Volume 3, Issue 3, March 2013 ISSN: 2277 128X
International Journal of Advanced Research in Computer Science and Software Engineering
Research Paper
Available online at: www.ijarcsse.com

UML Diagrams: an aid to Database Design Specification: A Review

Mohammad Ibrar
Department of Computer science,
Metas Adventist College, Ranchi, India

Abstract : UML now has become one of the important tools in software system modeling and implementation. Database is generally an interim part of most system. UML is every aspect of software engineering can be modeled. This covers requirement management, project analysis, database design, deployment phase and maintenance. It Provides standard for software development. Reducing of costs to develop diagrams of UML using supporting tools. UML is a language that has a number of potential industry applications. In addition to those who work in the software development field, UML is also beneficial to engineers. It is an excellent language for architects and designers as well.

Key words: UML, Software, Database, industry, architects, designer.

1. Introduction

The use of modeling to design databases far exceeds the use of modeling for applications and is generally done within most organizations. The issue is that modeling a database is generally just that modeling the database tables, columns, and relationships but not the entire database design. In this book, we look at many other parts of database design that can be modeled and how using the UML helps to model the entire database design. One part of database design that has not been covered very well in the past by the UML is the actual modeling of the database. Designers have primarily focused on using the UML for object-oriented application design and not for database design. With the use of the Profile for Database Design, this has helped open the UML to database design and database designers to the UML. While UML may seem amazingly simple on the outside, it is very rich, and it has a large number of visual elements. It is virtually impossible to memorize all the elements of UML.

2. UML

Unified Modeling Language (UML) is a standardized general-purpose modeling language in the field of object-oriented software engineering. The Unified Modeling Language includes a set of graphic notation techniques to create visual models of object-oriented software-intensive systems.

The Unified Modeling Language was developed by Grady Booch, Ivar Jacobson and Jim Rumbaing at Rational Software in the 1990s.It was adopted by the Object Management Group (OMG) in 1997, and has been managed by this organization ever since. In 2000 the Unified Modeling Language was accepted by the International Organization for Standardization (ISO) as industry standard for modeling software-intensive systems[7].

The Unified Modeling Language is fast becoming a required skill for virtually anybody involved in a software project. Requirements analysts, software developers, architects, UI designers, database professionals, testers and project managers are increasingly being asked to create and consume specifications written in UML.

It is not only used for business change software development projects. It clearly explains concepts like business modeling, enterprise architecture and objects oriented analysis and design without bogging the reader down in the kind of technical detail only a programmer would understand. UML is now the de-facto standard for software modeling. There are two important reasons the need of a modeling Language. First, the model provides a blueprint for developers so they know exactly what they need to build and for project managers so that they can precisely estimate the cost of given project. Secondly and more importantly, UML is the bridge between technical developers and their non technical users.

3. Databases

Databases are designed to offer an organized mechanism for storing, managing and retrieving information. Databases are the integral part of any information system and there exists many database implementation tools. Thus the industry is in need of a standard approach that spawns all the phases of SDLC viz., requirement analysis, modeling, design, implementation and development. In computing, databases are sometimes classified according to their organizational approach. The most prevalent approach is the relational database, a tabular database in which data is defined so that it can be reorganized and

accessed in a number of different ways. A distributed database is one that can be dispersed or replicated among different points in a network. An object-oriented programming database is one that is congruent with the data defined in object classes and subclasses. Computer databases typically contain aggregations of data records or files, such as sales transactions, product catalogs and inventories, and customer profiles. Typically, a database manager provides users the capabilities of controlling read/write access, specifying report generation, and analyzing usage[2].

4. Relational Databases

A relational database is a collection of data items organized as a set of formally described tables from which data can be accessed easily. A relational database is created using the relational model. A relational database is the predominant choice in storing data, over other models like the hierarchical database model or the network model. It consists of n number tables and each table has its own primary key.

Relational databases are based on the relational model. The relational model has a strong mathematical foundation and it is based on set theory. Each relational table in a database is considered as a set and each row corresponds to an element of the set. The eight relational operations performed on the tables are union, difference, intersection, product, projection, selection, join and division.

