CHAPTER 1

INTRODUCTION

1.1 Background

With the invention of Internet and new technologies, learning style is changing and progressively expanding beyond the traditional system. There is a strong demand for learning, from individuals and groups, virtually anywhere and all the time and- these new technologies are helping to meet this demand. The digital world now is changing at even-faster pace, and long learning to be productive. Though Internet and Web, more than ever people have the ability to access online resources and can communicate with each other at the same time. These new technologies make a virtually changes in learning, which create a new type of learning- E-learning. However, how can these technologies be used to enhance and adapt the traditional methods of learning, and how can they transmit to new methods so that they are effective for this era? There are some of the fundamental challenges facing students and teachers alike in the digital age [1].

During the past two decades, the world has witnessed a technological evolution that has provided a totally new medium of communication entirely new to mankind. Through the use of networks, information in all forms has been disseminated throughout the world. One of the early applications of the Internet was its most popular application, the World Wide Web or sometime known as the "the web." The Web and recent technological advances have created the possibility for new ways of learning and teaching. Taking full advantage of the potential of the Web requires teachers to think about learning and teaching in new ways, as well as to master the technology itself. The Web-based classroom can support an existing teaching method or be used as a replacement [1].

One of the aims of Web-based class is to create new learning resources, managing existing learning resources and aggregating learning contents from a wide variety of academic and publishing sources. Hence, with this arises another question. How can the learning resources be made readily accessible to educators and learners?

Nowadays, most of software are based on object-oriented programming principles. Learning object-oriented programming languages have become required subject in everywhere but it is still considered difficult to the students. Because it requires a new way of thinking about programming and a teaching methodology is especially important for object-oriented systems, the teachers need to have a different approach. Hence, in this thesis, a flexible web-based teaching/learning framework is proposed to be used to enhance/implement an object-oriented programming course. From the review of educational software, teaching and learning models and learning objects, the model proposes to use the educational material which is transformed into reusable learning objects. Recent advances in learning technology have specified reusability and interoperability in Web-based courseware. Each interactive learning object represents an autonomous hypermedia entity, laborious to create, impossible to interlink and to adapt in a graduated manner, and hard to specify. Dynamic attributes, the look and feel, and functionality are predefined [1].

1.2 Problem Statement

The concepts of object-oriented design can be rather difficult to grasp for programming novices. Furthermore, it can be difficult for students to understand the purpose of object-oriented design. Though object-orientation is widely acknowledged as an area of computing, it is very difficult to learn. The principle themselves may not be very difficult to grasp but the deep understanding of the concepts needed to produce effective object-oriented solutions to problems is hard to achieve. The proposal outlined above is to provide a web-based learning environment for teaching object-oriented programming, with the intention of both making it easier to learn and instilling a greater understanding in the students. The model is based on the Q&A model and makes extensive use of the popular and successful BlueJ educational integrated development environment. It will be convenient to present the proposal as suggested modifications and extensions of BlueJ. Besides, it is built by the combination of different tools such as flash, Java applets and other animated tools- to bring the most effective visualizations to help student learn object-oriented programming.

C++ and Java have become very popular candidates for the teaching of object-oriented programming at university level. C++ is considered the language that makes object orientation become successful and spread all over the world of programming. Java is a pure object-oriented language. Java meets the constructive criterion of being an authentic topic for the study. It is a powerful, real world language that can be used to create applets for the Web or full software system. Tutors also like the language because it embodies the Object Oriented paradigm that is so influential in modern computing. There are thus good reasons for teaching object-oriented programming (OOP) with C++ and Java. Unfortunately, many students find it difficult to learn OOP. Even high rank universities report difficulties in teaching OOP with C++ and Java.

To tackle these problems, teaching object-oriented programming is approached by using C++ and Java as framework, to focus on the key concepts of OOP, and to try to help the students to program object orientation with both C++ and Java languages [1]. The course will focus on each concept in turn, isolating it, and putting it in context with previously learnt concepts. It will be emphasized how these concepts relate to the major factors of software engineering: reliability, extendibility and compatibility.

The following are the basic principles of object oriented paradigm:

- **Object:** Everything is an object. Think of an object as a fancy variable; it stores data, but you can "make requests" to that object, asking it to perform operations on itself. In theory, you can take any conceptual component in the problem you're trying to solve (dogs, buildings, services, etc.) and represent it as an object in your program. Each object defines:
  Its states/attributes -descriptive characteristics
  Its behaviors/operations -what it can do (or what can be done to it)
- **Class**: A class represents group of objects that have characteristics and behaviors in common. Objects that are identical except for their state during a program's execution are grouped together into "classes of objects" and that's where the keyword class came from.
- **Inheritance:** Inheritance is a process by which objects of one class acquire the properties of objects of another class. Inheritance occurs throughout all levels of a class hierarchy. Inheritance complicates the testing because bugs may propagate from a parent class to

each of its descendants. Some of the test cases for superclass can be reused for subclasses.

- **Abstraction:** Abstraction is a mechanism that enables the designer to focus on the essential details of a program component (either data or process) with little concern for lower level details.
- **Encapsulation:** Encapsulation is defined as the packing or binding together of a collection of items**.** Encapsulation complicates testing because operations must be added to a class interface (by the developer) to support testing. A member function of a class may invoke several other member functions from different object classes to achieve an intended functionality.

BlueJ is an integrated development environment for Java, developed by Michael Kölling and John Rosenberg. BlueJ enables students to easily create graphics to be incorporated into the project. This graphical representation is crucial, since it makes the assignment more interesting, thereby enhancing student engagement. It has become a popular choice for introductory programming courses because of the ease with which absolute beginners can learn how to use its features. Though admittedly the features of BlueJ are limited compared to full-fledged IDE alternatives such as jBuilder and Eclipse, its functionality is more than sufficient for an introductory-level course. Even “lightweight” IDEs such as jGRASP present a steeper learning curve for students, both in terms of installation and configuration of the software and the process of compiling and running their first Java program. Advantages of the BlueJ environment include its graphical representation of the classes and objects within a project, and the simplicity with which students can interact with them through a sequence of pop-up menus. In particular, using the *Inspect* option of the pop-up menu associated with objects, students can directly see the values of the fields of an object. This allows them to immediately see the effect of a method invocation on that object and also simplifies the debugging process. BlueJ is an integrated software development environment specifically developed for introductory teaching of object-oriented programming. BlueJ has a unique interface that supports a much greater degree of interaction than other environments and allows a fundamentally different teaching approach.

Examining this tool, exploiting the possibilities of BlueJ and demonstrating the potential inherent in the system are tasks that we have to do [15].

The Q&A model is the teaching and learning model, which aims to compliment preferred learning and teaching styles of students and teachers in order to have a better learning result. The model represents clear benefits. However, to apply our system to this model is another problem that we need to consider.

1.3 Research Goal

The main objective of this research would be to propose a teaching and learning model helps students to improve learning the fundamental principles of object-oriented programming languages and to analyze the differences in basic principles of C++ and Java . More specifically, the following sub-objectives can be defined:

- To review and analyze the state-of-art of learning objects and their use for improving teaching and learning styles.
- To study and select/create appropriate content of learning objects for teaching and learning programming of classes and instances in object-oriented programming languages.
- To propose a model of teaching and learning using the designed learning objects.
- To analyze the differences in basic principles of C++ and Java.

1.4 Scope of Work and Limitations

The scope of my thesis can be primarily defined as follows:

- The system will be designed to impart knowledge on OOP with Java and C++.
- Teaching/learning of abstraction, instantiation, data encapsulation, information hiding and message passing.
- Apprentice is believed to be in under-graduate level.
- Intranet environment will be the foundation for system development.

Object-oriented programming is a large field that cannot be fully presented in the scope of a Master thesis. It requires more comprehensive investing of time as well as of research with the help of more than one student, which is considered as the main limitation of this thesis. Hence, this thesis focuses only on the basic concepts and principles of object oriented programming that is stated in the part statement of problem.

1.5 Organization of Thesis

The rest of thesis report is organized as follows:

- Chapter 2 includes the basic principles of educational software, some styles of learning and teaching, some teaching and learning models that are applied for suitable styles learning and teaching and finally the principles of learning objects.
- Chapter 3 gives the details of the methodology used to achieve the research goal. it mainly focuses on analysis and specification of the system.
- Chapter 4 presents the implementation details of the system.
- Chapter 5 presents the experiments setup, results and analysis.
- Chapter 6 presents the conclusions of the thesis followed by future work.

CHAPTER 2

LITARETURE REVIEW

Learning using computers has increased for several decades, and many new models of learning have been invented. However, the development of new technologies and the World Wide Web create new ways of teaching that can be learned by students worldwide. This presents exciting new opportunities for educators. Before investing considerable time and effort to build such software for that kind of teaching, it is necessary to determine the effectiveness of materials used for Web-based teaching and how they can be designed and used best to apply to this model [1].

This chapter includes three main parts as follows.

- In first part, the basic principles of educational software are reviewed and considered how educational software that uses some of the new technology and uses the World Wide Web as a platform should be designed.
- In the second part, it is explored some main styles of learning and teaching and some teaching and learning models that are applied for suitable styles learning and teaching. Specifically, it is continued with online learning models and explore whether online learning models enhances learning and benefits certain types of students more than other.
- Finally, following an understanding of the concepts and models of teaching and learning, it is moved on to the third part where it is reviewed the matters and principles of learning objects, which effectiveness it provide to enhance learning their standards and present one learning objects approach to teaching programming.

2.1 Technology and its influence on education

Nowadays, we can see clearly that the use of educational software in class is increasing dramatically. It can be used in high schools and universities of higher education as a tool to help students understand more about the lectures that the teachers teach in the class or it can be considered an instructor.

Let us begin with some studies about educational software and specifically what role educational software has played and what role we expect it might play in the future. Then we also consider the new opportunities and challenges the World Wide Web is presenting for education in the future.

The different factors that can influence on education are as follows [1]:

- Educational Software
- The impact of World Wide Web
- The importance of tailoring teaching to learning styles
- Teaching programming in computer science education

2.2 Review of Learning Models

There are a lot of styles of learning. Each student often prefers one specific style. However, some students change their styles depending on the subject. Students can learn by ‘seeing and hearing; reflecting and acting; reasoning logically and intuitively; memorizing and visualizing and drawing analogies and building mathematical models; steadily and in fits and starts’. And it is also similar for teaching methods. Some teacher emphasizes in memory or understanding; some focus on principles while other on applications; some like presenting lecture and others like discussing. Therefore, the questions of how much and how fast a given student can learn depends on his own ability but also on his learning style and the teacher’s learning styles.

This part combines material from two themes that were considered separately.

- Learning styles
- Models of learning

The aim of this review is to examine the context and pedagogy for online learning. We will try to consider how different theoretical approaches can inform learning styles and why knowledge of theses styles and approaches is useful for teaching and the online tutor.

However, the focus in developing the subject was on how best the technology could enhance the student learning, so that the teaching and learning process was the driver of the subject development and not the technology.

2.2.1 Different Learning Styles

Each student has his own habits of study and practices. They also have their own motivation for studying. Therefore, teacher need to understand their students: who they are and how they learn. This can be done only when they have a clear understanding of the different approaches to learning. However a learning style will not be perfect for every individual, because human beings are complex. A student can change his habit of studying, depends on his environment, age, and so on. His individual learning style affects the way that he accepts and assimilates information. There are many ways to classify learning styles. In Online Tutoring E-book, *Cornelius* [21] chose to discuss a few of the most commonly used classifications.

- Activists, reflectors, experimenters and theorizers
- Deep and surface learners
- Visual, auditory, tactile and kinesthetic

2.2.2 Some specific teaching and learning models

There have been over a hundred different teaching and learning models developed during the last several decades. Here we will try to explore some specific teaching and learning models that are suited to online learning.

