Development of Intelligent Tutoring System Framework: Using Scaffolding Teaching Strategy

Dissertation Report

Submitted in partial fulfillment of the requirements

of the degree of

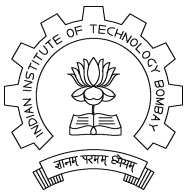
Master of Technology

by

Chandra Pal Singh
Roll No:10305075

Supervisor

Prof. Sridhar Iyer



Department of Computer Science and Engineering
Indian Institute of Technology Bombay
June 2012

Dissertation Approval Certificate

This dissertation entitled **"Development of Intelligent Tutoring System Framework: Using Scaffolding Teaching Strategy"**, submitted by **Chandra Pal Singh** is approved for the degree of **Master of Technology** in **Computer Science and Engineering** from **Indian Institute of Technology, Bombay**.

Prof. Purushottam Kulkarni
Dept. of CSE, IIT Bombay
Internal Examiner

Prof. Vijay Raisinghani
Dean and HoD of NMIMS, Mumbai
External Examiner

Prof. Sridhar Iyer
Dept. of CSE, IIT Bombay
Supervisor

Prof. Abhay Karandikar
Dept. of EE, IIT Bombay
Chairperson

Place: IIT Bombay, Mumbai
Date: 26^{th} June, 2012

Declaration

I declare that this written submission represents my ideas in my own words and where others' ideas or words have been included, I have adequately cited and referenced the original sources. I also declare that I have adhered to all principles of academic honesty and integrity and have not misrepresented or fabricated or falsified any idea/data/fact/source in my submission. I understand that any violation of the above will be cause for disciplinary action by the Institute and can also evoke penal action from the sources which have thus not been properly cited or from whom proper permission has not been taken when needed.

_____
(Signature)

_____
(Name of Student)

_____
(Roll No.)

Date:_____

Abstract

Over the past decade, distance education programs have developed at an extraordinary rate. Web-based distance education has emerged in higher education as a means for providing a variety of educational opportunities to a diverse community of individuals. Teaching programming to a student is always a challenge for the teacher. Teachers used many teaching strategies to teach a student. But when it comes to a computer based system it becomes very difficult. So delivering an effective Intelligent Tutoring System (ITS) is a challenge. While many approaches exist for solving this problem, this project is to developed an ITS framework to incorporate more than one teaching strategy at one place. The purpose of this project was to develop an interactive and adaptive (Intelligent Tutoring System framework) for the computer science under-graduate 1st year students for teaching programming languages.

In this effort design and architecture of our ITS framework is explained. We build an ITS from this framework. ITSs are required to follow at least one teaching-learning strategy to achieve the learning objectives. Our ITS follows four teaching strategies. Scaffolding teaching strategy is one of these strategies. Scaffolding is a teaching-learning strategy in which the instructor provides a temporary support to the student so that student will be able to do the question which he can't do without help. This ITS can be used to teach multiple subjects.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Learners usually have to attend classes physically to learn subject. Only a single instructor handles the entire class. Now a days as population is increasing so as number of students in single class. Instructor may not be able to take care of each individual student. Then teaching subject to students is a tough challenge for teacher. These day's technology is improving rapidly so integration of technology for teaching programming also increasing. As a result on-line learning and distance education systems are gaining attention which can be available at any place and at any time. So the education system should provide support to such place and time independent learning and treating each learner with equal care. In this effort we will discuss our approach to make a framework for developing an Intelligent Tutoring System (ITS). This chapter has the introduction of ITS, strategies and of ITS framework. This chapter also includes work summary and outline of our effort.

1.1 Intelligent Tutoring System (ITS)

Intelligent tutoring systems (ITSs) are computer based programs which knows two things what to teach and how to teach. ITSs has wide area of scope and has been used in many domain such as in traditional education, distance learning and training. Education research has shown that the best learning environment is one-to-one with an expert human teacher. Bloom (Bloom, 1984) proposed a theory that 98% students with private tutor performed better than classroom students. In classroom teachers are forced to suit their teaching to the average student. High achieving students may become bored due to slow pace and lack of challenges. So these students do not get what they deserve? And in second part low achieving students find the work difficult and never receive the level of attention they require. It means one-to-one teaching performed better than collaborative teaching. So the primary advantage of ITS is to provide one-to-one tutoring. There are so many strategies which work on one-to-one teaching methodology. Early computer assisted education tools were called Computer-Aided Instruction (CAI) systems. One of the early tutoring systems is the system by Uhr in the year 1969. This system generate problems on arithmetic and questions on vocabulary. This system has limitation that this had no adaptation logic. Adaptation logic means this system doesn't change its behavior according to students

response. Therefore it was not very successful. There is a difference between ITS and CAI , an ITS can provide individual attention to the student but CAI can't due to absence of adaptation logic in CIA. ITSs provide the feedback according to cognitive profile of individual student. Now a days so many ITSs are present like SQL tutor, C++ tutor, Algebra tutor, Auto tutor, Smart-tutor, Lisp tutor,Wayang Outpost tutor, thermo-tutor etc. Most of them has 4 main components domain model, student model, expert model and teaching model. Basically ITSs are of 2 types[1].

- **Standalone:** The ITS software is installed in learners system and learner can access the ITS when he runs the software.
- **Web-based:** The ITS software is installed and runs continuously on a system, which functions as server. The learners can access the ITS through Internet.

The web-based ITSs have following advantages over stand alone ITS.

- learners are not constrained to use specific machines in their schools, and can access ITS from any location and at any time.
- Distributing software to learners and hardware/software compatibility problems are minimized.
- Updated versions of the ITSs will be available to learners.

ITSs are being used to teach various subjects. ITSs have been developed in geography, circuits, medical diagnosis, computer programming, mathematics, physics, genetics, chemistry etc, subjects to help learners learn various subjects. All these ITSs are made to teach only one specific subject. Most of these ITSs teach concepts directly. Teaching the concepts directly makes the learner passive, the learner will memorize the concepts and he is not exposed to the real world problems. The learner should be able to discover the concepts and principles through interaction which increases the interest of learner, helps in remembering concepts for long time and teaches how to learn. Some subject or topics can be taught only with some specific instruction strategy. So an efficient ITS should worked on good teaching strategies. Above listed ITSs either uses only one teaching strategy or no strategy for all topics of subjects. Teaching with only one strategy is not a good approach as learner's performance can be very poor with one teaching strategy and very well with other teaching strategy. That's why these ITSs are not efficient. Some of teaching strategies are listed in next Section.

1.2 ITS framework & Strategies

Goal of this research project is to provide a generic framework which can support more than one teaching strategies and independent from subject domain. Then building an ITS from this framework that provides responsive and interactive teaching facilities for learners, tracks their progress , assesses their performance, sequences the curriculum and helps the learners to improve

without human instructor intervention.
For achieving this goal we have decided a pattern which will be independent from subject domain called multiple choice questions(MCQs). After that we have studied different teaching strategies and picked 4 different strategies which can support our architecture. These strategies are

1. Socratic Questioning
2. Scaffolding teaching strategy
3. Guided discovery and
4. Game based Learning.

1.3 Work Summary

We started working on ITS from reading about the origin of ITSs and did brief literature survey. We study some good Tutoring system like SmartTutor, Zosmat and Wayang Outpost. Then we started studying about the teaching strategies and the methods by which instructor teach the students. After this we decided to work on Scaffolding teaching strategy and did literature survey on that.
We were four people in this research project. When we finished literature survey about ITSs and teaching strategies, we decided to implement the ITS framework. ITS framework can be used to make an ITS for any specific subject and teaching strategy. New teaching strategies can be embedded easily in ITS framework. Each of us chooses one specific teaching strategy. I have chosen Scaffolding and did literature survey on it.
Finally we developed an ITS Framework architecture and completed the implementation. We build an ITS with the help of this framework. Current ITS is working on four strategies as follows.

1. Scaffolding : This strategy is implemented by me. We will have detailed discussion on Scaffolding in Chapter 3.
2. Guided discovery: This strategy is implemented by RajaShekhar. Details of this strategy can be available at [24].
3. Socratic Questioning: This strategy is implemented by Vikash Kumar. Details of this strategy can be available at [23].
4. Game based learning: This strategy is implemented by Praveen Dhanala. Details of this strategy can be available at [25].

In Scaffolding a proper support is provided to the student at each step in learning. In our system support is provided in form of hints. After implementation of the scaffolding strategy, questions from different sources are found and appropriate hints are added to all questions. We make a course for students to learn with the help of these questions in ITS. After that we present the demonstration of ITS with scaffolding.

1.4 Screenshot of Homepage of our ITS

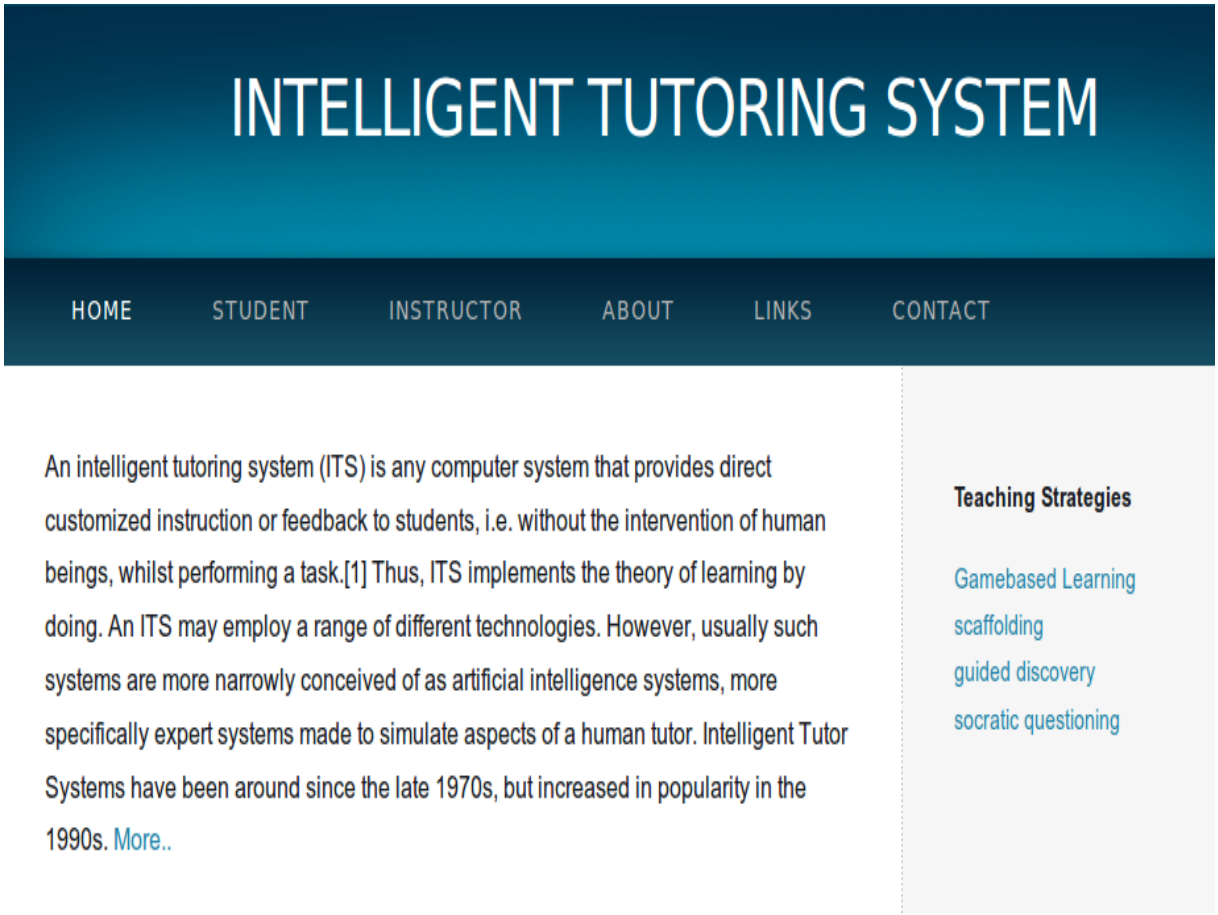


Figure 1.1: Home Page of Our ITS.

