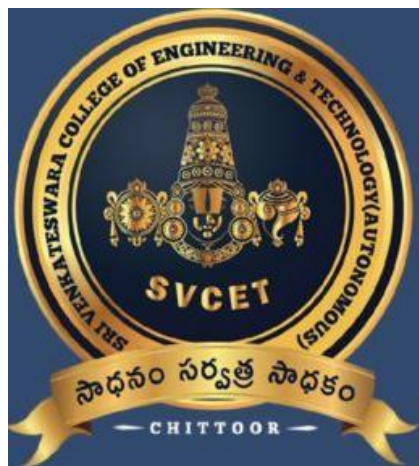


**SRI VENKATESWARA COLLEGE OF ENGINEERING & TECHNOLOGY**

**(AUTONOMOUS)**

RVS Nagar, Chittoor Andhra Pradesh - 517127



**DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING (DATA SCIENCE)**

**B.Tech II YEAR – II SEM – CSD-B**

**2022-2023**

**R20 REGULATION**

**LABORATORY MANUAL**

**SOFTWARE ENGINEERING LAB**

**(20AIT05)**

## **Vision and Mission of the Institute**

### **Vision**

- To carve the youth as dynamic, competent, valued and knowledgeable professionals who shall lead the Nation to a better future and to mould the institution into a Center of Academic Excellence and Advanced Research.

### **Mission**

- To Provide quality education, student-centered teaching-learning processes and state-of-art infrastructure for professional aspirants hailing from both rural and urban areas.
- To Impart technical education that encourages independent thinking, develops strong domain of knowledge, contemporary skills and positive attitudes towards holistic growth of young minds.

## **Vision and Mission of the CSE (DS)**

### **Vision**

- Evolve as centre of Proficiency in Data Analytics and develop ingenious professional as data analytics and researchers.

### **Mission**

- M1: To empower students with innovative and cognitive skills to expertise in the field of Data science.
- M2: To Inculcate the seed of knowledge by providing industry conducive environment and excel in data driven world.
- M3: To provide an excellent infrastructure, facilities and ambience to nurture the young professionals.
- M4: Committed to provide professionals with socio-disciplinary attitude and acquire professional ethics.

## **Program Educational Objectives (PEOs)**

- PEO1: To be able to solve wide range of computing related problems to cater to the needs of industry and society.
- PEO2: Enable students to build intelligent machines and applications with a cutting-edge combination of machine learning, analytics and visualization.
- PEO3: Produce graduates having professional competence through life-long learning such as advanced degrees, professional skills and other professional activities related globally to engineering & society.

## **Program Specific Outcomes (PSOs)**

- PSO1: Should have an ability to apply technical knowledge and usage of modern hardware and software tools related AI and ML for solving real world problems.
- PSO2: Should have the capability to develop many successful applications based on machine learning methods, AI methods in different fields, including neural networks, signal processing, and data mining.

# **SRI VENKATESWARA COLLEGE OF ENGINEERING AND TECHNOLOGY**

## **(AUTONOMOUS)**

II B.Tech II Semester (Common to CSE, IT, CSE (DS) & CSE(AI &ML)

L T P C

- - 3 1.5

### **20AIT05 SOFTWARE ENGINEERING LAB**

#### **Course Outcomes**

At the end of the course the student will be able to:

1. Acquaint with historical and modern software methodologies
2. Understand the phases of software projects and practice the activities of each phase
3. Practice clean coding
4. Take part in project management
5. Adopt skills such as distributed

#### **List of Experiments**

1. Draw the Work Breakdown Structure for the system to be automated.
2. Schedule all the activities and sub-activities Using the PERT/CPM charts.
3. Define use cases and represent them in use-case document for all the stakeholders of the system to be automated.
4. Identify and analyze all the possible risks and its risk mitigation plan for the system to be automated.
5. Diagnose any risk using Ishikawa Diagram (Can be called as Fish Bone Diagram or Cause& Effect Diagram).
6. Define Complete Project plan for the system to be automated using Microsoft Project Tool.
7. Define the Features, Vision, Business objectives, Business rules and stakeholders in the vision document.
8. Define the functional and non-functional requirements of the system to be automated by using Use cases and document in SRS document.
9. Develop a tool which can be used for quantification of all the non-functional requirements.
10. Write C/Java/Python program for classifying the various types of coupling.
11. Write a C/Java/Python program for classifying the various types of cohesion.
12. Write a C/Java/Python program for object-oriented metrics for design proposed by Chidamber and Kremer. (Popularly called CK metrics).
13. Draw a complete class diagram and object diagrams using Rational tools.

## References

1. Software Engineering? A Practitioner's Approach, Roger S. Pressman, 1996, MGH.
2. Software Engineering by Ian Sommerville, Pearson Edu, 5th edition, 1999.
3. An Integrated Approach to software engineering by Pankaj Jalote , 1991 Narosa.

### Online Learning Resources/Virtual Labs

<http://vlabs.iitkgp.ac.in/se/>

## CO, PO, PSO Mapping

[illegible]

## Experiment 1: Draw the Work Breakdown Structure for the system to be automated.

### **Introduction**

Dividing complex projects to simpler and manageable tasks is the process identified as Work Breakdown Structure (WBS).

Usually, the project managers use this method for simplifying the project execution. In WBS, much larger tasks are broken down to manageable chunks of work. These chunks can be easily supervised and estimated.

WBS is not restricted to a specific field when it comes to application. This methodology can be used for any type of project management.

A Work Breakdown Structure WBS proposes a graphical nature that helps project managers predict results based on various scenarios.

It is often described as a result-oriented tree that covers all project procedures in an organized way. However, WBS can also be displayed as a tabular list of tasks and elements in Work Breakdown structure Gantt Charts.

Following are a few reasons for creating a WBS in a project:

- Accurate and readable project organization.
- Accurate assignment of responsibilities to the project team.
- Indicates the project milestones and control points.
- Helps to estimate the cost, time and risk.
- Illustrate the project scope, so the stakeholders can have a better understanding of the same.

### **Construction of a WBS**

Identifying the main deliverables of a project is the starting point for deriving a work breakdown structure.

This important step is usually done by the project managers and the subject matter experts (SMEs) involved in the project. Once this step is completed, the subject matter experts start breaking down the high-level tasks into smaller chunks of work.

In the process of breaking down the tasks, one can break them down into different levels of detail. One can detail a high-level task into ten sub-tasks while another can detail the same high-level task into 20 sub-tasks.

Therefore, there is no hard and fast rule on how you should breakdown a task in WBS. Rather, the level of breakdown is a matter of the project type and the management style followed for the project.

In general, there are a few "rules" used for determining the smallest task chunk. In "two weeks" rule, nothing is broken down smaller than two weeks' worth of work.

This means, the smallest task of the WBS is at least two-week long. 8/80 is another rule used when creating a WBS. This rule implies that no task should be smaller than 8 hours of work and should not be larger than 80 hours of work.

One can use many forms to display their WBS. Some use tree structure to illustrate the WBS, while others use lists and tables. Outlining is one of the easiest ways of representing a WBS.

Following example is an outlined WBS:

Project Name	Task 1	Subtask 1.1	Work Package 1.1.1
			Work Package 1.1.2
		Subtask 1.2	Workpackage 1.2.1
			Workpackage 1.2.2
	Task 2		
		Subtask 2.1	
			Workpackage 2.1.1
			Workpackage 2.1.2

There are many design goals for WBS. Some important goals are as follows:

- Giving visibility to important work efforts.
- Giving visibility to risky work efforts.
- Illustrate the correlation between the activities and deliverables.
- Show clear ownership by task leaders.

### WBS Diagram

In a WBS diagram, the project scope is graphically expressed. Usually, the diagram starts with a graphic object or a box at the top, which represents the entire project. Then, there are sub- components under the box.

These boxes represent the deliverables of the project. Under each deliverable, there are sub- elements listed. These sub-elements are the activities that should be performed in order to achieve the deliverables.

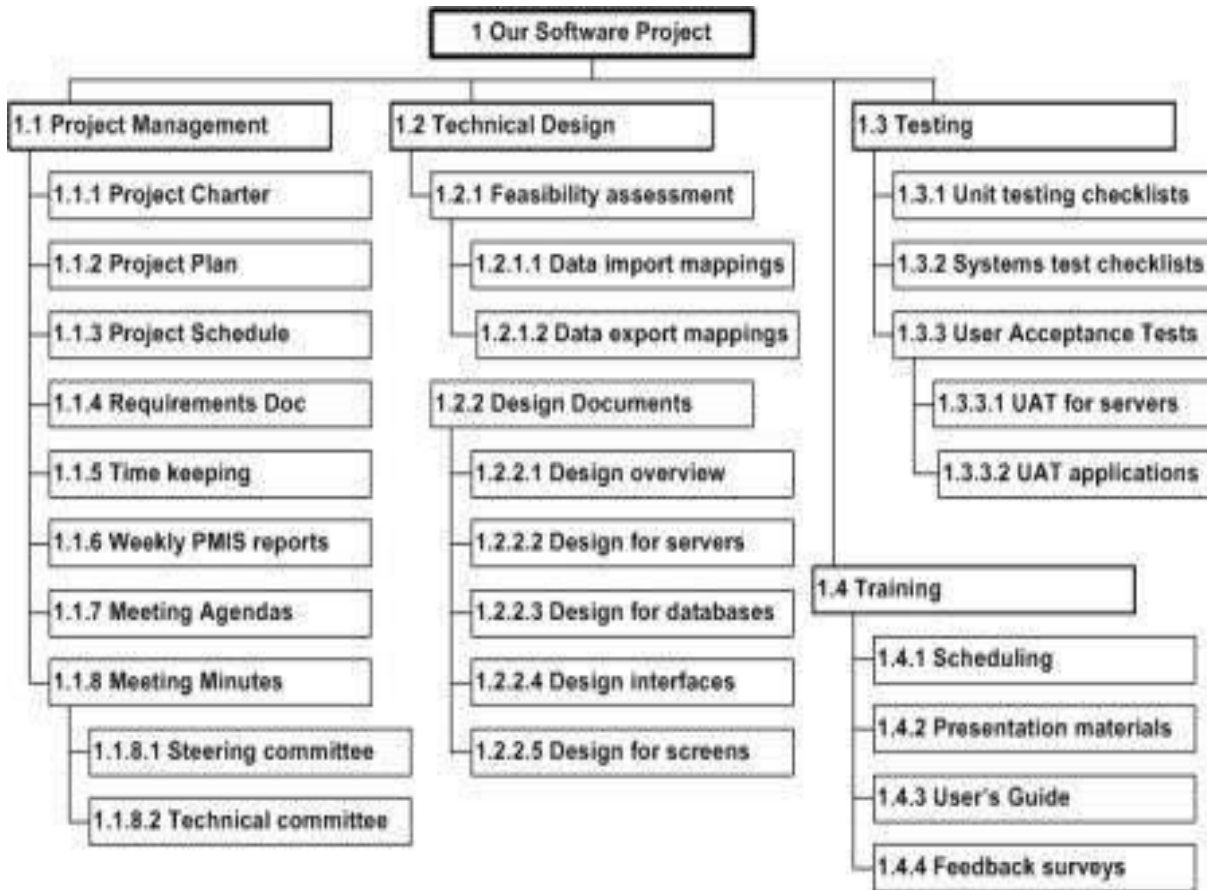
Although most of the WBS diagrams are designed based on the deliveries, some WBS are created based on the project phases. Usually, information technology projects are perfectly fit into WBS model.

Therefore, almost all information technology projects make use of WBS.

In addition to the general use of WBS, there is specific objective for deriving a WBS as well. WBS is the input for Gantt charts, a tool that is used for project management purpose.

Gantt chart is used for tracking the progression of the tasks derived by WBS.

Following is a sample WBS diagram:



### Work Breakdown Structure (WBS) using MS Excel:

A work breakdown structure (WBS) in Excel is used to visually represent the ordering of different tasks and project activities schedule of resources during project planning. It allows us to divide or split the project into more manageable parts by classifying the project tasks into a hierarchy of events that are further split into a series of tasks that are not dependent on any other task for completion. Also, it is used to allocate matching equipment, costs, materials, labour, and time duration for the completion of each task.

#### Aim:

To decompose the project for accessing or receiving the data in Excel.

#### Procedure:

**Step 1:** Create the below columns across the top

- Task ID
- Task Description
- Predecessor
- Owner
- Role
- % Completion
- Start Date
- End Date

- Deliver To

**Step 2:** After creating these columns, the cells are to be formatted.

1. First, we format the cell Task ID.
  - a) To format it, we first highlight the column designated for Task ID numbers by clicking on the letter at the top of the column.
  - b) Next, right-click on the selected column, select “Format Cells” -> “Number,” and set decimal places to 1.
  - c) Next, we format column: Predecessor, which will track task dependencies.
2. We should set it up the same way as “Task ID.” “Start Date” and “End Date” are to be set up to accept dates entered.

**Step 3:** Enter the data.

	A	B	C	D	E	F	G	H	I
	Task ID #	Predecessor	Task Description	Duration	Start Date	End Date	Owner	Deliver To	% Complete
2	1.0		Deliverable		04-Mar	05-Mar	Malinda	Marvio	10.00%
3	1.1	1.0	Task 1			10-Mar			0.00%
4	1.2		Task 2			23-Mar			15.00%
5	2.0					17-Mar			0.00%
6	2.1	1.2							50.00%
7	2.2								100.00%
8	2.3	2.0							5.00%
9									4.00%
10									50.00%
11									20.00%
12									10.00%
13									

Next, we can use conditional formatting for various tasks. For instance, we wish to highlight tasks.

associated with a particular deliverable, assigned to a particular member, or tasks due within a given time range.

We can also see the “data bar” feature graphically representing a column like “% Complete.” We can access this feature in the “Conditional Formatting” dropdown.

### Result:

Thus, the WBS operation for project management work, including, planning, cost and effort estimation, resource allocation, and scheduling was created and verified successfully.

### Reference:

<https://www.wallstreetmojo.com/work-breakdown-structure-in-excel/>



## **Conclusions**

The efficiency of a work breakdown structure can determine the success of a project.

The WBS provides the foundation for all project management work, including, planning, cost and effort estimation, resource allocation, and scheduling.

Therefore, one should take creating WBS as a critical step in the process of project management.

## Experiment 2:      Schedule all the activities and sub-activities Using the PERT/CPM charts.

### **Introduction to CPM / PERT Techniques**

CPM/PERT or Network Analysis as the technique is sometimes called, developed along two parallel streams, one industrial and the other military.

A PERT/CPM is a project diagram, much like a flow chart, used to identify various stages in a project. This technique combines a Program Evaluation and Review Technique diagram, and a Critical Path Method diagram. A PERT diagram is used to illustrate worst-case, best-case, and most likely occurrences from a project start to finish. It can also illustrate the probability of a project being completed on a specific date. A CPM diagram, as its name implies, shows the path, or steps, through the project. You can use Microsoft Excel to create a PERT/CPM diagram using the text boxes and drawing tools much like those produced using project management software.

### **Basic Steps in PERT / CPM**

Project scheduling by PERT / CPM consists of four main steps.

#### **1.      Planning**

- The planning phase is started by splitting the total project into small projects. These smaller projects in turn are divided into activities and are analyzed by the department or section.
- The relationship of each activity with respect to other activities are defined and established and the corresponding responsibilities and the authority are also stated.
- Thus, the possibility of overlooking any task necessary for the completion of the project is reduced substantially.

