by identifying the necessary task, action or activity that must be accomplished. Functional requirements analysis will be used as the top level functions for functional analysis.

Non-functional Requirements

Non-functional requirements are requirements that specify criteria that can be used to judge the operation of a system, rather than specific behaviors.

Performance Requirements

The extent to which a mission or function must be executed; generally measured in terms of quantity, quality, coverage, timeliness or readiness. During requirements analysis, performance (how well does it have to be done) requirements will be interactively developed across all identified functions based on system life cycle factors; and characterized in terms of the degree of certainty in their estimate, the degree of criticality to system success, and their relationship to other requirements.

Design Requirements

The ―build to,‖ ―code to,‖ and ―buy to‖ requirements for products and ―how to execute‖ requirements for processes expressed in technical data packages and technical manuals.

Derived Requirements

Requirements that are implied or transformed from higher-level requirement. For example, a requirement for long range or high speed may result in a design requirement for low weight.

Allocated Requirements

A requirement that is established by dividing or otherwise allocating a high-level requirement into multiple lower-level requirements. Example: A 100-pound item that consists of two subsystems might result in weight requirements of 70 pounds and 30 pounds for the two lower-level items.

**Requirement Gathering Methods**

**Gathering Requirements From All Parties**

Techniques involving visualization of the requirements like storyboards, prototypes, scenarios are helpful when you have a business user who may not be worried about the ins and outs of technical solution or have long attention duration for legalizing the requirements with users to let the analyst drive his discovery efficiently than just reading a document with a prospective user.

[The requirement gathering techniques may differ](http://www.brighthubpm.com/project-planning/13669-gathering-requirements-for-a-feasibility-study/) from one project to another. Some [requirement gathering](http://images.brighthub.com/media/74C016_abcvideo_gatheringrequirements.pdf) techniques may prove highly beneficial for you in one project but may not be as productive in the other project or for some other company. Therefore the usefulness of a technique is determined by its need and the kind of advantages it offers in a particular project. There are 10 essential requirement gathering techniques that you must be aware of in order to manage the projects in a better way and run your business successfully are:

* 1. [Brainstorming](http://www.brighthubpm.com/risk-management/48947-effective-brainstorming-methods/)
  2. Document Analysis
  3. Focus Group
  4. Interface Analysis
  5. Interview
  6. Observation
  7. Prototyping
  8. Requirements Workshop
  9. Reverse Engineering
  10. Survey

**1. Brainstorming**

[](http://img.bhs4.com/f4/3/f43f6ab7901d5bede3f9e1706b4233b785a820a1_large.jpg)It is utilized in requirements elicitation to gather good number of ideas from a group of people. Usually brainstorming is used in identifying all possible solutions to problems and simplifies the detail of opportunities. It casts a broad net, determining various discreet possibilities. Prioritization of such possibilities is vital to locate needles in haystack.

**2. Document Analysis**

Document Analysis is an important gathering technique. Evaluating the documentation of a present system can assist when making AS-IS process documents and also when driving the gap analysis for scoping of the migration projects. In today’s world, you will also be determining the requirements that drove making of an existing system- a beginning point for documenting all current requirements. Chunks of information are mostly buried in present documents that assist you in putting questions as a part of validating the requirement completeness.

**3. Focus Group**

A focus group is actually gathering of people who are customers or users representatives for a product to gain its feedback. The feedback can be collected about opportunities, needs, and problems to determine requirements or it can be collected to refine and validate the already elicited requirements. This type of market research is different from brainstorming in which it is a managed process with particular participants. There is a risk in following the crowd and some people think that focus groups are at best unproductive. One danger that we usually end up with is with least common denominator features.

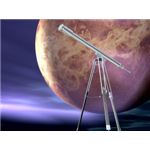
**4. Interface Analysis**

Interface for any software product will either be human or machine. Integration with external devices and systems is another interface. The user centric design approaches are quite effective to ensure that you make usable software. Interface analysis- analyzing the touch points with another external system- is vital to ensure that you do not overlook requirements that are not instantly visible to the users.