The Structured Query Language (SQL) is the most widely used language for relational databases. SQL is broadly categorized into Data Definition Language (DDL) and Data Manipulation Language (DML). DDL is used for specifying and manipulating the structure of the database and DML is used

for performing operations on the database itself. In this paper, we focus on the SQL DML constructs only.

5. Why UML

Since UML is accepted by the Object oriented Management Group (OMG) as the standard for modeling object oriented programs, it becomes one of the automatic choices. UML has quickly become one of the popularly used tools for modeling business and software application needs [2].

The UML became popular due to following reasons

(1) It is very flexible. It allows for many different types of modeling. Like

a. Business Process modeling event workflow

b. Sequencing of events

c. Defining database architecture etc.

(2) Platform-independent: It allows software architecture to model any application on any operating system in any programming language or network

(3) Supports Object-Oriented Methodologies: UML supports Object-Oriented methodologies. UML can be used to design both relational and object oriented models.

(4) The integration of various SDLC stages through UML tools has brought the analyst, modeler, designer and the software application developers closer to each other.

6. Types of UML diagrams

Each UML diagram is designed to let developers and customers view a software system from a different perspective and varying degrees of abstraction [1][6].

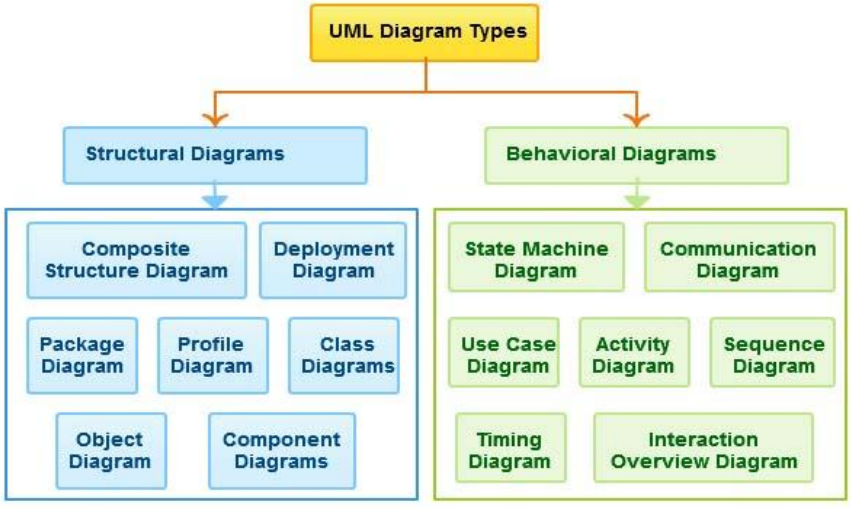

Figure 1: UML Diagram Types

6.1 Structural Diagrams:

The structural diagrams represent the static aspect of the system. These static aspects represent those parts of a diagram which forms the main structure and therefore stable. These diagrams are used to model the structural or static relationship among various objects or their components.

6.1.1 Composite Structure Diagram

- Composite structure diagrams are used to show the internal structure of a class.

6.1.2 Deployment Diagram

- Depicts the physical resources in a system
- Includes nodes, components and connection.

6.1.3 Package Diagram

- As the name suggests a package diagrams shows the dependencies between different packages in a system.

6.1.4 Profile Diagram

- Profile diagram is a new diagram type introduced in UML 2. This is a diagram type that is very rarely used in any specification

6.1.5 Class Diagram

- Backbone of almost every object-oriented method including UML
- Describes the static class structure of a system.

6.1.6 Object Diagram

- Subset of class diagrams, sometimes treated as separate techniques
- Organizes elements of a system into related groups to minimize dependencies

6.1.7 Component Diagram

- Describe the organization of physical software components including source code, run time code and executables.

6.2 Behavioral Diagrams

Behavioral diagrams basically capture the dynamic aspect of a system. These diagrams are used to model the dynamic relationship among various objects or their components.

6.2.1 State machine diagram

- State machine diagrams are similar to activity diagrams although notations and usage changes a bit.
- These are very useful to describe the behavior of objects that act different according to the state they are at the moment.

6.2.2 Communication Diagram

- Communication diagram was called collaboration diagram in UML 1.
- Represents interactions between objects as a sequence of messages.
- Describes both static structure and dynamic behavior.

6.2.3 Use Case Diagram

- Models the functionality of system using Actors and use cases.
- Use case diagrams gives a graphic overview of the actors involved in a system, different functions needed by those actors and how these different functions are interacted.

6.2.4 Activity Diagram

- Activity diagrams represent workflows in an graphical way.
- They can be used to describe business workflow or the operational workflow of any component in a system.

6.2.5 Sequence Diagram

- Describes interaction between classes in terms of an exchange of messages over time

6.2.6 Timing Diagram

- Timing diagrams are very similar to sequence diagrams.
- They represent the behavior of objects in a given time frame.

6.2.7 Interaction overview Diagram

- Interaction overview diagrams are very similar to activity diagrams.
- While activity diagrams shows a sequence of processes Interaction overview diagrams shows a sequence of interaction diagrams.
- In simple term they can be called a collection of interaction diagrams and the order they happen

7. Use of UML Diagrams

During the construction of an application system design we use all the major UML diagrams. These diagrams are very useful for the database design. The Use Case Diagrams are used to gather information during the requirement analysis phase. The sequence diagram is used to visualize the execution of the cases. While the collaboration diagram defines the sequence the message. The activity diagram is used to graphically represent some of the states and activities of a system.

Thus we can say that these diagrams are very useful for providing the functional information about database. In transaction and processing design these diagrams are very useful.