#### **2.      Scheduling**

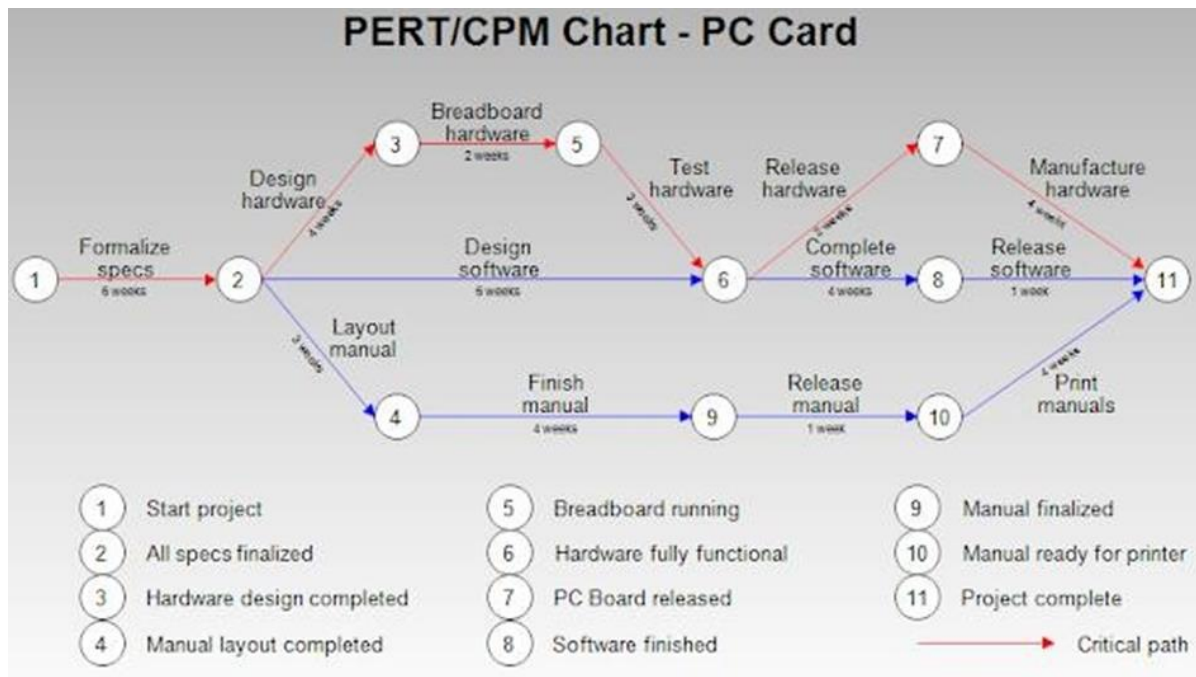
- The ultimate objective of the scheduling phase is to prepare a time chart showing the start and finish times for each activity as well as its relationship to other activities of the project.
- Moreover, the schedule must pinpoint the critical path activities which require special attention if the project is to be completed in time.
- For non-critical activities, the schedule must show the amount of slack or float times which can be used advantageously when such activities are delayed or when limited resources are to be utilized effectively.

#### **3.      Allocation of resources**

- Allocation of resources is performed to achieve the desired objective. A resource is a physical variable such as labour, finance, equipment and space which will impose a limitation on time for the project.
- When resources are limited and conflicting, demands are made for the same type of resources a systematic method for allocation of resources become essential.
- Resource allocation usually incurs a compromise, and the choice of this compromise depends on the judgment of managers.

#### 4. Controlling

- The final phase in project management is controlling. Critical path methods facilitate the application of the principle of management by expectation to identify areas that are critical to the completion of the project.
- By having progress reports from time to time and updating the network continuously, a better financial as well as technical control over the project is exercised.
- Arrow diagrams and time charts are used for making periodic progress reports. If required, a new course of action is determined for the remaining portion of the project.



#### Advantages of PERT/CPM

- A PERT/CPM chart explicitly defines and makes visible dependencies (precedence relationships) between the elements.
- PERT/CPM facilitates identification of the critical path and makes this visible.
- PERT/CPM facilitates identification of early start, late start, and slack for each activity.
- PERT/CPM provides for potentially reduced project duration due to better understanding of dependencies leading to improved overlapping of activities and tasks where feasible.

#### Disadvantages of PERT/CPM

- There can be potentially hundreds or thousands of activities and individual dependency relationships.
- The network charts tend to be large and unwieldy requiring several pages to print and requiring special size paper.
- The lack of a timeframe on most PERT/CPM charts makes it harder to show status although colours can help (e.g., specific colour for completed nodes).
- When the PERT/CPM charts become unwieldy, they are no longer used to manage the project.

## **PERT/CPM charts using Excel**

### **Aim:**

To Schedule all the activities and sub-activities using the PERT/CPM charts in Excel.

### **Procedure:**

#### **Step 1: Open Excel**

Launch MS Excel on your desktop. For this guide we will use MS Excel 2013.

#### **Step 2: Select Blank Worksheet**

Once MS Excel has launched, select a blank worksheet.

#### **Step 3: Create PERT Chart**

To create a PERT chart in MS Excel, go to the Insert Tab on the Excel ribbon and click on the text box under the text section. Click on the area of the worksheet where you want to place a text box. Resize the text work by selecting intent and dragging by the corners. Keep putting different text boxes in the order of your process.

#### **Step 4: Add Details**

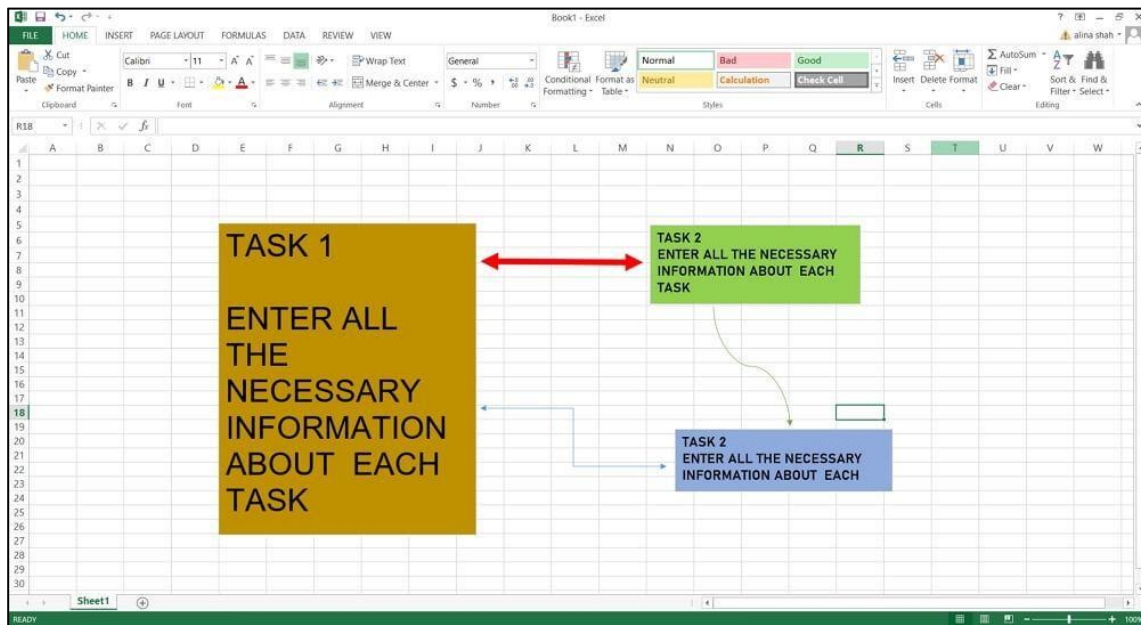
Once you have arranged different textboxes on the worksheet, go to the Format Tab under the drawing tools on the Excel ribbon and use various tools to change the features of the text box such as style, color, background, etc.

Now, add the text within each text box. In each textbox, enter the details of a single task and make sure to include all the necessary information that needs to be displayed.

Once all the tasks have been labeled and displayed, go to Shapes in the “Illustrations” sections under Insert tab and select the lines or rows to connect your textbox. Once you have placed the characters you will get a basic looking PERT chart.

#### **Step 5: Save**

When your PERT chart is complete click on the File tab and save your document.



**Figure 2.1: Drawing PERT/CPM Chart**

**Result:**

Thus the procedure for scheduling all the activities and sub-activities for personal trainer using the PERT/CPM charts in Excel was created and verified successfully.

**Reference:**

<https://www.youtube.com/watch?v=8NHJTRUiX6A>

### Experiment 3: Define use cases and represent them in use-case document for all the stakeholders of the system to be automated.

#### **‘Use Case’ Documents:**

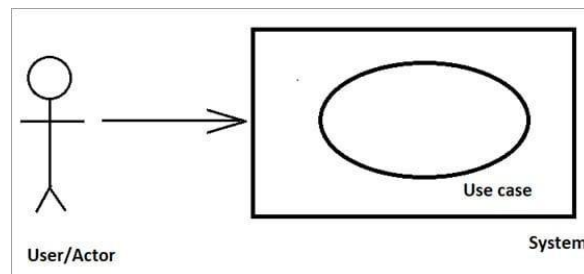
- This documentation gives a complete overview of the distinct ways in which the user interacts with a system to achieve the goal. Better documentation can help to identify the requirement for a software system in a much easier way.
- This documentation can be used by Software developers, software testers as well as Stakeholders.

#### **Uses of the Documents:**

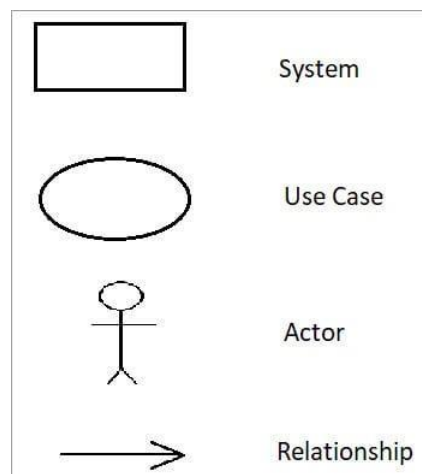
1. Developers use the documents for implementing the code and designing it.
2. Testers use them for creating the test cases.
3. Business stakeholders use the document for understanding the software requirements.

#### **Use Case Diagram:**

Use Case Diagram is a pictorial representation of a user(s) Actions in a system. It does provide a great tool in this context, if the diagram is containing a lot of actors, then it is very easy to understand.



**Figure 3.1:** Simple Use Case Diagram



**Figure 3.2:** Use Case Diagram Symbol

Figure 3.1 represents a diagram where Rectangle represents a ‘System’, oval represent a ‘Use Case’, Arrow represents a ‘Relationship’ and the Man represents a ‘User/Actor’. It shows a system/application, then it shows the organization/people who interact with it and shows the basic flow of ‘What the system does?’.

### Use Case Testing:

A Functional Black Box Testing ensures that the path used by the user is working as intended or not. It makes sure that the user can accomplish the task successfully.

### Aim:

To define use cases and represent them in use-case document for all the stakeholders of the School Login system to be automated.

### Procedure:

**Step 1:** Explain the cases for Login to School Management System.

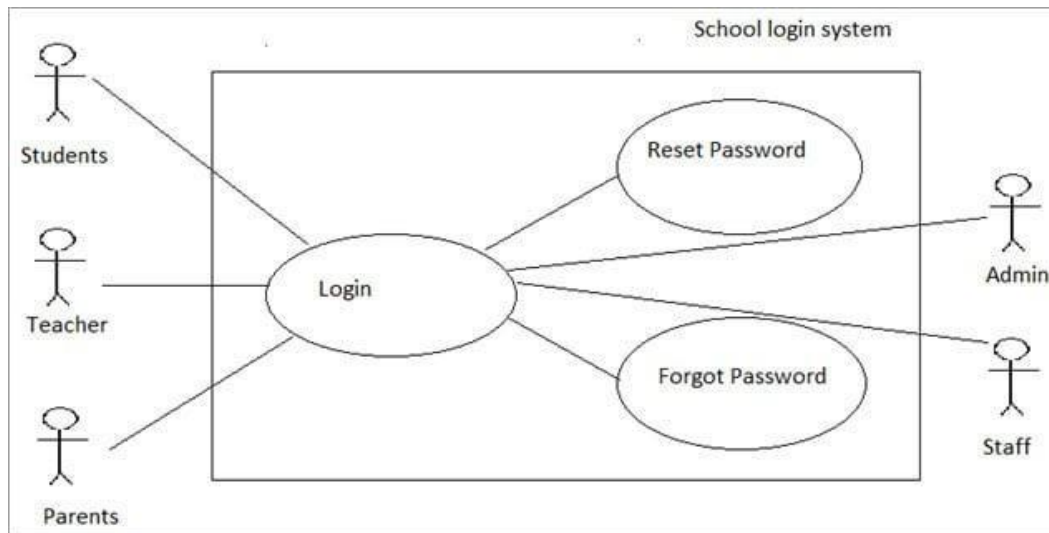
**Table 3.1** Use Cases for Login in School Management System

Use Case Name		Login
Use case Description	A user login to System to access the functionality of the system.	
Actors	Parents, Students, Teacher, Admin	
Pre-Condition	System must be connected to the network.	
Post -Condition	After a successful login a notification mail is sent to the User mail id	
Main Scenarios	Serial No	Steps
Actors/Users	1	Enter username Enter Password
	2	Validate Username and Password
	3	Allow access to System
Extensions	1a	Invalid Username System shows an error message
	2b	Invalid Password System shows an error message
	3c	Invalid Password for 4 times Application closed

**Step 2:** Draw Use case Diagram for Login to School Management System using Visio Tool.

- Create a new use case diagram.
- On the File tab, point to New.
- in the Search box, type UML use case.
- From the search results, select UML Use Case.
- In the dialog box, select the blank template or one of the three starter diagrams. (A description of each one is shown on the right when you select it.) Then select either Metric Units or US Units.
- Select Create.

- g. The diagram opens. You should see the Shapes window next to the diagram. A UML Use Case stencil is open in the Shapes window.
- h. Add shapes and connectors to the diagram.
- Drag Use Case shapes from the UML Use Case stencil and place them inside the subsystem boundary, and then drag Actor shapes to the outside of the subsystem boundary.
  - Use connector shapes to indicate relationships between shapes in the diagram. There are five connectors available.



**Figure 3.3:** Use Case Diagram for School Login System

**Step 3:** Write test cases for each normal flow and alternate flow.

Consider the 'Show Student Marks' case, in a School Management System.



**Use case Name:** Show Student Marks

**Actors:** Students, Teachers, Parents

**Pre-Condition:**

- 1) The system must be connected to the network.
- 2) Actors must have a 'Student ID'.

**Use Case for 'Show Student Marks':**

Main Scenario	Serial Number	Steps
A: Actor/ S: System	1	Enter Student Name
	2	System Validates Student Name
	3	Enter Student ID
	4	System Validates Student ID
	5	System shows Student Marks
Extensions	3a	Invalid Student ID S: Shows an error message
	3b	Invalid Student ID entered 4 times. S: Application Closes

**Corresponding Test Case for 'Show Student Marks' case:**

Test Cases	Steps	Expected Result
A	View Student Mark List 1 -Normal Flow	
1	Enter Student Name	User can enter Student name
2	Enter Student ID	User can enter Student ID
3	Click on View Mark	System displays Student Marks
B	View Student Mark List 2-Invalid ID	
1	Repeat steps 1 and 2 of View Student Mark List 1	
2	Enter Student ID	System displays Error message

## Result:

Thus, the Use case document for School Login Management System was created and verified successfully.

## Reference:

1. <https://www.softwaretestinghelp.com/use-case-testing/>
2. [https://support.microsoft.com/en-us/office/create-a-uml-use-case-diagram-92cc948d-fc74-466c-9457-e82d62ee1298#OfficeVersion=Newer\\_desktop\\_versions](https://support.microsoft.com/en-us/office/create-a-uml-use-case-diagram-92cc948d-fc74-466c-9457-e82d62ee1298#OfficeVersion=Newer_desktop_versions)

## Experiment 4: Identify and analyze all the possible risks and its risk mitigation plan for the system to be automated.

What is Risk?

Put simply, a risk is a chance of something negative happening. In the context of Six Sigma, risks are things that can delay, halt, or harm your project.

There are a few different types of risk:

- **External:** Occurrences outside your company that you have little to no control over. For example, natural disasters, social changes, government policy modifications.
- **Consequential:** Things that happen because of something the company does. For example, a new product receives a negative reaction and public opinion declines.
- **Personal:** Issues arising from staffing issues, management problems and the like. For example, a subject matter expert resigns, leaving you without a key source of knowledge.
- **Technical:** Problems with technology and automation. For example, a company might find that their current automated production hardware can't be configured to use a new algorithm that would decrease production time.
- **Supply:** Every project requires some resources. In most cases, you'll need these resources – whether physical or intangible – to be supplied. For example, you might need material for producing a new part. Conversely, you might be reliant on getting enough cloud resources to test a new SaaS product.

