NMS

(NGO Management System)

System Analysis and System Description

- 14CS30026 Pranjal Shankhdhar

- 14CS10055 Yash Agrawal

Table of Contents

1.0 Introduction

1.1 Purpose

This document describes all specifications for the NMS developed for the NGO for helping poor school children.

1.2 Scope

- This software is designed to provide the NGO a system for its working.
- It allows helpers to register poor students and maintain their info including marks. Also, they can accept donations of paraphernalia like books, bags, dress etc.
- Donors can register and store the amount of the money they intend to contribute and their frequency of donation per year.
- Manager can access various records and contact donors for money. Manager can see expenditure made by NGO. Manager can also register and remove helpers.

1.3 Glossary

Title	**Description**
NMS	NGO Management System
Helpers	Register poor students and maintain their info. Also accept small money donations and item donations.

Manager	Looks over all activities/lists and contacts donors for money.
Donor	Pledge to NGO to help it financially.
Database	Where all information is stored
DFD	Data Flow Diagram
GUI	Graphical User Interface
ERD	Entity Relationship Diagram
I/O	Input/Output
OOD	Object Oriented Architecture
OOA	Object Oriented Design

1.4 References

1. [IEEE] the applicable IEEE standards are published in
"IEEE StandardsCollection", 2001 Edition.
2. Class slides.

1.5 Overview

Section 2 consists of Feasibility study which discusses the problem in depth and discusses alternatives.

Section 3 consists of Requirements Analysis in which review the problem and find out requirements(functional and non functional).

In Section 4, we define architecture and platform.

2.0 Feasibility Study

2.1 Understanding the Problem

The aim of the NMS is to help NGO register students and track their performance. NGO helps them with items.

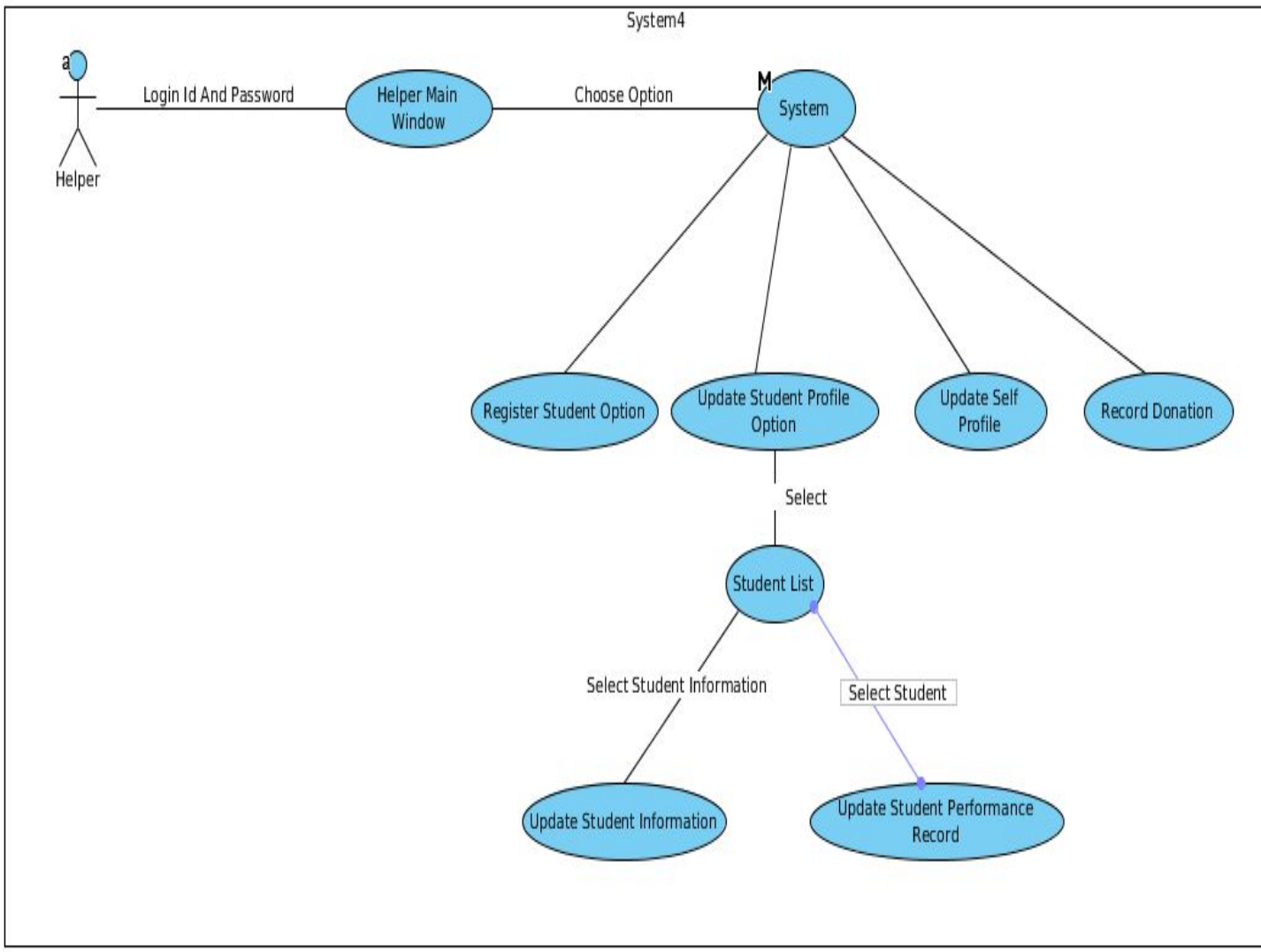
Donors with small amount of money or various items can contact helpers to make a donation.