**5. Interview**

Interviews of users and stakeholders are important in creating wonderful software. Without knowing the expectations and goal of the stakeholders and users, you are highly unlikely to satiate them. You also have to understand the perspective of every interviewee, in order to properly address and weigh their inputs. Like a good reporter, listening is a quality that assists an excellent analyst to gain better value through an interview as compared to an average analyst**.**

**6. Observation**

[](http://img.bhs4.com/d8/d/d8d32678ee0a7f48b2f62bc72f79695034fd40f5_large.jpg)The observation covers the study of users in its natural habitat. By watching users, a process flow, pain points, awkward steps and opportunities can be determined by an analyst for improvement. Observation can either be passive or active. Passive observation is provides better feedback to refine requirements on the same hand active observation works best for obtaining an understanding over an existing business process. You can use any of these approaches to uncover the implicit requirements that are often overlooked.

**7. Prototyping**

Prototyping can be very helpful at gathering feedback. Low fidelity prototypes make a good listening tool. Many a times, people are not able to articulate a specific need in the abstract. They can swiftly review whether a design approach would satisfy the need. Prototypes are very effectively done with fast sketches of storyboards and interfaces. Prototypes in some situations are also used as official requirements.

**8. Requirements Workshop**

Popularly known as JAD or joint application design, these workshops can be efficient for [gathering requirements.](http://www.brighthubpm.com/monitoring-projects/50286-introduction-to-requirements-management/) The requirements workshops are more organized and structured than a brainstorming session where the involved parties get together to document requirements. Creation of domain model artifacts like activity programs or static diagrams is one of the ways to capture the collaboration. A workshop with two analysts is more effective than one in which on works as a facilitator and the other scribes the work together.

**9. Reverse Engineering**

Is this a last resort or starting point? When a migration project is not having enough documentation of the current system, reverse engineering will determine what system does? It will not determine what the thing went wrong with the system and what a system must do?

**10. Survey**

When gathering information from many people: to many to interview with time constraints and less budget: a questionnaire survey can be used. The surveyinsists the users to choose from the given options agree / disagree or rate something. Do not think that you can make a survey on your own but try to add meaningful insight in it. A well designed survey must give qualitative guidance for characterizing the market. It should not be utilized for prioritizing of [requirements](http://images.brighthub.com/media/74C016_abcvideo_gatheringrequirements.pdf) or features.

**Feasibility Study**

**What is a Feasibility Study?**

A feasibility study is an analysis of the viability of an idea through a disciplined and documented process of thinking through the idea from its logical beginning to its logical end.

A feasibility study provides an Investigating function that helps answer “Should we proceed with the proposed project idea? Is it a viable business venture?”

A feasibility study should be conducted to determine the viability of an idea BEFORE proceeding with the development of a business.

**Why feasibility study?**

**Objectives:**

* To find out if an system development project can be done:
* ...is it possible?
* ...is it justified?
* To suggest possible alternative solutions.
* To provide management with enough information to know:
* Whether the project can be done
* Whether the final product will benefit its intended users
* What the alternatives are (so that a selection can be made in subsequent phases)
* Whether there is a preferred alternative

**Types of feasibility**

* Technical
* Economic
* Schedule
* Operational

**Four Types of feasibility**

1. **Technical feasibility**

* Is the project possible with current technology?
* What technical risk is there?
* Availability of the technology:
* Is it available locally?
* Can it be obtained?
* Will it be compatible with other systems?

1. **Economic feasibility**

* Is the project possible, given resource constraints?
* What are the benefits?
* Both tangible and intangible
* Quantify them!
* What are the development and operational costs?
* Are the benefits worth the costs?

1. **Schedule feasibility**

* Is it possible to build a solution in time to be useful?
* What are the consequences of delay?
* Any constraints on the schedule?
* Can these constraints be met?

1. **Operational feasibility**

* If the system is developed, will it be used?
* Human and social issues…
* Potential labour objections?
* Manager resistance?
* Organizational conflicts and policies?
* Social acceptability?
* legal aspects and government regulations?

**Steps of Feasibility Study**

**1. Purpose & scope of the study**

* Objectives (of the study)
* who commissioned it & who did it,
* sources of information,
* process used for the study,
* how long did it take,…

**2. Description of present situation**

* Organizational setting, current system(s).
* Related factors and constraints.

**3. Problems and requirements**

* What’s wrong with the present situation?
* What changes are needed?

**4. Objectives of the new system.**

* Goals and relationships between them

**5. Possible alternatives**

* …including ‘do nothing’.