### **Note:**

You'll find that every industry and company tends to define its risk categories differently, according to need. You'll rarely use the exact categories mentioned above. What matters is that you understand the broad varieties and sources of risk – then you can adapt as needed.

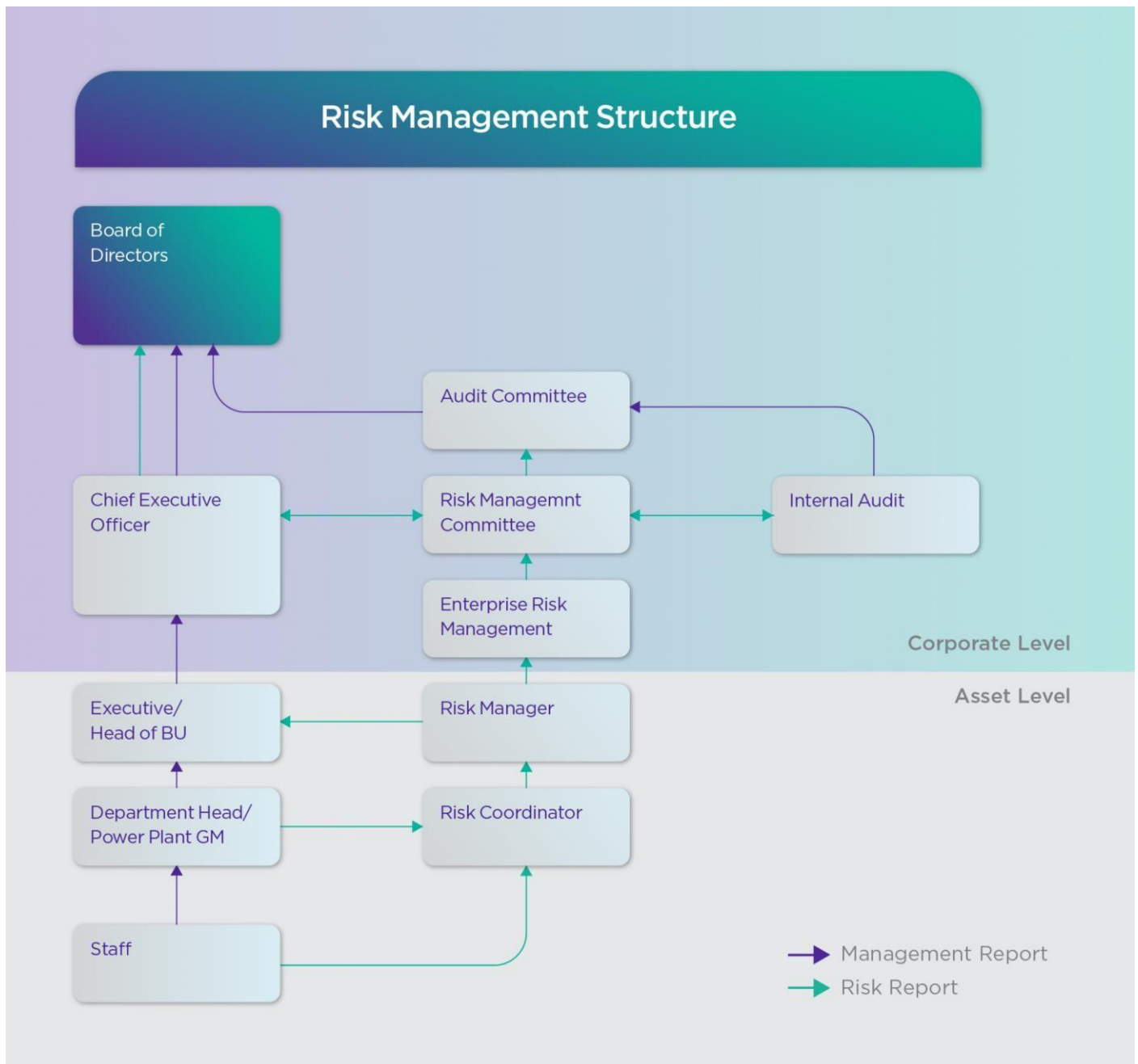
### **What We Can Do About Risk**

Every project will contain risks. Some of them you'll be aware of; some of them will hit out of the blue. No matter how hard you try, you can't eliminate all risk from a project. So why are we even talking about this? Because through risk analysis and mitigation, you can decrease the chances of big negative effects on your project results. Or in other words: you can give your projects their best possible chance of success. Risk analysis and mitigation requires identifying your risks, understanding how they might affect your project, and then figuring out what you can do to minimize their effects.



Image: Risk analysis and mitigation process

## Risk management structure



## How to Identify Risks?

So, you know what risk is, and you know the main types of risk you could encounter. How do you actually identify the risks your particular project could face? There are a few methods you could use:

- **Brainstorming:** Sit down with team members and subject matter experts to throw around some ideas for risks in the major risk categories.
- **Learned lessons:** What did you learn from previous projects? Are any of the negatives from these applicable to the new project?

- **Keystones:** What are the important supports on which the project relies? These could be people, regulatory requirements, a piece of technology, an environmental situation. For example, say you're designing a new piece of hardware for the building industry. Your keystones might be a supply of cheap steel from overseas and a continued boom in the commercial building industry.
- **Assumptions:** This is a tricky one, because it requires you to question points that you unconsciously consider factual. Sit down with your team to identify assumptions that you have about the project. For example, that all members of your team will always be available to work in the office.
- **Fault tree analysis:** This tool allows you to chart a process and its potential failure points. See Fault Tree Analysis for more information.

## How to Analyse Risks!!!

Once you've identified your risks, the next step is to analyze them. Analysis allows you to understand the risks that your project faces. You'll then be able to prioritize them by severity and likelihood.

Properly analysing your risks at this stage ensures that:

- The most dangerous risks can be dealt with appropriately.
- You don't waste resources countering risks that aren't important.
- Your team can channel their efforts into getting the most benefit.
- The project is safeguarded wherever practical.

Good risk analysis improves stakeholder confidence, minimizes unexpected expenditures, and allows for appropriate resource allocation.

## Example: Risk Analysis and Mitigation

A construction company has a large CBD building project beginning soon. It needs to identify, analyze, and mitigate risks to the project.

### Identify

The project team brainstorms and looks at similar past projects to gather a list of risks for the project.

### Analyse

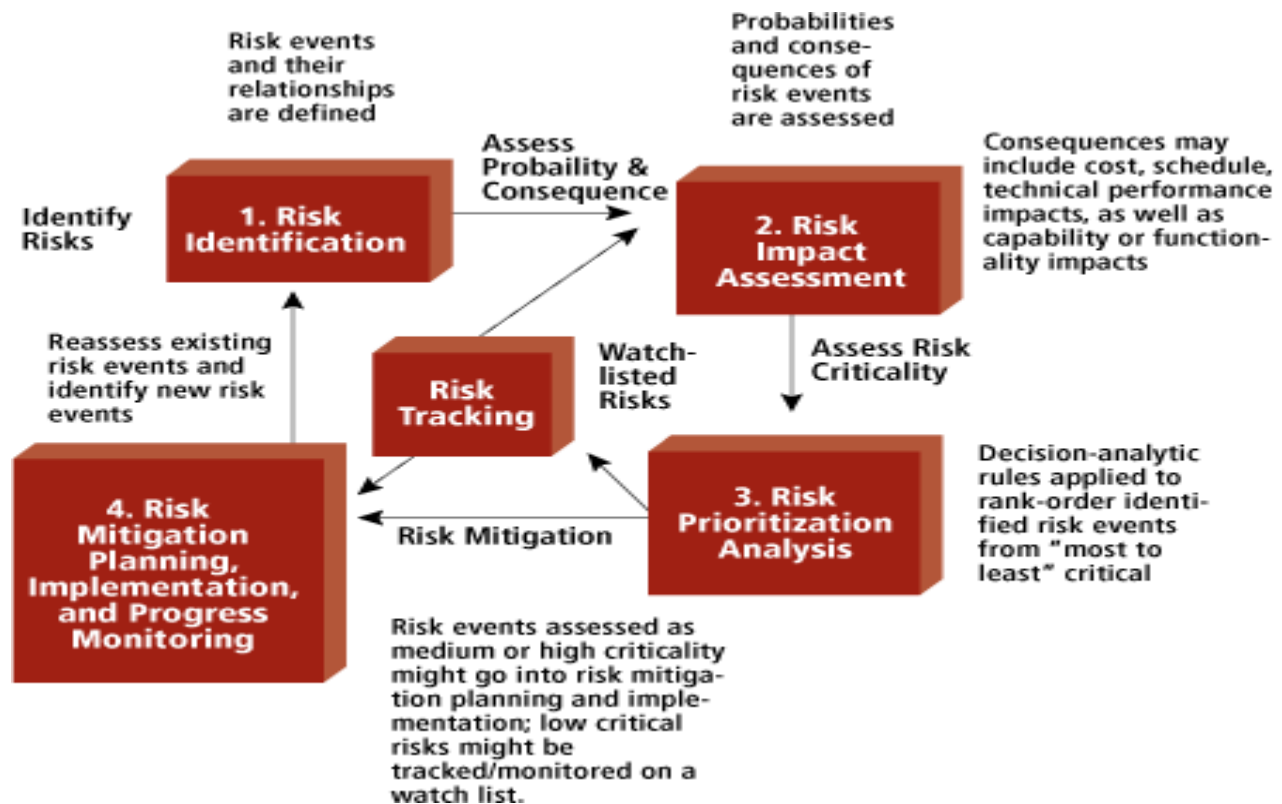
The team uses a feasibility study and risk assessment matrix to analyze and prioritize the risks that the project faces. It determines that the highest priority risk is that issues with shipping will cut supply lines and halt construction on the site.

### Mitigate

While changing to a local supplier would eliminate that risk, it would also increase costs dramatically. The project team decides that the cost outweighed the benefit of eliminating the risk. However, it decides to make local suppliers an alternative option if the shipping issues actually occur. Getting a new supplier signed off requires quite a bit of paperwork and due diligence. If the problem hits, the team doesn't want the project held up while

this sign-off process happens. Instead, it starts the process at the beginning of the project, so that if it can't get material from international suppliers, it can quickly and easily switch to the local suppliers instead.

Note that this solution does not in any way change the potential event. The chances of international shipping disruption remain exactly the same. What the team has changed, though, is the negative impact that the event would have. The project might have higher costs, due to higher prices from local suppliers, but it shouldn't experience a time delay.



## Experiment 5: Diagnose any risk using Ishikawa Diagram (Can be called as Fish Bone Diagram or Cause& Effect Diagram).

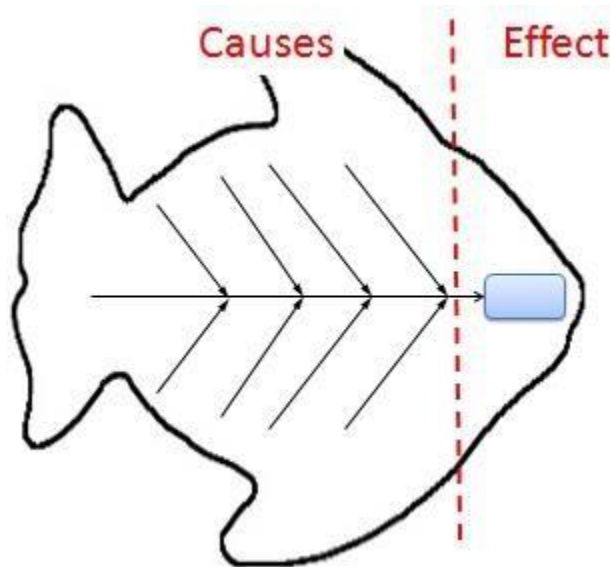
A **fishbone diagram** is a tool that can help you perform a cause and effect analysis for a problem you are trying to solve. This type of analysis enables you to discover the root cause of a problem.

This tool is also called a **cause-and-effect diagram** or an **Ishikawa diagram**. These names can be used interchangeably.

### Ishikawa Diagram Structure

The left side of the diagram is where the causes are listed. The **causes** are broken out into major cause categories. The causes you identify will be placed in the appropriate cause categories as you build the diagram.

The right side of the diagram lists the **effect**. The effect is written as the **problem statement** for which you are trying to identify the causes.



Ishikawa Fish Bone Diagram

The diagram looks like the skeleton of a fish, which is where the **fishbone** name comes from.

### How to Create a Cause-and-Effect Diagram

A cause-and-effect diagram can be created in six steps...

1. Draw Problem Statement
  2. Draw Major Cause Categories
  3. Brainstorm Causes
  4. Categorize Causes
  5. Determine Deeper Causes
  6. Identify Root Causes
1. DRAW PROBLEM STATEMENT

The first step of any problem-solving activity is to define the problem. You want to make sure that you define the problem correctly and that everyone agrees on the problem statement.

Once your problem statement is ready, write it in the box on the right-hand side of the diagram.



Fishbone Diagram - Problem Statement

## 2. DRAW MAJOR CAUSE CATEGORIES

After the problem statement has been placed on the diagram, draw the major cause categories on the left-hand side and connect them to the "backbone" of the **fishbone chart**.

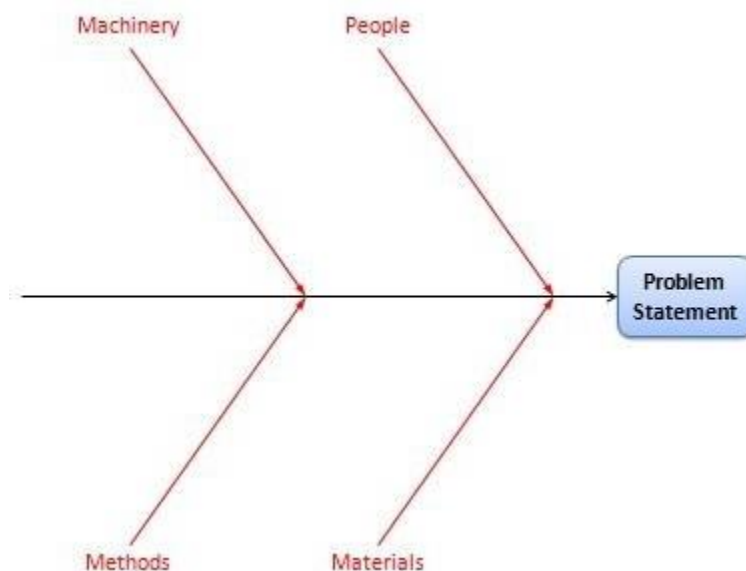
In a manufacturing environment, the traditional categories are:

- Machines/Equipment
- Methods
- Materials
- People

In a service organization, the traditional categories are:

- Policies
- Procedures
- Plant
- People

You can start with those categories or use a different set that is more applicable for your problem. There isn't a perfect set or specified number of categories. Use what makes sense for your problem.



Cause and Effect Diagram - Major Cause Categories



### 3. BRAINSTORM CAUSES

Brainstorming the causes of the problem is where most of the effort in creating your Ishikawa diagram takes place.

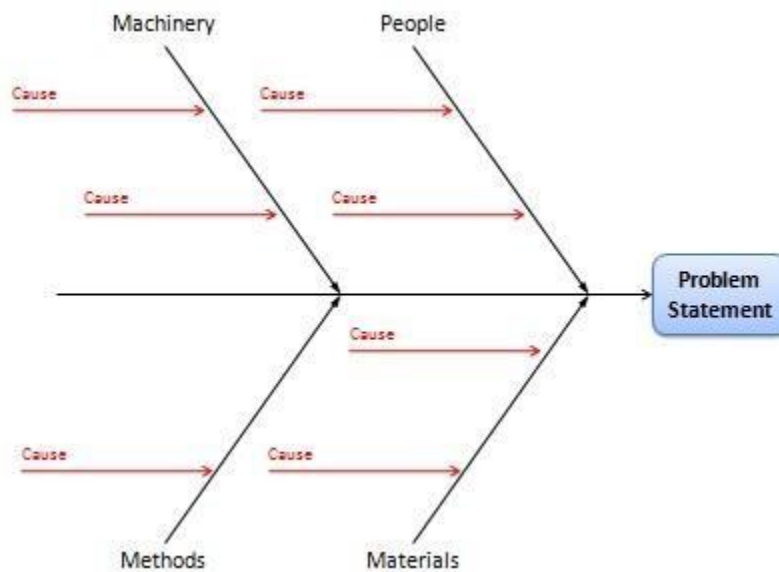
Some people prefer to generate a list of causes before the previous steps in order to allow ideas to flow without being constrained by the major cause categories.

