Chapter 3: Project Management

Table of Contents

Project Initiation

There are mainly two reasons to develop a project, either to solve a problem, or to improve an existing system. Before jumping in though, we need to be able to identify the problems. There are mainly four ways to do this:

1. Check output against performance criteria - Too many errors, work completed slowly, work done incorrectly, work done incompletely, work not done at all, etc.
2. Observation of employees (ergonomic analysis) – High absenteeism, high job dissatisfaction, high job turnover, etc.
3. Feedback from external vendors, customers and suppliers – complaints, suggestions, loss of sales, lower sales, etc.
4. Interviewing users

Problem Definition

Once we have found the problem we want to resolve, we need to document it. There are a few parts to the documentation:

- **Problem Statement** – A paragraph or two stating the problem or opportunity
- **Issues** – Independent smaller pieces that contribute to the problem, e.g. unable to take input properly on data entry
- **Objectives** – Address each of the issues listed above, e.g. if an issue is that students cannot access their grades, an objective may be to provide an online portal
- **Requirements and Constraints** – Things that must be accomplished along with the possible solutions and constraints that limit the system. These may include government requirements, security features, budget limitations etc.

Another major point is that each issue has some weight associated with it to signify its importance. For example, if we have two issues, one that incorrect results are shown, and another that students cannot view results online but must physically go to administrative offices to view their results, the first would have the higher weight, since it is a far bigger problem. This allows us to identify which objectives are most critical.

Example

Consider a catering service that starts out small. At that stage, the owner was able to talk to customers to determine the size of events, the types of meals and other information. Spreadsheets and word processing were used to keep track of these details. As the business grew, it became difficult to manage the endless phone calls with inquiries about the meals the service provided, changes by customers and scheduling part time employees.

Problem Statement:
The business is growing and it is becoming difficult to manage phone calls about basic enquiries, changes and scheduling employees.

Issues:	**Weight**
1. Too many calls about basic inquiries	10
2. Changing details such as number of people in an event manually	5
3. Scheduling employee time tables via phone calls	7

Among these, the problem of dealing with too many phone calls should have the most weight. This is because it affects the entire workflow, slowing it down. Solving this would ease things significantly.

Objectives:
1. Create a website to provide basic information
2. System to deal with customer data to make things less manual
3. System to schedule timetables for employees

Determining Feasibility

Feasibility essentially means how possible a project is. We need to check if the objectives we want to fulfil are practically possible. The three main categories of feasibility are operational feasibility, technical feasibility and economic feasibility. In some cases, other categories such as time feasibility or legal feasibility are also added.

Technical feasibility refers to whether or not we have the technology required to fulfil the objectives. If we do not, does the technology exist and can we obtain it? Essentially, technical feasibility tries to answer the question 'Can we build it?'. We also need to address other technical issues such as:

- Are our employees and workers familiar with the technology? If they are not, there is a higher risk the project will fail.
- How large is the project? Larger projects are more likely to fail.
- Would the new system be compatible with the current one? Can the current system be upgraded to make it compatible with the new system?

Economic feasibility is essentially a cost-benefit analysis. We need to ensure that the value of the investment exceeds the time employees need to spend on the project and the cost of the software we need to buy or develop for the project. The value does not need to be monetary profit, since not all projects are for economic gains, but there must be some benefit the project gives to society. Economic feasibility addresses the question 'Should we build it?'.

Operational feasibility addresses the question 'If we build it, will it be used?'. We need to check

- Will we face problems installing the system? If yes, can the problems be overcome?
- Are users knowledgeable enough to use the system? If not, can we train them?

- Can the users be convinced to shift to the new system?
- Do we have the manpower to keep the system running?

Now that we have determined what problems we want to solve and how we plan to solve them, we need to consider if what we want to do is actually doable. If, at any time in the development cycle, we determine that the project is not feasible, we need to stop working. This is where feasibility analysis comes in.

The following topics are not necessary for our current level of work, but in a professional environment, they are essential.

Ascertaining Hardware and Software Needs

Before beginning working on our project, we need to figure out what hardware and software we need. We need to know what we already have and what we need to obtain from elsewhere. Regarding obtaining hardware and software, there are three methods of doing this – buying, leasing and rental.

Buying is simply providing the full amount of money that something is worth. Say a piece of equipment is $\$50,000$. Buying it would involve paying the full amount.