Pledged donors register and pledge to help NGO with a particular amount of money (semi)annually.

Manager handles contact with pledged donors. He/She contacts them for financial help. Manager can also register and remove helpers. He/She can check all records. including expenses made by the organization.

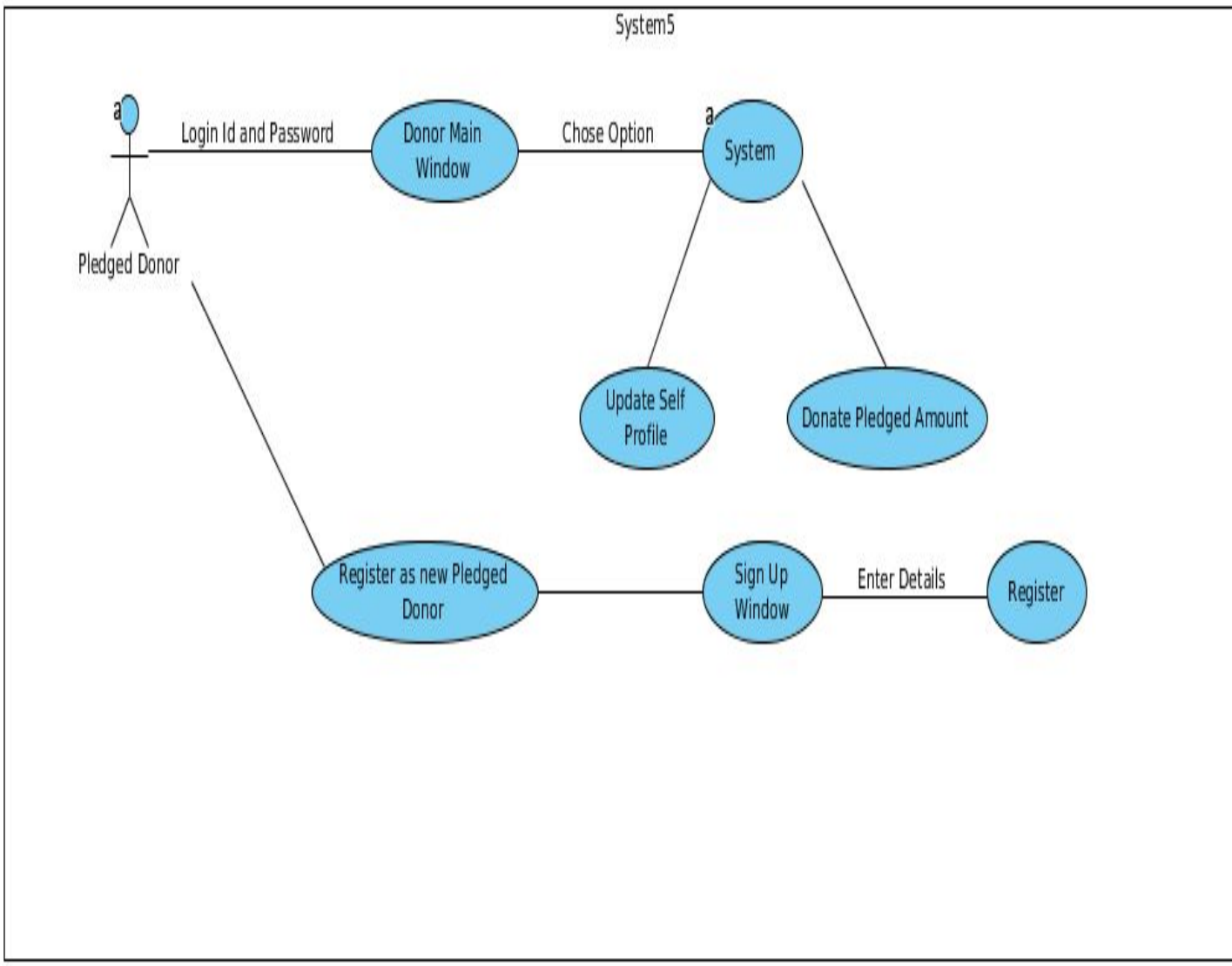
2.2 Analyzing Stakeholders

Helper :