However, sometimes the major cause categories can be used as catalysts to generate ideas. This is especially helpful when the flow of ideas starts to slow down.

### 4. CATEGORIZE CAUSES

Once your list of causes has been generated, you can start to place them in the appropriate category on the diagram.

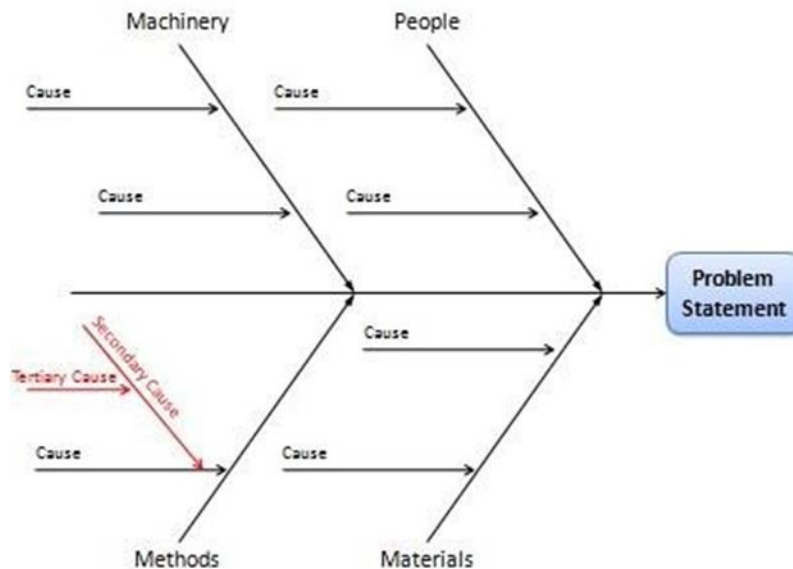
Ideally, each cause should only be placed in one category. However, some of the "People" causes may belong in multiple categories. For example, Lack of Training may be a legitimate cause for incorrect usage of Machinery as well as ignorance about a specific Method.



### 5. DETERMINE DEEPER CAUSES

Each cause on the chart is then analyzed further to determine if there is a more fundamental cause for that aspect. This can be done by asking the question, "Why does it happen?"

This step can also be done for the deeper causes that are identified. Generally, you can stop going deeper when a cause is controlled one level of management removed from your group. Use your judgment to decide when to stop.



## 6. IDENTIFY ROOT CAUSES

The final step for creating a fishbone diagram is to identify the root causes of the problem. This can be done in several ways...

- Look for causes that appear repeatedly
- Select using group consensus methods
- Select based on frequency of occurrence

## Conclusions

Fishbone diagrams are an excellent way to explore and visually depict the causes of a problem. They enable the root causes of a problem to be determined. This will help you be more effective by focusing your actions on the true causes of a problem and not on its symptoms.

## Experiment 6: Define Complete Project plan for the system to be automated using Microsoft Project Tool.

### **Aim:**

**To define a project task for the system to be automated using Ms-Project.**

### **Introduction about Microsoft Project:**

Microsoft Project is a project management software program, developed and sold by Microsoft, which is designed to assist a project manager in developing a plan, assigning resources to tasks, tracking progress, managing the budget, and analyzing workloads. Microsoft Project can be used in a variety of industries including construction, manufacturing, pharmaceuticals, government, retail, financial services and health care.

Microsoft Project is a software application sold by Microsoft that provides project management tools to manage projects. The program, which has many different versions, allows users to:

- Understand and control project schedules and finances.
- Communicate and present project information.
- Organize work and people to make sure that projects are completed on schedule.

### **Features:**

Project creates budgets based on assignment work and resource rates. As resources are assigned to tasks and assignment work estimated, the program calculates the cost, equal to the work times the rate, which rolls up to the task level and then to any summary tasks and finally to the project level. Resource definitions (people, equipment and materials) can be shared between projects using a shared resource pool. Each resource can have its own calendar, which defines what days and shifts a resource is available. Resource rates are used to calculate resource assignment costs which are rolled up and summarized at the resource level. Each resource can be assigned to multiple tasks in multiple plans and each task can be assigned multiple resources, and the application schedules task work based on the resource availability as defined in the resource calendars. All resources can be defined in label without limit. Therefore, it cannot determine how many finished products can be produced with a given amount of raw materials. This makes Microsoft Project unsuitable for solving problems of available materials constrained production. Additional software is necessary to manage a complex facility that produces physical goods.

The application creates critical path schedules, and critical chain and event chain methodology third-party add-ons also are available. Schedules can be resource leveled, and chains are visualized in a Gantt chart. Additionally, Microsoft Project can recognize different classes of

users. These different classes of users can have differing access levels to projects, views, and other data. Custom objects such as calendars, views, tables, filters, and fields are stored in an enterprise global which is shared by all users.

## Introduction about Gantt Chart

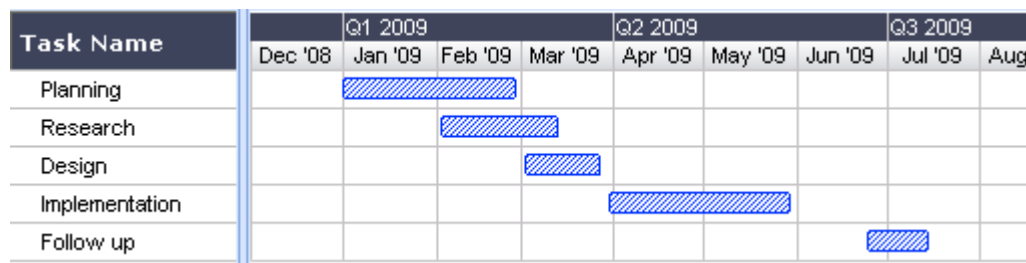
A Gantt chart, commonly used in project management, is one of the most popular and useful ways of showing activities (tasks or events) displayed against time. On the left of the chart is a list of the activities and along the top is a suitable time scale. Each activity is represented by a bar; the position and length of the bar reflects the start date, duration and end date of the activity.

A Gantt chart is a type of bar chart, developed by Henry Gantt in the 1910s, that illustrates a project schedule. Gantt charts illustrate the start and finish dates of the terminal elements and summary elements of a project. Terminal elements and summary elements comprise the work breakdown structure of the project. Modern Gantt charts also show the dependency (i.e., precedence network) relationships between activities.

This allows you to see at a glance:

- What the various activities are
- When each activity begins and ends
- How long each activity is scheduled to last
- Where activities overlap with other activities, and by how much
- The start and end date of the whole project

To summarize, a Gantt chart shows you what has to be done (the activities) and when (the schedule).



Sample of Gantt chart

## Introduction about Network Diagram

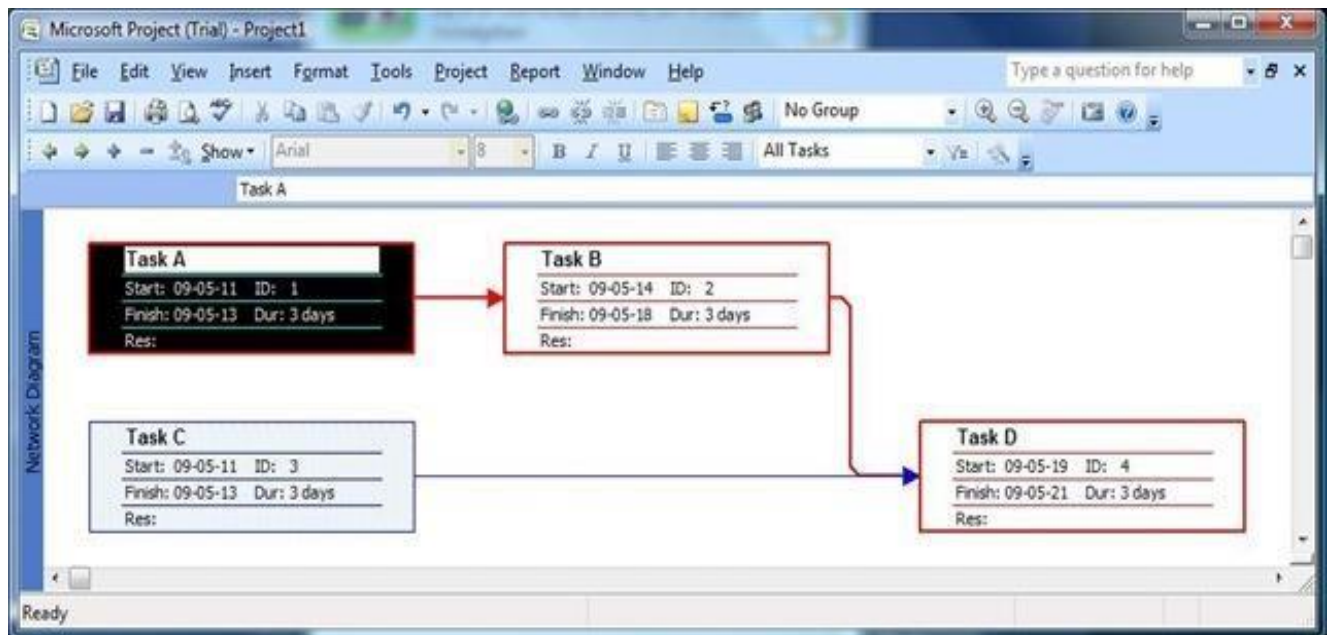
The Network Diagram view was called the PERT Chart in earlier versions of Project. This view shows the dependencies between tasks in a graphical manner. Gantt chart is primarily meant to view the schedule time line, whereas Network diagram to view the all type of dependencies in the project.

Each task shown in the box called node and a line connecting two boxes represents the dependency between those tasks.

You can also create new tasks in the Network Diagram. It's always easy to manage the tasks from Gantt chart view, but if you are concerned on view the dependencies instantly while you are creating the tasks, Network Diagram will be the right one.

Network Diagram also can display a record of progress:

- If the task is completed, the task node shows cross diagonal lines
- If the task is in progress but not completed, a single diagonal line drawn through node
- No diagonal line appears in tasks that are not yet started



Sample of Network Diagram

## Create Gantt chart and Network Diagram

### Gantt Chart:

#### 1. Adding tasks and milestones to a Project File:

- On the View menu, click Gantt Chart.
- In the Task Name field, type a task name, and then press TAB. (Microsoft Project enters an estimated duration of one day for the task followed by a question mark)
- In the Duration field, type the amount of time each task will take in months, weeks, days, hours, or minutes, not counting nonworking time. (By default the time period will be days, but that can be changed to hours, months, etc.)
- Press ENTER. It should look like the figure below:

	Task Name	Duration	Jan 19, '03	S	M	T	W	T	F	S
	Activity 1	2 days								

- To add a milestone the only difference is that the duration of the activity must be zero (below is an example):

	Task Name	Duration	Jan 19, '03	S	M	T	W	T	F	S
	Activity 1	0 days								

**Note:** By double clicking on a Task or milestone, you can modify its information with a form that prompts

## 2. Grouping Tasks in Logical Order:

Outlining helps organize your tasks into more manageable chunks. You can indent related tasks under a more general task, creating a hierarchy. The general tasks are called summary tasks; the indented tasks below the summary task are subtasks. A summary task's start and finish dates are determined by the start and finish dates of its earliest and latest subtasks.

- Click once on the first activity of the group of activities you want to group. For the example Activities 4 and 5

	Task Name	Duration	Jan 19, '03	S	M	T	W	T	F	S
	Activity 1	1 day								
	Activity 2	1 day								
	Activity 3	1 day								
	Activity 4	1 day								
	Activity 5	1 day								

- Then click on the option “New Task” in the “Insert” Menu to insert a new task that will represent the name of the group (“Group 1” for this example)

4	Group 1	1 day?								
5	Activity 4	1 day								
6	Activity 5	1 day								

- Then select the tasks below (4 and 5) and then click in the option “Indent” in the “Project

4	Group 1	1 day								
5	Activity 4	1 day								
6	Activity 5	1 day								

### 3. Creating Relationships Between Tasks:

A network of tasks in a project must be connecting activities from the start to the end, to establish these relationships we need to use the field “Predecessors” of each task, where we can designate which activity will be preceding the one we are updating, in the example below we will indicate MS project that “Activity 5” can start once “Activity 4” is completed (**Finish to Start** relationship).

	i	Task Name	Duration	Predecessors	9, '03
1		Activity 1	1 day		T W T F S
2		Activity 2	1 day		
3		Activity 3	1 day		
4		Group 1	2 days		
5		Activity 4	1 day		
6		Activity 5	1 day	5	

Notice that by establishing the relationship now the Group 1 takes 2 days to be completed, because before, the activities were set to be performed in parallel, and now they are in series (**Finish to Start** relationship)

**Note:** MS project will calculate dates based on the durations of the tasks, their relationships and the start date set for the project, however it is possible to change the starting date of a task (if necessary) By double clicking on a Task or milestone, and using the fields related to the dates (Start or Finish)

### 4. Assigning Resources to Tasks:

You can use the Resource Sheet in Microsoft Project to create a list of the people, equipment, and material resources that make up your team and carry out the project tasks. Your resource list will consist of work resources or material resources. Work resources are people or equipment; material resources are consumable materials or supplies, such as concrete, wood, or nails.