In the leasing process, the full amount is not paid at once. Instead, we pay a certain agreed upon amount every week or month or other agreed upon time period. At the end of this time period, when an equivalent amount to the full price has been paid in total, the product is considered to have been fully purchased. Alternatively, we could pay a smaller amount at every time period and return the product at the end of the contract instead of taking ownership of it. This depends on the contract.

Rental is similar to leasing, except the periodic payments tend to be far lower and there is no possibility of gaining ownership. Another difference is that for rental products, we are responsible for any damages that occur, but under leasing, the current owners whom we are leasing from are responsible for the damages.

	Advantages	Disadvantages
Buying (More than 5 years)	• Cheaper than leasing or renting over a long time period • Ability to change the system • Tax advantages • Full control	• Initial cost is high • Risk of obsolescence • Risk of getting stuck if incorrect choice is made • Full responsibility
Leasing (3 – 5 years)	• No capital tied up • Cheaper than renting (considering damage costs)	• Company may not own the system at the end of the lease • Heavy penalty for terminating lease • More expensive than buying
Renting (6 months)	• No capital tied up • Easy to change systems • Maintenance and Insurance usually included	• Company does not own the system • Costs are high due to assuming the risk

Regarding software, there are three things we could do – create custom software, purchase commercial off-the-shelf (COTS) software, or buy software provided by an application service provider (ASP).

If the software we need is unique to our problem, such that no one else has that type of software, it is always best to just create it ourselves.

If the software problem we have is very general, such as an office automated system problem like needing a word processor, it is always best to just buy existing software. It is better to buy Microsoft Word than to try developing our own software.

Finally, application service providers provide software that we use through cloud networks, such as Gmail. It is software that is not on our own devices, but is accessible from our devices. Again, if such software is what we need, it is better to use the existing ones rather than try and create our own one.

	Advantages	Disadvantages
Creating Custom Software	• Specific to special requirements • Innovation may give firm a competitive advantage • In-house staff available to maintain software • Pride of ownership	• Significantly more expensive that COTS or ASP • Necessary to hire and work with a development team • Ongoing maintenance
Purchasing COTS	• Perfected over time so less likely to be buggy, so more reliable • Greater functionality • Lower initial cost • Help and training available	• Limited customization • Uncertain future of vendor
Using an ASP	• No need to hire, train or retain a large IT staff • No time lost to non-essential IT tasks	• Loss of control over data, systems, IT employees and schedules • Security and privacy concerns

Software Evaluation

When evaluating software to ensure it is good enough for our needs, we need to consider several factors:

- How well it accomplishes the goals, i.e. its performance effectiveness
- How much work needs to be done to achieve the goals, i.e. its performance efficiency
- How easy it is to use
- How flexible it is, for example how many different ways there are to login to the system so different people can use whichever method they prefer
- How well it has been documented
- How much support is available from manufacturers

We will study this section in more detail while discussing different evaluation methods, but for now, knowing the criteria is enough.

Cost and Benefit Analysis

After we have decided on everything we want to do, we need to study the financial aspects of our project. Forecasting is done, which is a prediction of the costs the system will accrue over some time period and the benefits it will provide over that time period.

The forecasting is done based on historic data. This data could be conditional, which happens due to some specific circumstance, or unconditional, which is guaranteed to always occur regardless of circumstance. For example, say we are studying the benefits of something and we come to the conclusion that if the headquarters of our business is in a specific location, the benefits increase by a significant amount. This benefit is conditional.

The costs and benefits can also be split into two categories, tangible and intangible. Things that can have specific monetary amounts attached to them, such as hardware costs, maintenance costs, etc., are tangible, while things that do not have monetary amounts attached to them, such as job satisfaction, or the gain or loss of reputation, are intangible.