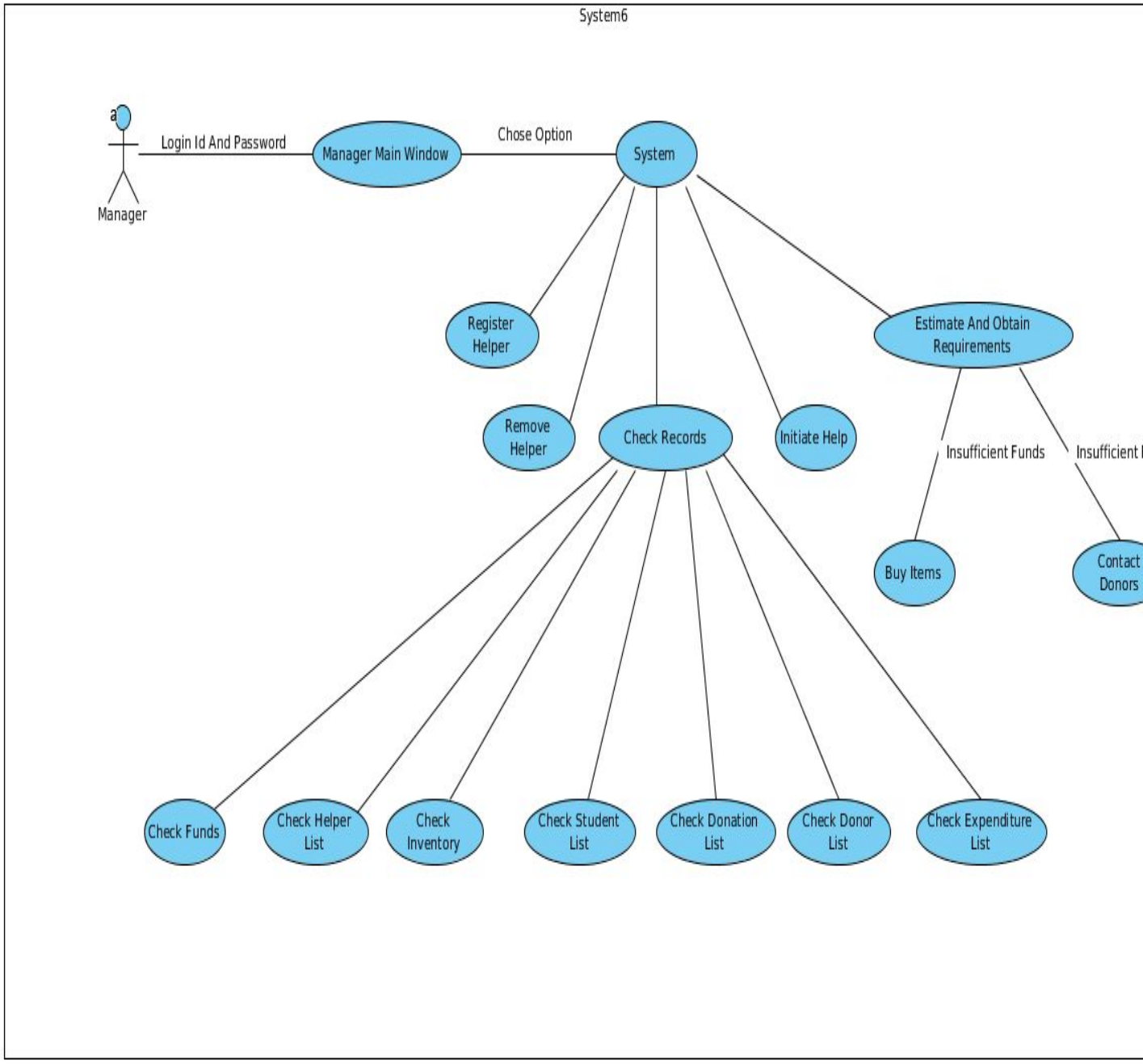


➔ Enter login ID and password provided by manager.
➔ Select what to do :
    Register student
    Edit student info
    Edit profile (own)

Pledged Donor :



- ➔ Register (new) / Login(existing). Main window will open.
- ➔ Donor can pledge money and change frequency of donation.
- ➔ Donor can also edit his details



Manager :

- ➔ Login with id and password provided by our firm.
- ➔ Manager can access all records which exist(students, donors, helpers)
- ➔ Manager can contact donors for funds.
- ➔ Manager can see an account of financial expenses.
- ➔ Manager can exclusively add/remove helpers.

2.3 Defining Alternatives

2.3.1 Connection between students and helpers

Students contact helpers for registration at NGO. Alternatively, helpers might visit them.

2.3.2 Hardware Infrastructure

The software can be designed to run on MacOS.
Instead of using internal memory we can use external hard disk.
Software can be designed for 32 bit computer instead of 64 bit.

2.3.3 Software Infrastructure

Instead of password login we an implement finger print scan recognition.
We can user other languages like C++ with QT for designing GUI.

2.4 Defining Criteria to Evaluate

- Cost of technology .
- Cost of infrastructure .
- Lifetime of technology .
- Stability of technology.

2.5 Assessment of Unusual Circumstances

Data should not be lost in any case(hardware or software failure). We can design a MASTER system for storing backup of all data so we can update and restored data.

2.6 Evaluation of Alternatives

2.6.1 Connection between students and volunteers

This alternative will not affect the software.

2.6.2 Hardware Infrastructure

Using an external hard disk to save the database will indirectly imply a backup of the database along with the master system which can be retrieve the data at any point of time desired. Thus, this is better than

using the system hard disk to store the data.

The NMS can be designed for a 32-bit system as well as a 64-bit system. Designing the NMS for a 64-bit system should be preferred for the graphical interface . 64-bit is preferred for GUI.

2.7.4 Software Infrastructure

Finger print scan is irrelevant as tight security is not required and would unnecessarily add to the cost. Any language works for implementation and JAVA has better compatibility.

2.8 Report

The objectives of the NMS have been laid out and the various scopes have been discussed in detail. Firstly we understood the complete problem and found the various functions that the software performs such as registering students, updating their performance, registering donors, accepting donations, estimation of funds, allocation of funds and maintaining expenditure. Then, various cases were analyzed with the help of use case diagrams along with the brief description followed by a step-by- step description of each use case. Each of the use cases were supported by the