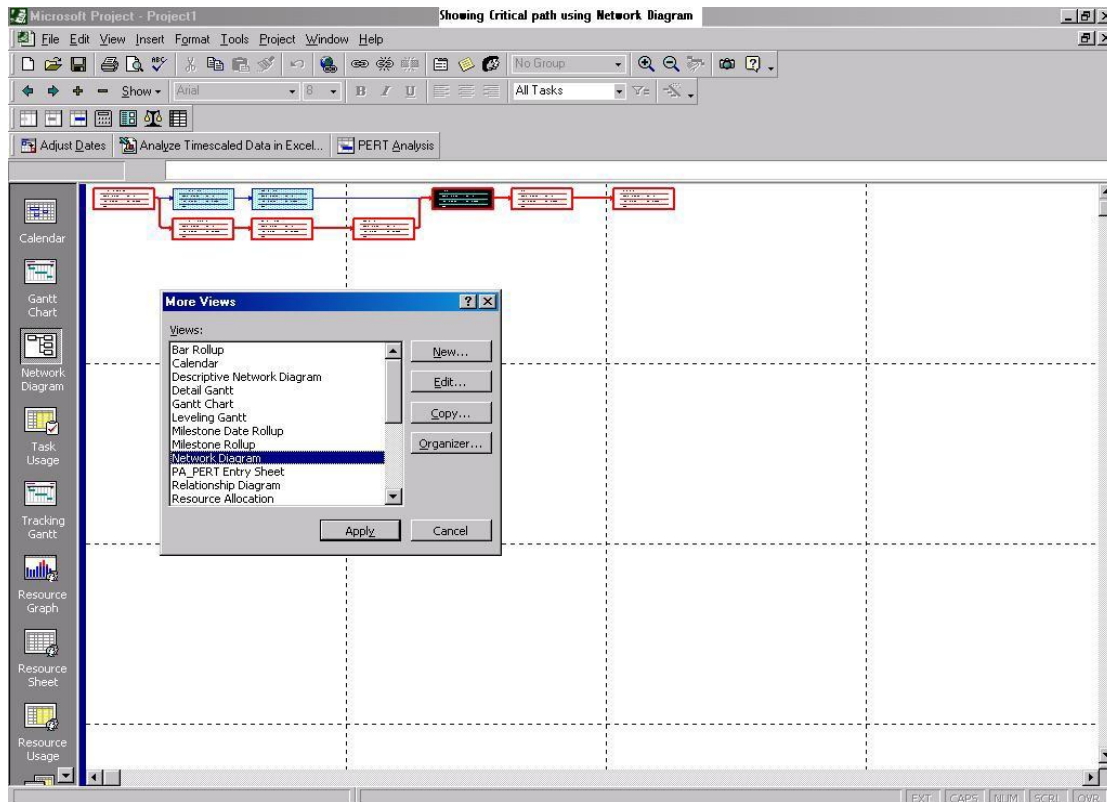
- On the View menu, click Resource Sheet.
- On the View menu, point to Table, and then click Entry.
- In the Resource Name field, type a resource name.
- You can go through the fields in the sheet, but for the simplicity of the example just focus on the name and initials of the Resource
- Below is an example of some Human resources added to the Resource Sheet (We could add also other type of resources such as Equipments, Consumables, etc.)

	i	Resource Name	Type	Material Label	Initials	Group	Max. Units	\$
1		Project Manager	Work		P		100%	
2		Team Leader	Work		T		100%	
3		Developer	Work		D		100%	
4		Tester	Work		T		100%	

- Once the resources are created, you can go back to the View menu, and click Gantt Chart to see again the tasks, and then when you double click a task you can add a resource to this task by using the tab “Resources”

### Network Diagram:

- Click Gantt chart view and go to the sub options in Gantt Chart
- Click Network diagram



Now let's try a small example, step by step to practice each of the options we have seen so far about how to create a project using MS Project. We are going to use a small set of tasks (Table Below) related to the initial phases of a System Testing Plan (Definition and Design)

Activity	Predecessor	Responsibility	Effort	System Testing Phase
1.RSD Analysis	Requirements Specification Document completed. (Not part of System Testing Plan	<ul style="list-style-type: none"> <li>Test Manager</li> <li>Project Manager</li> <li>Test Leader</li> </ul>	3 days	Definition Phase
2. Develop Test Plan	Activity 1	<ul style="list-style-type: none"> <li>Test Manager</li> <li>Test Leader</li> <li>Tester E</li> </ul>	5 days	Design Phase

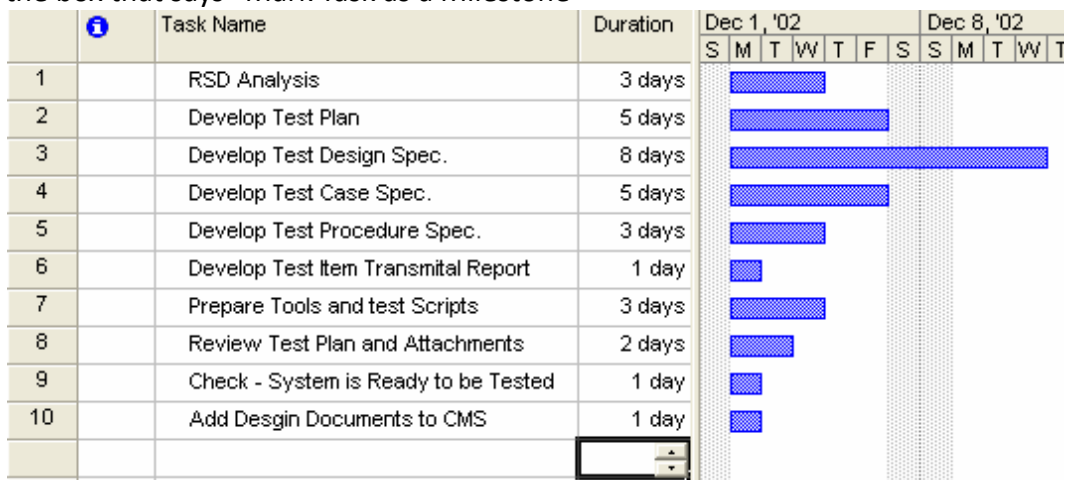


3. Develop Test Design Specification	Activity 2	•Tester A and B	8 days	Design Phase
4. Develop Test Case Specification	Activity 3	•Tester A and B	5 days	Design Phase
5. Develop Test Procedure Specification	Activity 4	•Tester A and B	3 days	Design Phase
6. Develop Test Item	Activity 5	•Tester A and B	1 days	Design Phase
7. Prepare Tools and Test Scripts	Activity 6	•Tester A and B	3 days	Design Phase
8. Review Test Plan and Attachments	Activity 7	•Test Manager •Test Leader •Tester E	2 days	Design Phase
Check that System is Ready to be tested	Activity 8	•Tester E, A and B	1 days	Design Phase
10. Add Design Documents to CMS (Milestone)	Activity 9	•Test Leader •Tester E, A and B	1 days	Design Phase

## Step by Step Create Gantt Chart of System Testing Plan

### Adding Task:

- Write the name of each task in the spreadsheet using the column “**Task Name**”
- Write the duration in days of each task in the spreadsheet using the column “**Duration**”
- Group the tasks by the Phase according to the table of tasks shown before, and add a group that encloses the phases named “**System Testing Plan MCY-ADTT-ST-2002-01**” this will represent the plan as a whole
- Write the predecessors of each task in the spreadsheet using the column “**Predecessors**” (If you can’t see the column, try to expand the vertical bar that divides the spreadsheet to the Gantt Chart)
- To convert a Task in a Milestone, just double click the Task and go to the tab “Advanced” then check the box that says “Mark Task as a Milestone”



## Grouping:

- Insert a new task at the beginning that will group everything. Highlight the tasks that are going to be added as subtasks and use 'Indent' option.

	Task Name	Duration	Dec 1, '02							Dec 8, '02						
			S	M	T	W	T	F	S	S	M	T	W	T	F	S
1	System Testing Plan MCY-ADTT-ST-200	1 day?														
2	RSD Analysis	3 days														
3	Develop Test Plan	5 days														
4	Develop Test Design Spec.	8 days														
5	Develop Test Case Spec.	5 days														
6	Develop Test Procedure Spec.	3 days														
7	Develop Test Item Transmittal Report	1 day														
8	Prepare Tools and test Scripts	3 days														
9	Review Test Plan and Attachments	2 days														
10	Check - System is Ready to be Tested	1 day														
11	Add Design Documents to CMS	1 day														

1.1 Insert new task

	Task Name	Duration	Dec 1, '02							Dec 8, '02						
			S	M	T	W	T	F	S	S	M	T	W	T	F	S
1	System Testing Plan MCY-ADTT-ST-200	1 day?														
2	RSD Analysis	3 days														
3	Develop Test Plan	5 days														
4	Develop Test Design Spec.	8 days														
5	Develop Test Case Spec.	5 days														
6	Develop Test Procedure Spec.	3 days														
7	Develop Test Item Transmittal Report	1 day														
8	Prepare Tools and test Scripts	3 days														
9	Review Test Plan and Attachments	2 days														
10	Check - System is Ready to be Tested	1 day														
11	Add Design Documents to CMS	1 day														

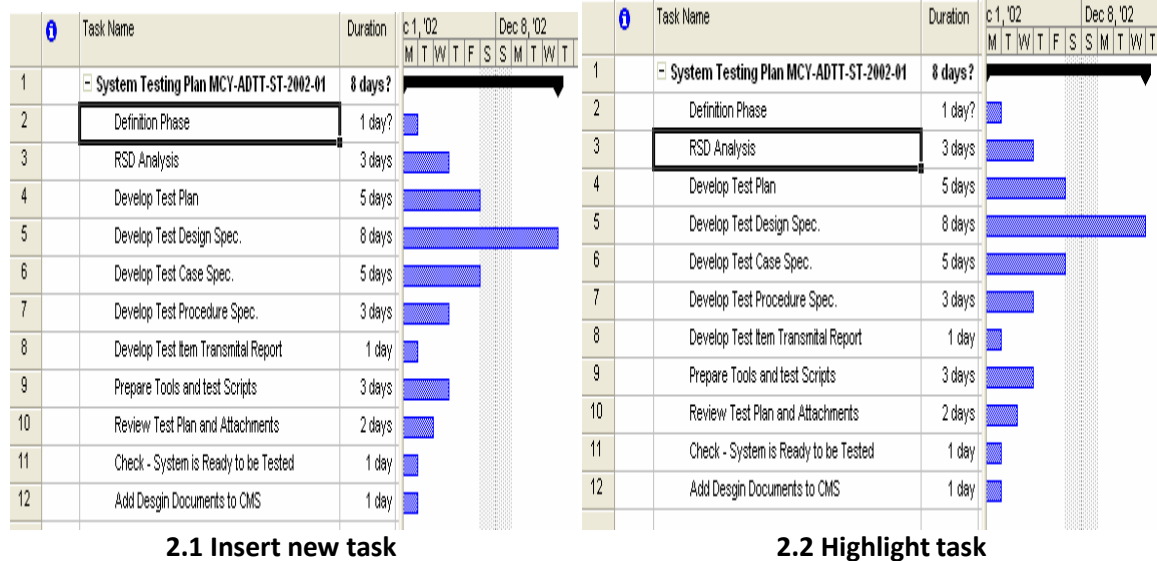
1.2 Highlight task

The final result should look like this, now repeat this steps to create the Subgroups that will represent the phases (Definition and Design).

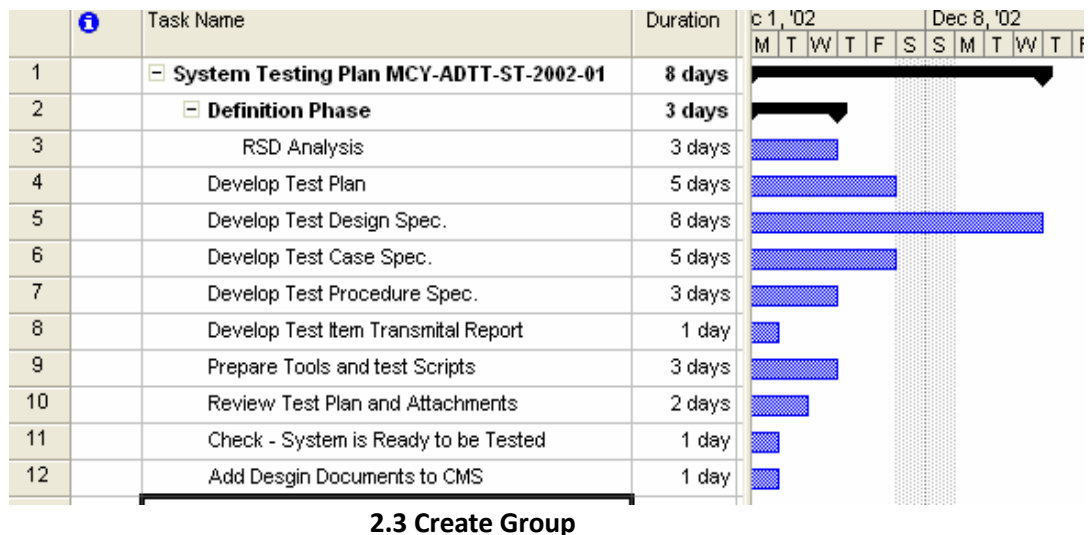
	Task Name	Duration	Dec 1, '02							Dec 8, '02						
			M	T	W	T	F	S	S	M	T	W	T	F	S	S
1	<b>System Testing Plan MCY-ADTT-ST-2002-01</b>	<b>8 days</b>														
2	RSD Analysis	3 days														
3	Develop Test Plan	5 days														
4	Develop Test Design Spec.	8 days														
5	Develop Test Case Spec.	5 days														
6	Develop Test Procedure Spec.	3 days														
7	Develop Test Item Transmittal Report	1 day														
8	Prepare Tools and test Scripts	3 days														
9	Review Test Plan and Attachments	2 days														
10	Check - System is Ready to be Tested	1 day														
11	Add Design Documents to CMS	1 day														

1.3 Create Group

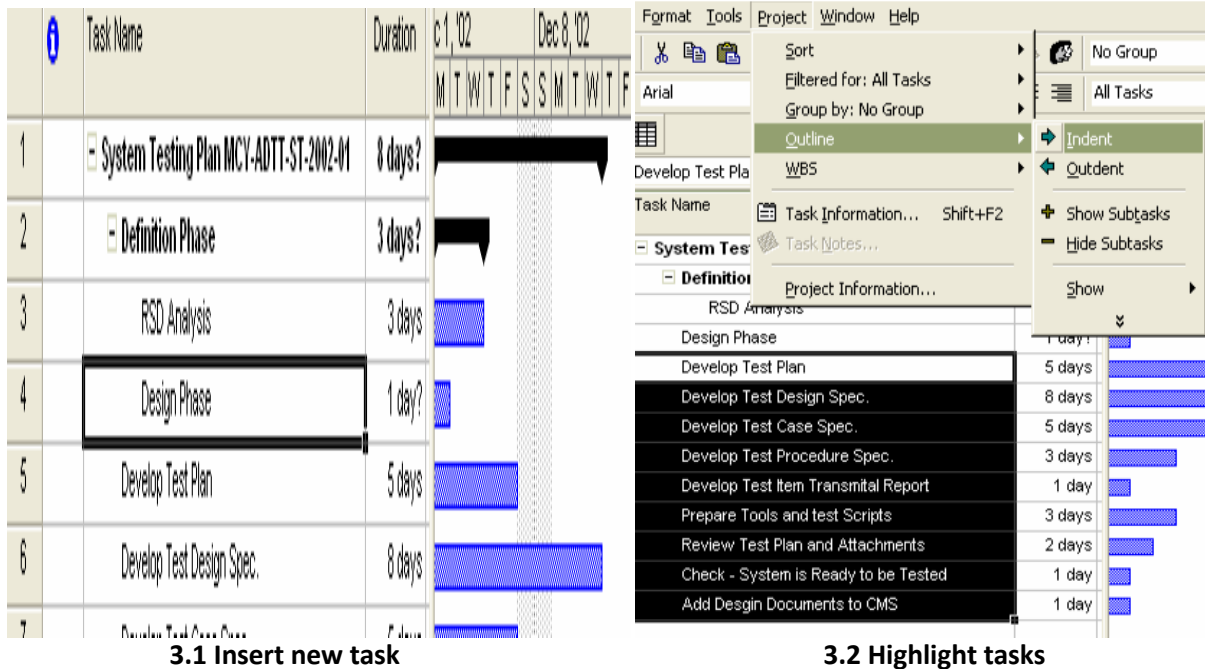
- Insert a new task at the beginning of the definition tasks. Highlight the tasks that are going to be added as subtasks.