Web-based learning model

The invention of World Wide Web helps us to find out another type of learning in which we can learn without a teacher. Web-based learning model enables much more student to learn at the same time and it allows learners flexibility to interact with each other at anytime and anywhere through synchronous and asynchronous modes. That is, it allows for self-directed learning. Another advantage is that Web-based learning can allow students to choose their learning path freely. This feature is called non-linearity. Students can have more control over their learning because they can jump freely from one idea to another, without concern for a predetermined order or sequence.

Casey [8] suggested some Web-based learning models as follows:

- The Web as Source of Information: this model uses the Web only as a place to store supporting information for a traditional class. The information here can be requirements for assignments or projects.
- The Web as Electronic Book: students follow screen instructions to read material, activate multimedia demonstrations, and take self-correcting quizzes or other activities…………there is no interaction between teacher and the students through the web.
- The web as a teacher: the Web-based courses can include some personal communication between students and teacher or between students by using chat rooms or email.

Questions and Answer (Q&A) learning model

Q&A model is a learning model in which it is built as a basis for the development of educational software. It is used to 'identify possible questions covering any subject based on the Bloom's Taxonomy of learning in a cogitative domain [7]. According to *Batanov*, the benefit student and teacher would receive are teachers still can retain their teaching styles and students also can retain their learning styles. This can help learners improve their learning results. In teaching and learning, it is proven that asking question is one of the most important skills that relate to learning quality. However, in the traditional class, students cannot ask teacher at anytime they want and teachers cannot always answer all questions from each student because of time constraints. Therefore, how to ask the right questions with the right person, and how to get the right answers at the right time is one of the problem which can be solved by Q&A learning model.

In Q&A model, questions and answers are collected and grouped following one given subject. Teachers can receive question at anytime because the question will be stored in the database and when the teachers are available he can answer. It is also the same for students' role, they cannot ask the questions and receive the answers, but also can get the questions form teachers and answer. The Q&A model is responsible to record the qualified questions and answer systematically and ready to be expanded and revised at anytime by authorized users.

The Q&A model should be implemented on Web-based Internet environment so that the students and teachers can access to the model anywhere and anytime. This also enables a chance for a large number of participants in forming questions and suggesting different answers.

2.3 Learning Objects

Recently, there is a new concept appeared in the world of e-learning. This concept is learning object- an item of content, a learning resources or an entity of learning capable of being reused from one course to another.

As we know, the content of related courses taught at different universities are often similar. This occurred a new idea of reusing the learning materials and resources that are developed for the course content. It will help us reduce the time and cost of developing content and make course material more readily available and more easily updated.

Wiley [3] defines a learning object as any digital resources that can be reused to support learning. However, not all of learning objects may be reused. Here we also consider about the digital resources that are used for educational purpose. Therefore, we apply the following definition for our learning objects: ‘A learning object is a digital resource that facilitates a single learning objective and which may be reused in a different context.’

2.3.1 Authoring Learning Objects

A learning object contains:

- Overview (Introduction, statement of subject taught)
- Series of information objects
- Summary
- Assessment (i.e. Quiz, consisting of assessment questions)

There are two major facets to authoring learning objects. The first is the content of the learning object itself; the second is the metadata describing the learning object.

Content

The most common medium for content is hypertext markup language (HTML). HTML can combine the content and present the static information. HTML pages can be built by a lot of authoring tools. Some popular tools are Microsoft's FrontPage or Macromedia's Dreamweaver. These tools allow authors to embed interactive applications into the page such as Java applet and flash. However, they have some limitations. For example, FrontPage can only interact with Microsoft server and in some situation it cannot display well in multiple formats. This limits the portability of HTML pages. Another medium for creating a document's content but more portable is XML. XML textual resources, such as video clips, applets, animations, we need to author them using other more complicated software as Java editors or Macromedia's Director.

Metadata

To store information about the learning objects for re-use purpose, we need to create an associated metadata. The content of the metadata is determined by the type of object. The authoring metadata should be easy for most course designers. For example, the authors just have to access a form I which they can enter the appropriate metadata information. And this form will send information to a metadata page editor. All author needs to know is what type of metadata is being created, and that type is defined by the type of object.

2.3.2 Standards and Specifications for Developing Learning Objects

As LO development is a collective enterprise among the programmers, graphic designers and subject experts, a standardized approach can accelerate and establish efficiency in the LO strategy. It is important that the developers agree to a set of specifications for development of LOs covering such areas as technology, editorial requirements, and stylistic considerations. A commonly agreed on standard will enable genuinely sharable and reusable content objects, without which we will revert back to current iterations: static Web pages [5].

Technical Standards

The technical specifications should address the interoperability of LOs and the physical structure to facilitate easy manipulation of the elements. Success depends on the flawless execution of LOs in all operating systems and delivery media. What is crucial for achieving interoperability is the selection of language for producing LOs. The language should be chosen with the future in view, that is, we should not attempt to make objects compatible with all earlier generation technologies, but instead we should focus on the next generation technology. XML is ideal for achieving both purposes since it is being endorsed as the standard for all future applications. Additionally, separation of structure, content and presentation, which is the fundamental logic of XML, will allow the flexibility required for deploying and manipulating LOs.

Editorial Requirements

Every discipline has its own discourse and rules of construction for its discourse. Subject experts in a discipline should agree beforehand on editorial standards to ensure the consistency of language across the LOs. Similarly, a common terminology should be created for referring to concepts in a discipline. To accommodate the requirements of LO strategy and individual stylistic preferences, a LO can have a glossary that explains the terminology used in the LO and which cross-references other terms used in the discipline to refer to similar concepts.

Stylistic Considerations

Appearance and style are extremely important for an effective presentation of LOs. Developers should draw up specifications for use of color, fonts, and layout of images and text. These structural elements should be consistent across LOs so that they can be easily combined for instruction. Since XML separates content, structure and appearance, stylistic considerations can be excluded from the domain of subject experts. Due to the flexibility of XML, the same style sheets can be used for different contents without additional development, and each can be modified without affecting the other. An ideal situation would be to develop several interface and stylistic environments that are user-controlled, which would enable the user to choose the most suitable form of interacting with and exploring the knowledge.

2.3.3 Learning Management System (LMS) and Learning Content Management System (LCMS)

A LMS covers the domains administration, learning, and authoring with the objective to simplify the management of learning/training programs. A LMS contains a database which is a repository to store learning objects. Administration job includes managing courses, users, and institutions. It also tracks the user's activities and provide self-tests and evaluation. A LMS must provide users with suitable input facilities to communicate with database interfaces. Online wizards are responsible for this job. They allow modifying the courseware on the fly. For example, when user input data, it will be versioned and be verified by editorial staff. Finally, it can be integrated to the system automatically.

What is LCMS (Learning Content Management System)?

"LCMS is a system (mostly Web-based) that is used to author, approve, publish and manage learning content (more specifically referred to as learning objects)." This means that a LCMS includes a LMS and the content management system (CMS). It has libraries of Los which are assembled by a CMS and administrated by a LMS. *Nichani* [11] stated the workflow process around a LCMS as following:

- New LOs are created by instructional designers and submitted to LCMS. Each LO has to have its own objective. Designers also can create new courses by assembling already created LOs.
- Editors are responsible for approving or rejecting the submitted LOs.
- "Personalization rules would set in, targeting the new LOs/courses to those who fit its profile."
- The useless LOs and courses can either be backed up or achieved, or just be deleted from the database.

2.3.4 Nature of Learning Objects

It is often suggested that learning objects are really like the objects of object-oriented software. Let us see the following. Consider audio-visual media files and files of other formats (e.g., GIF, JPEG, PowerPoint Presentations, etc.) which are simply collections of bits. These files are not software, much less software objects. So, it is clear that not every learning object is software. However, files such as HTML files containing markup and code written in scripting languages such as JavaScript and Java applets may be considered software. However, HTML files do not support object-oriented concepts, so are not object-oriented. Java applets are certainly software. From the perspective of its developers, a Java applet may be considered object-oriented. From the perspective of an instructional designer or other programmer wishing to use the applet (without access to the source code), the applet is just like another piece of software, and indeed, its object-oriented features are irrelevant. Thus, despite the claims being made, a cursory examination reveals that most learning objects today are not truly object-oriented from the computer science perspective. However, from an instructional designer's point of view, there is no real benefit in viewing learning objects as object-oriented and it is best to look at learning objects in terms of their content, context of use, and relationship with other learning objects. To require one to understand object-oriented theory to put together meaningful content packages is perhaps akin to asking a good carpenter to understand the internal structures of the materials that are used to fabricate furniture. From a computer science perspective, there is much to gain by treating learning objects as object-oriented software artifacts. Object-oriented technology can be used to take learning objects out of their current static form and imbue them with behaviors that allow them to contribute more meaningfully to an instructional situation. Object-oriented learning objects are capable of overcoming several of the limitations mentioned in the previous section. Moreover, instructional designers and other users do not have to be aware of this technology, much like users of Microsoft Office are not aware of the agent-like technology underlying the Office Assistant.

2.3.5 Limitations of Current Efforts with Learning Objects

LOM metadata allows searching for learning objects based on keywords and other basic pedagogic metadata such as *LearningResourceType*, *TypicalAgeRange*, or *Language*. To determine the suitability of a learning object, an instructional designer must carefully examine each learning object. This can be very time consuming, given that many learning objects may satisfy the query and each one must be examined individually. The Digital Repositories Interoperability (DRI) Specification from the IMS makes it easier for computers to query on-line repositories for learning objects. However, the problem still remains, since the metadata is the same and computers cannot perform the level of filtering required.

Making instructionally principled decisions with learning objects requires deep pedagogic information in the metadata. Current learning object metadata say nothing about how to combine learning objects with others, and indeed, whether such a combination is useful in the first place. Metadata also say nothing about the types of learners for which a learning object is best suited and the kinds of teaching and learning strategies it employs. Worse yet, metadata do not even identify the learning outcome(s) that will be achieved if the content in a learning object was successfully learned. Putting together content packages based on reusable learning objects is a very difficult problem. However, this problem must be solved otherwise it will severely limit the usefulness of the numerous repositories of learning objects that are being developed.

In addition to the problem of locating the 'right' learning objects, the full set of 86 elements in the IMS Metadata Specification is not suited to direct implementation since it entails a huge classification effort. Also, widely varying interpretations of the utility, scope and purpose of individual elements threaten to cause considerable interoperability problems, which the specification was designed to solve in the first place. Thus, even if the metadata specifications were very useful in automatically generating content packages (and they are not), creating the metadata itself entails a huge effort. We therefore contend that the future of learning objects is at stake. Current development efforts do not address the really important issues associated with using learning objects for e-learning. There is much excitement about metadata, the Semantic Web, RDF, ontologies, and other technology that promise a bright future for locating information on the Web. These technologies are poised to profoundly influence the way in information is used in the future Web. However, they do not address many of the problems related to successfully using learning objects in an instructionally meaningful manner.

2.4 Teaching and Learning Programming

2.4.1 Challenges of teaching programming

The first challenge of teaching programming course is how to bring all students to a common learning environment within a specific period of time. However, students who take programming course often have different background and different levels of programming experiences. Therefore, the matter here is how to "provide the right information in the right context at the right time to the right person." Other problem is the lack of an effective communication mechanism between instructor-student and student-student. When students misunderstand concepts or have errors in the program that takes a lot of time for debugging, they need help from instructors or other students. However, be the time they get help it may be too late than it wastes time. Another problem is that teaching programming concepts and principles are sometimes very difficult. The effective way is using examples for explaining the concepts and principles. However, it is not easy to present the execution of examples. Even the examples are executed, students still don't understand because they lack of interaction with the programming.

