An Efficient Approach on Object Oriented Design using Genetic Algorithm

Seema Sharma
Asstt Prof
MRIU, Faridabad

Rashmi Aggarwal
Asstt Prof
MRIU, Faridabad

Anupriya Jain
Asstt Prof
MRIU, Faridabad

Sachin Sharma
Asstt Prof
MRIU, Faridabad

ABSTRACT
Object oriented design is a method where developers think in terms of objects instead of procedures or functions. The overall system is designed keeping in view the interaction among objects that maintain the states and provide implementation on them. For implementing object oriented design three methodologies are adopted Object Modeling technique (OMT), Structure Analysis/Structured Design (SA/SD), Jackson Structured Design (JSD). SA/SD approach is based on the data flow and JSD approach is action-oriented. Since SA/SD is easy to understand but it focuses on well defined system boundary whereas JSD approach is too complex and does not have any graphical representation. In this paper, we will apply the concept of Genetic Algorithm to minimize the above said problem.

Genetic Algorithm method is based on natural evolution. Through this algorithm, we can improve the performance of the system. Genetics Algorithm start with a fixed size of data structure which are used to perform such given task. There are basically three processes of Genetic Algorithm. 1. Crossover 2. Mutation 3. Inversion

This paper presents a method to improve efficiency of Object Oriented Design. Through a method of Crossover in Genetic Algorithm, it will optimize and reduce the complexity.

Keywords: SASD, JSD, Objects, Crossover, Genetics

1. INTRODUCTION
OOD [3] has clearly become buzzword of the choice in the industry. Almost everyone claims to be doing it and every one says it is better than traditional Functional Oriented Design[6]. Object Design deals with detailed design of objects and interaction. It follows the agreed guidelines and protocols defined during System Design. Object Oriented mainly concerned with classification of attributes, their operation and association between objects. Objects may be distributed and execute either sequentially or in parallel. Object Oriented Design follows 3 approaches SASD, OMT and JSD [1]. In this paper we will concentrate on 2 approaches SA/SD and JSD.

SA/SD is diagrammatic notation which is design to help people understand the system. It is pervasive applicable to many problems and well documented. The basic goal of SA/SD is to improve quality and reduce the risk of System failure. It establishes concrete management specification and documentation. It focuses on reliability, flexibility and maintainability of system. In this system it involves 2 phases.

1) Analysis Phase: It uses DFD, Data Dictionary, State Transition diagram and ER diagram.

- Data Flow Diagram: In a DFD model describe how the data flows through the system.
- Data Dictionary: The content that is missing from DFD is described in the data dictionary. Data Dictionary defines the Data store and there relevant meaning.
- State Transition Diagrams is similar to Dynamic model .It specifies how much time function will take to execute and data access triggered by events.
- ER Diagram: It specifies the relationship between data store.

2) Design Phase: Structure Chart, Pseudo Code

- Structure Chart: It is created by the DFD. Structure Chart specifies how DFD's processes are grouped in to task and allocate to CPU's.
- Pseudo Code: Actual implementation of the System.

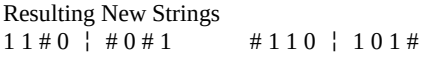
Although SA/SD have several advantage but it is still suffers from some drawbacks. One major disadvantage is that it supports well defined system boundaries. Therefore many changes can not be done once the design process is over.

1.1 Genetic Algorithm
It produces more effective problem solutions. Genetic Algorithm[2] is most widely used in applied problem solving as well as in Scientific Models. When Genetic Algorithm is used in solving a problem it follows three stages. There are a number of Generic operators that produce offspring having feature of the parents. 1. Crossover 2. Mutation 3. Inversion

The most common of these is Crossover[4]. It takes two candidate solution and divides them, swapping component to produce two new components. Mutation takes a single candidate and randomly changes some aspect of it that is it can be applied only with the same data structure. In this paper, we will use the Crossover function of Genetic Algorithm[5]. Let us consider an example on bit stream pattern of length A. This operator splits them in the middle and forms two children whose initial segment comes from one parent and whose tail comes from another. It is to be noted that splitting the candidate solution in the middle is arbitrary. This split at any point in the representation and may be randomly adjusted during the solution.

Input Bit String

Resulting New Strings



Inversion transformation is applied to a single Data Structure. In this operation 1 bit is selected randomly and at that position inversion is applied .this operation concatenates the tail of the string to the head of the string.