Comparing Costs and Benefits

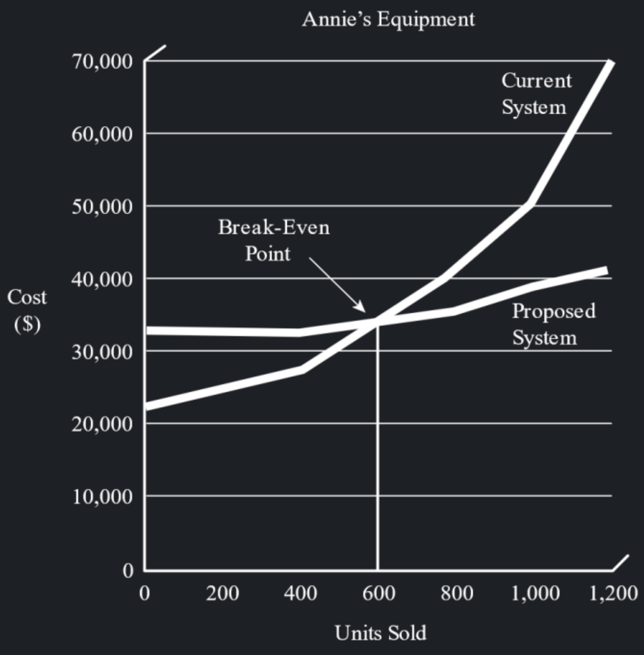
When discussing the economic feasibility of our project, it is not enough to make a vague assumption that the project will be economically feasible. We need to use specific analytical methods to prove the feasibility.

There are four analytical methods we will be covering:

1. Break-Even Analysis
2. Payback Period
3. Cash-Flow Analysis
4. Present Value Analysis

Break-Even Analysis

In break-even analysis, we try to find a particular break point at which we become profitable.



In the diagram above, we are comparing the costs associated with different amounts of sales for the current system and the proposed one. Note that here, costs do not refer to just monetary costs, but also things like time spent developing and maintaining the systems.

We can see that the costs of the current system increase exponentially with the amount of sales. This could be due to things like maintenance costs, which increase with the number of units sold.

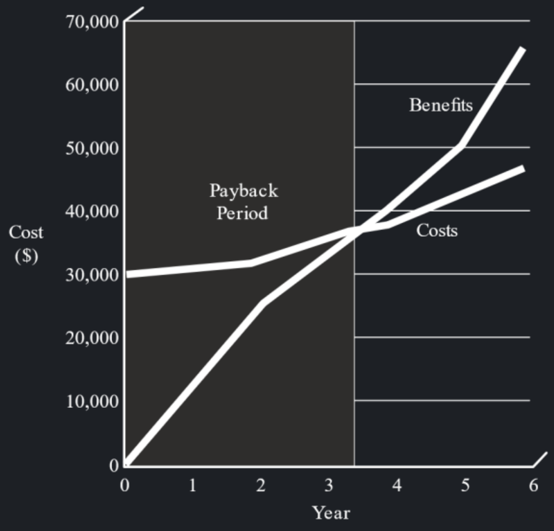
On the other hand, the proposed system is much flatter, meaning the costs increase less as we sell more units. Since it is a new system, it has a significant initial cost.

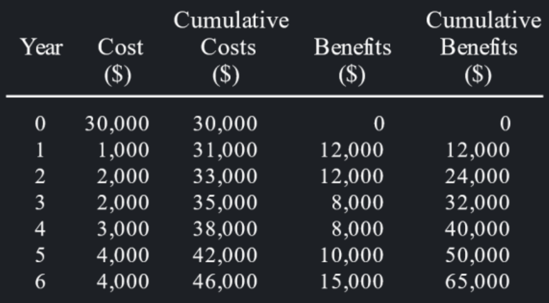
The point where the two curves meet is of interest to us. This is called the break-even point. After this point, we can see that we would be spending more money on the current system than we would on the proposed system.

The disadvantage of break-even analysis is that it assumes that profits remain the same under both the old and new systems. This is not necessarily true.

Payback Period

The payback period tells us the amount of time we need to earn back the money that we spent developing the new system. It is the time taken for the system to become profitable.