Analyzing all challenges above, it is necessary to propose a solution for teaching programming course. The proposed solution should allow different students with different background and levels to comprehend the course content, understand the concepts of programming. It also should provide an interactive communication platform for discussions and exchange of learning material.

2.4.2 Advantages and disadvantages of teaching programming online

Teaching programming also includes all the advantages of teaching any course online which are stated in the introduction part of this thesis. Moreover, programming is really more suitable for teaching online. The reason of this is that learning programming trends to be self-placed. Students can learn how to program without the direct instruction of teachers. If the courseware is designed with visualization and east to understand, it is really a good tool for teaching programming outline.

However, teaching programming also has some limitations. Teaching online requires two kinds of staff: the course team and the tutors. The course team is responsible for developing the course.

The tutor is responsible for teaching the course developed by the course team. The tutor can be the courseware itself or a real instructor who can communicate directly with the student, as well as marking and providing feedback on assignments. Because the courseware is taught online, when students have problems in understanding and need discussion, it will be difficult for tutors to understand what the problem is. The tutor sometimes has to guess the problems or students have difficulty in explaining what it is while they do not know very well.

Within a conventional university, a programming course often has some lab hours that tend to be occurred in departmental laboratories where all students are using a uniform platform. When the course is taught online, students will have different platforms, such as different hardware, or possibly different operating systems running on similar hardware. Therefore, the courseware should be designed so that it can run in all platforms because making a particular hardware and software configuration mandatory will vastly reduce the number of available students. Additionally, teaching online often requires run-of-the mill computers and efficient bandwidth.

2.4.3. Teaching and Learning Object-Oriented Programming

Recently, object-oriented programming has become the most influential programming paradigm. It is offered in almost computer science and related programs. It is now used in al large number of applications in education and industry. The utilities of object-oriented programming are clear evidences that OOP is a good thing and should be taught. OOP has changed the way of programming, modularization and program design. It also supports techniques for approaching problems: programming in teams, maintenance of large systems and software reuse. In short, object-oriented programming seems to be a good tool for teaching those programming methodologies that we consider important.

Moving from structured programming to object-oriented programming requires a different way of thinking. Structural programming follows the thinking of the sequence of programs executed by the computer, and object-oriented programming is based on representation the real world. Therefore, in order to teach object-oriented programming essentially, we need to develop a new suitable approach of teaching and learning.

Becker [19] stated “many introductory programming courses focus on syntax instead of concepts, code instead of models and construction instead of structure.” Teaching object-oriented programming by concepts, models and structure is the heart of object-oriented development. The important thing is to understand how to design the software following the principles of object-orientation before it can be focused on how to program with a specific language.

There are at least three important factors that make the teaching and learning of object-oriented development different and more difficult:

- Some high-level and abstract concepts/principles such as abstraction and encapsulation need to be taught early. Because it allows us to understand the way of thinking in object-orientation.
- Some object-oriented concepts are highly interrelated, making it impossible to teach them in isolation; such as encapsulation and information hiding or some relationships
- There is typically no central control in object-oriented programs, making it difficult to find the right location for a piece of code.

The challenge of teaching and learning object-oriented programming is how to put together aspects as theory of OOP, data and logical structures of programs, chosen programming languages, syntax commands of this language and also the information from the environment.

Learning object-oriented programming for a long time was considered more difficult that structural programming. It was often taught late than teaching structural programming in the curriculum. Many students who started with procedural style first often felt that OOP seems to be difficult to understand. The main reason for this problem is the paradigm shift. Because of changing the ways of thinking in programming, it often takes programmer long time to switch his/her mind-set from a procedural to an object-oriented view of world. However, this is changing now: object oriented programming has been stared to teach as the first programming course. Evidence indicates that students do not seem to have any difficulty in understanding object-oriented principles if they encounter them first. It is the switch that is difficult, not object-orientation.

Therefore, the new approach of teaching object-oriented programming here is clear: if we want to teach object-orientation, we should do it first. We don’t need to go by a path through procedural programming to object-orientation.

Object-orientation is an underlying paradigm that shapes our whole way of thinking about how to map a problem onto an algorithmic model. It determines in fundamental ways the structure of even simple programs. It cannot be added on to other language constructs; rather it replaces the fundamental structure of procedural programming. Because of this, we firmly believe that object-oriented concepts should be taught form the very beginning. If it is considered as necessary that students can also program in a procedural style, then a procedural language can be introduced later. The paradigm shift backwards (from object-orientation to procedural) is much easier. Programmers can just think about it as writing the complete solution within one class (a solution an object-oriented programmer might not be very happy about, but easy enough to understand)

CHAPTER 3

RESEARCH METHODOLOGY

In literature survey, the basic principles of educational software were reviewed and considered how educational software that uses some of the new technology and uses the World Wide Web as a platform were studied. Some main styles of learning and teaching were reviewed and some teaching and learning models that are applied for suitable styles learning and teaching were seen. Specifically, it is continued with online learning models and explore whether online learning models enhances learning and benefits certain types of students more than other. following an understanding of the concepts and models of teaching and learning, it is moved on to the third part where it is reviewed the matters and principles of learning objects, which effectiveness it provide to enhance learning their standards and present one learning objects approach to teaching programming. It is the objective of this research to design and develop a web-based model using learning objects for teaching and learning classes and instances in C++ and Java

To achieve the objective of this thesis, it is needed to study, analyze, design and implement the system with following phases:

Phase 1: System Analysis

- Study the Web-based teaching and learning model, Q&A framework to specify the requirements for the system.
- Study the principles, concepts of object-oriented programming with Java and C++.
- Study Learning Objects and tools to create Learning objects..
- Discover BlueJ educational software tool, how to extend Blue to integrate to the system.
- Study visualization approaches and learning object approaches to find out the best design for the study.
- Choose the tools to develop the system and study them in detail.
- Study the requirements of the object-oriented programming class.

Phase 2: Design

- Specify the detail requirements of the system: inputs, outputs, interface and functions
- Design for each requirements
- Design the web-based object-oriented programming class
- Design the content of OOP class
- Design the structure to integrate to Q&A framework.

Phase 3: Implementation

- Install the tools and software which are needed to develop the system: BlueJ, Java environments, database software and tools to build learning objects.
- Setup and modify Q&A framework
- Implement the web-based teaching and learning environment for object-oriented programming.
- Implement to integrate with Q&A framework.

Phase 4: Testing

- Test my own software,
- Test the whole system of the group.
- Run the demonstration.

Phase 5: Documentation

- Write user manual.
- Write reports of the thesis.

Before coming to implementation phase, it is needed to analyze and design the system in details. This chapter presents the necessary analysis and specifications for the implementation of the system. First, we examine the criteria for the design and implementation, the criteria of functionality the system should have, and the criteria of system's architecture. Then we continue to describe the general approach to implement the system and the functionality should be fulfilled. After that we present the actors interacting with the system and some use cases respectively. This chapter will be ended with the architecture of the whole system which have been implemented [1].

3.1 Criteria for Design and Implementation

One of the most important parts of system analysis phase is deciding which criteria the system should meet. The criteria are defined depending on the environment of the system, and the actors interacting with the system. Two most important criteria which every information system should meet are functionality and architecture. The functionality criteria define the goal of the system, which will be implemented. The architecture criteria define how the system should be constructed.

3.1.1 Functionality criteria

The goal of this thesis is to construct a system based on a web-based environment to teach and learn the basic concepts of object-oriented programming in Java and C++. Therefore the functionality criteria which the system should meet are:

- Identify the actors interacting with the system
- Understand the web-based environment which the system will be applied. For example, the system should allow learners to access the system with multiple browsers.
- Provide the functions which help learners understand the lessons easily, and meet learner's need
- Provide the manual of the system to help learners and teachers use and interact with the system easily.

3.1.2 Architecture criteria

The architecture of the system should be designed such that the learners can learn step-by-step, from the concepts to the principles, from easy to difficult parts, from definitions to examples, from lessons to practices and tests in the course.

- Because the course will be applied in a web-based environment, the client/server model should be considered as the core of the system

- The system should be implemented without depending on the platform so much and it can be integrated with other system easily. Therefore, applet, flash and other web-based application can be the suitable choices among other technologies.
- The web server should be used for providing server side environment
- Graphical lessons to explain the concepts and principles of object-oriented programming with animations and highly interacted with users should be implemented
- The program demonstration and simulation should be provided. Because it is necessary to show the student how the objects are created and stored in the memory, how the methods are invoked by the objects, and why abstraction, encapsulation and information hiding principles should be applied. All these jobs can be displayed by simulating the actual program.
- The learning material should be designed so that optimal learning is facilitated and curricular redundancy minimized.
- The system should be organized to a level of competency that the students should expect to learn, knowledge, comprehension, application, analysis, synthesis and evaluation
- Time to learn, how long does it take a learner to learn the course?
- The system should be implemented in modularity so that it is possible to link to other sources without influencing to the current functionalities

3.2 Description of the system

This study is aimed toward teaching and learning the basic concepts and principles of object-oriented programming in Java and C++. To help the students understand the concepts and principles in OOP, it's definitely necessary to design the lecture with high visualization of definitions and examples. The feasible way to do it is through web with flash and java applet technology. Java^{TM} 2 SDK, Standard Edition supports and provides Application Programming Interfaces (APIs) for building applet-based application. Macromedia Flash MX 2004 is the authoring tool that allows creating interactive web application and developing multimedia content.

The system will be constructed by Learning Objects so that they can be reused in multiple contexts. The flash, java applets and accompanying materials should be developed to illustrate OOP concepts. The flash serves as a tool which visualizes the learning content. It also instructs the learners to go from step-by-step. The java applets serve as a visual representation of the program with respect to go through the sequence of the course.

3.3 Functionality of the system

After examining all the criteria of functionality and architecture, the system now should be designed with its functionalities. Based on the idea of using and re-using learning objects, a course of teaching and learning the basic concepts and principles of OOP applied in a web-based environment should be implemented in terms of functionalities are:

- The main actors of the system are learners. They interact and use the courseware.
- The system enables teachers and learners to view course outline through Q&A system. Besides, they also can submit their questions and answers through this foundation.
- The system explains in detail the concept of OOP such as class, object, field, method, access level control, abstraction, encapsulation and information hiding. The explanation should be visualized and animated with flash. Teachers and learners can go step-by-step from easy lessons to difficult ones.
- However, the explanation of the concepts of OOP cannot be clear without seeing the examples of program codes. These examples will be animated with java applet. Teachers and learners can see how the corresponding source code running, for example, how an object is created, and how the methods are invoked.
- Teachers and learners also should be able to review the source code and the process of running program.
- Learners should be able to review the whole lesson and they can practice with the exercise in each lesson.
- At the end of the lesson, there should be a test so that teachers can assess the comprehension of learners.

Learners should study OOP, using a top-down approach, starting with the general concepts and after that go in details to the examples and practices, then continue with the more difficult ones. This approach helps them to study OOP without knowing procedure programming before.