1.2 Jackson Structured Development (JSD)

It is another approach for object oriented design in which the time factor is the vital factor and is described by using sequence of events. JSD is applicable to the application where efficiency plays a vital role. JSD was introduced by Michael Jackson in 1983. The fundamental principle of JSD[6] is that it focuses on describing the real world by the system i.e. its main focus is to map the progress in the real world rather than specifying the functions performed by the system i.e its main focus is to map the progress in the real world. JSD consist of three stages:

1. Modeling Stage – This Stage comprises of Entity Action and Entity Structure.
2. Network Stage – Initial model step, Function Step, and System Timing Step.
3. Implementation Stage – Implementation Step.

In modeling stage the entity structure diagram has been created and identifies the entities and their actions. In a network phase a set of sequential is expanded into a process network by addition of process for handling messages for external environment and generating system output.

In Entity Action the entities perform actions in time. The action is atomic and cannot be decomposed further.

Entity Structure Step partially sequences the order of action with respect to time. In initial model step we concentrate on state vector and data stream communication. A data stream is an unbounded queue where read operator extracts the next record from the queue, if there are no records when the process executing the read operation waits until the record is available whereas in write operation data stream places the next record in the queue. In state vector we are present all the possible states of an entity. Function step consist of pseudo code to the output of action generated from states. In this step the developer requires a complete specification of the system.

System Timing Step relates with the performance constraints of the system. The implementation step focuses on process scheduling and allocates the number of processors required. The process may be different form the processors. So it is the job of the developer who decides how to get several processes to share the same processors.

Since JSD is useful for concurrent software, real time software and parallel programming computers but when we talk about high level analysis JSD is ill suited. JSD approach is difficult to understand because it is less graphical oriented as compared with SA/SD.

Since SA/SD and JSD both suffers from some limitations we will try to formulate a new approach using cross over operation of genetic algorithm which can minimize the limitation to a certain extent.

2. RESULTS AND DISCUSSIONS

2.1 Problem Statement

In order to overcome the limitations of SA/SD and JSD by using cross over operation (GA) we will consider a case study on library management system. The system should be stand alone and it should be designed to provide the following functionality. The processes that can be used are

1. Issue of Books
2. Return of books
3. Join the membership.
4. Leave the membership.
5. Acquire the books.
6. Sell the books.

JSD approach for designing Library Management System : To design library management system by JSD following stages should be taken

2.2 Modeling Stage – Entity Action and Entity Structure.

The processes used in entity action are

ACQUIRE	The library acquires a book. (book-id, date, ISBN)
JOIN	A new member joins the library. (member-id, name, address, date)
LEAVE	A member leaves the library. (member-id, date)
BORROW	A member borrows a book from the library. (book-id, date, member-id)
RETURN	The borrowing member returns a book to the library. (book-id, date, member-id)
SELL	The library sells one of its books. (book-id, date, price)

Table 1.1 Entity Action

2.2.1 Entity Structure Step

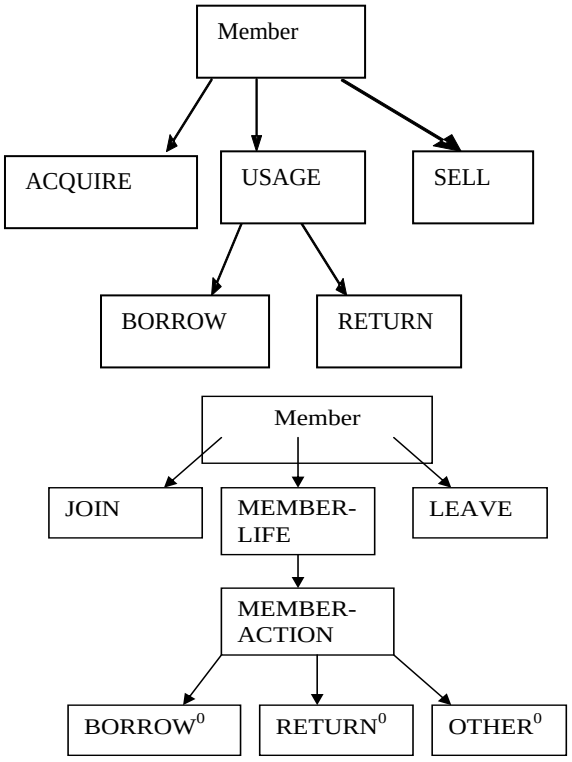


Fig:1.1 Structure chart of Entity

The * indicates the multiplicity in an operation and circle '0' indicates that they are the part of selection. Each Member action can borrow, return and other.

2.3 Network phase: In this phase, we connect model processes and functions.

2.3.1 Initial Model Steps:

2.3.1.1 Data Stream: The data stream diagram is shown below

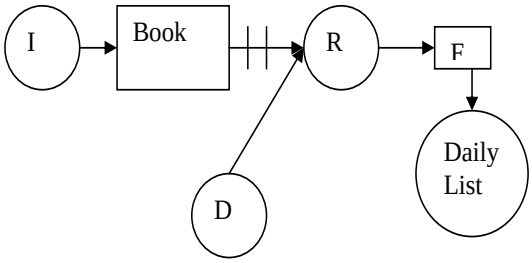


Fig 1.2: Data Stream

The library may require daily list of return of books. For each return event, each book process writes a record on output stream R. These streams are merged with the message D to generate another day's list. Then the function process F reads this merged stream and whenever it encounters a D message daily list is produced.

2.3.1.2 State Vector Connection: In this system a state vector function is used to query the availability of a particular book. It checks the state vector of book processes and output the nature of its presence in the state vector at recent time.

2.3.2 Function Step: It specifies the pseudo code to state output of action. In this the following output may be produced.

ACQUIRE
JOIN
LEAVE
BORROW
RETURN
SELL

2.3.3 System Timing Step: In the library management system performance constraints are noted through which the model is permitted to lag the real world. The various constraints are:

- Number of books issued to a member.
- Books out of stock or already issued cannot be issued.

2.4 Implementation Step: Based on hardware and timing constraint we design a system implementation diagram as shown below.

Diagram

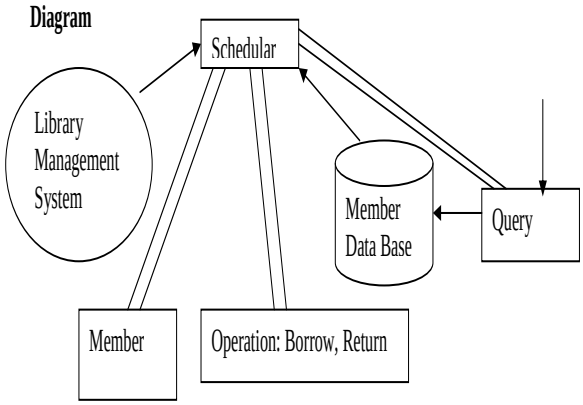


Fig 1.3: Implementation Procedure

The above approach suffers from major limitations.

- Difficult to comprehend.
- Less Graphical.
- High level analysis.
- Complex process scheduling.

Since JSD approach is difficult, complex and less graphical so we switch over to another approach of Object Oriented Methodology i.e SA/SD.

2.5 SA/SD approach of Library Management System: The processes that can be used are

- **Issue of Books**
- **Return of books**
- **Join the member**
- **Leave the member**
- **Acquire the books**
- **Sell the books**

2.6 Analysis Phase:

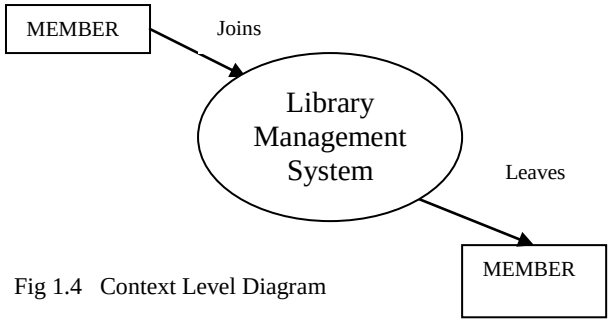


Fig 1.4 Context Level Diagram

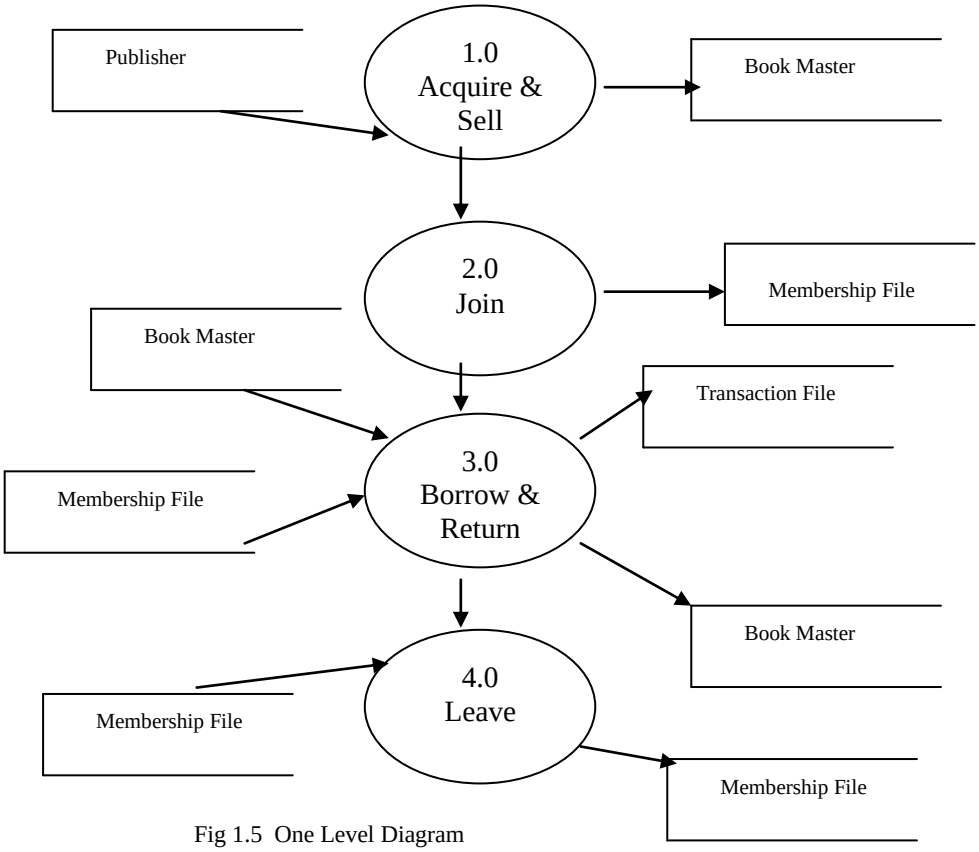


Fig 1.5 One Level Diagram

2.6.1 Data Dictionary

Attribute	Table	NULL	P Key	Domain
Book-Id	Book Master	N	Y	Alpha Numeric
Book Name	Book Master	N	N	Character String
No of Books	Book Master	N	N	Numeric
Member-Id	Membership	N	Y	Numeric
Member Name	Membership	N	N	Character String

Table1.2

2.6.2 ER Diagram

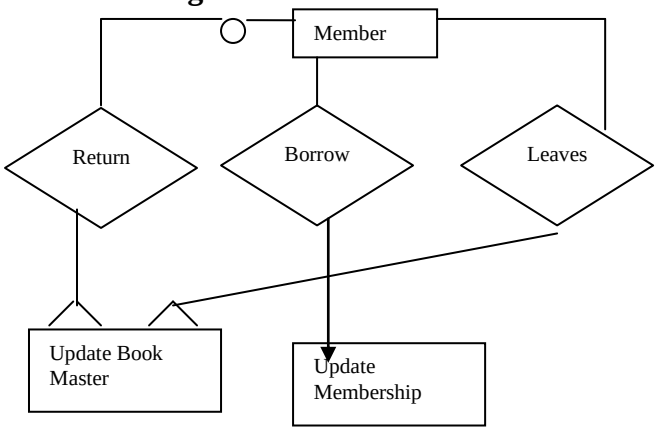


Fig 1.6 ER Diagram

2.6.3 State Transition Diagram

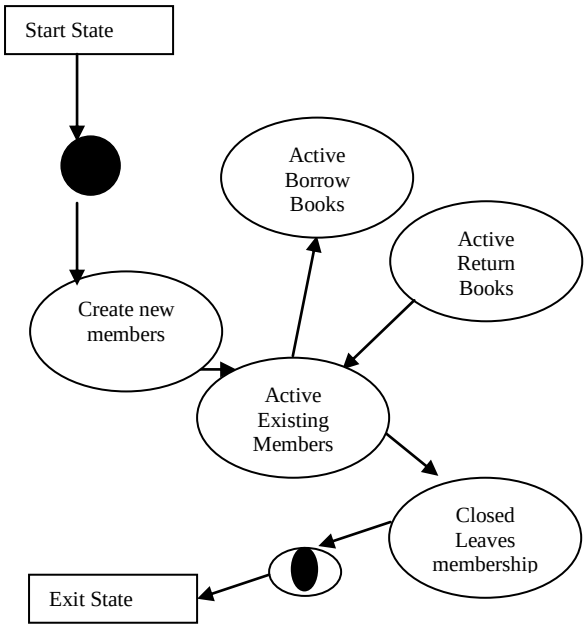


Fig 1.7 : State Transition Diagram

2.7 Design Phase

2.7.1 Structure Chart

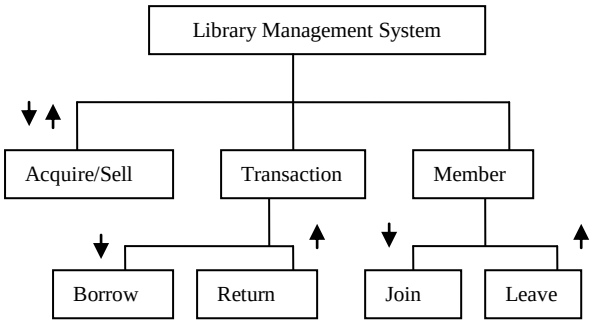


Fig 1.8 : Structure Chart

Although this approach overcomes the limitations of JSD approach but it has a well defined system boundary problem which means if we want to add a new process CLASSIFY in our library management system we have to start the process from scratch. So to overcome this problem we apply genetic algorithm's cross over approach.

2.8 Cross over Approach over JSD with SA/SD: In cross over operation two strings are used for mating which selects a bit position and concatenates the head of one parent with the tail of second parent to produce offspring. Using this concept we will apply crossover on JSD with SA/SD. The steps of JSD are

1. Entity Action
2. Entity Structure
3. Initial Model
4. Function Step
5. System Timing Step
6. Implementation Step.

The steps of SA/SD are

1. DFD
2. Data Dictionary
3. State Transition Diagram
4. Entity Relationship Diagram.
5. Structure Chart
6. Pseudo code.

Let us suppose cross over is applied on JSD at position 3 Therefore the formulated approach will have the following steps:

1. Entity Action
2. Entity Structure
3. State Transition Diagram
4. ER Diagram
5. Structure Chart
6. Pseudo code

Let us assume that we want to add a new entity CLASSIFY (used for classification of books) it can be easily modified through the above formulated approach which can be shown as below-

2.8.1 Entity Action

ACQUIRE	The library acquires a book. (book-id, date, ISBN)
CLASSIFY	Stock Updation
JOIN	A new member joins the library. (member-id, name, address, date)
LEAVE	A member leaves the library. (member-id, date)
BORROW	A member borrows a book from the library. (book-id, date, member-id)
RETURN	The borrowing member returns a book to the library. (book-id, date, member-id)
SELL	The library sells one of its books. (book-id, date.

Table 1.3

2.8.2 Entity Structure

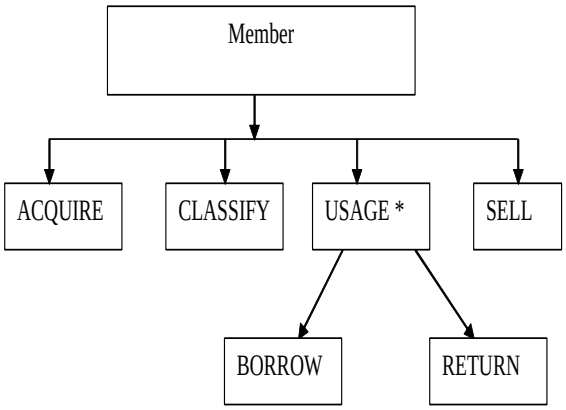


Fig 1.9 Structure

2.8.3. State Transition Diagram

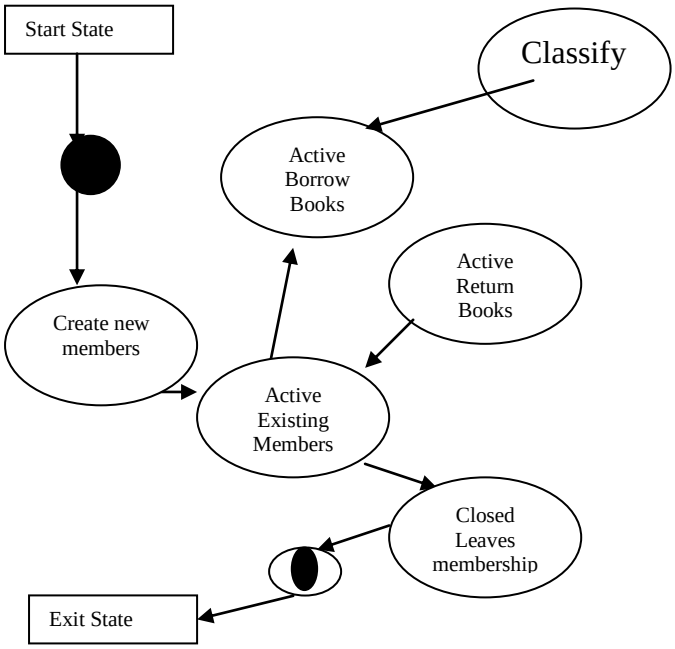


Fig 1.10 : State Transition Diagram

2.8.4 ER Diagram

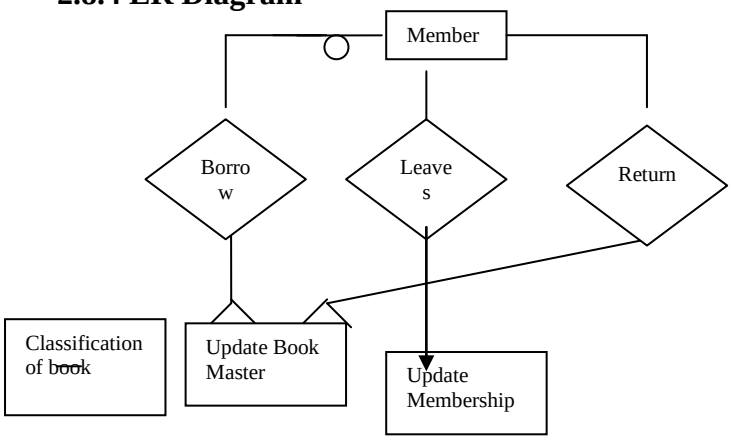


Fig 1.11: Entity Relationship Diagram

2.8.5 Structure Chart

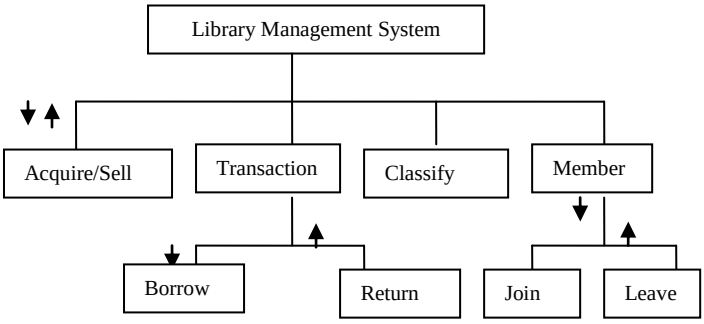


Fig 1.12 : Structure Chart

3. CONCLUSION

Although the basic object oriented methodology approach, i.e. SA/SD and JSD are complete in their own aspects but they suffer from certain limitations with the application of genetic algorithm we have tried to formulate a new approach which can remove the major limitation of the above said approaches. The limitations that are focused and removed are well defined system boundary (SA/SD), complexity (JSD) and less graphical and difficult to comprehend (JSD)

4. REFERENCES

[1] Rumbaugh James,Blaha Michael,Premerlani Williamm,Eddy Frederick, "Object Oriented Modeling and Design" Pearson Printing Hall.

[2] Luga George F,"Artificial Intelligence" V Edition Pearson Education

[3] Pressman Roger S "Software Engineering" V Edition McGraw Hill Edition

[4] *Artificial Intelligence* (2nd ed.) *Rich, Knigh*Introduction

[5] Artificial Intelligence And Expert Systems, Patterson by Dan W. Patterson. McGraw-Hill. 1991

[6] *Object-Oriented Analysis and Design* with Applications (2nd Edition) by Grady *Booch*.

[7] Object-Oriented Analysis and Design By: Brett McLaughlin, Gary Pollice, David West