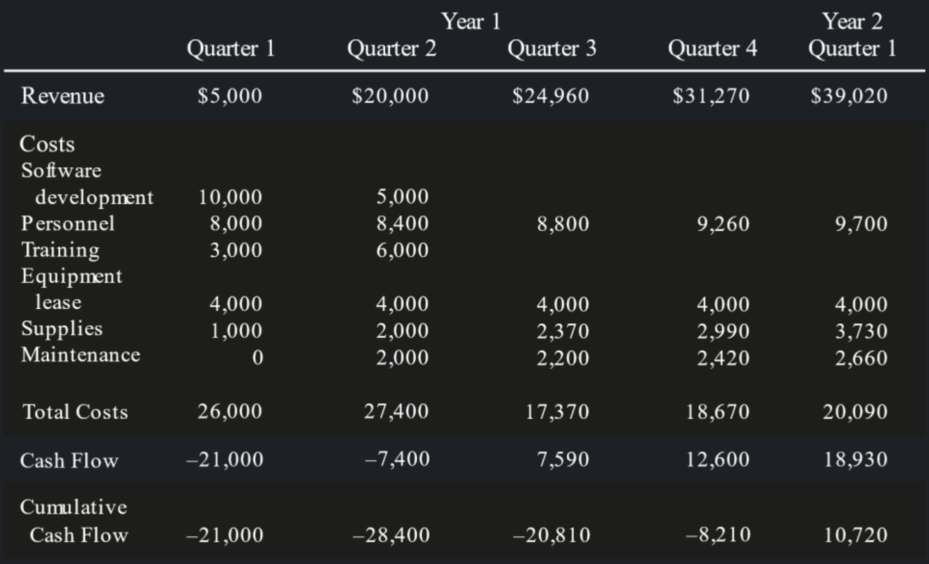


Year	Cost ($)	Cumulative Costs ($)	Benefits ($)	Cumulative Benefits ($)
0	30,000	30,000	0	0
1	1,000	31,000	12,000	12,000
2	2,000	33,000	12,000	24,000
3	2,000	35,000	8,000	32,000
4	3,000	38,000	8,000	40,000
5	4,000	42,000	10,000	50,000
6	4,000	46,000	15,000	65,000

The point at which the two curves meet is when we have reached the break-even point. Thus, the time taken to get there is the payback period.

Notice that the graph is based on the cumulative costs and benefits.

Cash-Flow Analysis



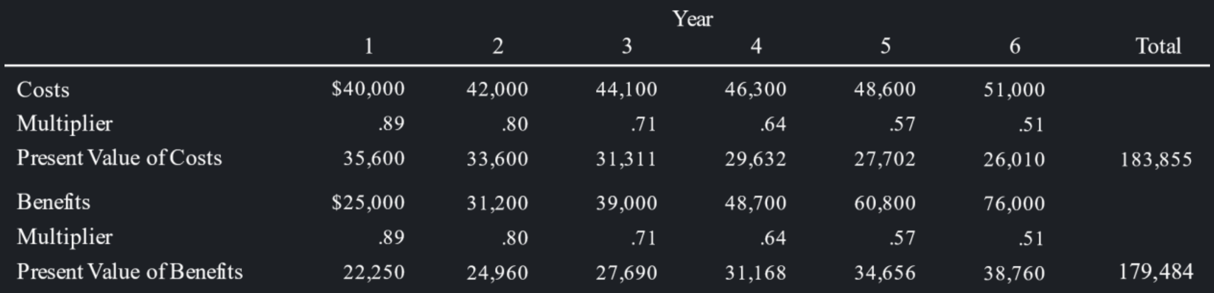
	Year 1				Year 2
	Quarter 1	Quarter 2	Quarter 3	Quarter 4	Quarter 1
Revenue	$5,000	$20,000	$24,960	$31,270	$39,020
Costs					
Software development	10,000	5,000			
Personnel	8,000	8,400	8,800	9,260	9,700
Training	3,000	6,000			
Equipment lease	4,000	4,000	4,000	4,000	4,000
Supplies	1,000	2,000	2,370	2,990	3,730
Maintenance	0	2,000	2,200	2,420	2,660
Total Costs	26,000	27,400	17,370	18,670	20,090
Cash Flow	–21,000	–7,400	7,590	12,600	18,930
Cumulative Cash Flow	–21,000	–28,400	–20,810	–8,210	10,720

In cash-flow analysis, we care less about the exact break-even point and more about the general costs and benefits. We look at the revenue for a particular time period and the different types of costs incurred over that time. Based on this, we find a cash flow. If the cash flow is negative, it means we did not profit in that time period.

The cumulative cash flow keeps track of the total cash flow from the beginning of the project. The point at which the cumulative cash flow becomes positive is when the project has become profitable.

Present Value Analysis

Money decreases in value over time. $\$100$ today is worth more than it will be 10 years from now. In present value analysis, we compare costs and benefits somewhat like we did for cash-flow analysis, but we take the fact that money decreases in value over time into account.