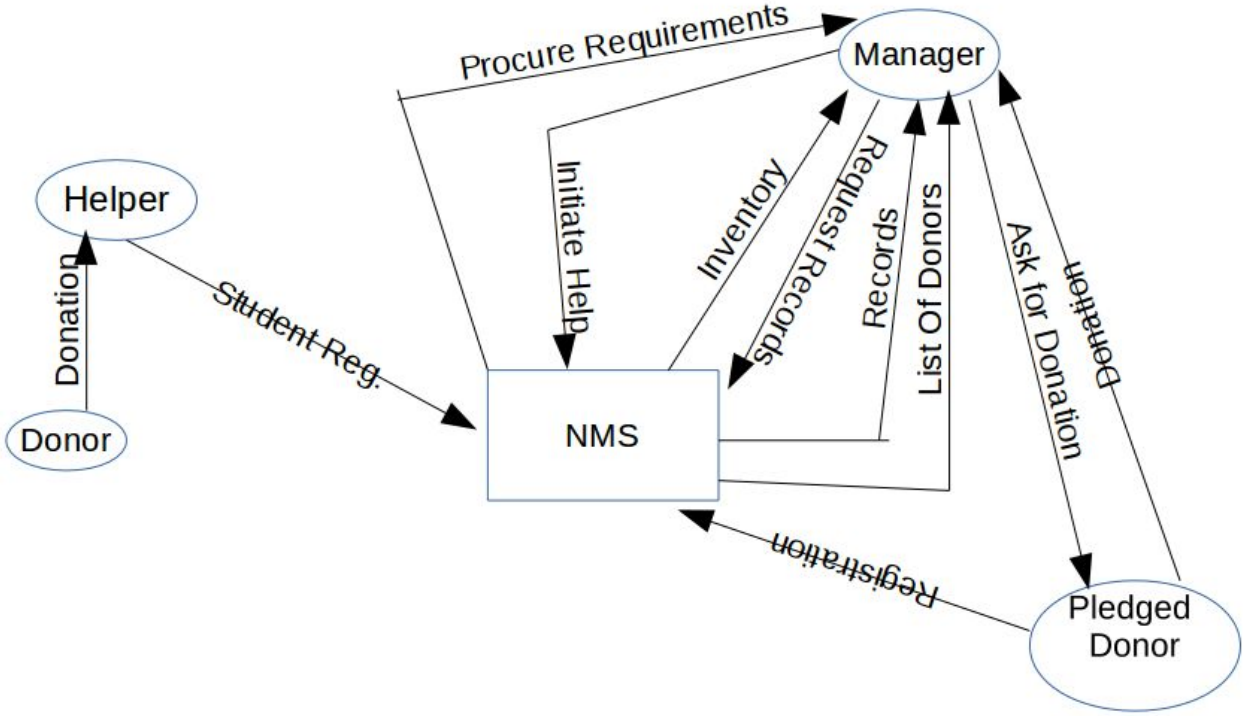
use case diagrams for ease of understanding. Then, the various alternatives were developed keeping in mind the cost and the lifetime of the components the alternative brings with it and hence the advantages and disadvantages were highlighted. These alternatives included the hardware, software, technology, security and many other aspects which form an integral part of the software and which could be incorporated in the NMS, if desired.

The primary criteria for evaluation were expected lifetime,cost, stability, and instability of the technology. The unusual circumstances like loss of data due to hardware or software failure or hacking were taken care of by certain concepts of data backup, cryptography etc.
At last, all the alternatives proposed earlier were analyzed in depth and their advantages and boon to the NMS were clearly mentioned. A very vivid comparison was made between the NMS development without the alternatives and the BAS as it would function with the alternatives if incorporated in the software.

3.0 Requirements

Functional :

3.1 Data Flow : Context Diagram



Inputs :