**6. Criteria for comparison**

* definition of the criteria

**7. Analysis of alternatives**

* description of each alternative
* evaluation with respect to criteria
* cost/benefit analysis and special implications.

**8. Recommendations**

* what is recommended and implications
* what to do next;
* E.g. may recommend an interim solution and a permanent solution

# Cost Benefit Analysis

A cost benefit analysis is done to determine how well, or how poorly, a planned action will turn out. Although a cost benefit analysis can be used for almost anything, it is most commonly done on financial questions. Since the cost benefit analysis relies on the addition of positive factors and the subtraction of negative ones to determine a net result, it is also known as running the numbers.

A cost benefit analysis finds, quantifies, and adds all the positive factors. These are the benefits. Then it identifies, quantifies, and subtracts all the negatives, the costs. The difference between the two indicates whether the planned action is advisable. The real trick to doing a cost benefit analysis well is making sure you include all the costs and all the benefits and properly quantify them.

Should we hire an additional sales person or assign overtime? Is it a good idea to purchase the new stamping machine? Will we be better off putting our free cash flow into securities rather than investing in additional capital equipment? Each of these questions can be answered by doing a proper cost benefit analysis.

**Net Present Value**

**Introduction**

When deciding to invest or not, a firm or an individual has to decide what to do with the money tody.

So, we need to compare money today with money in the future.

What’s the relationship between $1 today and $1 tomorrow?

Is it worth the same $1 today as $1 tomorrow /yesterday?

Time t Time t + 1

$1?

$1 more?

Less?

This is called “time-value-of-money” concept.

Example: A financial analyst at a leading real estate firm is thinking about recommending that Kaufman & Broad invest in a piece of land that costs $85,000. She is certain that next year the land will be worth $91,000, a sure $6,000 gain. Given that the guaranteed interest rate in the bank is 10%, should Kaufman & Broad undertake the investment in land?

**Future value (or compound value):** the value of a sum after investing over one or more periods.

If the money is invested in the bank, next year they would have

$85; 000 (1 + 0:1) = $93; 500

Since

Future value $93, 500 > $91, 000

Then

**Invest everything in the bank.**

The general formula is

F V = C0 (1 + r) -------------- (1)

whereC0 is cash áow today (at time 0),

and r is the appropriate interest rate

**Present value:** the amount of money that should be put in the alternative investment (the bank) in order to obtain the expected amount next year.

In our example, present value is,

P V (1 + 0:1) = $91; 000

Solving for P V

P V = $91, 000/1:1= $82; 727:27

Since

Present value $82; 727:27 < $85; 000

Then

**Do not to buy the land**

The general formula is

PV= C1/(1+r)

Where C1 is cash flow at date 1,

And r is the appropriate interest rate, also called discount rate.

**Net Present Value (NPV)**

The present value of future cash flows minus the present value of the cost of investment.

The formula is NPV=PV-Cost

In our case, we would have

NPV= $91,000/1.1 -$85,000= -$2,273

Because NPV < 0

**Purchase of the land should not be recommended.**

**Present Value for multiple years**

Formula

Present Value (n)= 1/ (1+i)n

e.g. If the discount rate is 12% then,

Present value (1) = 1/(1+0.12)1 = 0.893

Present Value (2) = 1(1+0.12)2 = 0.797

Net Present Value (NPV) = Cumulative PV of all benefits – Cumulative PV of all costs