	Year 1	2	3	4	5	6	Total
Costs	$40,000	42,000	44,100	46,300	48,600	51,000	
Multiplier	.89	.80	.71	.64	.57	.51	
Present Value of Costs	35,600	33,600	31,311	29,632	27,702	26,010	183,855
Benefits	$25,000	31,200	39,000	48,700	60,800	76,000	
Multiplier	.89	.80	.71	.64	.57	.51	
Present Value of Benefits	22,250	24,960	27,690	31,168	34,656	38,760	179,484

Rather than just directly comparing costs and benefits for different years, the monetary amount for each year is multiplied by a particular multiplier which gives us the actual worth of the cost/benefit at that period of time.

The multiplier is calculated using the following formula:

$$M = \frac{1}{(1 + i)^n}$$

Here, n is the time period. For example, in the image above we can see that time is represented in years, so if we wanted to calculated the value in the 6^{th} year, n would have the value 6. If instead time is represented in quarters, as it was in the example before this, if we wanted to calculated the value in the 4^{th} quarter, n would have the value 4.

i is called the discount rate. It is the rate of decrement in value of money. If this rate is very high, the multiplier will be low which means the actual value of money will also be lower.

For example, if the present value is $\$300$ and the discount rate is 13%, the value 6 years from now is

$$\frac{1}{\left(1+\frac{13}{100}\right)^{6}} \times \$300 = \$144$$

If instead we want to know the value 6 months from now, the value will still be $\$144$, since n is the value of the time period in whatever scale is given.

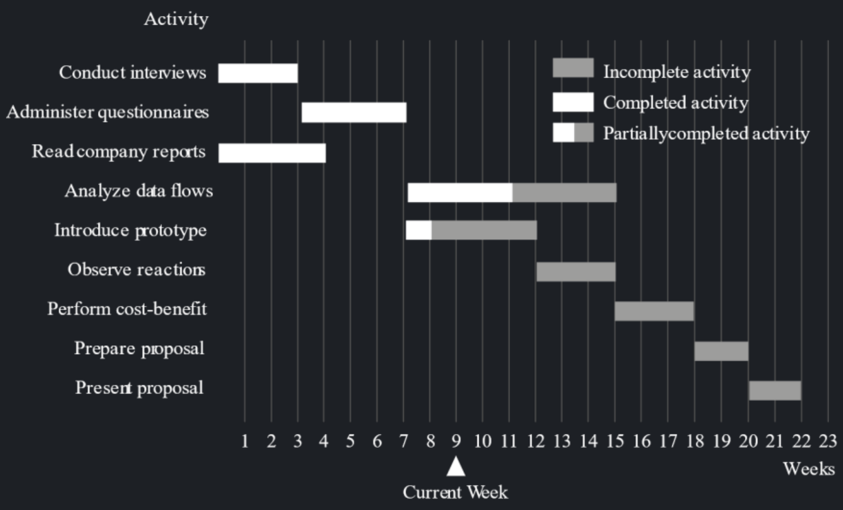
This may seem confusing. In reality, the value of i would adjust accordingly. It would not be so high for 6 months, but it has been kept the same here to reinforce the point that n does not depend on the scale in which we are measuring time.

Project Scheduling

There are two methods to represent a project schedule, Gantt Charts and PERT Diagrams.

Gantt Charts

Between the two, Gantt charts are much simpler and make it easy to communicate the information with end users.

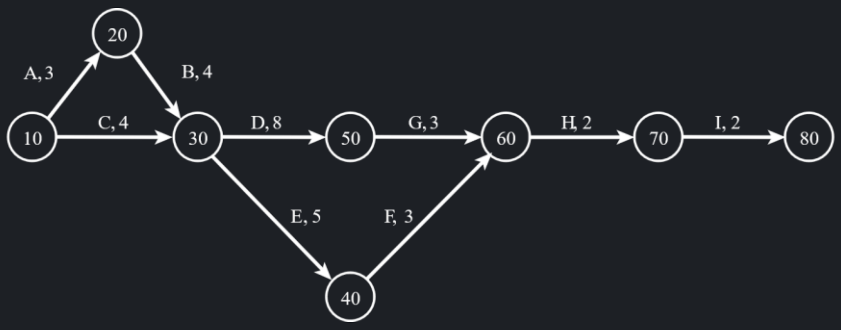


Gantt charts are drawn to scale, meaning longer bars represent longer periods of time. Usually, the horizontal direction represents time while the vertical direction represents activities. The start and end of the bars tell us when a particular activity starts and when it ends. The current week is also shown, so we can tell if we are lagging behind, are on time or are ahead of our work.