A use Case diagram

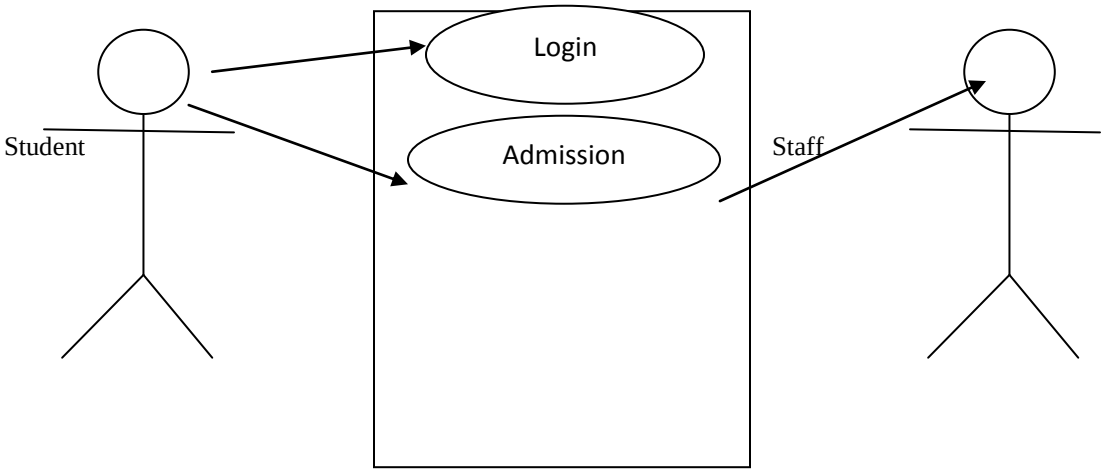

Figure 2: A sample use case diagram for a university

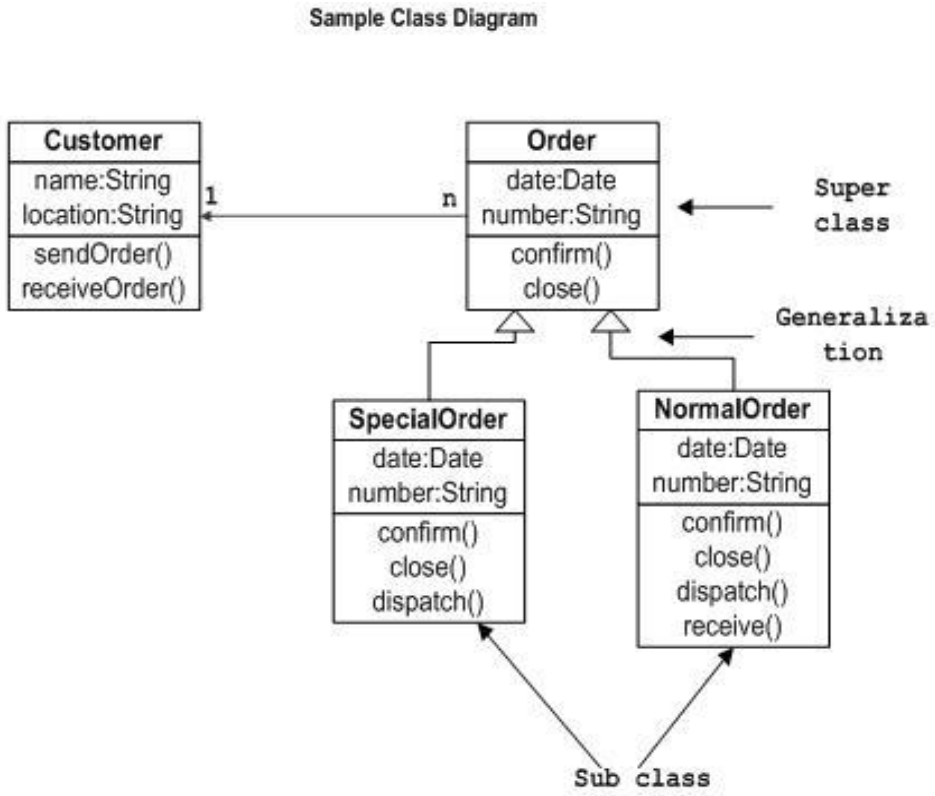

Figure 3: A class diagram

8. UML class diagram and Database design

The UML diagram is closely associated with database structure design in the class diagram. The class diagrams sometimes may be considered as an alternative notation to E-R diagram. In UML diagrams contains:

- Class; is displayed as a box. It consists of three sections.

Top Section : Class Name
Middle Name : Attributes
Last Section : Operations

- Association: Relationship types
- Links: Relationships instances
- Binary association: Represented as a line connecting the participating classes.
- Linked Attribute: Connected to the Association's line by dashed line
- Multiplicities : The minimum and maximum constraints
- Reflexive association: A recursive Relationship
- Aggregation: To represent a relationship between a whole object and its component parts.
- Different unidirectional and bi-directional associations.

The database table design from the above diagram:
Customer (Name, Location)
Order (date, number)
Special order (date, number)

9. Conclusion

UML is being used as the de-facto standard in the software industry. There are many tools used in the industry to develop information system. These tools provide the initial specification in UML that eventually leads to the database development.UML provide for conceptual, logical and physical database modeling design. Ofcourse it plays important roles in reverse engineering i.e., it allows the user to create a data model based on the already existing database structure. UML base tools can read the schema of the data models and create the database and the data storage model and generate appropriate code DDL code for the same.

References

[1]. G. Booch , J. Rumbaugh, and I. Jacobson , "The Unified Modeling Language User Guide", Addision Wesley, Reading, MA, 1999.
[2]. Blaha, M. & Premerlani, W. 1998, Object-Oriented Modeling and Design for Database Applications, Prentice Hall, New Jersey.
[3]. Booch G., Rumbaugh, J., Jacobson, I. The Unified Modeling Language User Guide, Twelfth Indian Reprint, 2004, Pearson Education
[4]. R. Bourdeau and B. Cheng. A Formal Semantics for Object Model Diagrams, IEEE Transactions on Software Engineering, 21(10):799- 821, 1995.
[5]. G. Booch, J. Rumbaugh and I. Jacobsen. UML Semantics, Version .1, Rational Software Corp., 1997
[6]. http://www.altova.com/umodel/uml-database-diagrams.html
[7]. http://en.wikipedia.org/wiki/Unified_Modeling_Language