1. Student details
2. Donated item
3. Donor details
4. Request records
5. Initiate Help

Process : NMS

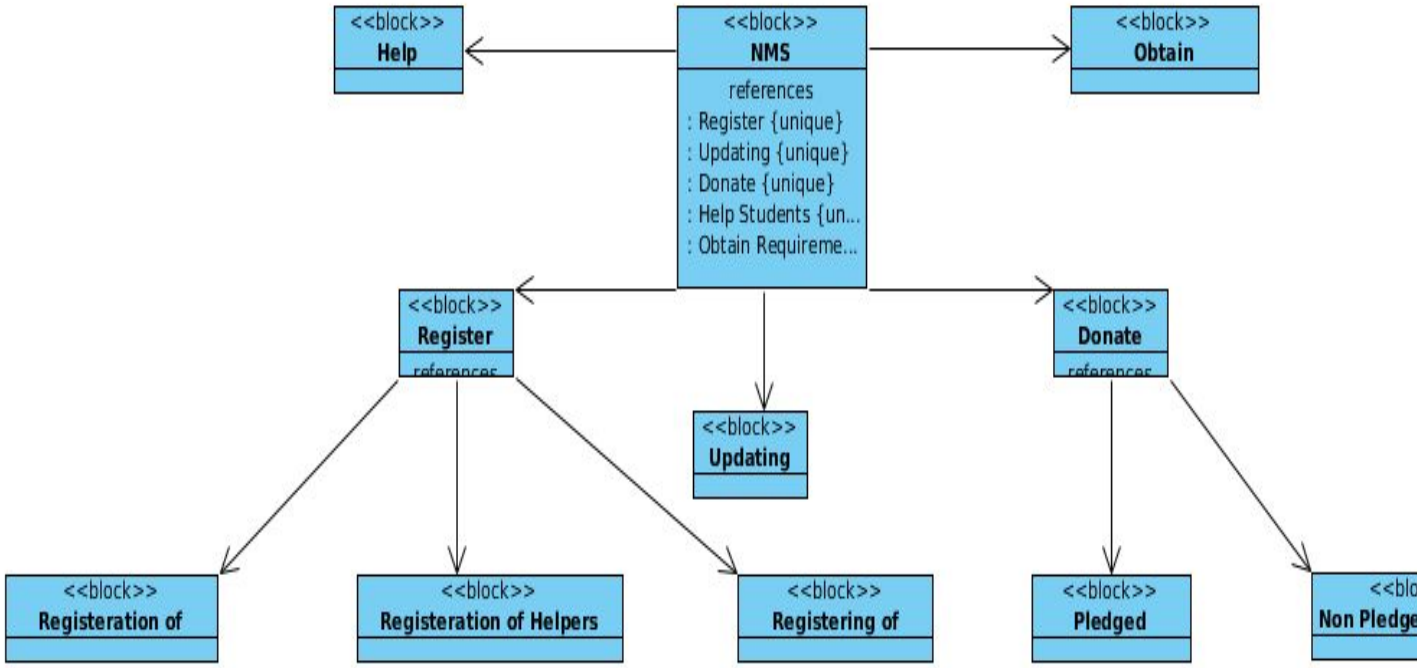
Output :

1. List of donors to be contacted
2. Inventory
3. Procure Requirements
4. Records

3.2 Data Dictionary

Record : ArrayList<Record>
Inventory : Inventory
Funds : Integer
Student List : ArrayList<Student>
Volunteer List : ArrayList<Volunteer>
Donor List : ArrayList<PledgedDonor>
Donation List : ArrayList<Donation>
Expenditure list:ArrayList<Expenditure>

3.3 Structure Chart



3.4 Non-Functional Requirements

Database Requirements: The database requires space which increases with number of students/donors/helpers/expenses.

Legal Requirements: NMS has a software license agreement according to which any user making illegal copies will be severely punished.

Physical security: A person can close the window when leaving the system which requires password to login again. So an

intruder cannot use the software behind his/her back.

3.5 Report

In the requirements analysis section, we started with the functional requirements of the NMS and explained detail it using data flow diagram, Structure chart and data dictionary. The data flow diagram graphically represent the "flow" of data through the information system, modeling the process aspects of the NMS. They are a preliminary step to create an overview of the NMS. Thus the DFD along with the structure chart and the data dictionary have been used for the visualization of data processing in the NMS, i.e. the structural design of the software being developed.

The second part of the requirements analysis deals with the non- functional requirements of the NMS. These include

the database requirements, the legal requirements, and the availability of the NMS over a day and the physical security of the software being developed. The various non-functional requirements ensure the delivery of an operable and manageable system which provides the required functionality in a reliable fashion, uninterrupted and with minimal time of interruption even under the unusual circumstances

4.0. Detailed design

4.1. Global System Architecture

A 2-tier architecture is implemented which includes client at one end and the database at the other.

4.2. Platform

Minimum Systems Requirements:
Hardware Requirements : Operating System Windows XP/98 or later versions , Linux
Processor Pentium II processor or equivalent .

Hard Disk Space: RAM 512 MB
Software Requirements :
JDK 7 or above

4.3 Software Architecture

Object-oriented architecture forms the basis of the RRTS.
In this style data representations and their associated primitive operations are encapsulated in an abstract data type or object. The components of this style are the objects—or instances of the abstract data types. Objects interact through function and procedure invocations. Two important aspects of this style are that an object is responsible for preserving the integrity of its representation and that the representation is hidden from other objects.

4.4 Report

Under the detailed design section of the software design,
the global system architecture was discussed. The NMS has a 2-tier architecture comprising of the client and the database. Then the platform requirements for the NMS was discussed in terms of

the operating system, the processor required, the minimum and recommended hard disk space and RAM requirements, etc. The software architecture of the NMS was later stated to be of the object-oriented type using JAVA as the core technology. The important aspects of OOD used for the NMS are data abstraction and the preservation of integrity of the software.