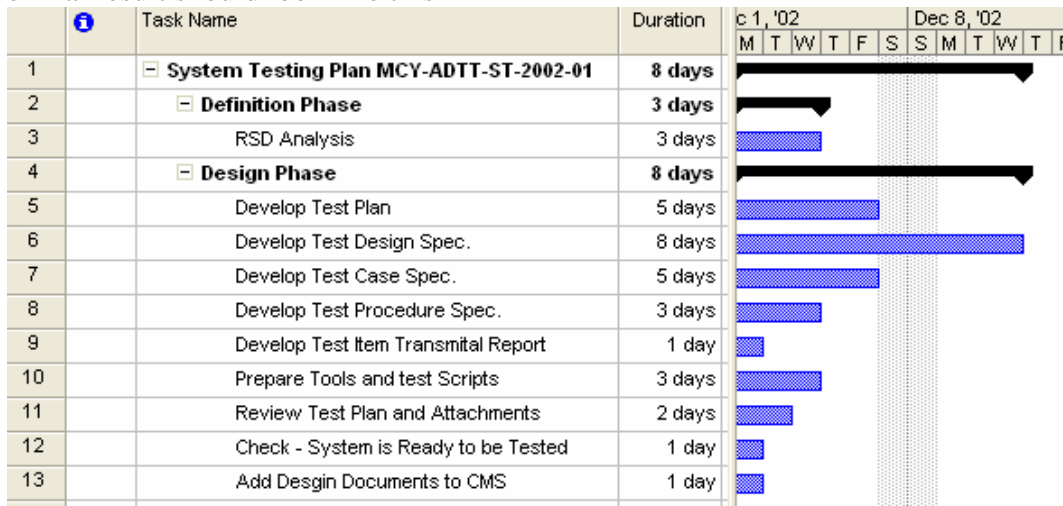
The final result should look like this, now repeat this steps to create the Subgroup that will represent the phase “Design



- Insert a new task at the beginning of the Design tasks. Highlight the tasks that are going to be added as subtasks in the design phase and then Click on the option ‘Indent’



The final result should look like this



**3.3 Create Group**

### Add Resource:

- Got to the view “Resource Sheet”
- Add the necessary resources to the “Resources Sheet”, we are going to use only the Name, Initials and Standard Rate in \$/hr. The resources are going to be taken from the table showed at the beginning of the example, more specifically from the column “Responsibilities”

- Now, with the Resources already register in the project file, go back to the View “Gantt Chart”

Microsoft Project - Project1 - Milton Hurtado

File Edit View Insert Format Tools Project Window Help

No Group

Arial 8 B I U All Resources

PM

		Resource Name	Type	Material Label	Initials	Group	Max. Units	Std. Rate	Ovt
1		Project Manager	Work		PM		100%	\$40.00/hr	\$
2		Test Manager	Work		TM		100%	\$35.00/hr	\$
3		Test Leader	Work		TL		100%	\$30.00/hr	\$
4		Tester D	Work		T_D		100%	\$23.00/hr	\$
5		Tester E	Work		T_E		100%	\$22.00/hr	\$
6		Tester F	Work		T_F		100%	\$22.00/hr	\$
7		Tester G	Work		T_G		100%	\$18.00/hr	\$
8		Tester J	Work		T_J		100%	\$24.00/hr	\$

### Assign Resources:

- Double click the task you want to link to resources available in the “Resource Sheet”
- Then got to the Tab “Resources” and look up the resources you want to relate to the activity (For the example let’s keep the amount of effort of each Resources as 100%, Leveling Resources won’t be covered in this tutorial), finally Click the “Ok” button to finish the assignment.
- Repeat steps 1 and 2 for the rest of the tasks.

Task Information

General Predecessors **Resources** Advanced Notes

Name: RSD Analysis Duration: 3d Estimated

Resources:

Resource Name	Units

Help OK Cancel

Task Information

General Predecessors **Resources** Advanced Notes

Name: RSD Analysis Duration: 3d Estimated



















Resources:

<input checked="" type="checkbox"/>	Test Leader	
	Resource Name	Units
	Test Manager	100%
	Project Manager	100%
	Test Leader	100%
	Project Manager	
	Test Manager	
	Test Leader	
	Tester D	
	Tester E	
	Tester F	
	Tester G	
	Tester J	

Cancel

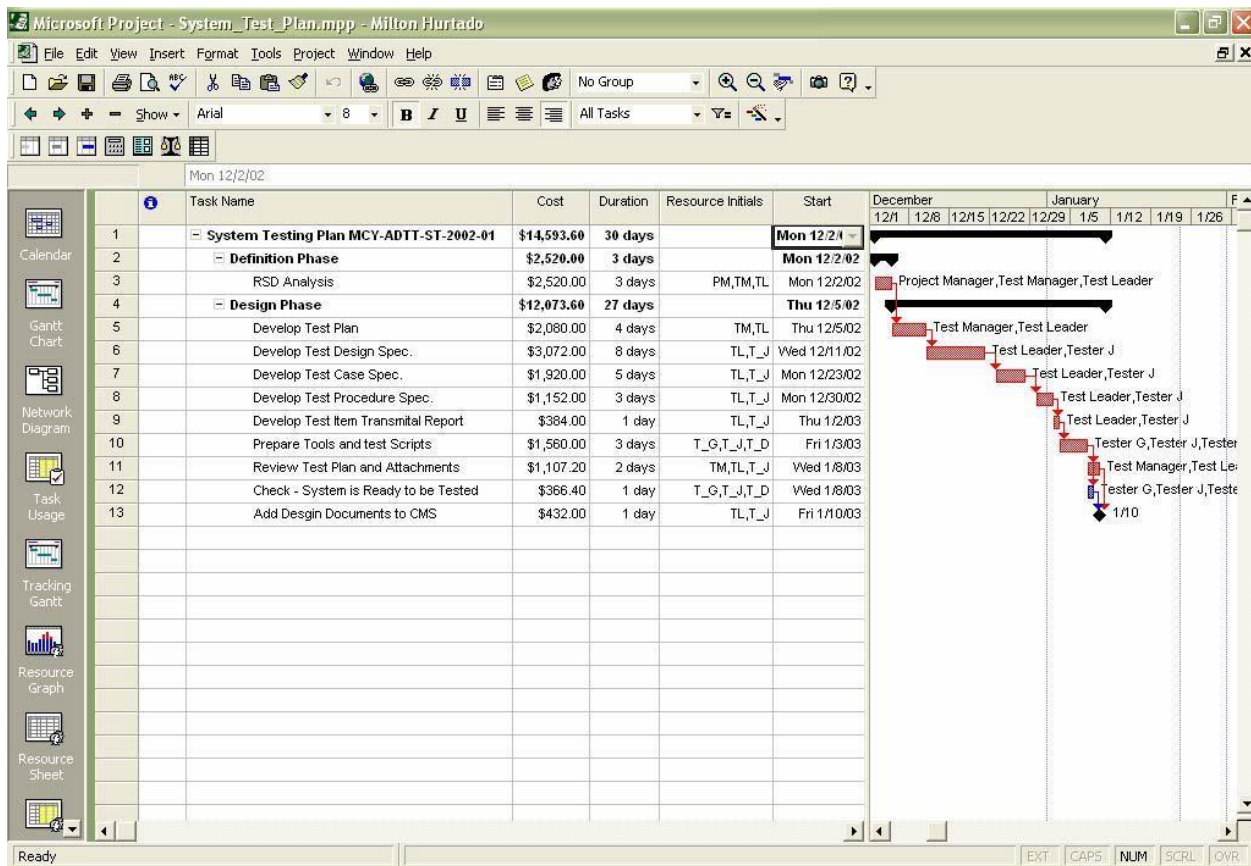
## Insert more Column:

- We can show more information, related to the tasks, in the spreadsheet, one column that might be of general interest is the cost, to do this first perform a Right Click on top of the spread sheet (Specifically In the titles of the Columns), a pop-up menu should appear showing several options, chose the one that says "Insert Column"
- 2. Then lookup the column named "Cost" and then press the "Ok" button

		Task Name	Cost	L	December				January		
					12/1	12/8	12/15	12/22	12/29	1/5	1/12
1		 <b>System Testing Plan MCY-ADTT-ST-2002-01</b>	<b>\$14,593.60</b>								
2		 <b>Definition Phase</b>	<b>\$2,520.00</b>								
3		RSD Analysis	\$2,520.00								
4		 <b>Design Phase</b>	<b>\$12,073.60</b>								
5		Develop Test Plan	\$2,080.00								
6		Develop Test Design Spec.	\$3,072.00								
7		Develop Test Case Spec.	\$1,920.00								
8		Develop Test Procedure Spec.	\$1,152.00								
9		Develop Test Item Transmittal Report	\$384.00								
10		Prepare Tools and test Scripts	\$1,560.00								
11		Review Test Plan and Attachments	\$1,107.20								
12		Check - System is Ready to be Tested	\$366.40								
13		Add Desgin Documents to CMS	\$432.00								

As explained before, you can add and hide columns from the Spread sheet, this lets you show exactly what the people needs to see, below is a view with selected fields: Name, Cost, Duration, Resource initials and Start Date. The reader is welcome to experiment with this features and to explore more views that are offered by MS Project, such as resources usage, cost reports, etc.

	Task Name	Duration	Start	Finish	Predecessors		December				
							11/24	12/1	12/8	12/15	12/22
1	<b>System Testing Plan MCY-ADTT-ST-2002-01</b>	<b>31 days</b>	<b>Mon 12/2/02</b>	<b>Mon 1/13/03</b>							
2	<b>Definition Phase</b>	<b>3 days</b>	<b>Mon 12/2/02</b>	<b>Wed 12/4/02</b>							
3	RSD Analysis	3 days	Mon 12/2/02	Wed 12/4/02							
4	<b>Design Phase</b>	<b>28 days</b>	<b>Thu 12/5/02</b>	<b>Mon 1/13/03</b>							
5	Develop Test Plan	5 days	Thu 12/5/02	Wed 12/11/02	3						
6	Develop Test Design Spec.	8 days	Thu 12/12/02	Mon 12/23/02	5						
7	Develop Test Case Spec.	5 days	Tue 12/24/02	Mon 12/30/02	6						
8	Develop Test Procedure Spec.	3 days	Tue 12/31/02	Thu 1/2/03	7						
9	Develop Test Item Transmittal Report	1 day	Fri 1/3/03	Fri 1/3/03	8						
10	Prepare Tools and test Scripts	3 days	Mon 1/6/03	Wed 1/8/03	9						
11	Review Test Plan and Attachments	2 days	Thu 1/9/03	Fri 1/10/03	10						
12	Check - System is Ready to be Tested	1 day	Thu 1/9/03	Thu 1/9/03	10						
13	Add Desgin Documents to CMS	1 day	Mon 1/13/03	Mon 1/13/03	11,12						



## Conclusion:

Hence the initial phases of a System Testing Plan (Definition and Design) created successfully using MS-Project.



## Experiment 7: Define the Features, Vision, Business objectives, Business rules and stakeholders in the vision document.

### **The Vision Document**

The Vision document is the Rational Unified Process artifact that captures all of the requirements information that we have been discussing in this chapter. As with all requirements documentation, its primary purpose is communication.

You write a Vision document to give the reader an overall understanding of the system to be developed by providing a self-contained overview of the system to be built and the motivations behind building it. To this end, it often contains extracts and summaries of other related artifacts, such as the business case and associated business models. It may also contain extracts from the system use-case model where this helps to provide a succinct and accessible overview of the system to be built.

The purpose of the Vision document is to capture the focus, stakeholder needs, goals and objectives, target markets, user environments, target platforms, and features of the product to be built. It communicates the fundamental "whys and whats" related to the project, and it is a gauge against which all future decisions should be validated.

#### **Features:**

The Vision document is the primary means of communication between the management, marketing, and project teams. It is read by all of the project stakeholders, including general managers, funding authorities, use-case modelers, and developers. It provides

- A high-level (sometimes contractual) basis for the more detailed technical requirements
- Input to the project-approval process (and therefore it is intimately related to the business case)
- A vehicle for eliciting initial customer feedback
- A means to establish the scope and priority of the product features It is a document that

gets "all parties working from the same book."

Because the Vision document is used and reviewed by a wide variety of involved personnel, the level of detail must be general enough for everyone to understand. However, enough detail must be available to provide the team with the information it needs to create a use-case model and supplementary specification.



The document contains the following sections:

- **Positioning:** This section summarizes the business case for the product and the problem or opportunity that the product is intended to address. Typically, the following areas should be addressed:
  - **The Business Opportunity:** A summary of business opportunity being met by the product
  - **The Problem Statement:** A solution-neutral summary of the problem being solved focusing on the impact of the problem and the benefits required of any successful solution
  - **Market Demographics:** A summary of the market forces that drive the product decisions.
  - **User Environment:** The user environment where the product could be applied.
- **Business Rules are statements (or conditions) that tell a person whether they can perform a specific action that relates to how the business operates. Business rules also supply the criteria and conditions for making these decisions.**
- **Business objectives** may include what needs to be done to enable the business rule to be implemented. In other words, a business requirement may not be valid if it contradicts or breaks an existing business rule.

In other words, **business rules** are statements defining or limiting any side of the business (Wiegers). Generally, those are the rules adopted in business practice, either in legislation or in the industry, and in any case we must comply with them when implementing the software.

They often involve very specific criteria or conditions for compliance. Moreover, rules can apply to not only to business processes and standards, but also to individuals, procedures, general corporate behavior, or anything else. The good news is that in fact most business rules are fairly static.

Behind those rules, there are usually some documents that regulate the business at the level of law or industry standard (or some other standard) about how the business or its software should work. Certain laws may regulate the industry itself or govern the rules of development, hosting, etc. If, for example, we are developing a system for a real estate domain, then there are a huge number of rules that are generated by different authorities (f.e. NAR in USA).

Some domain regulators (the same real estate) even introduce rules for API, building

databases and calculation algorithms in the system.

Though, a BA pro should remember that while many business rules originate outside of the business and are easy to identify and follow, others originate internally and are not always evident.

Business rules are extremely important as they define entities, attributes, relationships, and constraints. They allow the team to design rules and constraints and create the correct data model. They also enable the developer and business analyst to understand business processes as well as the nature, role, and volume of data.

**Business objectives** are what we use to enforce business rules and ensure compliance with them. They indicate what the end result or the future state is and are usually developed when looking for solutions to meet a business need/rule or to achieve a business goal/objective.

Business rules form the basis for making process decisions and eliciting business requirements. Thus, changing one word in a business rule can mean different or additional requirements.

However, a business rule can exist without business requirements, and business requirements exist in the context of a broader framework such as business rules, goals, objectives, mandates, or company vision and mission. There may be many different alternative business requirements for implementing/enforcing a set of business rules. The relevant business requirements depend on the business strategy, business risk tolerance, and budget.

Let us provide some examples here.

Business Rule:

*Users must pay for subscription before they gain access to the system.*

*System subscription must be individual, members of one office shouldn't share the same credentials.*

Business Requirement (remember that we can comply with business rule using different strategies, these are just examples):

*As an administrator, I want the system to check confirmed payment in the CRM before granting access.*

*As an administrator, I want the system to allow usage of no more than two browser windows with the same credentials simultaneously.*

*OR As an administrator, I want the system to allow usage of no more than two devices with the same credentials simultaneously.*

### **Stakeholders:**

- **Stakeholders and Users:** This section describes the stakeholders in, and users of, the product. The stakeholder roles and stakeholder types are defined in the project's Vision document—the actual stakeholder representatives are identified as part of the project plan just like any other resources involved in the project.
- **Key Stakeholder and User Needs:** This section describes the key needs that the stakeholders and users perceive the product should address. It does not describe their specific requests or their specific requirements, because these are captured in a separate stakeholder requests artifact. Instead, it provides the background and justification for why the requirements are needed.
- **Product Overview:** This section provides a high-level view of the capabilities, assumptions, dependencies (including interfaces to other applications and system configurations), and alternatives to the development of the product.
- **Features:** This section lists the features of the product. Features are the high-level capabilities (services or qualities) of the system that are necessary to deliver benefits to the users and satisfy the stakeholder and user needs. This is the most important, and consequently usually the longest, section of the Vision document.
- **Other Product Requirements:** This section lists any other high-level requirements that cannot readily be captured as product features. These include any constraints placed on the development of the product and any requirements the planned product places on its operating environment.

In many cases, a lot more work is put into uncovering the business opportunity and understanding the market demographics related to the proposed product than is reflected in the Vision document. This work is usually captured in-depth in business cases, business models, and market research documents. These documents are then summarized in the Vision document to ensure that they are reflected in the ongoing evolution of the products specification.

We recommend that the Vision document be treated primarily as a report and that the stakeholder types, user types, stakeholder roles, needs, features, and other product requirements be managed using a requirements management tool. If the list of features is to be generated, it is recommended that they be presented in two sections:

- In-Scope features
- Deferred features

Appendix B, Templates for Artifacts, contains a comprehensive Vision document template that contains a full definition of the structure and contents of a typical Rational Unified Process vision document. Appendix C, Case Study, contains a completed Vision document that complements the examples that appear within this chapter.

## Experiment 8: Define the functional and non-functional requirements of the system to be automated by using Use cases and document in SRS document.

**Aim:** To define functional and non-functional requirements of SVCET-LIS System.  
(Library Information System)

### **Problem Statement**

SVCET as the size and capacity of the institute is increasing with the time, it has been proposed to develop a Library Information System (LIS) for the benefits of the student and employees of the institute.

LIS will enable the members to borrow a book (or return it) with ease while sitting in a desk/chamber. The system also enables a member to extend the date of his borrowing if no other booking for that particular book has been made. For the library staff, this system aids them easily handle day-to-day book transactions.