1.5 Outline

Chapter 2 and 3 will have discussion about the literature survey on ITSs and on scaffolding strategy respectively. **Chapter 4** will have discussion about the ITS Framework which we are developing. **In Chapter 5** implementation of scaffolding strategy will be explained. We build an individual ITS for only Scaffolding. This chapter will have information of source code, information about modules and screen-shots of tables. **In Chapter 6** we will explain the format of questions used in ITS for scaffolding. In **Chapter 7**, Integration of our system will be explained. **Chapter 8** will have limitation of our system and conclusion of this project. **In Chapter 9** future work needed for scaffolding and for our ITS will be explained.

Chapter 2

Literature Survey

2.1 Intelligent Tutoring System (ITS)

Computer-based intelligent tutoring systems (ITS) provide one promising option for helping students prepare for high stakes assessments. Research on intelligent tutoring systems has clearly shown that users of tutoring software can make rapid progress and dramatically improve their performance in specific content areas. Specifically, studies of the Carnegie Tutor (CMU) for algebra and the AnimalWatch tutor (UMass-Amherst) for arithmetic indicate that student users successfully master specific skills and that their attitudes towards math become more positive as a result of working with the software[22]. Below are some popular definition of ITSs.

- The wikipedia definition of ITS is **"**An intelligent tutoring system (ITS) is any computer system that provides direct customized instruction or feedback to learners, i.e. without the intervention of human beings, whilst performing a task[7]**"**.
- Tom Murray defines an ITS as **"**Intelligent Tutoring Systems are computer-based instructional systems with models of instructional content that specify **what to** teach, and teaching strategies that specify **how to** teach[8].**"**
- The Association for the Advancement of Artificial Intelligence defines ITS as **"**Broadly defined, an intelligent tutoring system is educational software containing an artificial intelligence component. The software tracks learners' work, tailoring feedback and hints along the way. By collecting information on a particular learner's performance, the software can make inferences about strengths and weaknesses, and can suggest additional work[5].**"**

2.1.1 General Component

We studied many ITSs and found nearly all of them have four common components. According to Linda Lawson[21] components that represent student tutoring and communication knowledge are outlined below.

- **Domain knowledge** represents expert knowledge, or how experts perform in the domain. It might include definitions, processes, or skills needed to multiply numbers (Animal-Watch), generate algebra equations (PAT), or administer medications for an arrhythmia (Cardiac Tutor).

- **Student knowledge** represents students' knowledge of the domain. It contains both stereotypic student knowledge of the domain (typical student skills) and information about the current student (e.g., possible misconceptions, time spent on problems, hints used, correct answers, and preferred learning style).

- **Tutoring knowledge** represents teaching strategies, (examples, and analogies) and includes methods for providing appropriate feedback.

- **User Interface or communication knowledge** represents methods for communicating between students and computers (graphical interfaces, animated agents, or dialogue mechanisms). It includes managing communication, discussing student reasoning, sketching graphics to illustrate a point, showing or detecting emotion, and explaining how conclusions were reached.

The combination of these four components are available in all ITSs.

2.1.2 Specific Case Studies

We studied various ITSs and their architectures for example wayang outpost[15], zosmat[17] and smartTutor[11]. Here we will discuss two of them.

SMARTTUTOR

SmartTutor is a subsystem within the SOUL system. The SOUL platform is developed by the SOUL project team. It is an online learning platform. The system architecture is shown in Figure-2.1. This architecture is called the PowerEdBuilder. The secure e-course exchange (eCX) is used to provide a secure layer for protecting the copyrighted materials. The communication and searching infrastructure (CSI) is used to provide efficient communication channels among administrators, instructors, and learners. The content engineering system (CES) is mainly used by instructors to create or provide online course materials and launch online courses. The e-institute is the administration center of the platform which monitor all the other parts of system. The e-learning platform is a platform where students will interact with system for learning and downloading relevant learning materials. SmartTutor is the core part in the e-learning platform. SmartTutor is developed for tutoring of mathematics. This system can be used for either individual learning or for classroom with guidance by teacher. This paper[11] claims that this feature of SMARTTUTOR made this different from others. While teaching a topic in the classroom, a human tutor do many steps. He explain the core knowledge on the topic then tells how this knowledge is applied to solve problems. he supports student in solving the problems. He analyses student's solution and explain errors and suggest next activity to the student to do. There can

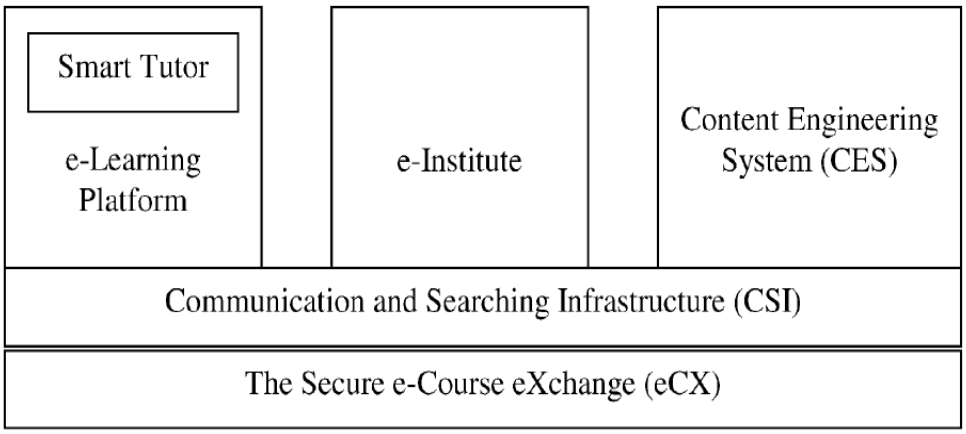


Figure 2.1: An overview of PowerBuilder source: (B. Cheung, March 2002)[11] .

be other steps also which vary from teacher to teacher. SmartTutor is made to follow all these steps efficiently. SMARTTUTOR is designed to explain the core knowledge on the topic, tell which kinds of knowledge can be applied to solve problems in the given area, provide examples of problem solving (by solving several problems and explaining each step of the solution) with examples module, suggest appropriate test paper from past performance, and suggest the next most efficient activity for the student to do. SMARTTUTOR achieves the flexibility and generality of a tutor in ways that is required to individual students' needs and abilities.
The main architecture of SMARTTUTOR consists of six components: Course manager, question bank, student model, content structure, expert model, and user interface (UI). As we can see in Figure-2.2. The Architecture of SmartTutor is very simple.

- **Course manager:** The course manager is the control center of SmartTutor. It analyses the content structure of each courses. It invokes the expert model to create a planned course using the domain knowledge provided by human tutor and developer system, for each student, or to give individualized instructions and suggestions during students' learning process. It delivers all the course via the UI.

- **Question bank :**Question bank is a warehouse of questions that is used in generation of test. It is also an essential part of an online course system because only through the tests generated from question bank, SmartTutor can get learning status of each student and record them in student model for future use.

- **Student model:** This component is responsible for all student related information. This component have student's personal information as well as professional information. This component keep track of student learning process. Student's activities are stored in this component.

- **Expert model :**The Expert Model is a one of the core components of SmartTutor. Two human tutor like modules, Advisor and Planner, provide main intelligent functions of Smart-

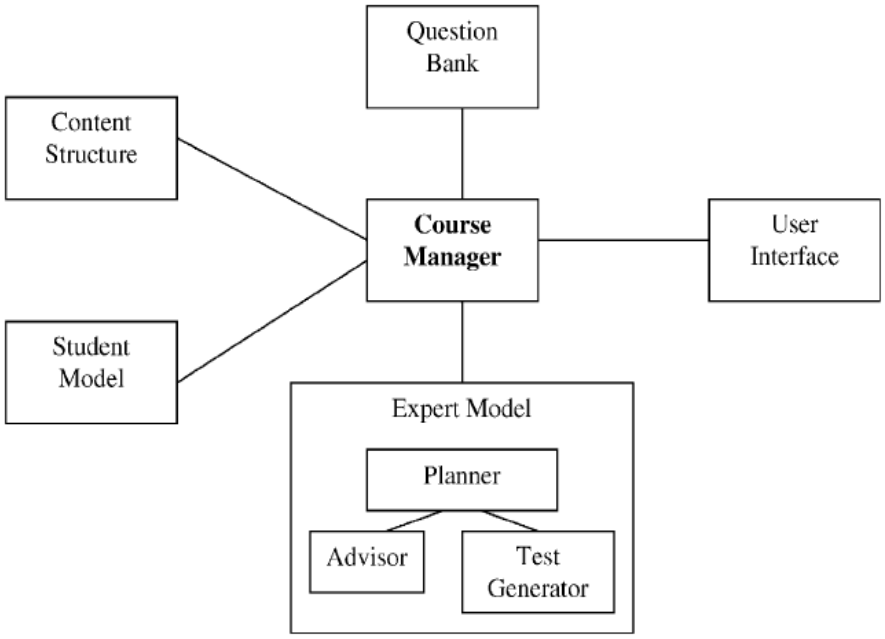


Figure 2.2: Architecture of SmartTutor .

Tutor. **Advisor** focuses on test results and assessment analysis. **Planner**, on the other hand, is like a resource manager and will provide more resource-oriented but personalized planning for the individual students based on the advices from the Advisor. Both the Advisor and Planner need the support from another module, called Test Generator.

- **Content structure :** In SmartTutor content structure consists of various learning materials only. Content Structure is shown in Figure-2.2. In this figure all arrows shows the dependency of one topic/chapter on other topic/chapter. A content graph is maintained by ITS for each student from this content structuer and whole course is delivered to student according to that content graph.
- **User Interface :** SmartTutor provide a web-base GUI to the user. The work of GUI is displaying results of other SmartTutor components.

WAYANG OUTPOST

A brief summary of Wayang outpost is given here. This ITS is discussed in details in Vikash's dissertation[23]. Wayang Outpost is a web-based tutoring system that teaches students to solve geometry SAT problems. Wayang Outpost uses multimedia to help students in their problem solving process, directing their attention, animating parts of the solution, and emphasizing concepts with sound. Multimedia is also used to motivate students, by adding a video-game style

to the tutor. So in other words this paper[10] describes "Wayang Outpost", an Intelligent Tutoring System to prepare students for high stakes achievement test. This tutoring system has improvement to others as it used multimedia content like audio and video to provide the instruction. Wayang contains short-term transfer problems as well as long-term transfer problems. Short-term transfer problems means variations of the operands of an previous high stakes exam problem and minor changes in figures to assess learning. Long-term transfer problems present real-world math problems following a story line . Wayang outpost created two kinds of hints, one based on an analytical approach, the second based on a visual estimations approach . It also provide other type of hint as required by student. Student can adopt the approach according to his simplicity.

2.2 Strategies

This section includes the teaching strategies which are used by our ITS to teach learner.

2.2.1 Scaffolding

Scaffolding is well known term in building construction. When we hear the word "scaffolding" we first think of new office buildings going up, or else aging buildings needing repair. The structure that is built outside a tall building to facilitate workers to climb up and repair building is referred to as **Scaffolding**. Scaffoldong appear to be an external skeleton of the building it surrounds when viewed from ground, but actually it has nothing to do with supporting the actual weight of the building it surrounds. Instead, what is evident is the short-lived nature of it's framework, individual pieces of which are designed to disassemble quickly. Person who passes frequently can see the changes in vertical and lateral movement. One day the scaffolding spreads north or retreats east; the next day, it stretches higher or drops lower. Scaffolding means; as soon as it's no longer needed, it disappears.
Instructional scaffolding can be defined similarly. Scaffolding in context of educational is a process by which a teacher provides students with a temporary support for learning. If this scaffolding is done properly, such structuring encourages a student to develop his or her own initiative, motivation and resourcefulness. Once students build knowledge and develop skills on their own, elements of the support are removed.

According to McKenzie (1999), the defining features of successful scaffolding include **clear direction, purpose, and expectation**. Results include **on-task activity; better student direction; reduced uncertainty, surprise, and disappointment; increased efficiency; and palpable momentum**.
We will have detail discussion about scaffolding in next chapter.

2.2.2 Socratic Questioning

This teaching strategy is fully explained in Vikash's[23] dissertation.

2.2.3 Guided discovery

This teaching strategy is fully explained in Rajashekhar's[24] dissertation.

2.2.4 Game Based Learning

This teaching strategy is fully explained in Praveen's[25] dissertation.

Chapter 3

Scaffolding Teaching Strategy

3.1 Introduction

As discussed above building construction scaffolding is defined as the structure built around the building when a brand new building is being built or when a building is being repaired. After the building is completed or the repairs are made, the scaffolding is removed.
In teaching strategies Vygotskian theory works as a ground for today's scaffolding. One of the main tenets of Vygotskian theory is the notion of a **zone of proximal development** (ZPD), which was conceptualized as: the distance between the actual developmental level as determined by independent problem solving and the level of potential development as determined through problem solving under adult guidance, or in collaboration with more capable peers"[12]. In the

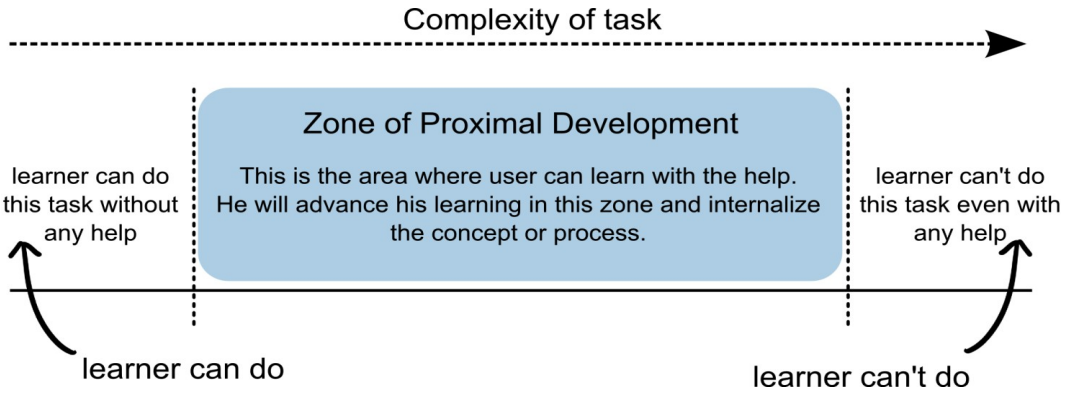


Figure 3.1: Zone of Proximal Development adapted from [16].

figure-3.1 it is shown that if we teach a learner the concept which he already know then learner will feel boredom. If we teach a learner the concept which he can not understand even after giving many hints then in that case learner will become hopeless and his anxiety will be finished about that concept. So with scaffolding student can learn only those things which comes in ZPD of that student.

In Vygotsky's view, the learner does not learn in isolation. Instead learning is strongly influenced by social interactions, which take place in meaningful contexts. Children's social interaction with more knowledgeable or capable people and their environment significantly impacts their ways of thinking and interpreting situations. Studies[8] have actually shown that in the absence of guided learning experiences and social interaction, learning and development are hindered[13]. So it has been proved that scaffolding is helpful to teach learner very effectively.

3.2 Definitions of Scaffolding strategy

Many authors and scholars have given different definitions of scaffolding but the main focus of all of them was on providing support to the students. We read many definitions of scaffolding. Some ITSs which have been studied from many sources are listed as follows.

- Vygotsky defined scaffolding instruction as "the important role of teachers and others in supporting the learner's development and providing support structures to get to that next stage or level" (Raymond, 2000, p.176)
- Scaffolding teaching strategy provides individualized support based on the learner's ZPD (Chang, Sung, and Chen, 2002).
- Scaffolding instructions are provided in the activities that are just beyond the level of what the learner can do alone (Olson and Pratt, 2000).

So the goal of the educator when using the scaffolding teaching strategy is for the student to become an independent and self-regulating learner and problem solver (Hartman, 2002). According to Saye and Brush, there are two levels of scaffolding[7]: soft and hard.

1. **Soft Scaffolding :** An example of soft scaffolding in the classroom would be when a teacher circulates the room and converses with his or her students (Simon and Klein, 2007). The teacher may question their approach to a difficult problem and provide constructive feedback.
2. **Hard Scaffolding :** Hard scaffolds are developed in order to assist students with a difficult task (Saye and Brush, 2002). Difficult task means that task which has only one solution and can be explained only by expert and there is no other method to solve them. The key is that the assistance is planned in advance. In this hard scaffolding, the teacher in the classroom is considered as the expert and responsible for the scaffolding of his or her students.

3.2.1 Our Definition of Scaffolding

Our definition is followed by Raymond's definition of scaffolding. We replaced Teacher with our ITS. Soft scaffolding is very difficult to achieve using our ITS. In soft scaffolding scaffold is provided on the fly by teacher's experience and according to the need of students. In our

approach **all assistance is planned in advance and our system is considered as the expert and responsible for the scaffolding of student**. So it provides more or less hard scaffolding to the student. We will provide the support to the student in form of hints. This support will be removed after a student become expert in particular topic. The logic by which we will achieve this will be explained in **Chapter 4**.

3.3 How scaffolding strategy is helpful?

Education research [3] [4] has proved that scaffolding is very helpful in learning programming concepts. We are listing here eight characteristics[10] of web-based educational scaffolding and the explanation of how our system is capable of these characteristic:

- **Scaffolding provides clear directions**.
  Step-by-step instructions are necessary to let students know what they need to accomplish to successfully meet the requirements of the task. Care should be taken by designers so that instructions produce as little confusion for students as possible. In our system we will provide an efficient interface to the instructor so that he is able to do this.

- **Scaffolding clarifies purpose**
  The objective of the activity is made clear at the outset and a ”big-picture” point of view dominates in each individual activity. In our ITS topics itself will clear the need of understanding the ongoing concept.

- **Scaffolding keeps student on task**
  The provided structure helps keep students from getting distracted and ”wandering off”. In our ITS whenever user is going to distracted from the question or being frustrated because of difficulty of problem he can ask for hint so he will get a clear direction.

- **Scaffolding offers assessment to clarify expectations**
  Rubrics and standards of performance are defined up front. This avoids confusion about what will be assessed at the end of an activity. Right now in our ITS we are assessing the student by right question only. But in future our ITS will be able to adapt the assessing technique defined by teacher.

- **Scaffolding guides students to worthy sources**
  Scaffolding can reduce wasted time and keep students on task because faculty can identify ”quality” sources on the web for students to use. In our ITS hints are provided by expert teacher. So student will get the question and hint of good quality. So learner have no need to search anywhere about that concept. Hints can be link to any web page also.

- **Scaffolding reduces uncertainty, surprise and disappointment**
  All distracting frustrations with site design should be eliminated. This is what McKenzie calls the ”Teflon lesson - no stick, no burn, no problem”. Our ITS will be so simple in looking but complex in functioning.

- **Scaffolding delivers efficiency**
  By eliminating boredom and irrelevance, scaffolding grants a sense that a larger amount of work can be completed in a shorter time. In our ITS we are achieving this by providing hints.

- **Scaffolding creates momentum**
  In our ITS student will follow an appropriate way to complete the course and keep student on doing task. So our ITS channeled the focus of student on study.

3.4 Scaffolding Instructions

Scaffolding instruction[14] includes a wide variety of strategies. According to Beth Lewis[14] **Teacher can offer hints or partial solutions to problems** as a scaffolding instructions. He can activate prior knowledge and offer a motivational context to excite student's interest or curiosity in the subject at hand. He can break a complex task into easier, more "doable" steps to facilitate student achievement. He can show students an example of the desired outcome before they complete the task. He can teach students chants or short tricks to ease memorization of key facts or procedures.He can use graphic organizers to offer a visual framework for assimilating new information. He also can guide the students in making predictions for what they expect will occur in a story, experiment, or other course of action. He can ask questions while reading to encourage deeper investigation of concepts. He also can ask students to contribute their own experiences that relate to the subject at hand. All these strategies are useful in teaching. All these strategies come under the scaffolding instruction category. Teacher can use one or more of the scaffolding instruction technique given above to teach. We are using only hints for scaffold. This is a limitation of our ITS.

Chapter 4

Overall ITS Framework

In this chapter challenges and components and architecture of our ITS will be discussed. This chapter includes figure of architecture of ITS framework. Workflow in our ITS is also discussed here.

4.1 Challenges we faced in developing ITS framework

We faced following challenges in developing this ITS framework.

- Finding an appropriate course structure which supports more than one teaching strategy.
- Finding the teaching strategies which can be compatible with the structure.
- Developing an algorithm for providing questions to learner according to his learning experience with strategies.
- Finding the common and uncommon modules between these strategy. According to this make a generalized architecture.
- Integration of DATABASE of all strategies.

First work was to decide the question pattern by which more than one strategy can teach. After studying architecture of many ITSs we decide MCQs pattern to make our system. So for supporting MCQs pattern we studied so many teaching strategies. Among them we have picked 4 strategies 1. Socartic Questioning 2. Scaffolding 3. Guided Discovery 4. Game Based Learning. Each strategy has its own requirements so we researched individual requirements for each strategy. We have developed single ITS for each strategy and after that we have figured out common module and non common module in all strategy. According to this common module and non common module we have developed our databases and user interfaces. Making the common database was tedious job to do. Finally we came up with our single ITS which teach with four strategy.

4.2 Architecture of our ITS

Figure 4.1 shows the architecture of our ITS. Our architecture has four main components as Course Controller, Domain/Expert model, Student Model and User Interface.

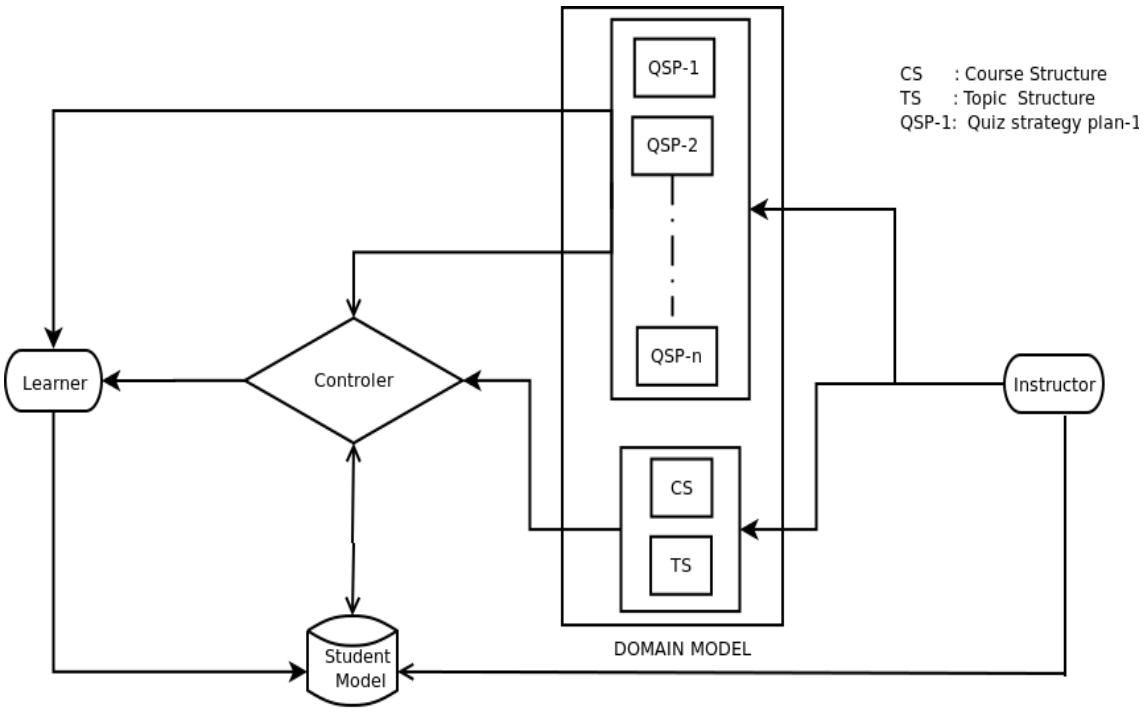


Figure 4.1: Top Level Architecture of Our ITS.

4.3 Components our ITS

1. **Course manager/Controller :** It will do main functioning of the ITS. Course manager will control all other components of system. It will provide the interface to the user. Interface will be provided according to the user type like student or instructor. It works behind the user interface. It takes input from the user via user interface and provides output according to the response of user again via user interface. It is responsible for providing appropriate hints to the student as required. It also takes decision of which topic should be given to the learner. All the logic and database part will be handled by this component. There will be some other module which will help to do the tasks.

2. **Domain/Expert Model :** It will manage the course structure. Student can be provided material like HTML page, videos, audios etc. in very articulate way. This component also responsible for providing the requirements needed by each strategy like hints and feedback in case of scaffolding teaching strategy.

3. **Student Model :** This component will have all the information of student. ITS required some information at the time of registration from user. All this information is stored in student model and can be accessed by instructor and learner. Student model also contain the present status of students' learning.

4. **User Interface :** This component is responsible for attractive interface to the learner and instructor to interact with ITS with the effective functionality. This component is not shown in figure 4.1. But learner and instructor is shown. They interact with ITS via user interface only.

4.4 Workflow in ITS

We explain the work-flow in ITS from three points of view as follows.

- **When a new user visit :** A simple home page is shown to the user explaining what is the ITSs and how to use this interface. User can see details of strategies used in ITS. User can see the contact details and useful links. User can register for learning.

- **When instructor visit :** Instructor first required to login with authenticated user-name and password. Then instructor can add courses, topics and subtopics. Then instructor provide question for appropriate subtopic with appropriate teaching strategy. He will have to choose the strategy by which he want to teach the student. Instructor then can upload required resources for specific subtopic. Instructor can see the details of student like his name, level and his progress report. After having information about the progress of students instructor can provide more resources and can switch to the new teaching strategy also.

- **When learner visit :** Student first required to login with authenticated user-name and password. Then student can see the courses offered and can select the subtopic of the course which he want to learn. Student can learn from the resources if available and after that try to attempt the questions given by teacher. Each strategy used by our system for each subtopic has priority over other. This priority is set by instructor in beginning. Suppose scaffolding has the highest for some subtopic over other strategy. Now ITS will present questions from the scaffolding module. Now as in scaffolding, If student is not able to give answer and tick wrong option then ITS will prompt for hint and students level will decrease by one point. If student choose the right option then his level number will increase by one point. ITS will store the number of hints used by student and his level number as well as his obtained marks in those questions. If student is not able to achieve the threshold of marks then ITS will switch that student to strategy of next highest priority and provide him questions from this teaching strategy for that subtopic. Thats why we say our ITS is adaptive to the learner progress.

4.5 Time Sequence Diagram

4.5.1 Time Sequence Diagram For Instructor

As shown in figure 4.2, Teacher is first asked by the module, called controller to specify the strategy for which he wants to create course or subtopic. When instructor specifies his strategy then controller provides an interface according to selected strategy because each strategy has different requirements. For example Scaffolding needs hints module, Guided discovery needs more than one level of questions. So controller manages these things. According to strategy specify by instructor it calls quiz-maker module of each strategy.

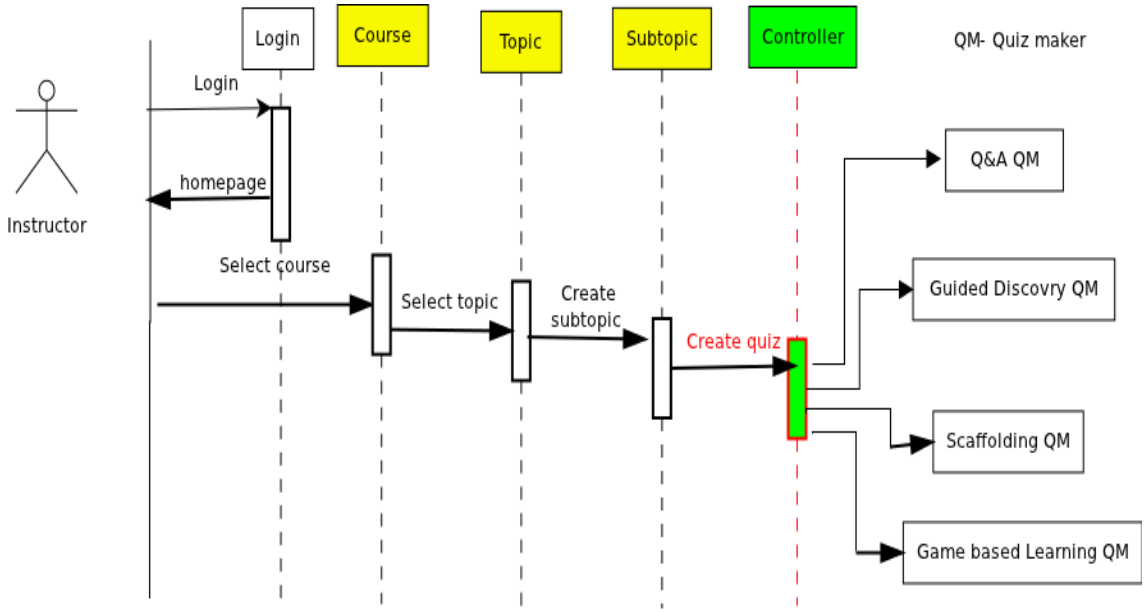


Figure 4.2: Time sequence Diagram for Instructor in Our ITS

4.5.2 Time Sequence Diagram For Learner

Figure-4.3 shows how learner will interact with system. After login learner will have some options like attempt exercise, see his level, see his performance subtopic wise etc. For example if he decides to attempt exercise then he has to select course, topic and subtopic respectively. Then Quiz Presenter module will call controller. After that controller will present question from the strategy which is best suitable for him. The decision over strategy is taken by seeing the previous learning experience of student and the priority of strategy. During attempting exercise learner is required to follow instruction given by system according to teaching strategy. As in case of scaffolding hints are given to learner.

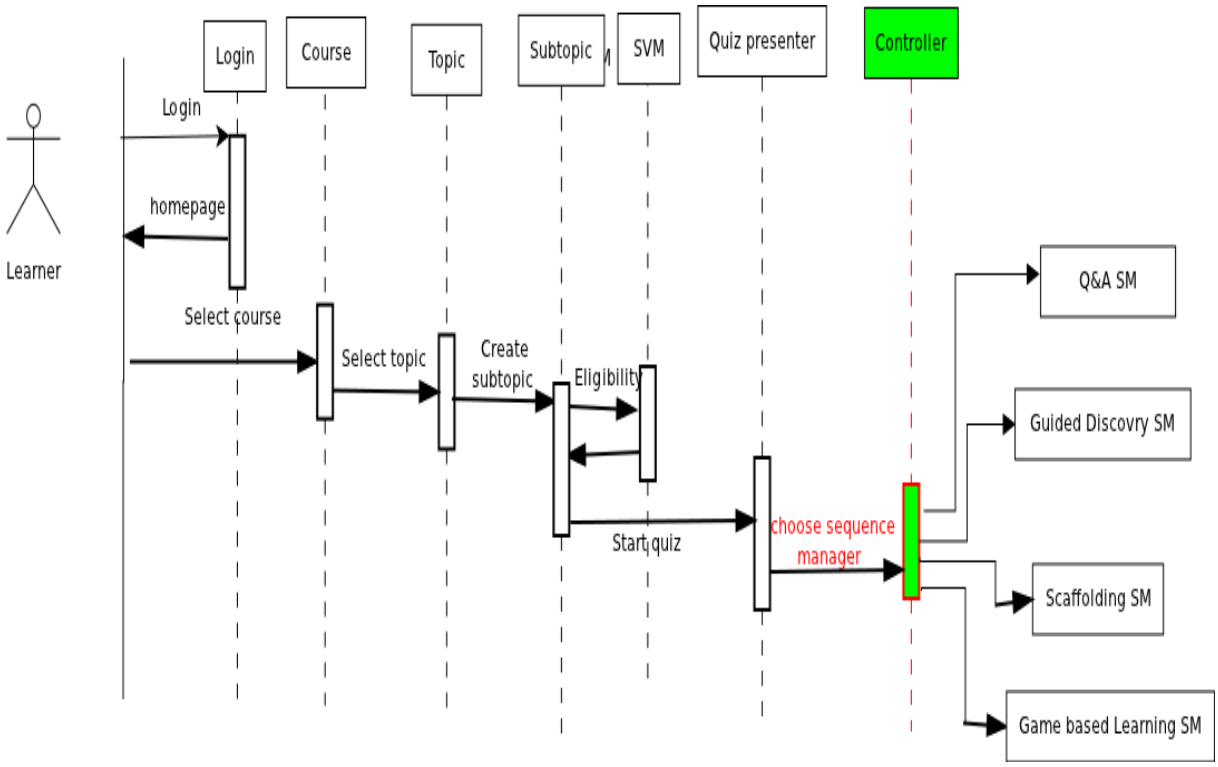


Figure 4.3: Time sequence Diagram for learner in Our ITS

Chapter 5

Implementation of Scaffolding Teaching strategy

This chapter has implementation details of scaffolding strategy for our ITS framework. Database of our ITS contains many courses, topics and subtopics. Each subtopic has some teaching strategy associated with it. Each subtopic of course is associated with at-least one teaching strategy. When ITS provide students the subtopic of scaffolding then my implemented modules gets activated. Logic behind my strategy is described below.

5.1 Architecture of our ITS with Scaffolding

Figure 5.1 shows the architecture of our ITS with scaffolding strategy only. This architecture has same components as explained in Section 4.3. Here course content and expert model is shown separately unlike Figure 4.1 where both are shown as domain model. Only difference is that expert model of this architecture provides materials for scaffolding strategy only. Expert teachers provide course content as course curriculum and hints and feedback to the student as scaffold. The approach of providing this scaffold to student is explained in next section.

5.2 Logic Used for Providing Scaffold

In scaffolding strategy hints and feedback are provided to the student as a scaffold. Each student has a level number which determine his learning progress. Logic of determining learning level of student, on the basis of hint is used in Wayang Outpost[15]. This tutor is highly successful in its work. So we are also using same logic here.
The work flow of system when user attempt the subtopic with scaffolding is given in figure-5.2. ITS is providing maximum 3 hints for each question. Number of maximum hints provided can be increased as per instructor requirement. Each students have a level number associated with him. This level number indicates how well that student is performing. At the starting all student

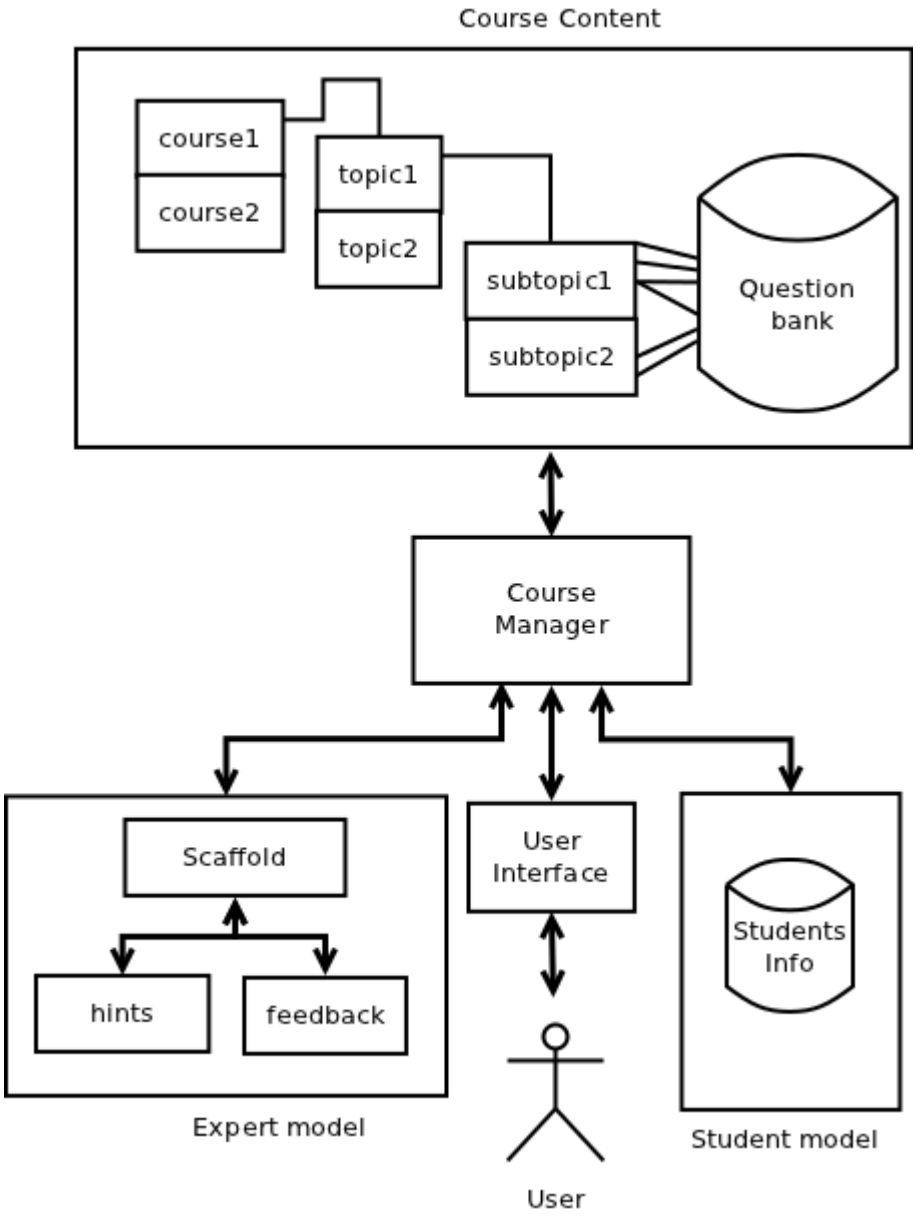


Figure 5.1: Architecture of our ITS with Scaffolding only.

are considered as beginner (level number 0) But when a student perform well without using hints then that student is moved to the expert level (level number highest). Highest level number is decided by teacher. Each time a student uses hint to solve the question his level number is **decreased** by one. Each time a student gives the correct answer of question his level number **increased** by one. The record of how much hints are used by particular student in completing the subtopic is stored in student database. This number can be further used in taking decision of changing the strategy. When a student is not obtained the threshold marks and used more hints

in the subtopic then ITS change its teaching strategy for that student automatically. Because now ITS will consider that this topic is out of his **ZPD** (Zone of Proximal Development).

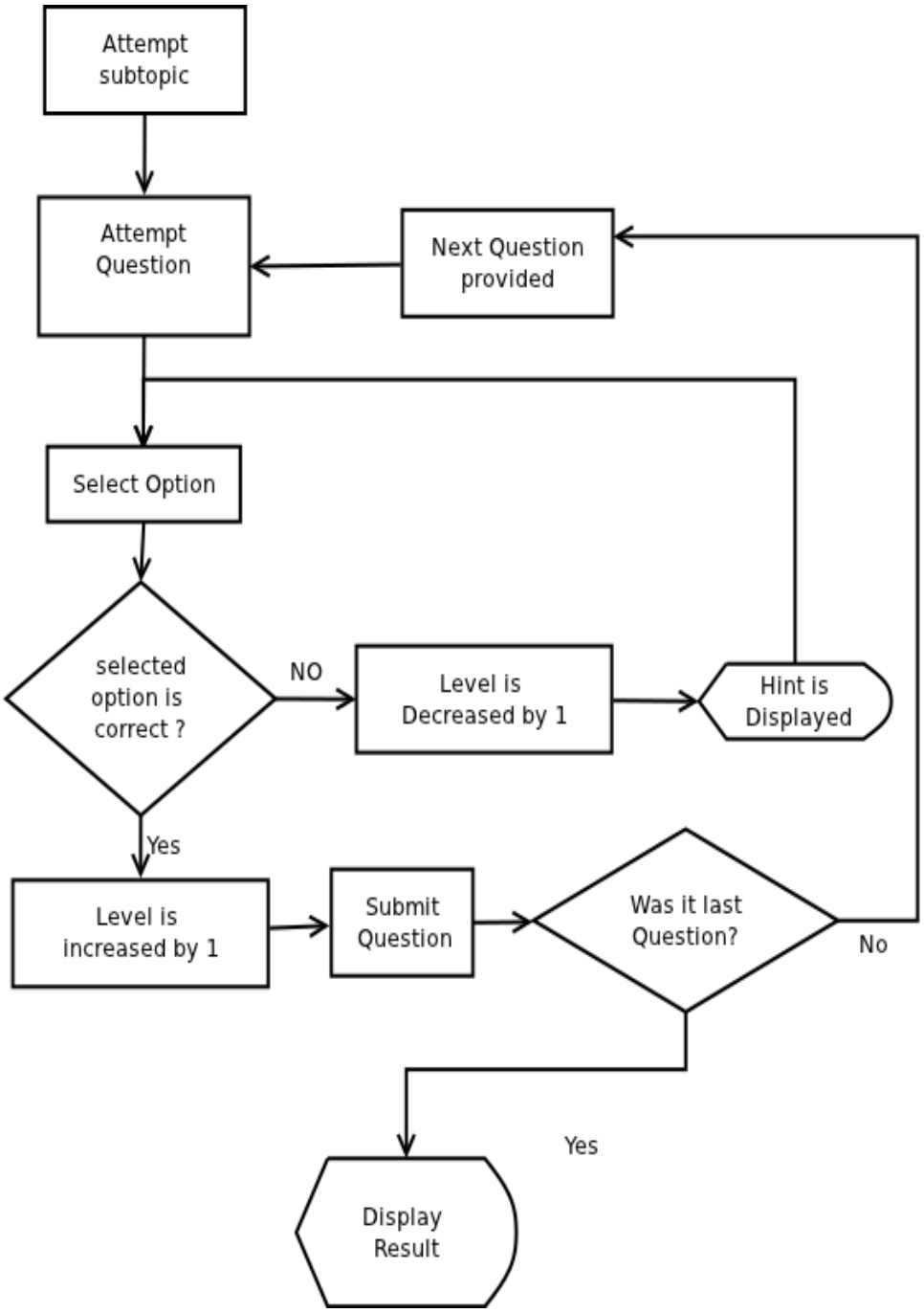


Figure 5.2: Control Flow of system when learner attempt subtopic with scaffolding.

5.3 E-R Diagram of ITS with Scaffolding

Entity-Relationship diagram of our ITS with scaffolding only is shown in figure-5.3. In this figure rectangle represent the entity or table in database, diamond represent the relationship between entity and circle represent the primary key or main attribute of entity. Explanation of figure 5.3 is given as follows. Instructor provides many courses and every student should enroll to some courses. Course has many topics and topic has many subtopics. Each subtopic has questions and hints according to these questions as scaffold. Student can attend the subtopic and attempt provided questions. Teacher can see progress of any student. Student can see his progress and his rank among other students.

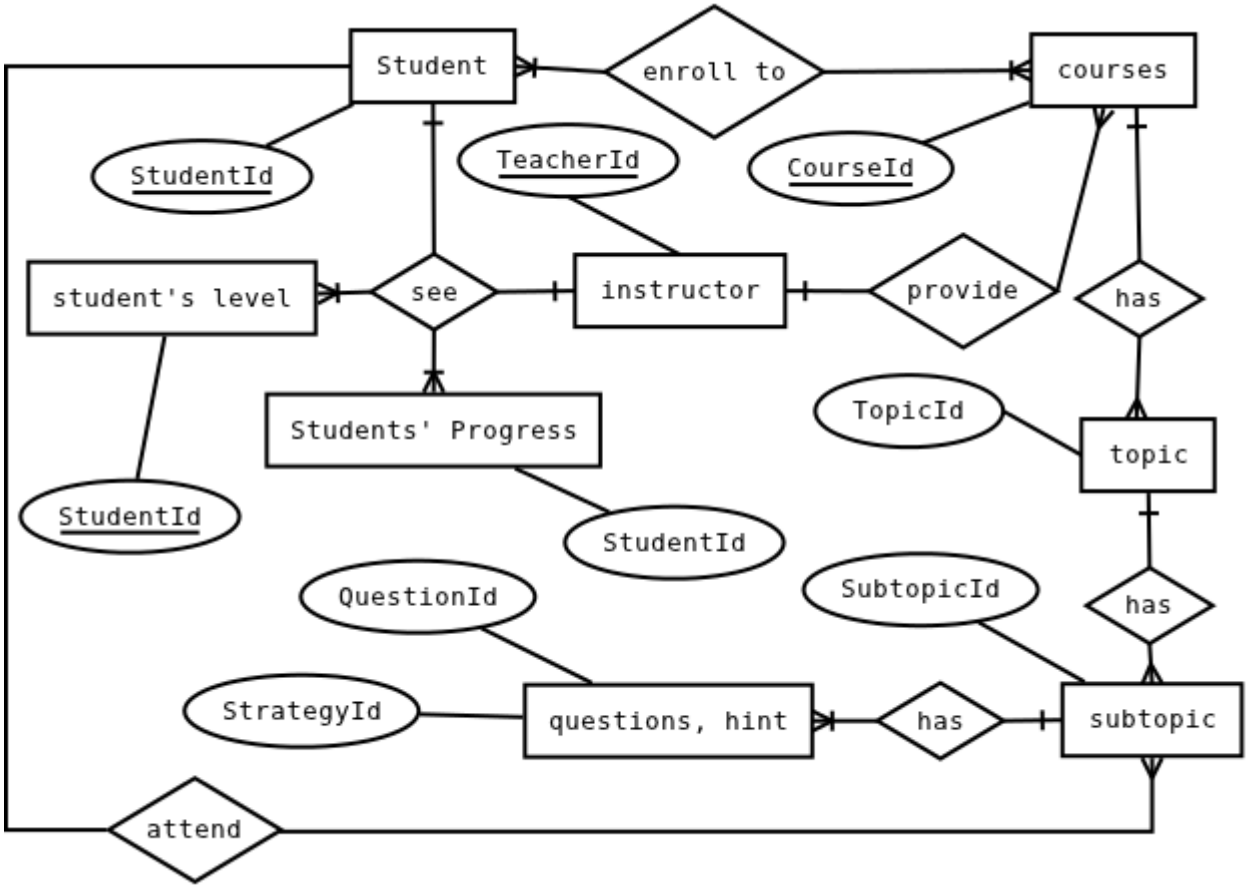


Figure 5.3: E-R Diagram of ITS with scaffolding.

5.4 Description of Modules

Here we will discuss the module which are essential for ITS framework and used by scaffolding strategy only. modules which are used by other teaching strategy will be discuss in chapter six. Questions like what functionality these modules provide to the user and how these modules interact with each other will be answered here.

All modules are explained below.

1. **GUI module**
   Functionality : Provide Login and Registration page for user depending on user type like student or instructor. Display interface to student and instructor after login. Handles all form related actions like displaying forms for storing question and hint in question bank and displaying question to the student from the question bank. Approximately all modules will communicate with this module.

2. **Login module**
   Functionality : This module will authenticate all the users. This module will allow only authorized user to access the course.

3. **Input Validation Module**
   Functionality : This module will be called whenever user is required to submit any form. This module will validate the input of user and also will ensure that no mandatory field is left empty.

4. **Course module**
   Functionality : This module maintains the course structure provided by instructor like course can have many topics.

5. **Topic module**
   Functionality : This module maintain the topic structure provided by instructor like topic can have many subtopics.

6. **Subtopic module**
   Functionality : This module maintain the subtopic structure provided by instructor like subtopic can have many Questions.

7. **Question module**
   Functionality : This module will be used to make the question for question database. This module will take input from instructor like Question, options and hints. All input will be stored by database handler module in the question database.

8. **Quiz presenter module**
   Functionality : This module will handle the issue of presenting questions to the student. This module will communicate with other module like hint module and question evaluator module. This module also call the result module if last question of subtopic has been attempted else next question will be presented to the student.

9. **Hint module**
   Functionality : This module is used to present the hints to the student when he choose wrong options. This module is also used to see and update the student level.

10. **Evaluate Question module**
    Functionality : This module will check the submitted option of student and store result like submitted option is wrong or right. After subtopic is finished then result module will use this result for preparing result.

11. **Database handling module**
    Functionality : This module will handle all the database storage, modification and accessing related task. For example this module will be called whenever instructor wants to add course, topic, subtopics etc. If instructor want to change logic then also this module will be called. We are using five databases for ITS with scaffolding listed below:

    - Content Structure Database: This database will handle all the queries regarding course, topic and subtopic. This database maintains the structure of all courses. This database will be used indirectly by course/topic/subtopic module.
    - Question Database: This database will have all the question and hints for presenting the subtopic to the student. This database will be used indirectly by question module and subtopic presenter module.
    - Student Database: This database will have all information about current status of student and his personal information. Report generator module will call database handler module to show the student's learning status which is stored in this module.
    - Login Database: This database will have password and user name of all student and teacher in it. Which is used by login module to authenticate the user.
    - Log Database: This database will have all user action stored in it. This database can be used to rollback the system after inappropriate shutdown.

12. **Result/report generator module**
    Functionality : This module will generate the report. It will use the result stored by evaluate question module to prepare report for student. This module also call database handling module to update student database according to this result. This module also provide interface to user to see the learning status of student by topic or course wise.

5.5 Description of Tables in Database

We used following tables to implement the scaffolding teaching strategy for our ITS. Screenshots of tables are given in the **Appendix A**.

- **login_table :** This table is common for both learner and instructor. This table is used for login in the system. As login for user need only user-name and password so that this table has these fields.
- **course_table :** This table has the available information about the courses like courseId, name and description of course. All these information about courses is used by system, student and instructor.
- **topic_table :** This table has the information about the topics available like TopicId, name and description of topic. This table also stored the courseId so their is an relation is established between course and its topic.
- **subtopic_table :** This table has the information about the subtopics available like SubtopicId, name and description of subtopic. This table also stored the courseId and TopicId to distinguish the subtopic from each others. So each subtopic is stored by associating it with its course and topic.
- **question_table :** This table contains all the questions w.r.t courses, topic, subtopic and strategy. As each question is made specifically for particular course, topic, subtopic and strategy. This table also contains four options and three hints. Teacher can also provide some comment to the students. This field is optional.
  This comment will be shown to the student if student does that question wrong.
- **upload_resource :** This table contains the information of uploaded resources w.r.t course,topic and subtopic. This table has field called URL which contains the path of folder in which uploaded file stored. Student can see all the download-able files. He can access all the uploaded files or resources by extracting the URL of that file. This table help teacher in providing the material which can be useful to the learner while attempting any course.
- **student_info :** This table stored all the information about the student which is given by the student at the time of registration. This information can be extracted by the instructor for various purposes.
- **student_progress_table :** This table keep the record of student's progress. Information, like how many subtopic he has done and what was his performance in these subtopics is stored in this table. This table stored the performance(in %) of student in subtopic by which ITS decide that this student is required to switch the strategy or not. Below are the formula we used for calculating the performance of student in the subtopic. This formula can be optimized further to calculate the students' performance.

Performance of student $= ((MO/TM) * 100 - (HU/TH) * 100)$

TM = Total marks in the subtopic
MO = Marks obtained in the subtopic
THU = Total Hints used by student
TAH = Total available hint for the subtopic

- **student_level :** This table keep track current level of student. Level of student shows how well student is performing in this strategy. This table has field called overall_level which is updated each time student attempt question.
- **student_response_table :** This table stored every action of students. This table stored the response of student for every question. If students system crash then this table is used to track his previous position. Suppose a student leave subtopic in middle and logout. When he login again and attempt that subtopic then he will not start from beginning. He will get the next question from his previous question. In this case system will use this table to track his last action.

5.6 Source code

We used **PHP, AJAX, JQuery, CSS, Javscripts and HTML** for implementation of the front-end. We used MySql database for storing the data. We are providing a table which contains the file name and information about what that file is used for in Appendix B.

Chapter 6

Content Used for Testing

In this Chapter we will discuss the format of questions on which we test this ITS for scaffolding. **Appendix C** will have all questions used for testing in our ITS.

6.1 Format of Question

whole course s provided to learner in three parts. Each course will have many topics. Each topic will have many subtopics. Each subtopic will have many MCQ questions. As we have mentioned in last chapter, only three hints are provided for each question. Learner will learn by attempting these questions. Below are given all field required to stored question with scaffold.

- **CourseId:** This field stores the course id of course. This field is the primary key of another table, called course table which stores the details of that course.
- **TopicId:** This field stores the topic id of topic. This field and courseID both are the primary key of another table, called topic table which stores the details of that topic. Each topic must be associated with some course so that that topic can be identified individually.
- **SubtopicId:** This field stores the subtopic id of subtopic. This field, TopicId and courseID are the primary key of another table, called subtopic table which stores the details of that subtopic. Each subtopic must be associated with some topic so that that subtopic can be identified individually.
- **QuestionId:** This field stores the question number of question. Each subtopic stores many questions then it is necessary to give question number for each question.
- **QuestionDesc:** This field stores the description of question, like what that question requires student to do?
- **Options1-4:** These four field contains all four options. Each question has four options but only one option is correct. Learner is required to choose that option only.

- **Hint1-3:** These three field contains three hint. Whenever learner choose wrong option one hint is displayed. Each hint is provide in such a way that learner will be able to solve that question.
- **comment:** This field is **optional**. This field stores the comments which is shown to student if he attempt question wrong even after having three hints.

6.2 Example Question

Course-Id: CS101
Topic-Id: C
Subtopic-Id: MSL (Miscellaneous questions)

Question Description: What will be the output of following program ?

```
main()
{
int x,y = 10;
x = y * NULL;
printf("%d",x);
}
```

Option1: error
Option2: 0
Option3: garbage value
Option4: 10
Correct answer: 0

Hint-1: NULL is a macro defined in stdlib.h
Hint-3: Macro is just replaced by its value at compile time
Hint-2: NULL is defined as zero
Comment: ...

Chapter 7

Integration of all Four Strategies

This chapter includes combined work of all four of us. First of all each of us implements his own strategy individually. Then we compare the modules of each strategy. We found many modules and tables are same in implementation of all strategies. Here we listed all common and non-common modules and tables. Then we will discuss the integration of all work together.

7.1 Modules

7.1.1 Common modules to all 4 Strategies

These module has been discussed in Section 5.4

- GUI module(GUIM)
- Input Validation module(IVM)
- Student module
- Authentication module(AM)
- Course Validation module (optional for my strategy)
- Quiz Maintainer Module
- Evaluation module
- Database handling module
- Log module
- Feedback module
- Result generator module
- Logic module

- Course Module
- Topic Structure Module
- Quiz maker Module
- Status Module

7.1.2 Non-common modules

1. Scaffolding Strategy
    - Hint Module: Discussed in Section 5.4
2. Socratic Questioning Strategy: Discussed in Vikash's dissertation[23].
    - Sequencing Module
    - Topic Validation Module
3. Guided Discovery Strategy
    - Sequence Module: Discussed in Rajashekhar's dissertation[24].
4. Game Based Learning
5. Module used for integration
    - Strategy module: This module is used to add/delete the strategy from ITS. It is also stored the information about strategies available.
    - Strategy sequencing module: This module is used to assign the priorities to strategies. Controller module used information provided by this module to decide questions from which strategy should be given to learner.
    - Controller module : This module is used to provide the questions for learner from appropriate strategy. It is also responsible for providing appropriate interface to instructor according to the strategy he wants.

7.2 Tables

7.2.1 Common tables

These tables has been discussed in Section 5.5

- Login table
- Course table

- Topic table
- Subtopic table
- Question table : we combine all fields required by each strategy in one table.
- Upload_file table
- Student_info table
- Student_progress table
- student_response table

7.2.2 Non-common Tables

1. Scaffolding Strategy
    - Student level : already discussed in Section 5.2
2. Socratic Questioning Strategy: Discussed in Vikash's dissertation[23].
    - Sequencing table
    - parsed table
3. Guided Discovery Strategy: Discussed in Rajashekhar's dissertation[24].
    - subtopic_threshold
    - topic_dependency
    - subtopic_dependency
4. Game Based Learning
5. Table used for integration
    - Strategy table : This table is used for storing the information of available strategies.
    - Strategy priority table : This table stores the priority of strategy over others.

7.3 Integration of modules and tables

We found common and non-common tables and modules. Then we put all tables and module together and implement the part which is useful for adequate functioning of system. We create a table, called strategy priority table which is used to store the priority number. Lower the strategy number higher the priority. Then we implement a strategy module. This module is used by teacher to create the new strategy and give this strategy some priority over other. Then we

implement a strategy sequencing module which is used by ITS. This module check the result of student in subtopic and take decision about switching to other strategy. We developed an algorithm for switching the student from one strategy to other. This algorithm will be discussed in next section.

7.4 Algorithm used for Switching from One strategy to Other

In our ITS instructor provides questions for each subtopic with many strategies. He gives priorities to each strategy over other for each subtopic. Suppose a student wants to attempt subtopic. Then his request with the subtopicId is sent to the controller via quiz presenter as shown in Figure 4.3. Then controller will follow an algorithm to choose the strategy by which student should learn. This decision is taken by considering the history of student performance in subtopics by all strategy. Terms used in algorithm are explained below followed by the algorithm.

Variables in algorithm: There are four variables used in this algorithm, called *currentStrategy*, *previousStrategy*, *Subtopic*, *found* and *Threshold*. *Subtopic* stores the subtopicId of the subtopic a student wants to learn. Other variable *currentStrategy* stores the StrategyId of current Strategy. *previousStrategy* is an intermediate variable to stores the *currentStrategy* variable for future use in algorithm and *found* is a boolean variable stores only true or false. *Threshold* is a very important variable. This variable stores the threshold value of student performance in a subtopic. It is set by the teacher at the time of subtopic creation.

Functions in algorithm: Following functions are used in this algorithm.

- **StudentPerformance(CurrentStrategy):** This function is used to get the result of last subtopic student attempt with the currentStrategy. This function take currentStrategy as arguments and returns the percentage result.
- **AvailableStrategy(Subtopic):** This function returns the number of available strategy for the Subtopc. It takes subtopicId as argument and returns integer number.
- **isStrategyUsed(Strategy):** This function is used to check whether this strategy has been used by student or not. It takes strategyId as argument and returns true or false.
- **NextPriorityStrategy(Subtopic, currentStrategy):** This function is used to get strategy of the next high priority for given subtopic. It takes subtopcId and currentStrategy as arguments and returns strategy of the next high priority. If there is no next high priority strategy available then it will return current strategyId.
- **provideQuestion(Strategy):** This function is used to provide the question from the given strategy. StrategyId is passed as the argument in this function.
- **getHighPrioStrategy(Subtopic):** This function is used to get the strategy of highest priority among all available strategies.

Algorithm 1 Choosing best strategy for student

Require: *Subtopic*

```
1: if AvailableStrategy(Subtopic)==0 then
2:     print "Please add strategy";
3: else
4:     currentStrategy ← getHighPrioStrategy(Subtopic)
5:     if (isStrategyUsed(currentStrategy)) ≠ 0 then
6:         if StudentPerformance(currentStrategy) ≥ Threshhold) then
7:             provideQuestion(currentStrategy);
8:         else
9:             previousStrategy ← currentStrategy
10:            currentStrategy ← NextPriorityStrategy(Subtopic, currentStrategy);
11:            while previosStrategy ≠ currentStrategy do
12:                if (isStrategyUsed(currentStrategy)) ≠ 0 then
13:                    if StudentPerformance(currentStrategy) ≥ Threshold) then
14:                        provideQuestion(currentStrategy);
15:                        found ← true;
16:                        break;
17:                    else
18:                        previousStrategy ← currentStrategy;
19:                        currentStrategy ← NextPriorityStrategy(Subtopic, currentStrategy);
20:                    end if
21:                else
22:                    provideQuestion(currentStrategy);
23:                    found ← true;
24:                    break;
25:                end if
26:            end while
27:            if found == false then
28:                currentStrategy ← maxPreformanceInallStrategy();
29:                provideQuestion(currentStrategy);
30:            end if
31:        end if
32:    else
33:        provideQuestion(currentStrategy);
34:    end if
35: end if
```

Explanation: First we need to find out that there is some strategy available for given subtopic. If only one strategy is available for the subtopic then all question will be provided with this strategy only. If there are more than one strategy available for subtopic then we first find out the strategy of highest priority. Now if student have used this strategy then we find out how was his performance with this strategy in last subtopic. Suppose Threshold is set at 70%. Decision over threshold value can be the future work. If student performance was greater than or equal to 70% then again questions from this strategy will be provide. If student performance was less than 70% then we find next high priority strategy and again check whether student has attempt subtopic with this strategy or not. If he doesn't then questions from this strategy will be provided. Otherwise again find the next high priority strategy. If there is no other strategy of high priority then we find the strategy with highest performance in any subtopic and provide questions from this strategy.

7.5 Screenshots of ITS with Scaffolding

Figure 7.1 shows the interface for the teacher to add question in a subtopic. Other functionalities like add/delete course, topic, subtopic etc. is also implemented. Here only important screenshots are provided.

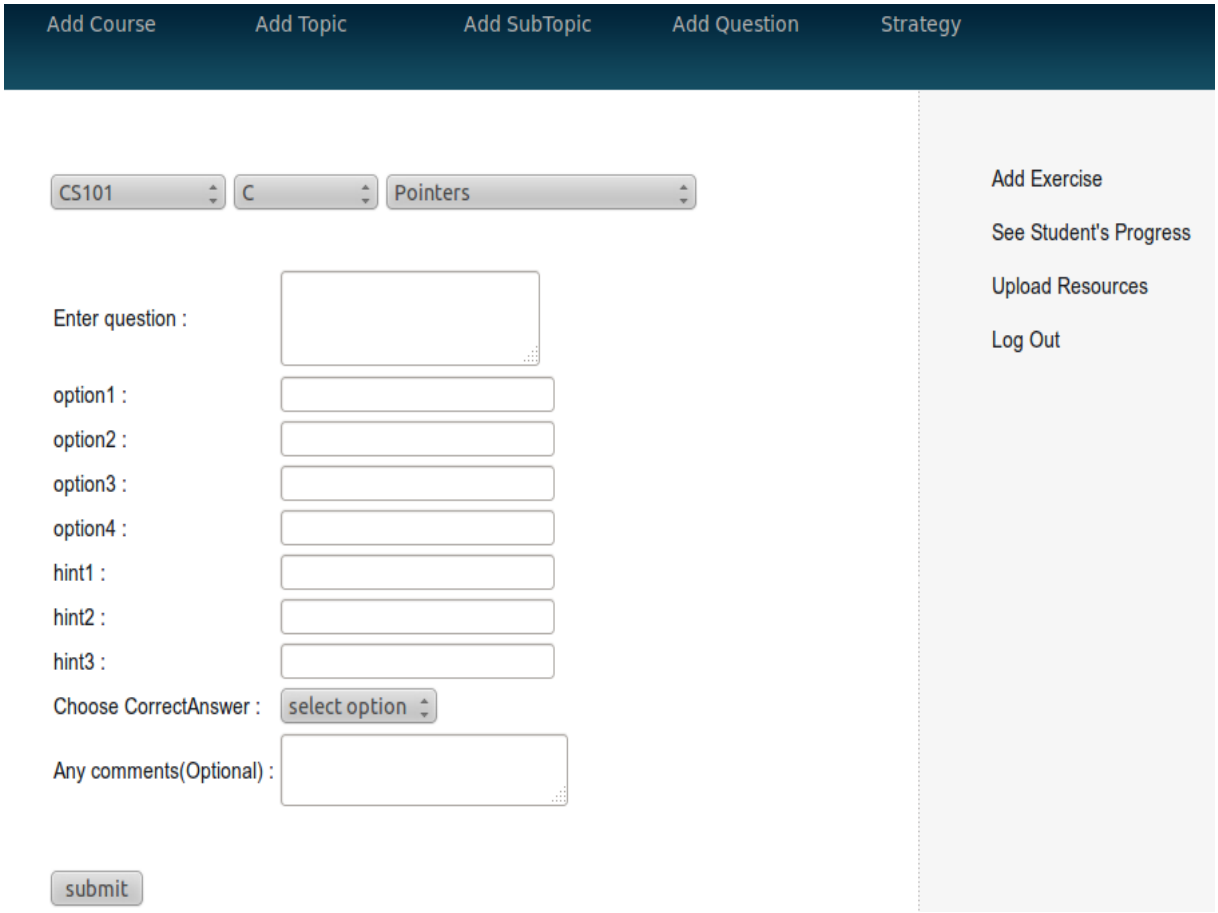


Figure 7.1: Instructor's Interface for Adding Question.