3.4 Actors and Use Cases

The main actors of the systems are learners and some use cases are as follows:

Study objects and classes concepts

Study how to declare classes and create, destroy objects in Java and C++. This part contains:

- Study how to declare classes in Java/C++
- Run the example of program codes to see how the classes are declared in Java/C++
- Study how to create objects in Java/C++
- Run the example of program codes to see how objects are created in Java/C++
- Study about the difference between object declaration and numerical data declaration in Java/C++
- View the animation of the difference between object declaration and numerical data declaration in Java/C++
- Study how to assign objects in Java/C++
- View the animation of comparing the assigning the content of primitive variables with the assigning objects in Java/C++
- Study about member access operators in Java/C++
- Run the example of program codes to see member access operators in Java/C++
- Study how to destroy objects in Java/C++
- Run the example of program codes to see how to destroy dynamic allocated objects in C++

Study about fields and methods of objects in Java/C++. This part contains

- Study about access level control in Java/C++, including default, private, protected, and public
- Run each examples of program codes to see how each corresponding access level control work

- Study how to define and use class fields and class methods in Java/C++
- Run the examples of program codes to see how to define and use class fields and class members in Java/C++
- Study how to define and use constructors in Java/C++
- Run the examples of program codes to see how to define and use constructor in Java/C++
- Study how to define and use destructor in Java/C++
- Run the examples of program codes to see how to define and use destructor in Java/C++

Study about abstraction, encapsulation and information hiding. This part contains:

- Study about abstraction principles and process
- Study about encapsulation and information hiding
- Run the examples of program codes to see the problems of designing without encapsulation and/or information hiding principles

Take a test after each step above

The Figure 3.1 illustrates the sequential relationships between the use cases.

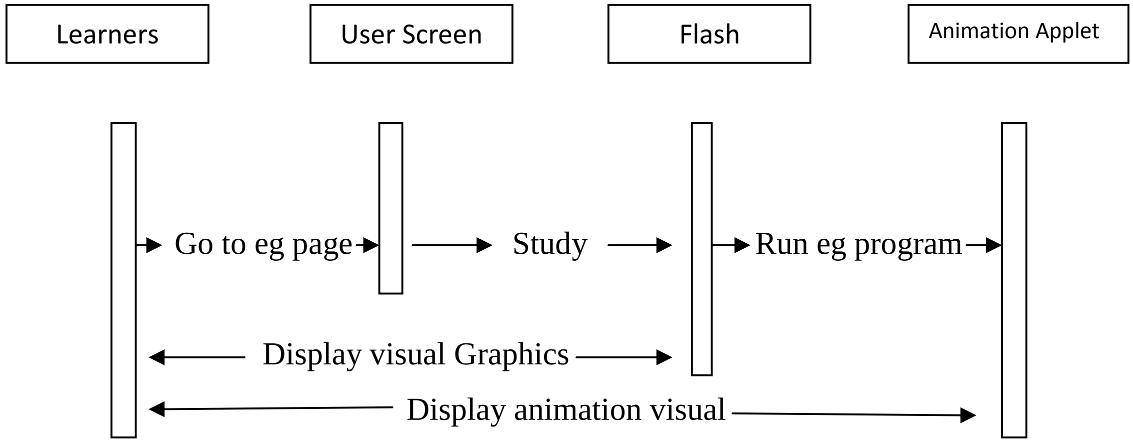

Figure 3.1 User sequence diagram

3.4.1. Actors

The main actors of the system are learners. Because this course is teaching introductory OOP, and it follows an object-first approach, learners are expected to have some knowledge about Java syntax of basic data types and commands, such as integer, float, double types and if-command, while-command. Learners don't be required any knowledge about procedure programming before entering to study this course.

Learners are able to use all the functionalities of the system. They can access the web page to see the outline of the course, choose the lesson they want to learn, study all information in the lesson, run the examples step-by-step to see how the program run, view the source code of the examples, and make a test at the end of each lesson to continue with the one.

3.4.2. Objects and Classes Concepts Learning Object

The first LO in this course is to teach concepts of objects and classes. The content of this LO is presented in detail in Appendix A. This LO is created by a flash with visual graphics to explain what an object is and what a class of object is. At the end of this lesson, learners can practice with some exercises and make a test.

The work flows of this learning object are as follows:

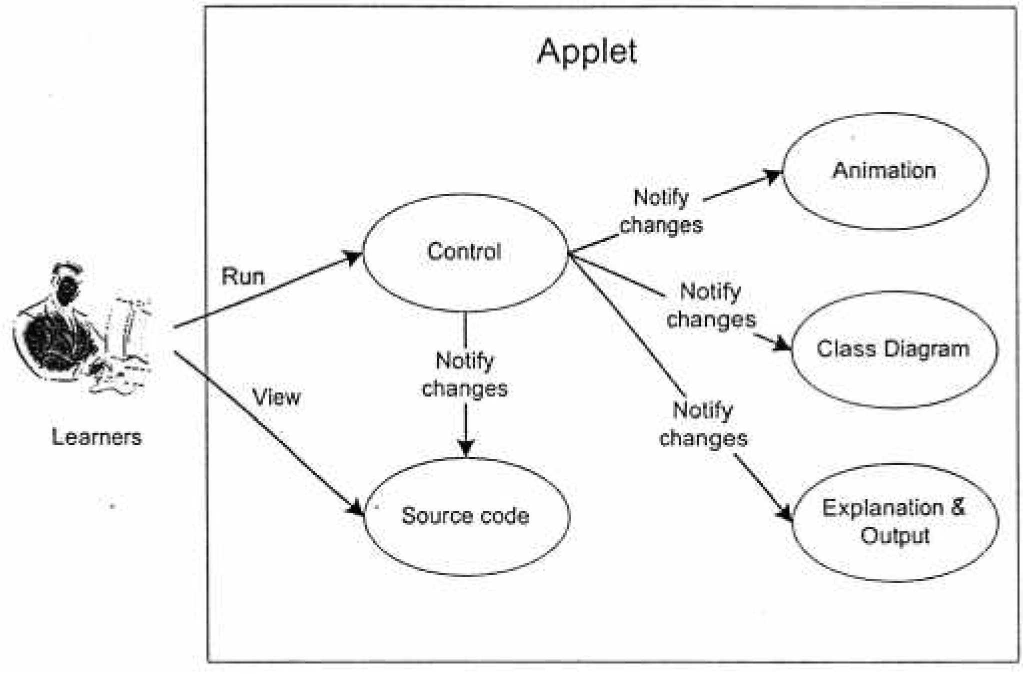

Figure 3.2 Object and Classes Concept LO

3.4.3. Declare Classes and Create, Destroy Objects Learning Objects

The second LO is to teach learners how to declare classes and create, destroy objects in java and C++. The content of this LO is presented in detail in Appendix B. In this LO there are one flash, which authors the content of whole lesson, and some examples, which are created by applets. The flash with a lot of animation instructs learners to understand the syntax to declare classes, how to create, destroy objects, how to access the members of an object, and how to assign objects in both Java and C++. The examples can be run with applets respectively. Each applet allows learners to see step-by-step the process which happens inside memory.

Applets are:

-Create object in Java

-Member Assess in Java

-Create static object in C++

-Create dynamic object in C++

-Member access of static object in C++

-Member access of dynamic object in C++

-Destroy object in C++

The work flows in each applet are described as follows:

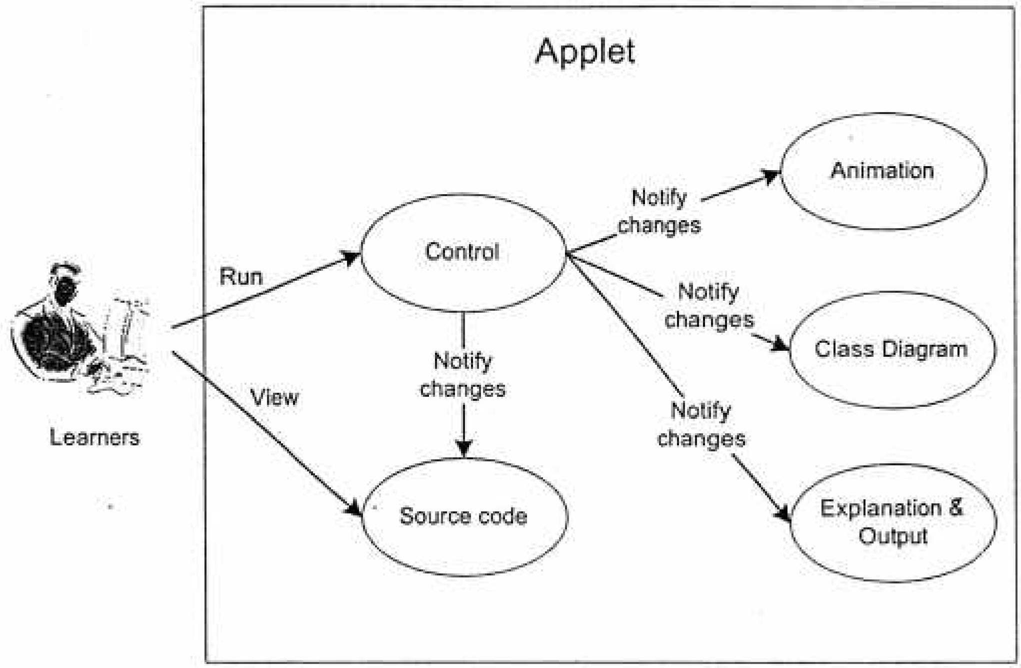

Fig 3.3 The work flows in Applet

The main goal of this thesis is teaching and learning classes and instances in Java and C++. Therefore, to help learners have a proper understanding of OOP, the applets solve two main scenarios:

Object creation: How the object is created, how the correct amount of memory allocated to the object, how the constructor is executed to initialize fields as part of object creation. Invocation of a method in an object: how operations can be invoked on objects, how the access specifiers control the accessibility of an object's member.

These scenarios are central to any understanding of OOP and must be understood by student. The applets also present the connection between execution of an operation and its source code, the state of an object and the fact that an operation will change the state.

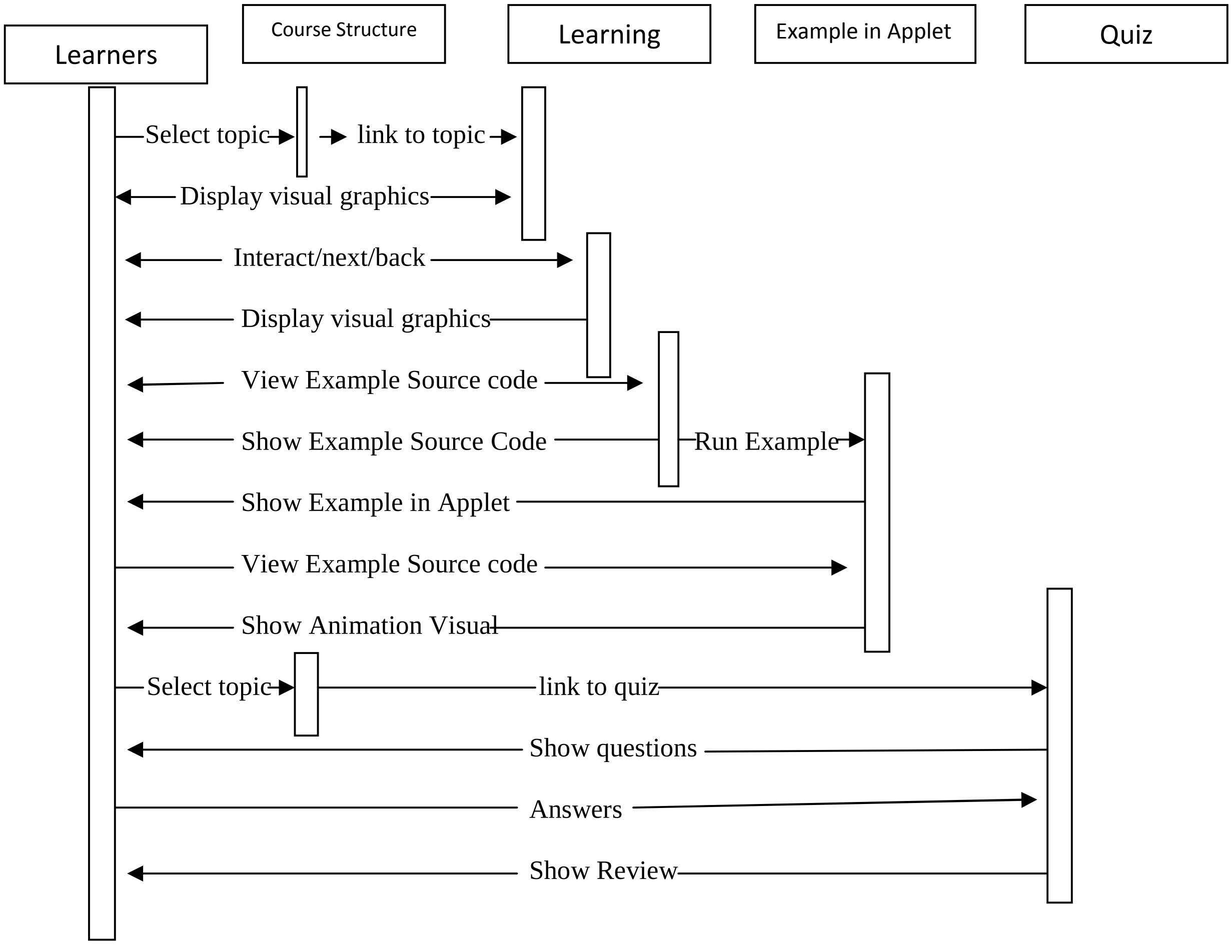

Figure 3.4 Sequence diagram of LO

The work flows of this LO are described as follows:

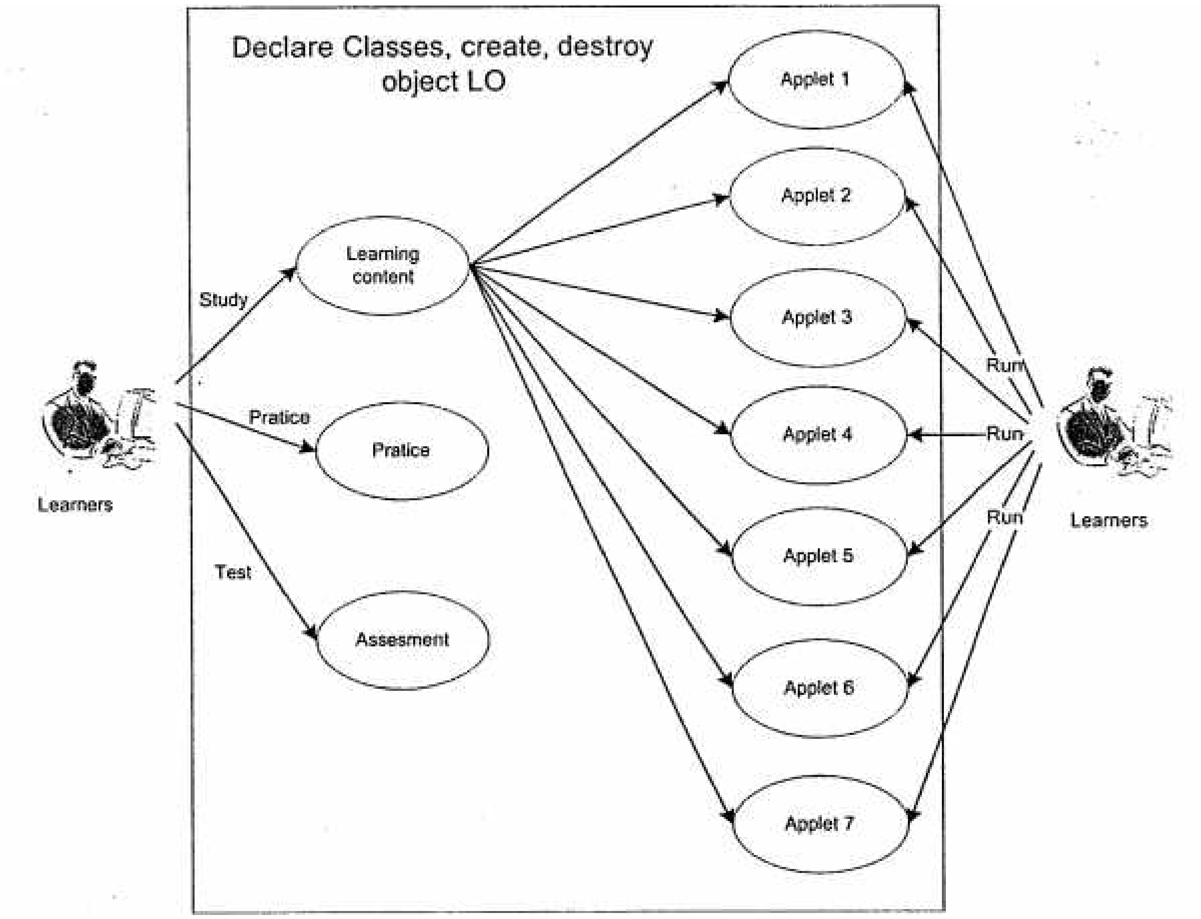

Fig 3.5 Declare Classes and Create, Destroy Objects LO

3.4.4 Fields and Methods Learning Object

The third LO is to teach learners how to declare access fields and methods of an object. There are four types of access level controls in Java, and three types of access level controls in C++. This lesson also teaches learners how to declare and use static class members, constructors and destructors in Java and C++. The content of this LO is presented in detail in Appendix C. Similar to the second LO, this LO also includes some Macromedia flash files, which authors the content of whole lesson, and a number of examples, which are created by applets. The applets are:

Default Access Level in Java
Private Access Level in Java

Protected Access Level in Java

Public Access Level in Java

Class Fields and Methods in Java

Multiple Constructors in Java

Private Access Level in C++

Create static object in C++

Protected Access Level in C++

Public Access Level in C++

Static Class Members in C++

Default Constructors in C++

Copy Constructors in C++

Multiple Constructors in C++

Destructors in C++

These applets also have the same user interface and work flows as the above. Each applet, together with its name, has its own goal to present the most visualizing graphics to help learners understand the use of the concepts which are taught in the flash. At the end of this lesson, learners can practice with some exercises and make a test to get the assessment. The sequence diagram of this LO also is similar to the second LO, but the difference in the number of applets.

3.4.5. Abstraction, Encapsulation and Information Hiding Learning Object

The fourth LO is to teach three of the most important principles of OOP. They are abstraction, encapsulation and information hiding. This LO explain how to go through the process of abstraction, how to assure encapsulation and information hiding principles in object-oriented design. This LO also shows some problems when encapsulation and information hiding criteria are not fulfilled, then it teach learners how to solve those problems.

CHAPTER 4

IMPLEMENTATION

After analysis and design phase all the requirements and criteria for the system are understood. This chapter will continue with the implementation of the system. The system is not a standalone system but it should be integrated with a Learning Management System and the Questions and Answer Model. However, the LMS and Q&A model are implemented by other members. So in this part it is found that how the system is integrated, and the implementation of creating learning objects for storing in LMS and Q&A model. Based on the analysis and specification in previous chapter, there are four separate learning objects developed using Java Applet technology and flash. Each one of the four can be considered as combined-open learning object whose constituent learning objects are directly accessible for reuse from the combined-open object.

Teaching object-oriented programming in C++ and Java are quite similar. In the scope of thesis it is just noticed about teaching classes and instances so that there are some lessons having the same content. This study includes four learning objects for teaching classes and instances in Java and four corresponding learning objects for teaching in C++. The differences of representing learning objects in Java and C++ will be analyzed and they will be presented and compared at the end of learning objects' representation.

There are four main learning objects created for teaching classes and instances in Java. They are:

- Objects and Classes Concepts Learning Object
- Declare Classes and Create, Destroy Object Learning Object
- Fields and Methods Learning Object
- Abstraction, Encapsulation and Information Hiding Learning Object

It is noticed that all the learning objects here are built based on the combination of some flash and Java applets. Their implementation is quite similar. In this chapter it is focused on the first and second learning objects. Because the components of these learning objects are also reused for the third and fourth ones, it is not necessary to present it again. In the third and forth learning objects, we will show differences added into the components.

4.1 Objects and Classes Concepts Learning Object

This object is the simplest one among four learning objects. The aim of this object is to teach the theory of concepts of object and classes. It is built with Macromedia Flash for creating visual graphics in the content. At the end of the lesson, there is one quiz built by Macromedia Captivate.

Object Summary	
Name: Object&ClassesConcepts	
Category: **General**	
Title	: Objects and Classes Concepts
Description	: Define the concepts of objects and classes and examples
Keyword	: Object Class Define State Behavior
Language	: English
Category: **Lifecycle**	
Status	: Final
Author	: Suresh Khatiwada
Date	: 2010-12-01
Category: **Technical**	
Location	: Flash Source
Format	: Macromedia Flash
Size	: 132 KB
Requirements	: Macromedia Shockwave
Category: **Educational**	
Resource Type	: Flash
Interact Level	: Medium
Context	: University level
Category: **Flash**	

Copyright	: No
Description	: For Educational Purpose

Table 4.1 Objects and Classes Learning Object Metadata

Each learning object has a metadata created in XML file describing the information about learning object. The metadata is used for LMS to locate and identify the learning object. The data elements and data structures used to describe this learning object is as Figure 4.1 above.

The other component of this learning object is one quiz to test students' understanding. The questions in the quiz are multiple choices. Each question has its own given weight. A student can pass the test if his/her correctness is above 80%. The result of the quiz will be returned to the LMS for managing the student's process.

4.2 Declare Classes and Create, Destroy Objects Learning Object

The second learning object corresponds with the lesson two of the courseware. This learning object for Java is combined by two main topics considered as smaller learning objects: declaring classes and creating objects, the one for C++ has an extra topic for destroying objects. There is also one quiz for assessment at the end of this learning object.

Let's look at the learning object for Java; the objective of this learning object is already presented in the previous chapter. The implementation of this learning object includes two flash files and two applets. Two flash files cover the two topics above. The metadata for these two flash objects are shown as follows:

Object Summary	
Name: Object&ClassesConcepts	
Category: **General**	
Title	: Objects and Classes Concepts
Description	: This object explains the syntax of declaring classes in Java and examples
Keyword	: Syntax Declare Java Class Identifier
Language	: English
Category: **Lifecycle**	
Status	: Final
Author	: Suresh Khatiwada
Date	: 2010-12-01
Category: **Technical**	
Location	: Flash Source
Format	: Macromedia Flash
Size	: 64 KB
Requirements	: Macromedia Shockwave
Category: **Educational**	
Resource Type	: Flash
Interact Level	: Medium
Context	: University level
Category: **Flash**	
Copyright	: No
Description	: For Educational Purpose

Table 4.2 Declare Java Class Learning Object Metadata

Object Summary	
Name: CreateJavaObject	
Category: **General**	
Title	: Create Objects in Java
Description	: This object explains the syntax of creating objects in Java and example. It also presents the differences between object declaration and numerical data declaration; object assignment and numerical data assignment; and member access operator in Java
Keyword	: Syntax Create Object Java Class Identifier
Language	: English
Category: **Lifecycle**	
Status	: Final
Author	: Suresh Khatiwada
Date	: 2010-12-01
Category: **Technical**	
Location	: Flash Source
Format	: Macromedia Flash
Size	: 188 KB
Requirements	: Macromedia Shockwave
Category: **Educational**	
Resource Type	: Flash
Interact Level	: Medium
Context	: University level
Category: **Flash**	
Copyright	: No
Description	: For Educational Purpose

Table 4.3 Create Java Object Learning Object Metadata

Two applets are also built as two learning objects working independently with their own objectives: one applet used for showing the animation of the program of declaring class and creating objects in Java, the other used for presenting the animation of accessing object's member program.

There are four panes built for presenting and controlling the animation: source code, performance, output, and control panes. Each pane and its responsibilities are presented on design phase. The source code pane is used for showing and highlighting the source code of the program. The performance pane is responsible for showing the animation of the program. Performance pane is divided into three areas: method area, explanation area, and instance and class area. Method area presents the visualization of the running method. Explanation area is used for extra explaining about the current command. Instance and class area is used to show the animation of objects or class members. The output pane as its name shows the output of the program. The last pane, control pane, is the only pane which students can interact with the applet. Through control pane, students run through the program or run step by step the code sequences.

Now we will look at the implementation of the applets. The source code pane is built using one open source package. The open source package is org.syntax.jedit. This package is being maintained by Tom Bradford and continues to be developed by Slava Pestov and others. For more information about jEdit, we can follow this link http://svntax.iedit.orgl The reason for using jEdit is that it is more suited for editing program source code than

JEditorPane class, because it drops the unnecessary features (images, variable-width lines, and so on) and adds a whole bunch of useful goodies such as:

- more flexible key binding scheme,
- supports macro reco-rders,
- rectangular selection
- bracket highlighting
- syntax highlighting

- command repetition
- block caret can be enabled

It is also faster and doesn't have as many problems. The copyright jEdit of allows everyone "may use and modify this package for any purpose. Redistribution is permitted, in both source and binary form, provided that this notice remains intact in all source distributions of this package" [Pes99] Package theater contains all the classes used for drawing animation on the performance pane. Animation means here any event that includes movement of actors or that is otherwise dependent of time. Examples of animation include moving an actor from one place to another or flashing the colors of an actor as it is introduced to the performance pane.

Two app1cts in this learning object are responsible for showing the animation of creating objects and accessing members. The creating object applet should have the following tasks:

- Presenting the method running: The instruction sequence of the method will be obtained.
- Presenting the creation of new object: The class declaration is consulted to determine the fields so that the correct amount of memory can be allocated to the object. After performing default or explicit initialization of the fields, the constructor is executed. The sequence of instructions of constructors will be obtained from the class declaration. The constructors are often used to initialize fields.
- Presenting the reference to the created object: An arrow representing the reference to the object will be shown.

All of these above tasks are controlled by theater package. Each task is handled by one class. For example:

- MethodStage.java: handle presenting the method running
- Object Stagejava: handle presenting the creation of new object
- RefenceActor.java: handle presenting the reference to the created object.

Output pane is controlled by the class OutputConsole.java. It is just a text area on which the output of a user's program is printed. .

Control pane is a panel containing five buttons: edit, compile, play, step, pause and rewind. Edit button allows us to return to compile the program again. Compile button checks if the program has the main method or not. If there is a main method, it will run the method automatically. If not, the applet will ask users for inputting a valid method call. If the program has main method or users input the right method, users can click the play button for running the program completely or run step by step by clicking the step button. The pause button can be clicked at any time when the program is running. Rewind button is used when users want to start the program again.

Now let's go back to the other applet of this learning object. To run an example of member access, we have to show first the object creation, then the way an object changes its fields or invokes its methods. Therefore, some new are required for this applet:

⌡ Presenting how an object accesses its fields

• Presenting how an object invokes its methods: The class declaration is consulted to obtain the instruction sequence of the method. The instructions will be then executed, where references to identifiers are to identifiers that are declared in the class, while the values themselves are in the fields of the object.

In summary, in the software of the applets, two scenarios are implemented, one for constructing objects and the other for invoking a method on an object. The source code pane will display the source code of the current line being visualized. The execution animation is performed on the Performance pane. When an object is created, a small frame will be shown on the instance area. Each object frame will directly show the fields and their current values.

Now we will describe the creation of an object.

1. A constructor is obtained and invoked, for example: calling a constructor for creating a new Circle

   ```
   new Circle(O, 0, 3);
   ```

2. A frame will appear, labeled with name and class of the object. In the Source Code panel, the source code for the class will appear, and it will scroll until the list of fields is shown. Each field declaration will be animated and it will appear with default value.

3. The constructor invocation Circle(0, 0, 3); will be highlighted and the source code will scroll until the code of the constructor is displayed. On the Performance panel, temporary cells containing the actual arguments will appear, they win be animated as moving into formal parameters and then into the fields in the object icon.
4. The fields of the object win be initialized corresponding with the formal parameters.

4.3 Fields and Methods Learning Object

This learning object is also a combined learning object. It includes three flash files and six applets. Three flash files cover three main topics: access controls, class members, and constructors. Six applets are corresponding with six examples: default access level, private access level, protected access level, public access level, class members and constructors. Because the implementation of the three flash files is similar with the ones above, we will skip describing them. Let's look at the applets. These applets are also similar with the applets presented above. Except class member applet, the others are built without main method, The reason is that after studying the first and second lesson now students already know about how to create new objects and how to access the fields and methods of the object. Because of this, the applets in this learning object allow students to have more interaction with the software. Students are required to input a method for execution. This is also reaJ1y necessary for practicing and getting more attention from students.

For example, the private access level applet has the source code as follows:

```
class Alpha {
private int iamprivatei
private void privateMethod(){
        System. out .println ("privateMethod") i
}
        boolean isEqualTo{Alpha anotherAlpha) {
if(this.iamprivate==anotherAlpha.iamprivate)     // legal
        return true;
else
```

```
                return false;
                 }
        }
        class Beta{
                static void accessMethod(){
                        Alpha a = new Alpha();
                        a. iamprivate = 10;                    // illegal
                        a.privateMethod();                     // illegal
                }
        }
```

Now students can input the method call and see the animation running. The inputting in this context helps students understand the scope of the access level. For example, if the method call is:

```
        new Alpha() .iamprivate = 10
```

The method call will be executed but the result is private field or method cannot be accessed from outside the scope of the class.

Default access level applet, private access level applet, protected access level applet, public access level applet are created with one more function checking the method call if it is allowed to access the members of the object or not.

Class member applet also added one more class for controlling creating static class instance into package "theater". ClassActorjava is the class responsible for that job.

Constructors applet also allows users to input 'constructor call. There are multiple constructors in program. Users can create new objects with any of the given constructors and they can see the differences between constructors."

Each flash or applet is created as an independent learning object which is considered as one component of the Fields and Methods Learning Object. Each of these components has a corresponding metadata file. Because the. metadata has the same format as shown above, we will skip including these metadata here in the report.

4.4 Abstractions, Encapsulation and Information Hiding Learning Object

The aim of this learning object is teaching three very important principles in object oriented programming: abstraction, encapsulation and information hiding. To help students understand principles, the visualization needs to be created. Therefore, in this learning object, the Java applets are not focused because the objective of Java applets is to show how the program is running, how a new object is created, or how the object's state changes. Applets in this situation don't play an important role. This learning object uses flash for creating the visualization of the process of abstraction, encapsulation and information hiding rules. Some applets are still applied just for students seeing the results of program's execution and comparing with the case without encapsulation and without information hiding.

Abstraction process can be created by flash with visual graphics. However) encapsulation and information hiding require students to analyze the examples themselves. Because they are the principles of designing and organizing the program, it is not very necessary to run the program and see the sequence of source code running.

Therefore, in this learning object, the applets just work for illustrations to help learners understand easier. The applets in this learning object don't add more features to the package "theater", it just uses the old ones to bring to learners more attraction to the lesson and illustrate for the lesson.

4.5 Learning Objects and Web-based model

The challenge of making learning resources 'interoperable' across different systems became a major goal. At a higher level, tutors often wished to reuse and repurpose.

All the learning objects are arranged following the schedule of the courseware: starting with Objects and Classes Concepts Learning Object to the last one Abstraction, Encapsulation and Information Hiding Learning Object", All the resources of this study are stored in a database. One learning object -corresponding with one lesson -is a collection of a number of objects. This complex learning object is called a package (IMS Global Learning Consortium, 2000b). Microsoft LMS's concept gives us a correct illustration of package (Figure 4.5) [6].

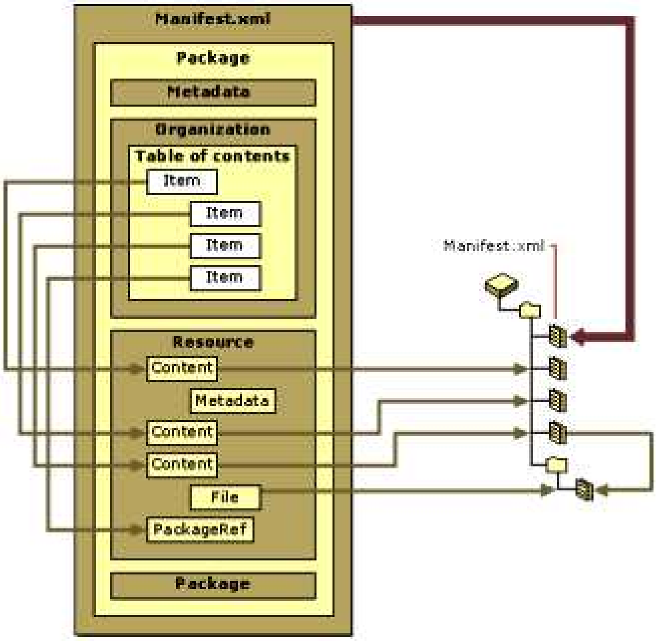

Figure 4. 1 LMS Conceptual Model

IMS has also developed a proposed standard for learning object packaging.

When: students choose a lesson, which corresponds with a package, the LMS knows that it is composed of several objects: some flash files and some java applets.

The course structure is as follows:

1. Lesson 1: Objects and Classes Concepts
    - Objects and Classes Concepts
    - Quiz

2. Lesson 2: Declare Classes and Create Objects
    - Declare Classes
    - Create Objects

- Quiz

3. Lesson 3: Fields and Methods
    - Access Control
    - Class Members
    - Constructors
    - Quiz

4, Lesson 4. Abstraction, Encapsulation and Information Hiding

- Abstraction
- Encapsulation
- Information Hiding
- Quiz

The LMS is responsible for managing all the information about users, their process of learning and their test's result. The implementation of the LMS is out of the scope of this thesis so it will not be presented in detail here. What we need to do is just to integrate the learning objects with the LMS as seen above.

4.6 Learning Object and Q&A model

In web-based learning objects are combined to' be lessons. However, in Q&A model, the learning objects are arranged following topics. Therefore, when a student asks a question, the answer will be linked to a suitable learning object.

In Q&A model, a lot of questions and answers are provided so that when users log into the forum, they can search all questions relating to their interests and follow the link of the answer to see the learning objects.

CHAPTER 5

EXPERIMENT, RESULTS AND EVALUATION

5.1 Comparative Study between Java and C++

5.1.1 Declaring classes in Java

Use the following Java syntax to declare a class in source code:

```
['public']   'class'  class_name
‘{‘
        //  behaviors and state variables are declared between the { and } characters
‘}’
```

Note:

-Specify keyword *class* and follow it with an identifier- represented by *class_name* - that names class.

-The identifier cannot be a reserved word.

-Specify *public* to make the class accessible to all classes in all packages. Do not specify *public* if the class is to be accessed only by classes in the same package.

Example:

// Declare a public class, the source filename must be MyClass. java

```
public class MyClass{
}
```

Another example:

// Declare a nonpublic class called MyClass, the source filenarne is sur.java

```
class MyClass{
 }
```

5.1.2 Declaring classes in C++

Use the following C++ syntax to declare a class in source code:

'class' *class_name*
‘{’
// behaviors and state variables are declared between the { and} characters
'}' [object_name);

where class_name is a name for the class and the optional field *object_name* is one or several valid object identifiers.

Example:

```
//Declare class  MyClass
        class MyClass{
        } ;
Another example
// Declare class MyClass and two objects called classl, class2 of / this MyClass
        c lass MyClass{
        } classl, class2;
```

5.1.3 Creating objects in Java

Use the following syntax to create an object:

class_name object_name = 'new' *constructor ‘;’*

or

class_name object_name ';'

object_name = 'new' *constructor* ’;‘

Explaination:

-The new keyword allocates an object's memory and initializes that memory to default values. An object's field values are stored in memory. The memory for objects in Java is allocated dynamically in the heap, that is, it is dynamic memory allocation.

-Because new is an operator, it takes an operand: constructor, which is the name of a special method that constructs the object. Once new finishes allocating and initializing memory, it cal1s the constructor to perform object initialization (we will explore constructor in next part)

-Once the constructor has finished building the object, creation operator returns the address of that object

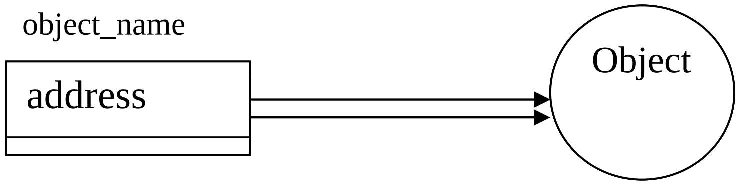

Figure 5.1 A conceptualization of object and its respective reference variable

Example:

```
class Hello{
}                    // HelloDerno class contains main() method to run the program
public class HelloDemo{
          public static void main (String [ ] args)        {
                    Hello objl new Hello();
          }
 }
```

5.1.4 Destroying objects in Java

Java assumes responsibility for destroying created objects. The JVM's (Java Virtual Machine) garbage collector, who runs at periodic intervals, examines each object to see if it is referenced by at least one variable. If not, the garbage collector destroys the object, freeing memory in the process. Therefore, with Java you don't need to care about destroying objects because the garbage collector will do that job for you.

5.1.5 Creating objects in C++

In Java, the only way to create an object is using dynamic allocating with new operator and store a reference to that object a variable. However, in C++, space in memory for objects may be either statically or dynamically allocated. Therefore, there are two ways of creating objects in C++.

Creating object statically

The syntax for creating objects statically is:

class_name object_name ['(' *arguments* ')'] ';'

or

class_name object_name constructor ';‘

Explaination:

-Declare an object of the *class_name* class named *object_name.*

- *arguments* are the values which are passed as the arguments of one constructor of the class.

-First, it allocates an object's memory and initializes that memory to default values. The memory for this object is typically allocated on the stack. The object is of fixed, known size at compile time.

-After finishing allocating and initializing memory, it calls the constructor to perform object initialization (we will explore constructor in next part).

-Once the constructor has finished building the object, it returns the object, and you can start using the object right away.

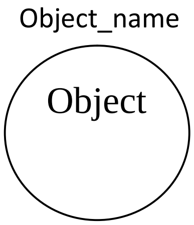

Figure 5.2 The conceptual view of an object which are allocated statically

Example:

```
class Hello{
 };
```

```
void main(){
        // Create an object with default constructor
        Hello obj 1;
        // from now you can use objl
}
```

Creating object dynamically

The syntax for creating objects dynamically is:

class_name ,*, *object-pointer* 'new' *constructor* ‘;.'
or
class_name ,*, *object-pointer* ';'
object-pointer = 'new' *constructor* ';'

Explanation:

-The new keyword allocates an object's memory and initializes that memory to default values. An object's field values are stored in memory. The memory for objects is allocated dynamically in the heap.

-Because new is an operator, it takes an operand: constructor, which is the name of a special method that constructs the object. Once new finishes allocating and initializing memory, it calls the constructor to perform object initialization (we will explore constructor in next part).

-Once the constructor has finished building the object, creation operator new returns a pointer to that object. This is different from Java. In Java the creation operator new will return the reference to that object to object_pointer, but in C++ object_pointer is a pointer to that object.

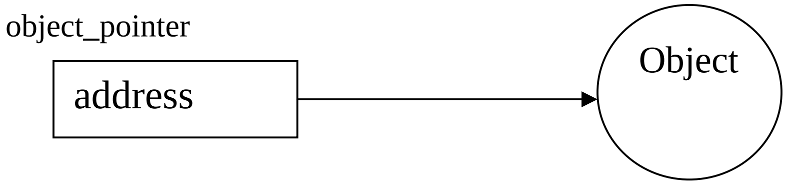

Figure 5.3 A conceptualization of object and its pointer variable

Example:

```
class Hello{
};
// main() function to run the program
void main () {
// create an object and a pointer points to that objects
Hello* obj1;
objl = new Hello();
// from now you can use objl to access the object
}
```

5.1.6 Destroying objects in C++

With static allocated object, the memory where the object is stored is automatically managed by the computer. When a local variable ever goes out of scope, the destructor will be called to release the memory (we will study about destructor in the next part) and the memory that it ties up will be freed up automatically.

However, if you create object dynamically in the heap the compiler has no knowledge of its lifetime, which means the lifetime of that memory can be as long as you choose it isn't determined by scope. Therefore you must determine programmatically when to destroy the object, and then perform the destruction using the *delete* keyword. The syntax for destroying dynamic allocated object

```
'delete '  object_pointer;
```

Explanation:

-First delete keyword calls the destructor of the object.

-Then it releases the storage for the dynamic allocated object which is pointed by new *object_pointer*

-*delete* can be called only for an object created by *new*

-*delete* must be applied to release any storage that is allocated with new. If you forget to use *delete*, that storage remains unavailable, it is called *memory leak.*

Example:

Return to the above example of creating dynamic object, we haven't destroyed the object. The correct program is following:

```
class Hello{
};
//main() function to run the program
void main () {
         // create an object and a pointer points to that object
        Hello* obj1;
        objl = new Hello();
        // from now you can use objl to access the object
        // ...
        delete objl;
}
```

5.1.7 Access Level Control in Java

Access specifiers

-It can be optionally declared a field and a method with an access specifier keyword: public, private or protected.

-The access specifier determines how accessible the field/method is to code in other classes. Access ranges from totally accessible to totally inaccessible.

The following chart shows the access level permitted by each speedier.

Specifier	**Class**	**Subclass**	**Package**	**All class**
Private	X			
Protected	X	X*	X	
Public	X	X	X	X
Default	X		X	

(X*: this access has a special caveat)

Default access level

-If it is not declared a field/method with an access specifier keyword, Java assigns the field/method a default access level

-The default access level makes the field/method accessible within its c1ass and to all classes within the same package.

5.1.8 Access level control in C++

Access specifiers

-You can optionally declare a block of fields and functions with an access specifier keyword: *public, private,* or *protected*.

-The access specifier determines how accessibiethe block of fields and functions is to code in other classes. Access ranges totally accessible to totally inaccessible.

The following chart shows the access_level permitted by each specifier.

Specifier	**Class**	**Subclass**	**All class**
Private	X		
Protected	X	X	
Public	X	X	X

Private or Default

-If you do not declare a block of fields and functions with an access specifier keyword, C++ assigns that block the private access level

-In *private* access level, only code contained in its class can access those fields/functions of the block.

5.1.9 Class Field and Class Method in Java

In Java, use *static* keyword to declare a *class field/method.* Class fields are associated with classes --not objects. When modified by a class's code, the class (as well as any created objects) sees the change.

A *class method* is a method that accesses the class fields declared in its class. A static class function is a function that accesses the static class variables declared in its class. It cannot access non-static data or functions.

Java Program:

```
    class Employee{
             static int numEmployees;
             static void setEmployeeCount (int numEmp) {
                     nurnErnployees = nurnEmp;          // legal
                     this . numEmployees = numEmp;
                     // Compile because this refers to the current instance, and a class method is
                        not associated with any class instance
             }
    }
```

To call a class fieldJmethod from another class's method:

```
    public class UseEmployee{
             public static void main (String [] args) {
                     Employee.numEmployees = 10;
                     Employee.setEmployeeCount (10);
             }
    }
```

C++ Program:

```
    class Employee{
```

```
                public:
                        static int numEmployees;
                        static void setErnployeeCount (int numEmp) {
                                numEmployees =  numEmp;                          // legal
                                ,this->numemp;
                                 // Campile Error because this refers to the current class instance,
                                and a class method is not associated with nay class instance
                        }
                };
```

To call a static class variable/function from another class's function:

```
         #include "Employee.h"
        class UseErnployee{
                void useStatic (){
                        Employee :: numErnployees = 10;
                        Employee: :setErnployeeCount (10);
                }
        }
```

5.1.10 Constructors in Java and C++

A *constructor* is a special method called by the creation operator ‘new’ to initialize an object. A constructor must be given the same name as the class in which it is A constructor must not be declared with a return data type.

The syntax to declare a constructor in Java is as following:

Class_name ' (' *parameter_list*   ') ’

The syntax to declare a constructor in C++ is as following:

class_name ‘ (‘ *parameter_list* ‘) ’

Example:

```
class Hello {
};
```

Hello looks empty, but in reality, the compiler sees *Hello* with a default constructor and a copy constructor .

```
class Hello{
private:
            int i;
           Hello () {
           }
           Hello(const Hello& h)  {
           i = h.i;
} ;
```

The compiler only automatically creates a default constructor and a copy constructor when no other constructors are declared.

5.1.11 Destructor in Java and C++

In Java, we don't need to make destructor because the JVM's garbage collector will assumes responsibility for destroying created objects.

In C++, an object's destructor is called whenever an object goes out of scope or when the delete operator is called on a pointer to the object. The destructor reuse memory for new objects before system reclaims object's memory.

-Same name as the class name prefixed by a tilde (~).

-It has no return type specified, no arguments and thus cannot be overloaded.

Example:

```
class Circle{
private:
		int *points;
	public:
		// Destructor
		~Circle () {
			delete points;
		}
	}
```

5.1.12 Abstraction, Encapsulation and Information Hiding

Abstraction

In the process of abstraction in object-oriented design, you will end up with abstractions in the form of objects, type, attributes, and It involves the following three tasks:

-dividing the problem domain into types of objects,

-modeling the relationships between the types, and

-modeling the attributes and behaviors of each types.

These tasks are not listed in any particular order.

Now let's practice with example to get the clear understanding of abstraction process.

Example:

Design a virtual café

Dividing the problem domain into types of objects

You identify the fundamental objects of the problem domain, the thing involved. You then classify the objects into identifying groups of objects that have common characteristics and behaviors. . .

Example: In what objects do you see? You may have different coffee cups in your distinct objects--but you place all 100 of them into the "coffee category. The other types of objects can be guests, tables, chairs, coffee, etc. You must decide objects you think be important to group .into categories. For example, the drawings hanging on the walls are not so important that they should be a type. Then the types of objects involved should be what you most concern yourself with in your solution: guest, tables, chairs, coffee cups, coffee, milk, sugar, and the itself.

Modeling the relationships between the types

As you divide the problem domain into types, you will to some degree model the relationships between the types as well. Objects can have three kinds of relationships:

-the *has-a relationship:* one type of object contains another

-the *relationship:* one type of object is a more specific version of a general type

-the *uses-a relationship:* one type of object uses another type of object.

Example:

The *has-a relationship:* A coffee cup has coffee or empty, tables have different numbers of chairs.

The *relationship:* a coffee cup is a cup; a tea cup is also a cup.

The *relationship:* guest sit on chairs at tables, guest buy cups of coffee, add milk or sugar to them.

With modeling the relationships between the types, you can realize more new types which are necessary also.

Modeling the attributes and behaviors of each types

The *attributes* of a type define the nature of the state of objects of that type.

The *behaviors* of a type define interactions between objects and the actions that objects take as a result.

You only focus on which attributes and behaviors are important to your particular solution.