The librarian, who administrative privileges and complete control over the system, new Record book entry and remove a record in case of any book is taken off the shelf, non member use this system to browse/ search books online.

The final deliverable would a web application (use HTML 5.0), which should run only the institute LAN. Although this reduce security risk of the software to a large extent, care should be taken no confidential information (eg. Passwords) is stored in plain text.

From the given problem statement we can identify a list of actors and use cases shown in tables 1 & 2. We assign an identifier to each use case, which we would be using to map from the software requirements identified earlier.

### **Identification of functional requirements**

The above problem statement gives a brief description of the proposed system. From the above, even without doing any deep analysis, we might easily identify some of the basic functionality of the system:

- **New user registration:** Any member of the institute who wishes to avail the facilities of the library has to register himself with the Library Information System. On successful registration, a user ID and password would be provided to the member. He has to use this credentials for any future transaction in LIS.
- **Search book:** Any member of LIS can avail this facility to check whether any particular book is present in the institute's library. A book could be searched by its:
  - Title
  - Authors name
  - Publisher's name

- **User login:** A registered user of LIS can login to the system by providing his employee ID and password as set by him while registering. After successful login, "Home" page for the user is shown from where he can access the different functionalities of LIS: search book, issue book, return book, reissue book. Any employee ID not registered with LIS cannot access the "Home" page -- a login failure message would be shown to him, and the login dialog would appear again. This same thing happens when any registered user types in his password wrong. However, if incorrect password has been provided for three time consecutively, the security question for the user (specified while registering) with an input box to answer it are also shown. If the user can answer the security question correctly, a new password would be sent to his email address. In case the user fails to answer the security question correctly, his LIS account would be blocked. He needs to contact with the administrator to make it active again.
- **Issue book:** Any member of LIS can issue a book against his account provided that:
  - The book is available in the library i.e. could be found by searching for it in LIS
  - No other member has currently issued the book
  - Current user has not issued the maximum number of books that can

If the above conditions are met, the book is issued to the member. Note that this FR would remain **incomplete** if the "maximum number of books that can be issued to a member" is not defined. We assume that this number has been set to four for students and research scholars, and to ten for professors. Once a book has been successfully issued, the user account is updated to reflect the same.

- **Return book:** A book is issued for a finite time, which we assume to be a period of 20 days. That is, a book once issued should be returned within the next 20 days by the corresponding member of LIS. After successful return of a book, the user account is updated to reflect the same.
- **Reissue book:** Any member who has issued a book might find that his requirement is not over by 20 days. In that case, he might choose to reissue the book, and get the permission to keep it for another 20 days. However, a member can reissue any book at most twice, after which he has to return it. Once a book has been successfully reissued, the user account is updated to reflect the information.

In a similar way we can list other functionality offered by the system as well. However, certain features might not be evident directly from the problem system, but which, nevertheless, are required. One such functionality is "User Verification". The LIS should be able to judge between a registered and non-registered member. Most of the functionality would be available to a registered member. The "New User Registration" would, however, be available to non-members. Moreover, an already registered user shouldn't be allowed to register himself once again.

Having identified the (major) functional requirements, we assign an identifier to each of them [v] for future reference and verification. Following table shows the list:

Table 01: Identifier and priority for software requirements

#	Requirement	Priority
R1	New user registration	High
R2	User Login	High
R3	Search book	High
R4	Issue book	High
R5	Return book	High
R6	Reissue book	Low

### **Identification of non-functional requirements**

Having talked about functional requirements, let's try to identify a few non-functional requirements.

- **Performance Requirements:**

- This system should remain accessible 24x7
- At least 50 users should be able to access the system altogether at any given time

- **Security Requirements:**

- This system should be accessible only within the institute LAN
- The database of LIS should not store any password in plain text -- a hashed value has to be stored

- **Software Quality Attributes**

- **Database Requirements**

- **Design Constraints:**

- The LIS has to be developed as a web application, which should work with Firefox 5, Internet Explorer 8, Google Chrome 12, Opera 10
- The system should be developed using HTML 5

Once all the functional and non-functional requirements have been identified, they are documented formally in SRS, which then serves as a legal agreement.

### **Conclusion:**

Hence the functional and non functional requirements of SVCET LIS system is successfully analysed and determined.



## Experiment 9: Develop a tool which can be used for quantification of all the non-functional requirements.

**Aim:** To Develop a tool which can be used for quantification of all the non-functional requirements

### **Introduction:**

Neoload is a low-priced high-efficiency load and stress testing tool that is used to measure the performance of web and mobile applications.

Neoload simulates traffic through virtual users to determine the application performance under load and analyze the transaction response times and pinpoint the number of the simultaneous users which the internet, intranet or the mobile application can handle.

I have worked on various performance testing tools that include LoadRunner, JMeter, RPT, and Neoload. Among all the performance testing tools, I feel comfortable with Neoload because of its user-friendly record and script enhancement options that make the tester's job much easier when compared to the other tools.

### **Performance Testing**

Performance testing is used to determine how fast a website or app will be responding to a user request when multiple users access it. It is also performed to check the stability of the system i.e. whether the server is able to handle thousands of users at a time.

### **Why Performance Testing?**

If the site or app is not performing well then it may lead to user drops i.e. the user may not be interested to use that site due to poor performance.

We need to do Performance testing in order to get the answers to the below questions:

1. *How fast is my system responding to load?*
2. *Is my system able to handle a large volume of users?*
3. *Is my system responding quickly? If no what will be the reason?*

### **Few definitions**

#### **#1) Load Testing**

- Testing the application with different workloads based on the usage pattern. It gives the probable workload application support under the normal working conditions.
- To determine the System's response time & resource utilization under load.

#### **#2) Stress Testing**

- Load testing executed to find issues due to low resources or competition for resources to find the Max capacity of the system.
- To identify the bottlenecks in the application like DB connection & to determine the Max no of user requests that a server can handle.

### #3) Endurance Testing

- To identify the stability of an application under constant load for an extended period of time.
- Helps to determine problems related to memory leaks, garbage collection etc.

## Neoload Installation

### Step #1:

Download the latest version and choose the OS bit version depending on your operating system from [here](#).

### Step #2:

Install the controller on the system.

- It is recommended to install “load generator agent” on the designated agent machines to handle more user load.
- Install Monitoring agent on the servers which are to be monitored.

### Step #3:

Once Neoload is installed successfully. Open the Neoload and create a new project.

### Step #4:

Once the project is created successfully then the tool looks as shown below.

**Neoload includes 3 components under one section. They are:**

- Design
- Runtime
- Results

**Design:** It is for script design like VUgen in LoadRunner.

Design again includes 3 sections as shown in the above screenshot.

- **User Paths:** Used to record and enhance the scripts.
- **Populations:** Used to add scripts to the scenario.
- **Monitors:** Used to add the monitoring servers.

**Runtime:** It is like the controller in LoadRunner. Runtime is used to create scenarios. **Results:** It is like

Analysis in LoadRunner. And is used to run the test and generate a report.

## Conclusion

So far we learned what is performance testing, why is performance testing done along with few types of Performance testing and how Neoload is different from other tools.

## Experiment 10: Write C/Java/Python program for classifying the various types of coupling.

### Coupling:

In order to create an effective application that is easy to develop, easy to maintain and easy to update, we need to know the concept of coupling and cohesion. Coupling refers to the extent to which a class knows about the other class.

*There are two types of coupling -*

Tight Coupling(a bad programming design)

Loose Coupling(a good programming design)

### *Tight Coupling*

If a class A has some public data members and another class B is accessing these data members directly using the dot operator(which is possible because data members were declared public), the two classes are said to be tightly coupled.

Such tight coupling between two classes leads to the bad designing. You would ask - Why? Well, here's the explanation.

Let's say, the same class A has a String data member, name, which is declared public and this class also has getter and setter methods that have implemented some checks to make sure -

A valid access of the data member, name i.e. it is only accessed when its value is not null, and

A valid setting of the data member, name i.e. it cannot be set to a null value.

But these checks implemented in the methods of class A to ensure a valid access and valid setting of its data member, name, are bypassed by its direct access by class B, due to tight coupling between two classes.

Let's understand this by a coding example -

//Java - Example of tight coupling

```
class A
```

```
{
```

```
public String name;           //public data member of A class
```

```
public String getName()
```

```
{
```

```
    if(name!=null) //Checking a valid access of instance variable, "name"
```

```
return name;
```

```
    else  
    return "not initiaized";
```

```
}
```

```
public void setName(String s)
```

```
{
```

```
    if (s==null) //Checking a valid setting of instance variable, "name" System.out.println("You  
        can't initialized name to a null");
```

```
    else
```

```
        name = s;
```

```
}
```

```
}
```

```

class B
{
public static void main(String... ar)
{
    //Creating an object of class A
    A ob = new A();

    //Directly setting the value of data member "name" of class A, due to tight coupling between
    the class A and B
    ob.name=null;

    //Direct access of data member "name" of class A, due to tight coupling between two classes
    System.out.println("Name is " + ob.name);
}
}

```

*Output is :*

Name is null

*Program Analysis*

Class A has an instance variable, name, which is declared public.

Class A has two public getter and setter methods which check for a valid access and valid setting of data member - name.

Class B creates an object of class A and directly sets the value of its data member, name, to null and directly accesses its value because it was declared public.

Thus, the validity checks for the data member, name, which were implemented within getName() and setName() methods of class A are bypassed, which shows that class A is tightly coupled to class B, and it is a bad design.

*Loose Coupling*

A good application designing is creating an application with loosely coupled classes by following encapsulation, i.e. by declaring data members of a class with the private access modifier, which disallows other classes to directly access these data members, and forcing them to call the public getter, setter methods to access these private data members.

Let's understand this by modifying our previous coding example of tight coupling to loose coupling.

//Java - Example of loose coupling

class A

{

private String name;               //data member "name" is declared private to implement loose coupling.

public String getName()

{

    if(name!=null)           //Checking a valid access to data member, name

        return name;

    else

        return "not initiaized";

}

public void setName(String s)

{

    if (s==null)   //Checking a valid setting of data member, name

        System.out.println("You can't initialize name to a null");

    else

        name = s;

}

}

```
class B
{
public static void main(String... ar)
{
    //Creating an object of class AA
    A ob= new A();

    //Calling setter method, as the direct access of "name" is not possible i.e. loose coupling
    between classes
    ob.setName(null);

    //Calling getter method, as the direct access of "name" is not possible i.e. loose coupling
    System.out.println("Name is " + ob.getName());
}
}
```

*Output is :*

You can't initialize name to a null

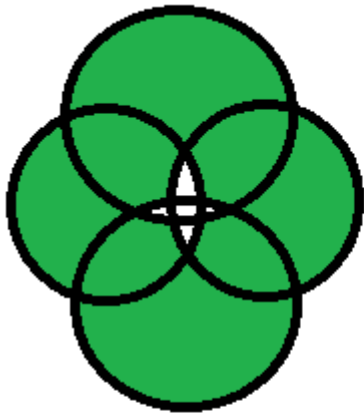
Name is not initialized

*Program Analysis*

Class A has an instance variable, name, which is declared private.

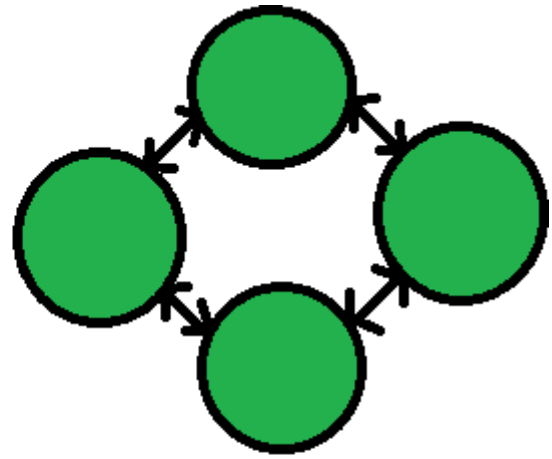
Class A has two public getter and setter methods which check for a valid access and valid setting of the data member, name.

Class B creates an object of class A, calls the getName() and setName() methods and their implemented checks are properly executed before the value of instance member, name, is accessed or set. It shows that class A is loosely coupled to class B, which is a good programming design.



#### **Tight coupling:**

- 1. More Interdependency**
- 2. More coordination**
- 3. More information flow**



#### **Loose coupling:**

- 1. Less Interdependency**
- 2. Less coordination**
- 3. Less information flow**

### **Difference between tight coupling and loose coupling**

Tight coupling is not good at the test-ability. But loose coupling improves the test ability.

Tight coupling does not provide the concept of interface. But loose coupling helps us follow the GOF principle of program to interfaces, not implementations.

In Tight coupling, it is not easy to swap the codes between two classes. But it's much easier to swap other pieces of code/modules/objects/components in loose coupling.

Tight coupling does not have the changing capability. But loose coupling is highly changeable.



## Experiment 11: Write a C/Java/Python program for classifying the various types of cohesion.

### Cohesion

Cohesion refers to the extent to which a class is defined to do a specific specialized task. A class created with high cohesion is targeted towards a single specific purpose, rather than performing many different purposes.

**There are two types of cohesion** - Low cohesion(a bad programming design) High Cohesion(a good programming design)

#### Low Cohesion

When a class is designed to do many different tasks rather than focus on a single specialized task, this class is said to be a "low cohesive" class. Low cohesive classes are said to be badly designed, as it requires a lot of work at creating, maintaining and updating them. Let's understand this by a code -

//Java - Example of a low cohesion

```
class class PlayerDatabase
{
    public void
    connectDatabase(); public
    void printAllPlayersInfo();
    public void
    printSinglePlayerInfo(); public
    void printRankings();

    public void
    printEvents(); public
    void closeDatabase();
}
```

## Program Analysis

Here, we have a class PlayerDatabase which is performing many different tasks like connecting to a database, printing the information of all the players, printing information of a single player, printing all the events, printing all the rankings and finally closing all opened database connections.

Now, such a class is not easy to create, maintain and update, as it is involved in performing many different tasks i.e. a programming design to avoid.

## High Cohesion

A good application design is creating an application with high cohesive classes, which are targeted towards a specific specialized task and such classes are not only easy to create but also easy to maintain and update.

// Java program to illustrate

// high cohesive behavior

```
class
```

```
    Name {
```

```
        String
```

```
        name;
```

```
        public String getName(String name)
```

```
        {
```

```
            this.name = name; return name;
```

```
        }
```

```
    }
```

```
class Age { int age;
```

```
    public int getAge(int age)
```

```
    {
```

```
        this.age = age; return age;
```

```
    }
```

```
}
```

```
class Number { int mobileno;

    public int getNumber(int mobileno)
    {
        this.mobileno = mobileno; return mobileno;
    }
}

class Display {
    public static void main(String[] args)
    {
        Name n = new Name();

        System.out.println(n.getName("Omsah"));

        Age a = new Age();

        System.out.println(a.getAge(10));

        Number no = new Number();

        System.out.println(no.getNumber(1234567891));

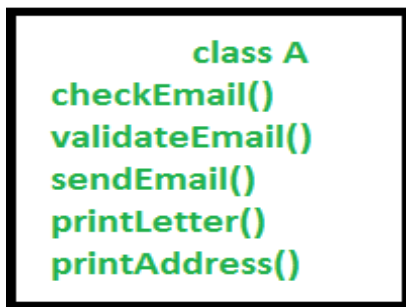
    }
}
```

#### Output

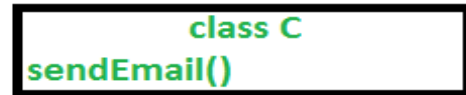
Omsah 10

1234567891

*Pictorial view of high cohesion and low cohesion:*



**Fig: Low cohesion**



**Fig: High cohesion**

## Experiment 12: Write a C/Java/Python program for object-oriented metrics for design proposed by Chidamber and Kremer. (Popularly called CK metrics).

### **Aim**

To Write a Java program for object-oriented metrics for design proposed by Chidamber and **Kremer**

### **Algorithm**

Step 1: Select the object-oriented design to measure the quality assessment.