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | | **Cash Flow** | **Year0** | **year1** | **year2** | **year3** | **year4** | | Dev. Costs | -100,000 |  |  |  |  | | Oper. Costs |  | -4000 | -4500 | -5000 | -5500 | | Present Value | 1 | 0.893 | 0.797 | 0.712 | 0.636 | | Time-adj.-Cost | -100000 | -3572 | -3586.5 | -3560 | -3498 | | Cumulative Costs | -100000 | -103572 | -107159 | -110719 | -114217 | |  |  |  |  |  |  | | Benefits | 0 | 25000 | 30000 | 35000 | 50000 | | Time-adj.-Costs | 0 | 22325 | 23910 | 24920 | 31800 | | Cumulative Benefit | 0 | 22325 | 46235 | 71155 | 102955 | |  |  |  |  |  |  | | **NPV** | **-100000** | **-81247** | **-60924** | **-39564** | **-11262** | |  |  |  |  |  |
|  |  |  |  |  |  |
| Payback Method  Payback method determines when (what date) you get your money back.  There are two ways to calculate the payback  method 1.)  Using undiscounted cash flows, which is the classic way or most often used way.  2.) Using discounted cash  flows.   Undiscounted Cash Flow Method   There are two different timing assumptions.  ***The first assumption*** is that the cash flows happen on the very last day  of the year (December 31) in a lump sum.  So using the cash flows below we can determine the payback or breakeven  point using this timing assumption.  To find the breakeven point, you must find the cumulative cash flow, which is  the current year cash flow, added to the pervious year cash flow.  For example YR0 is <90> and there is no previous  year cash flow so the cumulative cash flow would be <90>.  For YR1 the cash flow is 36 and the previous years cash  flow (YR0) is <90> and this equals <54>.                            YR0                 YR1                 YR2                 YR3                 YR4                 YR5  Cash Flow            <90>               36                    47                    57                    63                    77  $(000)    Cumulative  Cash Flow            <90>               <54>               <7>                 50                    113                  190                                                  36+<90>                  47+<54>                  57+<7>                    63+50                       77+113    So, the first year that a positive cash flow occurs is YR3 or assuming YR0 is the year 2000, then YR3  is the year 2003.    ***Using the second timing assumption***we assume that time has a straight-line cash flow from  January 1 to December 31.  Therefore we need to find out on what date in YR3 does the break even point occur. To  determine this you take the absolute value of the **cumulative cash flow** for the year before the breakeven year (BEYR-1)  divided by the **cash flow** of the **B**reak**E**ven **Y**ea**R**(BEYR).    From the information above we know that the absolute value of the cumulative cash flow for BEYR-1 is 7 and the *cash*  *flow* for BEYR is 57.  This is the fraction 7/57 or .1228.  Then you have to multiply .1228 by 365 (the number of calendar days in a normal year).  This gives you the number 44.8 or rounded to 45 days until you breakeven.  To determine the actual day, it is  suggested to make the following chart:                                         Jan                31                31                                                  Feb                28\*                59\*                                                  Mar                31                90                                            \*Remember to take into                                                  Apr                30                120                                            consideration Leap years.                                                  May                31                151                                                  Jun                30                181                                                  Jul                31                212                                                  Aug                31                243                                                  Sep                30                273                                                  Oct                31                304                                                  Nov                30                334                                                  Dec                31                365    So looking at the chart we can determine that the break even point will fall in February and we can determine that the  day will be 14  (45-31\*)  \*31 is the number of days in the month previous to the breakeven month, in this case January.    So the breakeven point or Pay Back date using the undiscounted cash flow method is 14Feb03.   Discounted Cash Flow Method   This method uses straight-line timing but also takes into consideration the discounted cash flow.  So, you first have to  determine the discount factors and the Present Value (PV) also called the Discounted Cash Flow.  Using the following  cash flow and the discount rate of 10% we will find the break even point using the discounted cash flow method.                              YR0                 YR1                 YR2                 YR3                 YR4                 YR5  Cash Flow            <90>               36                    47                    57                    63                    77  $(000)    Discount  Factor              1                      .909                 .826                 .751                 .683                 .621    PV                   <90>               32.724             38.822             42.807             43.029             47.817    Now you take the cumulative cash flows, which is the current year PV (or discounted cash flows) added to the previous  year cumulative cash flow.    Cumulative  Cash Flow            <90>               <57.276>           <18.454>        24.353              67.382              115.199                                                  32.724+<90>  38.822+<57.276> 42.807+<18.454>  43.029+24.353  47.817+67.382    So, again the BEYR is YR3.  Now we take the absolute value of the **cumulative cash flow** for BEYR-1 divided by the **PV (or discounted cash flow)** of BEYR. Which is 18.454/42.807= .4310 .  Multiply .4310 by 365 to get the day of the year that is the breakeven point, in this case 157.35 rounded to 157th day of the year.  Looking on the chart created earlier, we know that the breakeven month will be June and the day will be the 6th (157 –151).    The Pay Back date or breakeven point using the discounted cash flow method is 6Jun03. |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |