# Roundtrip Engineering using Unified Modeling Language with Rational Rose and JAVA

Dr.GSVP Raju
CS&ST Dept,Andhra Univ.
Vizag,Andhra Pradesh

K Koteswara Rao
CSE Dept.GMRIT
Rajam,Anhra Pradesh

M Sumender Roy
CSE Dept,GIET
Rajahmundry, Andhra Pradesh

## ABSTRACT
Sometimes documentation only available for the post delivery maintenance is the source code itself. This happens all too frequently when maintaining legacy systems, i.e. software in current use but developed not earlier than 15 or 20 years .Under these circumstances, maintaining the code can be extremely difficult. One way of handling this problem is to start with source code and attempt to recreate the design documents or even the specifications. This process is called Reverse Engineering. CASE tools can assist with this process. One of the simplest is a pretty printer, which may help display the code more clearly. Other tools construct diagrams such as flow charts or UML diagrams, directly from the source code, these visual aids can help in the process of design recovery.

Once the maintenance team has reconstructed the design, there are two possibilities, one alternative is to attempt to reconstruct the specifications, modify the reconstructed specifications to reflect the necessary changes, and reimplement the product the usual way, and with in the context of reverse engineering, the usual development process that proceeds from analysis through design to implementation is called forward engineering. The process of forward engineering and reverse engineering combinely called as roundtrip engineering. This paper explains about how it can be achieved using UML with Rational Rose and JAVA

**Keywords:** UML, Rational Rose, roundtrip engineering, source code, documentation

## 1. UNIFIED MODELING LANGUAGE
UML stands for Unified Modeling Language. UML is a language for visualizing, documenting the artifacts of software intensive system. James Rumbaugh defined Model as a simplification of reality Modeling is which captures essential parts of the system.

Visual modeling is a modeling with standard graphical notation

We build models so that we can better understand the system we are developing

- Models help us to visualize a system as it is or as we want to be
- Models permit us to specify the structure of system
- Models give us a template that guides us in constructing system
- Models can document the decisions we have made
- Visual modeling captures business process
- Visual modeling is a communication tool
- Visual modeling manages the complexity
- Visual modeling promotes reuse

In the late 1980s and early 1990s there were 3 methodologies:

**Booch Methodology**: This was designed by Grady Booch which is great in design

**OMT (Object Modeling Technique) Methodology**: This was designed by James Rumbaugh et' al which is great in analysis

**OOSE (Objectory) Methodology**: This was designed by Ivar Jacobson which is heart of UML i.e., use case

In 1994, James Rumbaugh joined in Rational with Booch and worked together and this is the beginning of unification method. In 1995, Jacobson joined in Rational with Booch and Jim. In 1996, matured unified method was released. In 1997, in January UML 1.0 was released. In 1997 on November 14th UML was accepted by OMG and accepted as a standard language. UML can be used in

1. Banking and Financial services 2.Telecommunications
3. Transportation 4 .Defense
5. Retail 6 Modeling Electronics
7. Scientific 8.Distributed Web services

### 1.1UML Diagrams
Diagram is a graphical representation of elements. UML diagrams can be classified into two types

1. Structural Diagrams 2. Behavioral Diagrams

**Structural Diagrams**:

These can be divided into 4 types:

i. Class Diagram iii.Component Diagram
ii. Object Diagram iv.Deployment Diagram

**Behavioral Diagrams**:

These can be classified into 5 types:

i. Use case diagrams iii. State Chart Diagram
ii. Activity Diagram iv. Sequence diagram v.Collaboration diagram

#### 1.1.1BEHAVIORAL DIAGRAMS
i. **Use case Diagram**:

Use case diagram is created to visualize the interaction of our system with the outside world. The components of use case diagram are:

**Use Case**: Scenarios of the system

**Actor**: Someone or something who is interacting with the system

**Relationship**: Semantic link between use case and actor. The forms of relationship are:

a. Association b.Dependency c.Generalization

**ii. Activity Diagram**

Activity diagram shows the flow of events within our system. The components are:

a) Start State d) Decision Box
b) Synchronization Bar e) End State
c) Transition f) Swim Lane

**iii. Interaction Diagram**

An interaction diagram models the dynamic aspects of the system by showing the relationship among the objects and messages they may dispatch. There are two types of interaction diagrams:

**1. Sequence Diagram**

Sequence diagram shows the step to step what mush happen to accomplish a piece of functionality provided by the system. The components are:

a) Actor d) Lifeline
b) Focus of Control e) Messages f)Object

**2. Collaboration Diagram**

Collaboration diagram displays object interactions organized around objects and their links to one another. The components are:

a) Actor b)Object c)Link

**iv. State chart Diagram**

State chart diagram show a life cycle of a single class. The state is a condition where the object may be in. The components are:

a) Start state c)State
b) Transition d)End state

**1.1.2STRUCTURAL DIAGRAMS**

**i. Class Diagram**

Class diagram shows structure of the software system. The class diagram shows a set of classes, interfaces and their relationships. The components are:

a) Class
b) Relationship:
The forms of relationship are:
1. Association 3.Composition
2. Dependency 4.Generalization
5.Aggregation

**ii. Component Diagram**

Component is a smallest individual physical replaceable part of the system. Component diagram shows the organization and dependencies among software components. The components present are:

a) Component
   a. Runtime component(.dll)
   b. Software components(.h)
   c. Executable components(.exe)
b) Dependency
c) Interface

**iii. Object Diagram**

Object diagram shows objects and links among objects. The components are:

a) Object b)Link

The object diagram cannot be model in rational rose.

**iv. Deployment Diagram**

Deployment diagram visualizes distribution of components across the enterprise

# 2. INTRODUCTION TO RATIONAL ROSE

Rational Rose is a software where the UML can be model. Here, Rational is the name of the software, ROSE stands for Rational Object Software Engineering.

**To draw the UML Diagram in Rational Rose**:

**Step 1**: Start Rational software, in that Rational Rose Enterprise Edition. After that Rational Rose Enterprise Edition will be activated. The Rational Rose window contains 5 parts.

1. **View Table**
   It contains:
   a. Use case view c. Component view
   b. Logical view d. Deployment view
2. **Diagram Tool Bar**
   This can contain the tools of the corresponding diagram in which we are going to draw
3. **Diagram Window**
   In this window we can draw the diagram
4. **Message Window**
   It contains the message of documentation of the corresponding diagram
5. **Log Window**
   This is the place where the errors can be displayed when we are drawing the diagram

**2.1 USECASE VIEW**

In this view we can draw two diagrams:

1. Use case diagram 2.Activity Diagram

**Steps to draw diagram**:

1. Select use case view and then right click on use case view
2. Select New, in that select use case/activity diagram
3. Name the diagram
4. After double clicking on the diagram name, the corresponding use case/activity will be opened
5. We can draw the diagram by drag and drop the components of the corresponding diagram

**2.2LOGICAL VIEW**

In this view we can model:

a. Class diagram c.Collaboration diagram
b. Sequence diagram d.State Chart diagram

**To draw the diagram:**

1. Select logical view and then right click on the logical view
2. Select new in that select class/ state chart/ sequence diagram
3. Name the diagram
4. After double clicking on the diagram name, the corresponding diagram will be opened
5. We can draw the diagram by drag and drop the components of the corresponding diagram

**2.3 COMPONENT VIEW**

In this view we can model Component Diagram

**To draw the diagram**:

1. Select component view and then click on the component view
2. Select New, in that select component diagram
3. Name the diagram

4. After double clicking on the diagram, the corresponding diagram will be opened
5. We can draw diagram by drag and drop the components of corresponding diagrams

**2.4 DEPLOYMENT VIEW**

In this we can model deployment diagram

**To draw diagram**:

1. Select deployment view, then right click on deployment view
2. Select New, in that select deployment diagram
3. Name the diagram
4. After double clicking on the diagram name, the corresponding diagram will be opened
5. We can draw diagram by drag and drop the components of corresponding diagrams

## 3. UNIFIED LIBRARY APPLICATION (ULAS) INTRODUCTION

Unified Library Application System emphasizes on the online reservation, issue and return of books. This system globalizes the present library system. Using this application the member can reserve any book from anywhere in the world. Still in nascent stages, this application soon revolutionizes present library system.

Let us just have an overview of the unified library application system:

- Librarian lends books and magazines and maintains list of members.
- Librarian maintains the list of all the members of library
- Borrower makes reservation online
- Borrower can remove reservation online
- Librarian issues books to the borrower and calculate bills.
- Borrower issues/returns books and/or magazines
- Librarian places order about the requirements to the master librarian
- Librarian updates system
- Master librarian maintains librarians

**3.1TEXTUAL ANALYSIS**

(a) ACTORS

i. Librarian
ii. Master Librarian
iii.Catalog
iv.Borrower

(b) VERBS

i. Borrower:
   1. Logs into the system
   2. Browses/searches for books or magazines
   3. Makes/removes reservation
   4. Views results and reports from the unified library application system

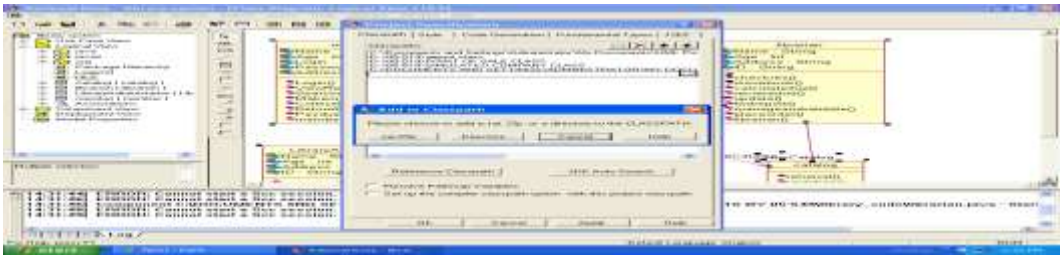
ii. Librarian:
   1. Manages and validates members
   2. View reports from the system
   3. Issues books
   4. Calculates dues
   5. Takes books
   6. Places orders to the master librarian
   7. Maintains list of books and magazine

iii. Master Librarian
   1. Maintains other librarians

**3.2FORWARD ENGINEERING:** steps to do the forward engineering by considering ULAS as Case Study

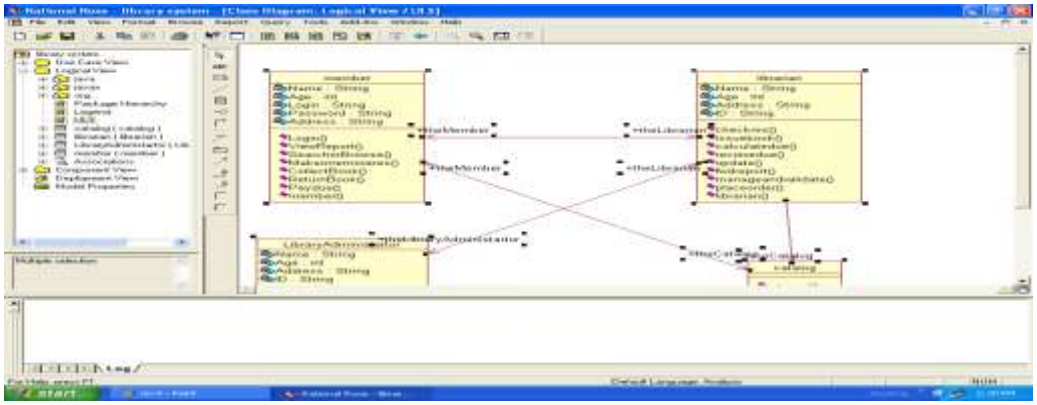
a. Step1 – Project Specification

b.Step2 – Set Path

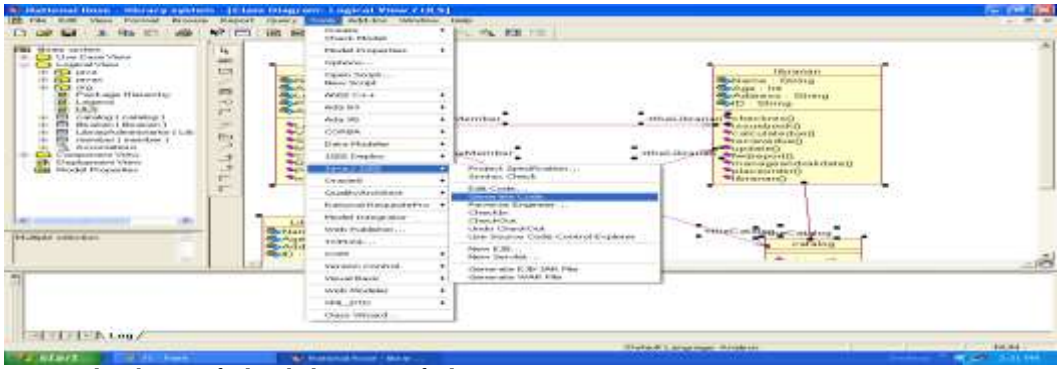


Step3:-.Browse the path

Step4 – Select the classes to be forward engineered



Step5:-Select Tools -> Java/J2EE -> Generate Code to forward engineer.



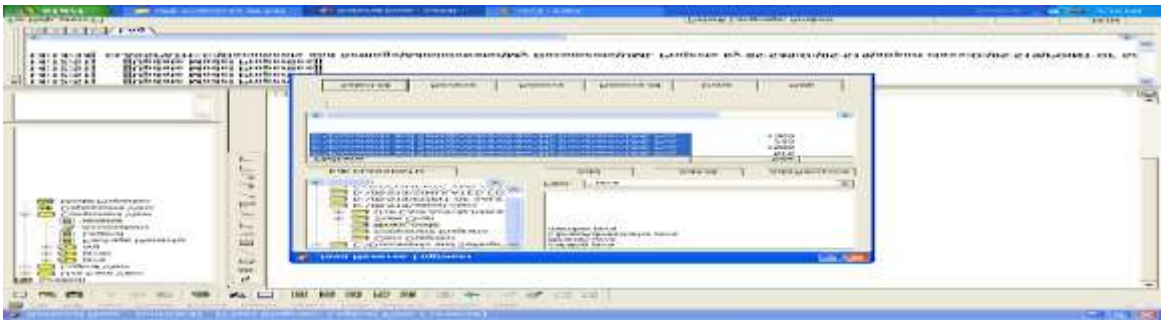
The code is generated at the specified path that is specified under Project Specification part as:

**Administrator**:

```
public class administrator {
   private String name;
   private String ID;
   public Librarian theLibrarian;
   public administrator() {   }
   public String receive_order() {
    return null;}
   public void manage_librarians() {}
   public void purchase_new_stock() {}}
```

**3.3REVERSE ENGINEERING:** steps in reverse engineering

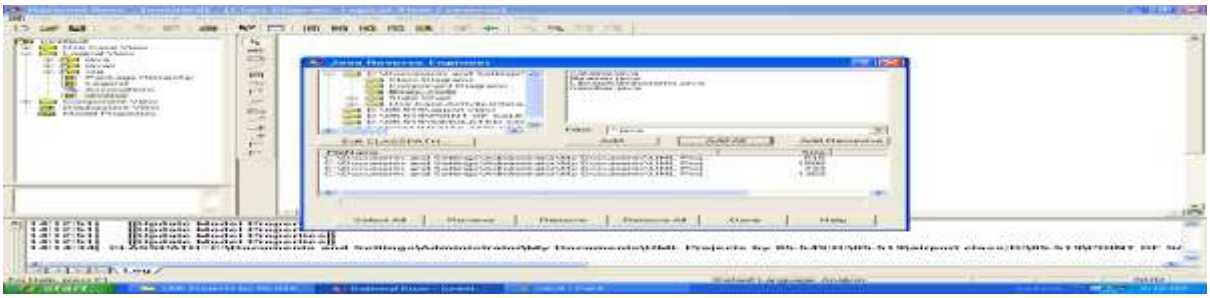
Step 1 :- Open new project -> Class Diagram under Logical View.

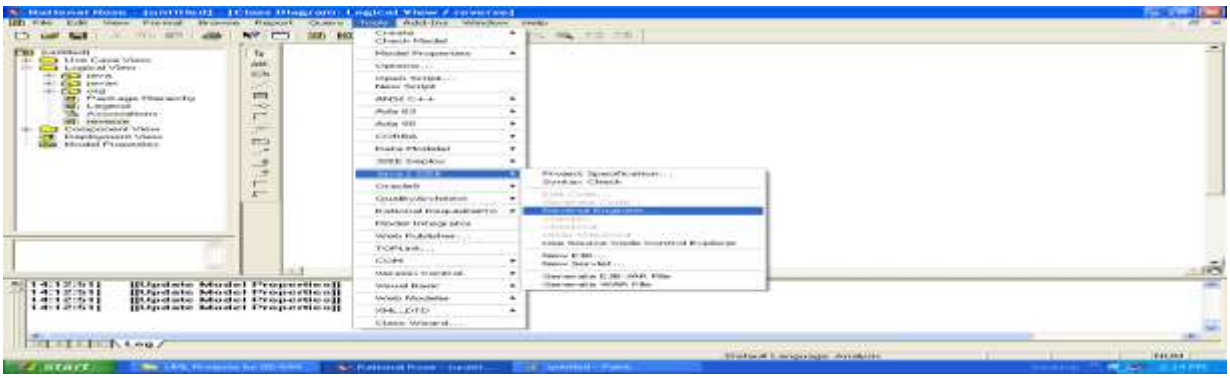


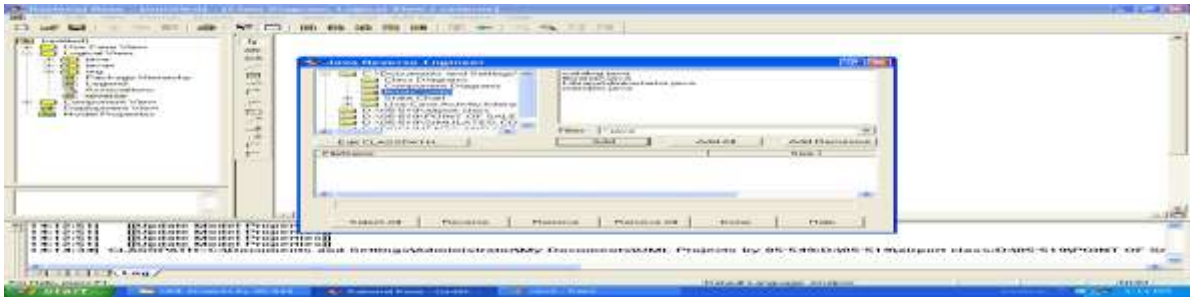
Step2 :-Select Tools -> Java/J2EE -> Reverse Engineer to reverse engineer the code.

Step3 :-Specifiy the path of code to reverse engineer.Select the files to be reverse engineered.

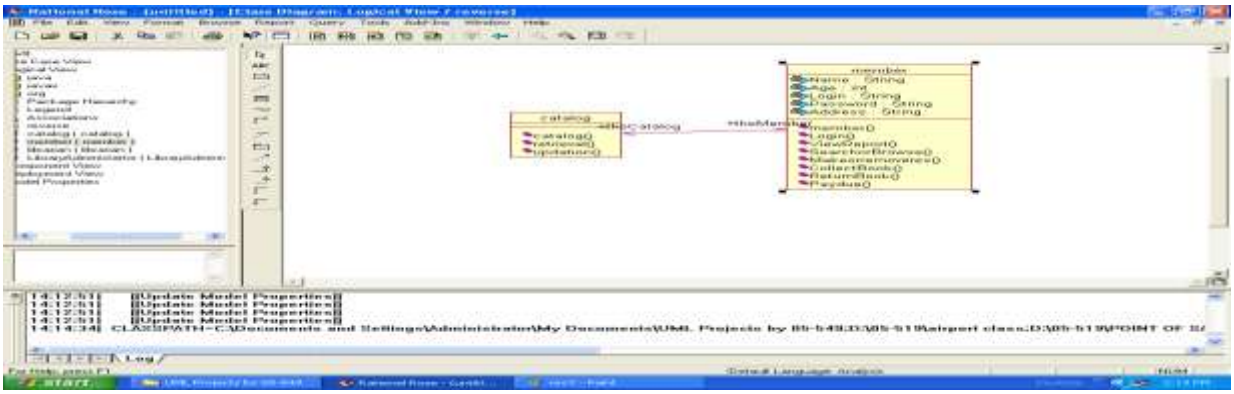
Step4 :-After adding all code files click Select All and then click Reverse.