Example:

There are many ways to describe a coffee cup, for instance,

-color

-current amount of coffee

-Size

-shape

-whether or not it has a handle

-material

-place of manufacture

You must choose the attributes that are most important, and model only those. For example, *only* the color, the size, and the current amount of coffee are the only attributes that are important in your solution.

Example: A coffee cup can do many things, for example;

-accept coffee

-release one sip of coffee

-spill

-move from one part of the to another

-shatter into pieces

You will not model all the possible behaviors a coffee cup can exhibit, only those that matter to your particular solution. For example, only accepting coffee, releasing coffee one sip, and spilling are the only behaviors that matter in your solution. When you model the coffee cup's behaviors, you should notice the ways it interacts with other objects.

Encapsulation and Information Hiding

A fundamental object-oriented concept is encapsulation, the bundling of data that represents the state of an object together with the code responsible for manipulating that data. Encapsulation is the basic process of forming objects. A class encapsulates the instance variables and methods that define an object.

Encapsulation rule 1:

Place data and the operations that perform on that data in the same class. Let us get back to the virtual example and see what encapsulation means in this case. After the process of abstractions, we identify the types of objects, the relationships between them, and their attributes and behaviors. Now look at coffee cup type, the chosen attributes are color, size, and current amount of coffee. The behaviors are accepting coffee, releasing coffee one sip, and spilling. The encapsulation here means enclosing all parts of coffee cup type within a class called *CoffeeCup*.

Encapsulation rule 2: Use responsibility-driven design to determine the grouping of and operations into classes.

C++ provides three ways returning types with functions; they are pointer, reference and value. You should be with returning by pointer and reference because they can allow clients access internal data ofthe class. Look at the above example, *getPositionl (int)* and *getPosition2 (int)* expose the actual internal data items .' maintained by Route. Clients can obtain a reference to any of Position objects along the route by *getPositionl (int)* or they can obtain a pointer to any Position objects by *getposition2 (int)* and freely change the state of that Position object. The *getPosition (int)* method is returning an equivalent Position object rather than the internal Position object. *getposition (int)* didn't expose the internal data, and it satisfies information hiding principles:

Information hiding rule 1: **Don't expose data items**

This enables objects to have responsibility for managing their own state. For example: with the above program, the clients can set value arbitrarily to any field of the objects.

```
CoffeeCup cup = new CoffeeCup();
cup. size = -1;
```

The size of a coffee cup must be a positive number. This assigned value is not acceptable. Therefore, an object's internal state cannot be directly manipulated by other objects, but they must send the object a by way of its interface. The object receives the messages and decides whether or not to change state.

There are degrees of information hiding. C++ and Java provides access levels to attach to the variables and methods of a class. We should normally declare as public only those methods that other objects need to invoke, and declare the data as private, so that the object's instance methods with have sole responsibility for maintaining the object's state.

Information hiding rule 2: Don't expose implementation details of a class
Go one step further when hiding design decisions concerning internal data. Don't allow clients to know or invisibly affect a class's implementation details. For example, a first-cut design of class Position and Route looks like the program given in example :

Though class Route encapsulates the array, the method getPositions ( ) returns the array of Position objects and the clients can freely change all Position objects. The clients shouldn't get the whole position array, but each position object's state, therefore we should remove this method.

5.2 Result

In addition to running a compiled Java program, computers running Java applications must also run the Java Virtual Machine JVM, while compiled C++ programs can be run without external applications. Early versions of Java were significantly outperformed by statically compiled languages such as C++. This is because the program statements of these two closely related languages may compile to a few machine instructions with C++, while compiling into several byte codes involving several machine instructions each when interpreted by a JVM. While C++ is faster than Java in most cases, there are several studies of mostly numerical benchmarks, which argue that Java could potentially outperform C++ in some circumstances. However, it was shown that numerical (micro-) benchmarks are not appropriate for evaluation of languages as compiler is able to optimize both cases equally, or eliminate benchmarked code entirely. If referring to a real world program, Java would suffer because of a number of reasons:

- All objects are allocated on the heap. For functions using small objects this can result in huge performance degradation as stack allocation costs essentially zero.
- Methods are by-default virtual. This increases memory usage as much as several times for small objects because of virtual function tables. Also, it induces performance penalty, because JIT compiler has to do additional optimization passes even for de-virtualization of small functions.

- A lot of casting required even using standard containers induces performance penalty as it is needed to walk through entire inheritance hierarchy.
- Virtual Java Machine increases memory usage even further, thus reducing memory locality and increasing chances of cache misses and therefore slowdown of the program.
- Lack of access to low level details does not allow developer to better optimize the program where the compiler is unable to do so.

It is argued, however, that compared to Java, C++ also has a number of downsides:

- Pointers make optimization difficult since they may point to arbitrary data. However this is obsoleted as new compilers introduced *strict-aliasing* rule and because of support of the C99 keyword *restrict*.
- Java garbage collection may have better cache coherence than usual usage of *malloc*/*new* for memory allocation as its allocations are generally made sequentially. Nevertheless, arguments exist that both allocators equally fragment the heap and no one exhibits better cache locality.
- Run-time compilation can potentially use additional information available at run-time to optimise code more effectively, such as knowing what processor the code will be executed on. However this claim is effectively made obsolete as most state-of-the-art C++ compilers generate multiple code paths to employ full computational abilities of the given system.

Furthermore, it is arguable that time spent debugging more complex C++ code could just as easily be spent optimizing existing Java code. Of course this is subject to other concerns such as how far it is possible to optimize bottleneck code in a given program. Ultimately, for processor-hungry cases such as video rendering, C++ has the direct access to hardware (being OS-native) and as such, will always be able to outperform Java given a particular hardware spec.

CHAPTER 5

CONCLUSION AND FURTHER RECOMMENDATION

5.1 Conclusion

The objective of this thesis is to analyze the differences of basic concepts used in C++ and Java. It builds a Web-based model using learning objects for teaching and learning some concepts and principle related to classes and instances in object-oriented programming in two languages C++ and Java. The content of teaching includes objects and classes concepts, how to declare classes and create objects, how to declare fields and methods of class, how to control access level controls, and three important principles of designing object-orientation: abstraction, encapsulation and information hiding in C++ and Java.

To implement the courseware, the approach is to create high visualization and animation with clear explanation. A series of Macromedia Flash file or Java applet is expressed as a learning object covering one small topic so that it can be reused in a different environment or other courses. Macromedia Flash with its benefits of highly visual graphics is used for creating the explanation theory. Java applets are the best choice for creating animation of the sequence of source code. The student can interact with the applets and see the effect of the changes. The main animation of the applets is showing object creation, object destruction, object's member access, and object's state changes. Animation, explanation and highlighting source code are implemented in parallel with flexible interactive manner. This helps student learn faster and more focusing on the lessons. Students can verify their understanding by testing with a quiz at the end of lesson.

Each lesson corresponds to one package which is a collection of learning objects built in Macromedia Flash or Java applet. The learning objects are stored in database of a Learning Content Management System which is used to simplify the administration of learning programs and learning object. As a result, a Web-based model is built.

In summary, the thesis has completed the following tasks:

- Creating all learning objects for teaching all the topics proposed above

- Embedding learning objects to Web-based model
- Creating questions and answers related to the learning objects for Q&A model
- Analyzing the difference in basic principle of Java and C++

5.2 Further Recommendation

The courseware of teaching and learning classes and instances in C++ and Java is quite completed. However, in order to have a more successful courseware, it is necessary to and some more practices. Students should be able to practice their comprehension with some exercise or other interactions. It should have some means for students to express their feedback such as questionnaires. Because based on the feedbacks, the developers can improve the model and enhance the course content.

Another recommendation for future work is that learning objects should be developed in several other techniques. In this thesis, learning objects are built by Java applets and Macromedia Flash only. Even in this flash files, only text effects are used to create animation and visualizations. They can be created with sound effects, video and other dynamic techniques.

Because of the limitation on time, this study includes some specific concepts and principles of object-oriented paradigm. The complete course should be implemented.

References

[1] Nguyen Dieu Huong, (2005) *A web-based model using learning objects for teaching and learning classes and instances in C++ and Java*, School of Engineering and Technology, Pathum Thani, Thailand : Asian Institute of Technology

[2] Beck, Robert J. (2009), *"What Are Learning Objects?", Learning Objects, Center for International Education, University of Wisconsin-Milwaukee,*, retrieved 2009-10-23.

[3] David A. Wiley, II, Utah State University Connecting learning objects to instructional design theory: A definition, a metaphor, and a taxonomy

[4] Mohan P., Brooks, C. *Engineering a Future for Web-based Learning Objects,* Department of Mathematics and Computer Science, The University of the West Indies, St. Augustine, Trinidad and Tobago

[5] Dorian Stoilescu, (2008) *Modalities of Using Learning Objects for Intelligent Agents in Learning*, Ontario Institute for Studies in Education, University of Toronto, Toronto, Canada

[6] Pithamber R. Polsani, *Use and Abuse of Reusable Learning Objects*, Learning Technology Center, University of Arizona, USA

[7] Batanov, D. N., Dimmitt, N. J., Chookottikull, W. (2000) Q&A *Teaching/Learning Model as a new basis for Developing Educational Software*, ASEE/IEEE Frontiers in Education Conference.

[8] Casey, D. (1998) *Learning "From" or "Through" the Web: Models of Web-based Education*, Monash University.

[9] Downes, S. (2001) *Learning Objects: Resources for Distance Education Worldwide*, International Review of Research in Open and Distance Learning.

[10] Northrup, Pamela (2007) (Book), *Learning Objects for Instruction: Design and Evaluation*, USA: Information Science Publishing.

[11] Rehak, Daniel R.; Mason, Robin (2003), "Engaging with the Learning Object Economy", in Littlejohn, Allison, *Reusing Online Resources: A Sustainable Approach to E-Learning*, London: Kogan Page, pp. 22–30, ISBN 9780749439491.

[12] Chiappe, Andres.; Segovia, Yasbley; Rincon, Yadira (2007), "Toward an instructional design model based on learning objects", in Boston, Springer, *Educational Technology Research and Development*, Boston

[13] Jennie L. Mitchell and Nicholas Farha, LEarning Object Metadata: Use and Discovery.

[14] Nichani, M. (2001) LCMS = LMS + CMS (RLOS)

[15] *Svetlana Kouznetsova, Ph.D, Department of Computing Science,Sam Houston State University,* USING BLUEJ AND BLACKJACK TO TEACH OBJECTORIENTED DESIGN CONCEPTS IN CS1

[16] David J Barnes, Michael Kölling, (1997). Teaching OOP - A Practical Approach using BlueJ

[17] Kolb, D. A., (1976). *Learning style Inventory Technical Manual*. McBeer, Boston, MA

[18] Maddock, P. (2003). *Enhancing the Learner Experience*. The Networker Article Archive

[19] Gillani, B., and Relan, A., (1997). *Incorporating Interactivity and Multimedia into Web-based Instruction,* Englewood Cliffs, New Jersey

[20] Guzdial, M. (1995) Centralized Mindset*: A Student Problem with Object-Oriented Programming.* Proceeding of the 26th SIGCSE Technical Symposium on Computer Science Education

[21] Cornelius, S. (2000) *Learning Online: Models and Styles*, *Online Tutoring E-book. Online Tutoring Skills e-Workshop.*

[22] Becker, B. W. (2000) Pedagogies for CS1: A Survey of Java Textbooks Manscript

[23] Carlos Argelio Arévalo Mercado, *The Effect of Learning Objects on a C++ Programming Lesson,* Autonomous University of Aguascalientes, México

[24] Norm Friesen, (2003), *Best Practice for Learning Object Metadata*

[25] Ásrún Matthíasdóttir (2004), *Learning objects in a multimedia interactive environment,* The Codewitz project