In Gantt charts, every activity can be put in one of three categories, incomplete, compete or partially complete.

PERT Diagrams

PERT diagrams are useful when activities can be done in parallel. Here, the sequence of activities is important, as opposed to Gantt charts where the order in which the activities were stated was mostly irrelevant.



Dummies

It is generally advisable to have just one starting and one ending point in a PERT diagram, so that it is clear where the project starts and where it ends.

However, this may not always be possible. For example, we could have two different activities on two different branches at the end of the project.

In these cases, we need to create a dummy activity. Dummy activities are not real activities, but blank ones created so that all the actual ending activities can be brought to a single ending point.

It is not necessary to have dummies at the end only. It is also possible for there to be dummies in the middle of the diagram if we need to bring multiple branches to a single point.

Critical Paths

From the diagram given above, we can see that there are multiple paths from the starting point to the ending point. The path that takes the longest time is called the critical path.

The different branches of a PERT diagram tell us which activities can be started in parallel. Activities that are on different branches are not dependant on one another and can thus run in parallel.

This means that the critical path tells us which series of activities takes the longest time to complete. These activities must be completed one after another and cannot be executed in parallel. Thus, the time taken by the critical path is the development time of the project. All the other branches can be completed in parallel and take a shorter time to complete.

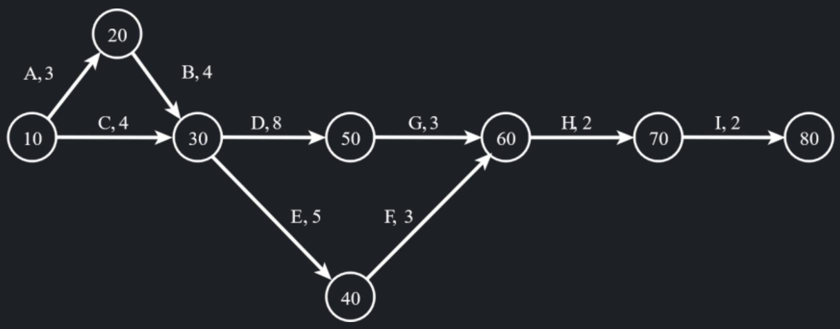
Example

Say we have the following activities:

Activity		Predecessor	Duration
A	Conduct interviews	None	3
B	Administer questionnaires	A	4
C	Read company reports	None	4
D	Analyze data flow	B, C	8
E	Introduce prototype	B, C	5
F	Observe reactions to prototype	E	3
G	Perform cost-benefit analysis	D	3
H	Prepare proposal	F, G	2
I	Present proposal	H	2

The list of predecessors tells us that a particular activity cannot be started without first completing the activities given as predecessors.

From this information, we can create a PERT diagram. Vertices are used to mark the start or the end of each activity, while the edges represent the actual activities and their durations.

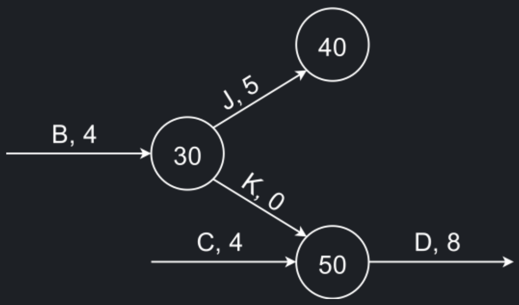


Notice that the vertices have been given different numbers. These numbers simply represent the level. They can be any sensible incrementing set of numbers.

A and C both started from 10. This is because neither of these activities had any predecessors. B has A as its predecessor, so B starts at 20, where A ends.

B and C both end at 30. This is done because D has both B and C as its predecessor, so D needs to start where these two activities end. Thus, it is easier to make these activities end at a single point.

If it were not possible to make B and C end at the same point, we could have used a dummy vertex. For example, if we had an activity that had only B as its predecessor but not C, instead of merging B and C at 30, we could create a dummy activity from 30 to a new vertex where it would meet with C. Thus, the new activity could start from 30 and D could start at the new vertex where the dummy activity and C meet.