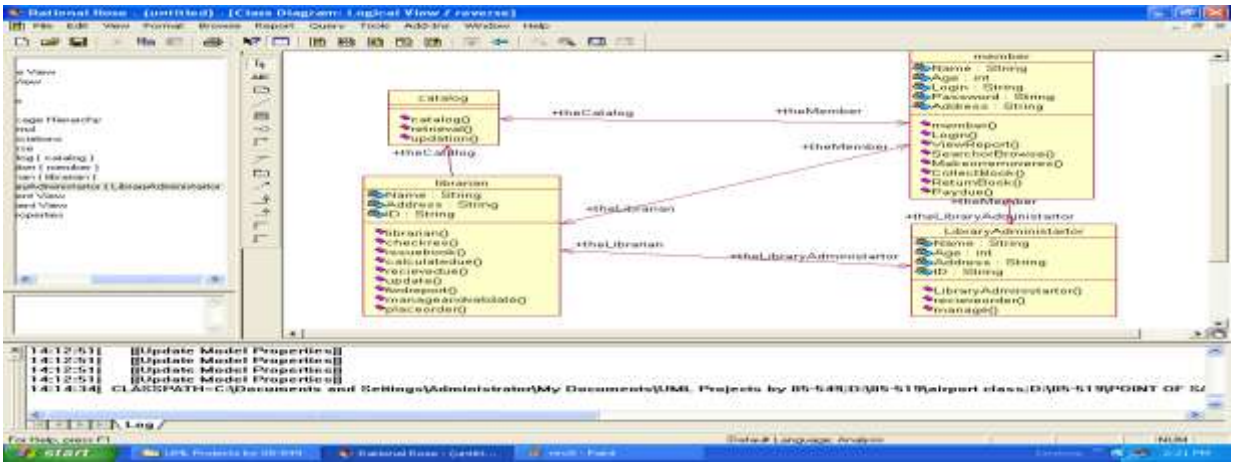






Step 5:-The classes generated in the tree window. Drag all the classes to the required area. The associations among the classes is generated automatically





Finally , the reverse engineering is complete.

## 4. CONCLUSION

The Roundtrip engineering is the combination of Forward Engineering and Reverse Engineering. This paper explains about how the skeleton source code can be generated from the detailed design using forward Engineering, and how the visual notations like UML diagrams can be obtained from the documentation called source code using Reverse Engineering in UML with Rational Rose and JAVA.

## REFERENCE

1. Pankaj Jalote ," An Integrated Approach to Soft ware Engineering" 2 nd Edition , Narosa Publishing House,2004,Chapter-4 (Planning a Software Project),Pg no. 166-170. ISBN – 81-7319-271-5

2. Roger S Pressmen, "Software Engineering - a Practitioner's Approch" 6th Eddition Mc Graw Hill international Edition, Pearson education, ISBN 007 - 124083 - 7

3. Waman S Jawdekar, "Software Engineering Principles and Practices" Tata Mc graw hill ISBN 0 -07 - 058371 – 4

4. Walker Royce "Software Project Management - A unified frame work" 2nd Edition, low price Edition, ISBN 81 - 7758 - 378 - 6, pearson education

5. Ian somarville, "Software Engineering" 5th Edition low price Edition, International Computer Science Series.

6. Shari, Laurance, Pfleeger, "Software Engineering theory and practies" 2nd Edition, ISBN 81 - 7808 - 4589 - 7, low price edition.

7. Carlo ghezzi, Mehdi Jazayeri Dino Mandrioli, "Fundamentals of Software Engineering", PHI, ISBN 81 - 203 - 0865 .

8. Grady Booch, James Rumbaugh,Ivar Jacobson," The Unified Modeling Language User Guide"Pearson Education,ISBN81-7758-371-7