Step 2: Select the quality attributes of the object-oriented design to measure the particular domain. Step 3:

Identify the design properties of an object-oriented design for identifying and related to the quality attributes selected in step 2.

Step 4: Form the related object-oriented design metrics to design properties and desired values for quantify the design properties of step 3 and find design metrics-design attributes relationship.

Step 5: Calculate the metric values of each metrics and tabulate their values for each classes of the entire system for easy operations and find quality assessment of the design.

Step 6: Find the design attribute effectiveness using the obtained metric values from step 5 and further using computation formula. And check the design attributes using the desired values and design properties weights.

Step 7: Closely examine the design attributes from computation formula and metrics values, modify and improve the individual classes of the entire object-oriented system.

### **Program:**

```
package gr.spinellis.ckjm.ant;

import gr.spinellis.ckjm.MetricsFilter; import
gr.spinellis.ckjm.PrintPlainResults; import
java.io.File;

import java.io.FileOutputStream; import
java.io.IOException; import
java.io.OutputStream;
```

```
import java.io.PrintStream;
import org.apache.tools.ant.BuildException; import
org.apache.tools.ant.DirectoryScanner;
import org.apache.tools.ant.taskdefs.MatchingTask; import
org.apache.tools.ant.types.Path;
public class CkjmTask extends MatchingTask { private
    File outputFile;
    private File classDir; private Path
    extdirs; private String format;
    public CkjmTask() {
        this.format = "plain";
    }
    public void setFormat(String format) {
        this.format = format;
    }
    public void setOutputfile(File outputfile) {
        this.outputFile = outputfile;
    }
    public void setClassdir(File classDir) {
        this.classDir = classDir;
    }
    public void setExtdirs(Path e) { if (extdirs
        == null) {
            extdirs = e;
        } else {
            extdirs.append(e);
        }
    }
}
```

```

public Path getExtdirs() { return
    extdirs;
}

public Path createExtdirs() { if (extdirs
    == null) {
        extdirs = new Path(getProject());
    }    return extdirs.createPath();
}    public void execute() throws BuildException { if
    (classDir == null) {
        throw new BuildException("classdir attribute must be set!");
    }
    if (!classDir.exists()) {
        throw new BuildException("classdir does not exist!");
    }
    if (!classDir.isDirectory()) {
        throw new BuildException("classdir is not a directory!");
    }

    if (extdirs != null && extdirs.size() > 0) {
        if (System.getProperty("java.ext.dirs").length() == 0)
            System.setProperty("java.ext.dirs", extdirs.toString());
        else
            System.setProperty("java.ext.dirs",
                System.getProperty("java.ext.dirs") + File.pathSeparator + extdirs);
    }

    DirectoryScanner ds = super.getDirectoryScanner(classDir); String
    files[] = ds.getIncludedFiles();

```

```

if (files.length == 0) {

    log("No class files in specified directory " + classDir);

} else {

    for (int i = 0; i < files.length; i++) {

        files[i] = classDir.getPath() + File.separatorChar + files[i];

    }

    try {

        OutputStream outputStream = new FileOutputStream(outputFile); if

        (format.equals("xml")) {

            PrintXmlResults outputXml = new PrintXmlResults( new

                PrintStream(outputStream));

            outputXml.printHeader();

            MetricsFilter.runMetrics(files, outputXml);

            outputXml.printFooter();

        } else {

            PrintPlainResults outputPlain = new PrintPlainResults( new

                PrintStream(outputStream));

            MetricsFilter.runMetrics(files, outputPlain);

        }    outputStream.close();

    } catch (IOException ioe) {

        throw new BuildException("Error file handling: "

            + ioe.getMessage());    }    } }}

```

### Print XMLResult

```

package gr.spinellis.ckjm.ant;

import gr.spinellis.ckjm.CkjmOutputHandler;

import gr.spinellis.ckjm.ClassMetrics;

```



```

import java.io.PrintStream;

public class PrintXmlResults implements CkjmOutputHandler {

private PrintStream p;

    public PrintXmlResults(PrintStream p) { this.p = ;

    } public void printHeader() { p.println("<?xml version=\"1.0\"?>"); p.println("<ckjm>");

    } public void handleClass(String name, ClassMetrics c) { p.print("<class>\n" +

        "<name>" + name + "</name>\n" + "<wmc>" + c.getWmc() + "</wmc>\n" + "<dit>" + c.getDit()

        + "</dit>\n" + "<noc>" + c.getNoc() + "</noc>\n" + "<cbo>" + c.getCbo() + "</cbo>\n" + "<rfc>"

        + c.getRfc() + "</rfc>\n" + "<lcom>" + c.getLcom() + "</lcom>\n" + "<ca>" + c.getCa() +

        "</ca>\n" + "<npm>" + c.getNpm() + "</npm>\n" + "</class>\n"); }

    public void printFooter () { p.println("</ckjm>"); }

}

```

#### **Result:**

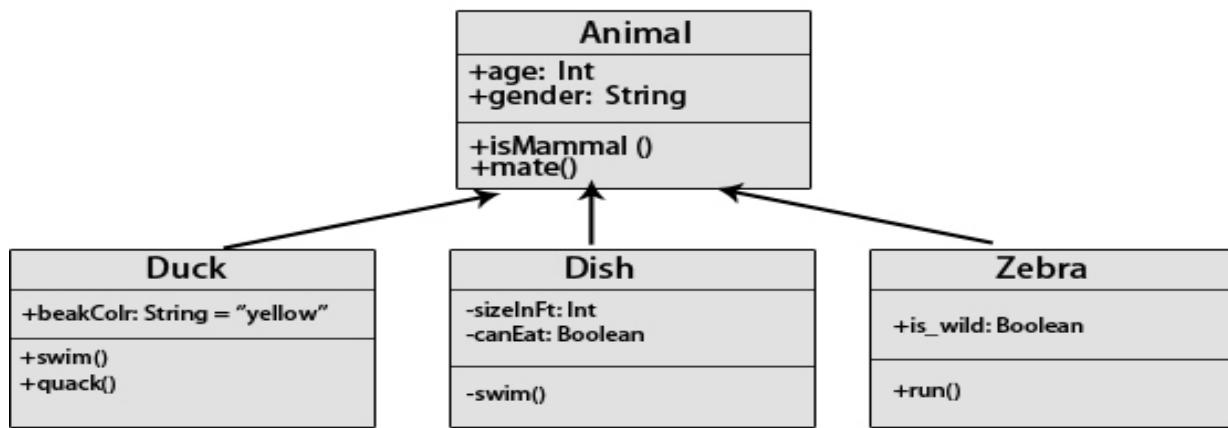
Thus the Java program for object oriented metrics for design proposed by Chidamber and Kremer executed.

## Experiment 13: Exp. No: 13- Draw a complete class diagram and object diagrams using Rational tools.

### *Introduction to Class Diagram*

The class diagram is one of the types of UML diagrams which is used to represent the static diagram by mapping the structure of the systems using classes, attributes, relations, and operations between the various objects. A class diagram has various classes; each has three-part; the first partition contains a Class name which is the name of the class or entity which is participated in the activity, the Second partition contains class attributes that show the various properties of the class, the third partition contains class operations which shows various operations performed by the class, relationships shows the relation between two classes.

## Class Diagram



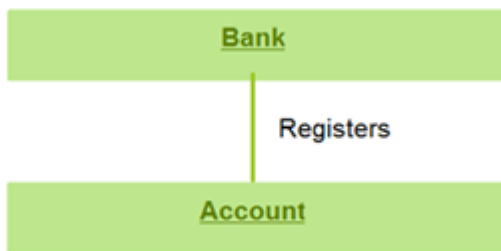
## Relationships

In a class diagram, it is necessary that there exists a relationship between the classes. Unfortunately, the similarity of various relationships often makes it difficult to understand them.

Below are the relationships which exist in a class diagram.

### 1. Association

Between two other classes in an association relationship, an association class forms a part of it. Additional information about the relationship could be obtained by attaching the association relationship with the association class. Various operations, attributes, etc., are present in the association class.

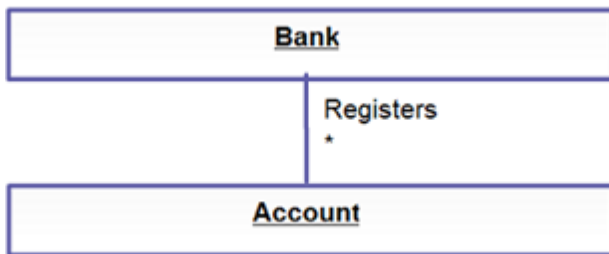


The below diagram shows an association between bank and account. class diagram (Association)

### 2. Multiplicity

The number of elements or cardinality could be defined by multiplicity. It is one of the most misunderstood relationships which describes the number of instances allowed for a particular element by providing an inclusive non-negative integers interval. It has both lower and upper bound. For example, a bank would have many accounts registered to it.

Thus near the account class, a star sign is present.



class diagram ( Multiplicity )

### 3. Directed Association

This is a one-directional relationship in a class diagram that ensures the flow of control from one to another classifier. The navigability is specified by one of the association ends. The relationship between two classifiers could be described by naming any association. An arrow indicates the direction of navigation.

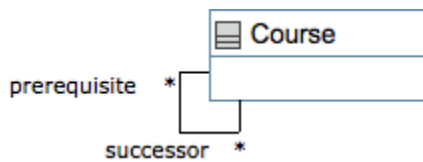
The below example shows an arrowhead relationship between the container and the contained.



class diagram (Directed Association)

### 4. Reflexive Association

The association of a class to itself is known as Reflexive association, which could be divided into Symmetric and Asymmetric type associations. In Symmetric reflexive association, the semantics of each association end has no logical difference, whereas, in Asymmetric Reflexive Association, the associated class is the same, but there is a semantic difference between the ends of the association.



## Reflexive Association

### 5. Aggregation

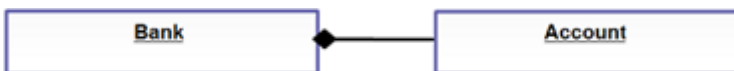
In this type of relationship, a more complex object is created by assembling different objects together. The interaction within the different groups of objects is defined by Aggregation. The integrity of the objects is protected, and the response of the assembled objects is decided by the control object. In aggregation, the classes nurture the 'has a relationship'.



## Aggregation

### 6. Composition

It is a form of aggregation which represents the whole-part relationship. Here, the part classifier lifetime is dependent on the whole classifier lifetime. In a class, a strong life-cycle is represented by the composition relationship. There is usually a one-direction flow of data here. It is generally indicated by a solid line.

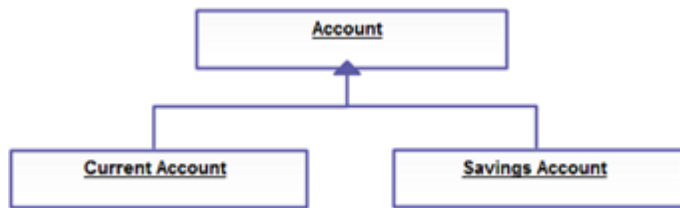


## Composition

### 7. Generalization

In this kind of relationship, the child model is based on the parent model. The relationship is used to describe various use-case diagrams and ensures that the child class receives the properties present in the parent. The child model could reuse the attributes of the parent.

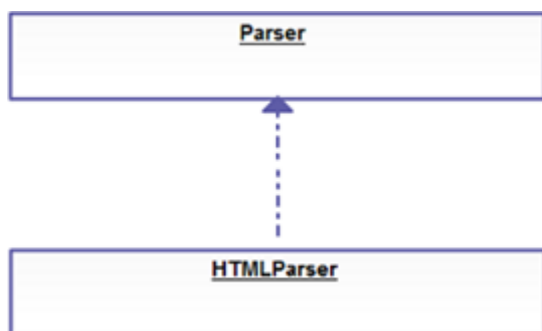
model with the help of the generalization relationship. Hence the distinct attributes need to be defined only in the child; the rest it would inherit from the parent. There could be single parents, multiple children, or multiple parents, single child characteristics in this relationship. There are no names in the generalization relationships. It is also known as the 'is a' relationship.



## Generalization

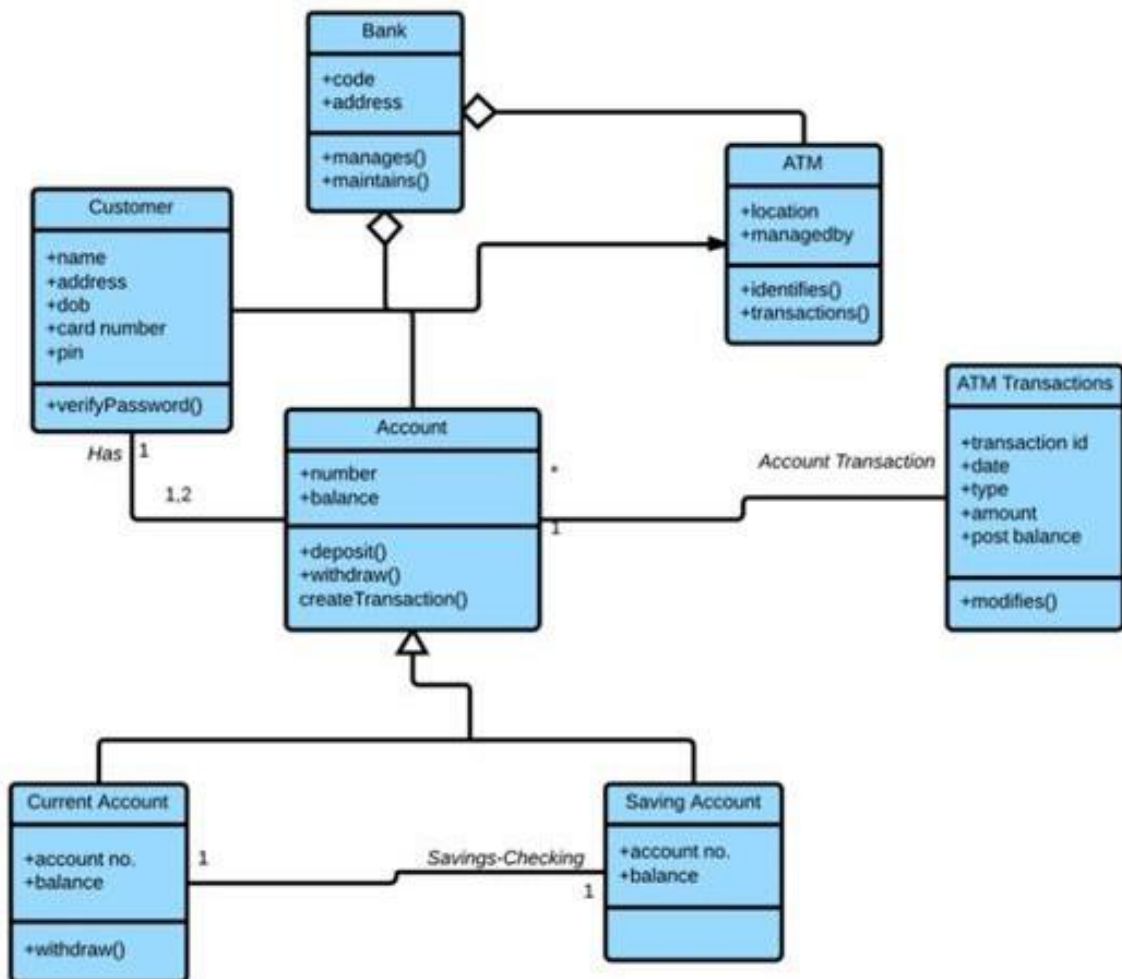
### 8. Realization

The behavior of one model element is realized by the specified behavior of another model element. This type of relationship doesn't have any names.



## Realization

# Example of Class Diagram



## Object Diagram

An Object Diagram can be referred to as a screenshot of the instances in a system and the relationship that exists between them. Since object diagrams depict behaviour when objects have been instantiated, we are able to study the behavior of the system at a particular instant. Object diagrams are vital to portray and understand functional requirements of a system.

In other words, "An object diagram in the Unified Modeling Language (UML), is a diagram that shows a complete or partial view of the structure of a modeled system at a specific time."

# ATM Object Diagram