In the above diagram, K is the dummy activity.

Alternatively, the new activity, J, could have started from 30 in the original diagram as well. Just because it is starting at 30 does not mean it is mandatory for both B and C to be its predecessors. This is not preferred though.

The rest of the diagram should be simple enough to understand.

The critical path for this diagram is $10 \rightarrow 20 \rightarrow 30 \rightarrow 50 \rightarrow 60 \rightarrow 70 \rightarrow 80$.

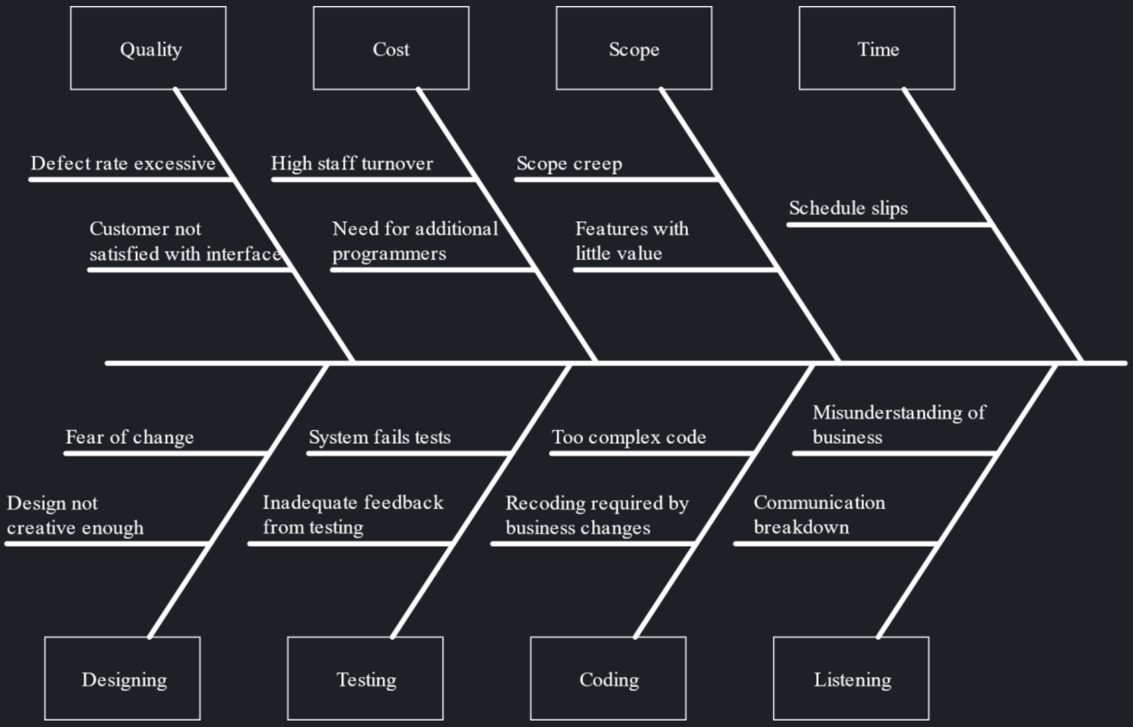
Advantages

PERT diagrams make it easy to:

- Identify the order of precedence of activities
- Identify the critical path, and thus the critical activities
- Identify the slack time, the difference between the earliest and the latest start or finish time of an activity. This tells us the maximum time we can delay an activity for.

Project Failures

Projects may fail due to a number of reasons. These reasons can be divided into different categories. All of this can be represented using a fish-bone diagram.


Quality
Cost
Scope
Time
Defect rate excessive
High staff turnover
Scope creep
Schedule slips
Customer not satisfied with interface
Need for additional programmers
Features with little value
Fear of change
System fails tests
Too complex code
Misunderstanding of business
Design not creative enough
Inadequate feedback from testing
Recoding required by business changes
Communication breakdown
Designing
Testing
Coding
Listening

In the above diagram, the rectangular boxes represent the categories and the text on the fishbones tell us what the specific errors in each of those categories are that could cause a failure. For example, regarding time, too many schedule slips could lead to project failure.

Avoiding Project Failures

Project failure can be avoided with:

- Proper training
- Experience
- Learning from other project failures and the reasons behind them