Figure 7.2 shows the interface for uploading the resources. These resources can be accessed by learner as needed.

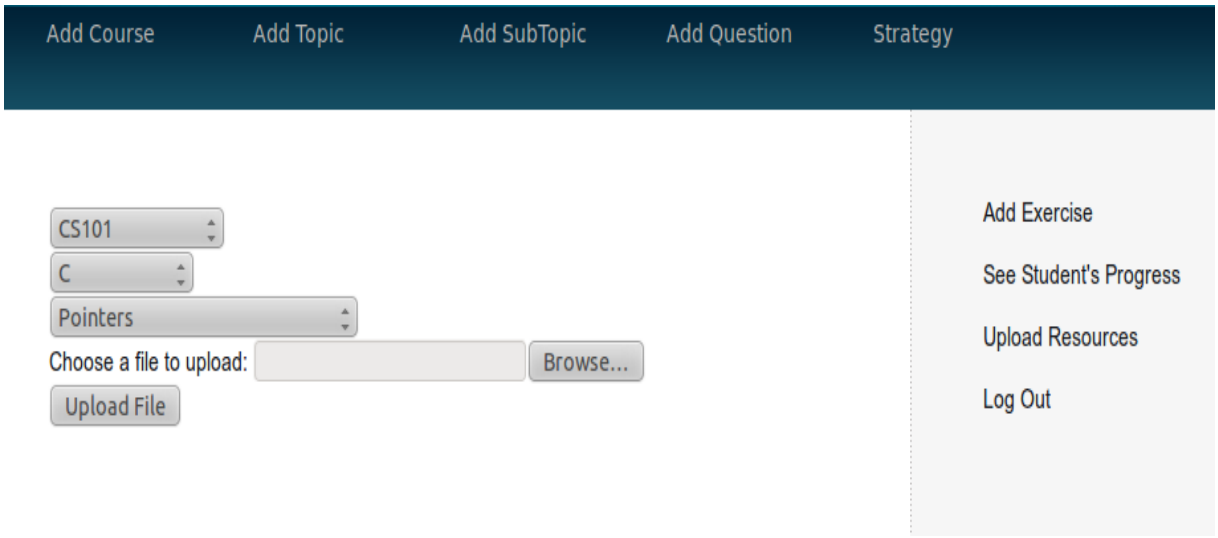


Figure 7.2: Instructor's Interface for Upoading Resource.

Figure 7.3 shows the interface for student to attempt question of a subtopic. Before attempting question learner is required to select the his interested subtopic to learn. This interface also has been implemented.

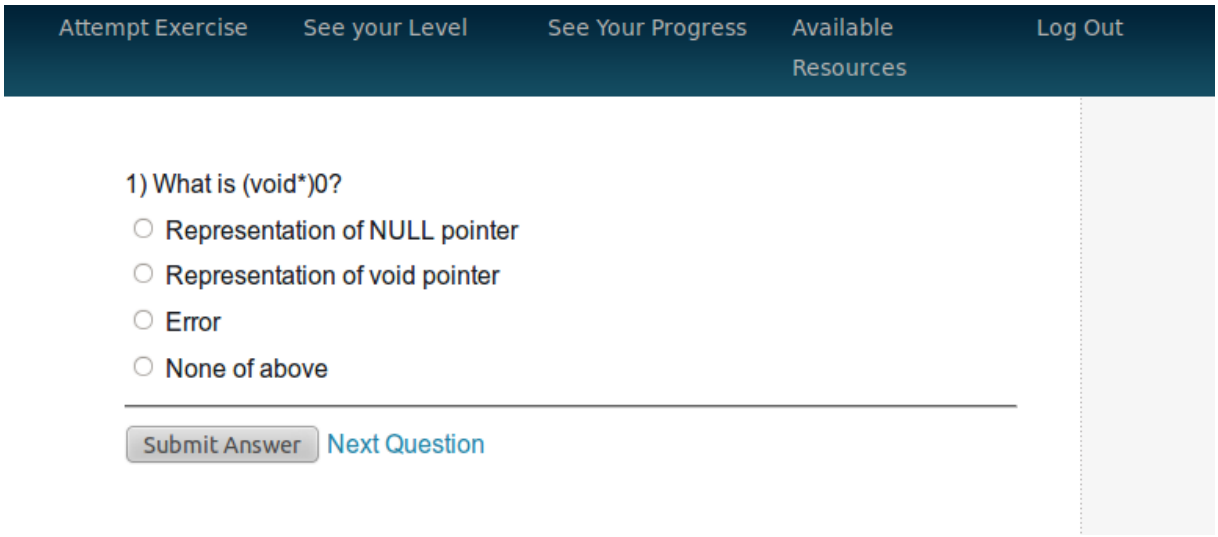


Figure 7.3: Student's Interface for Attempting Question.

Figure 7.4 shows the pop-up window which suggest learner to see hint. When learner attempt wrong question then this pop-up window guide the student towards the hints available for that question.

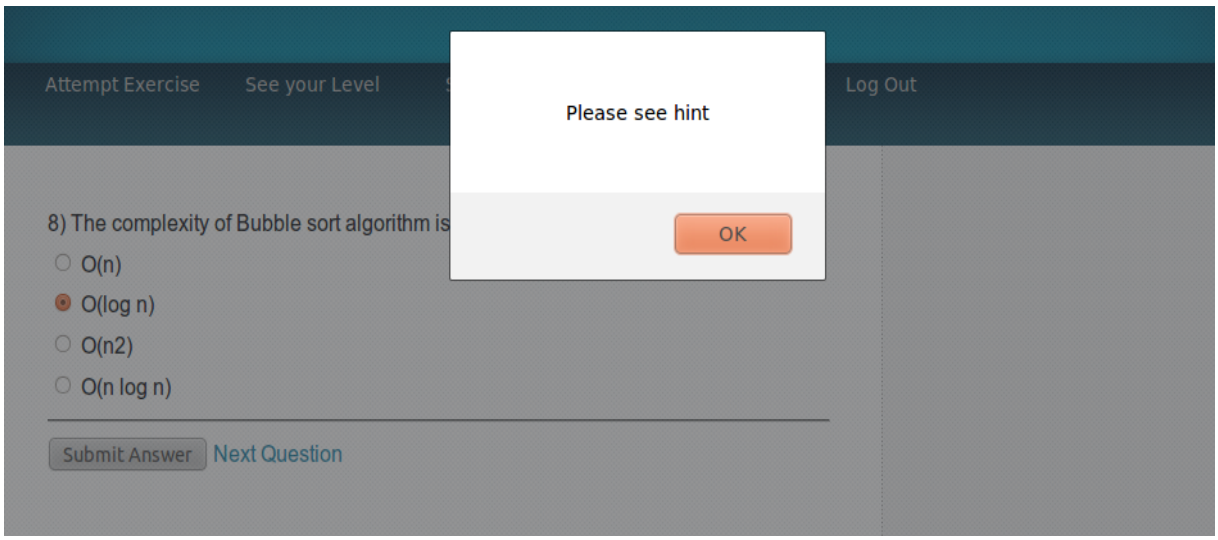


Figure 7.4: Showing Pop-up for Hint.

Figure 7.5 shows how a hint is shown to the student when needed by student.

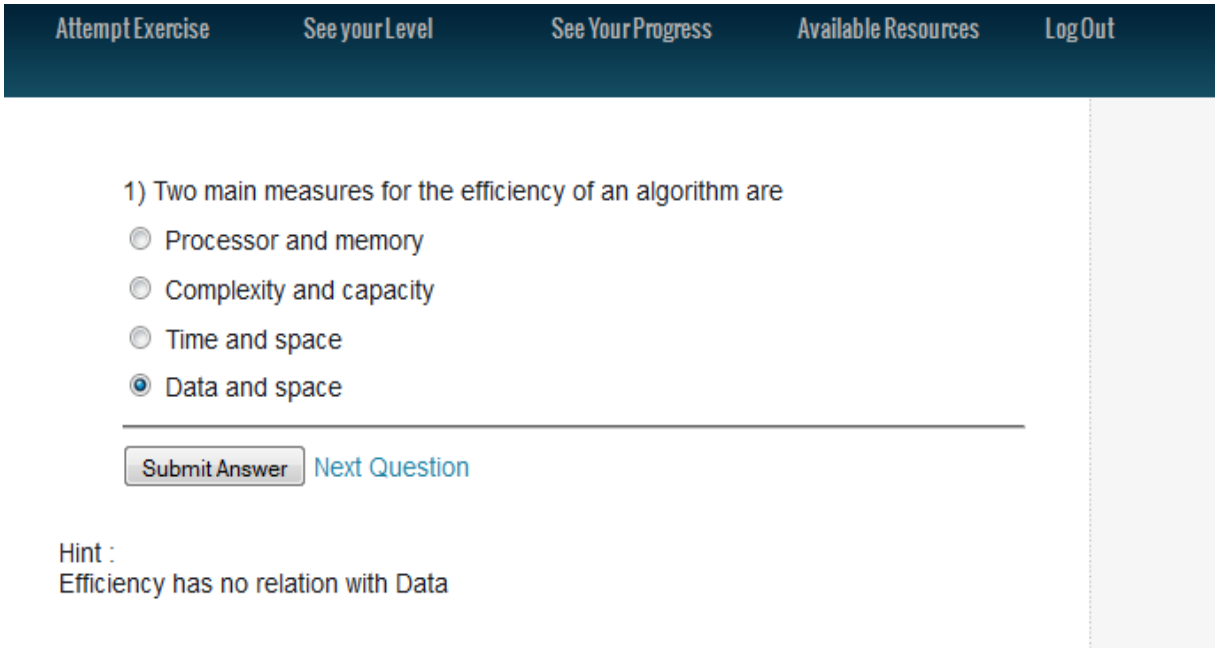


Figure 7.5: Showing Hint for Answer.

Figure 7.6 shows the feedback of students' actions. When student is finished attempting quiz then in last a feedback is generated by the system. It shows student how well he performed in this subtopic.

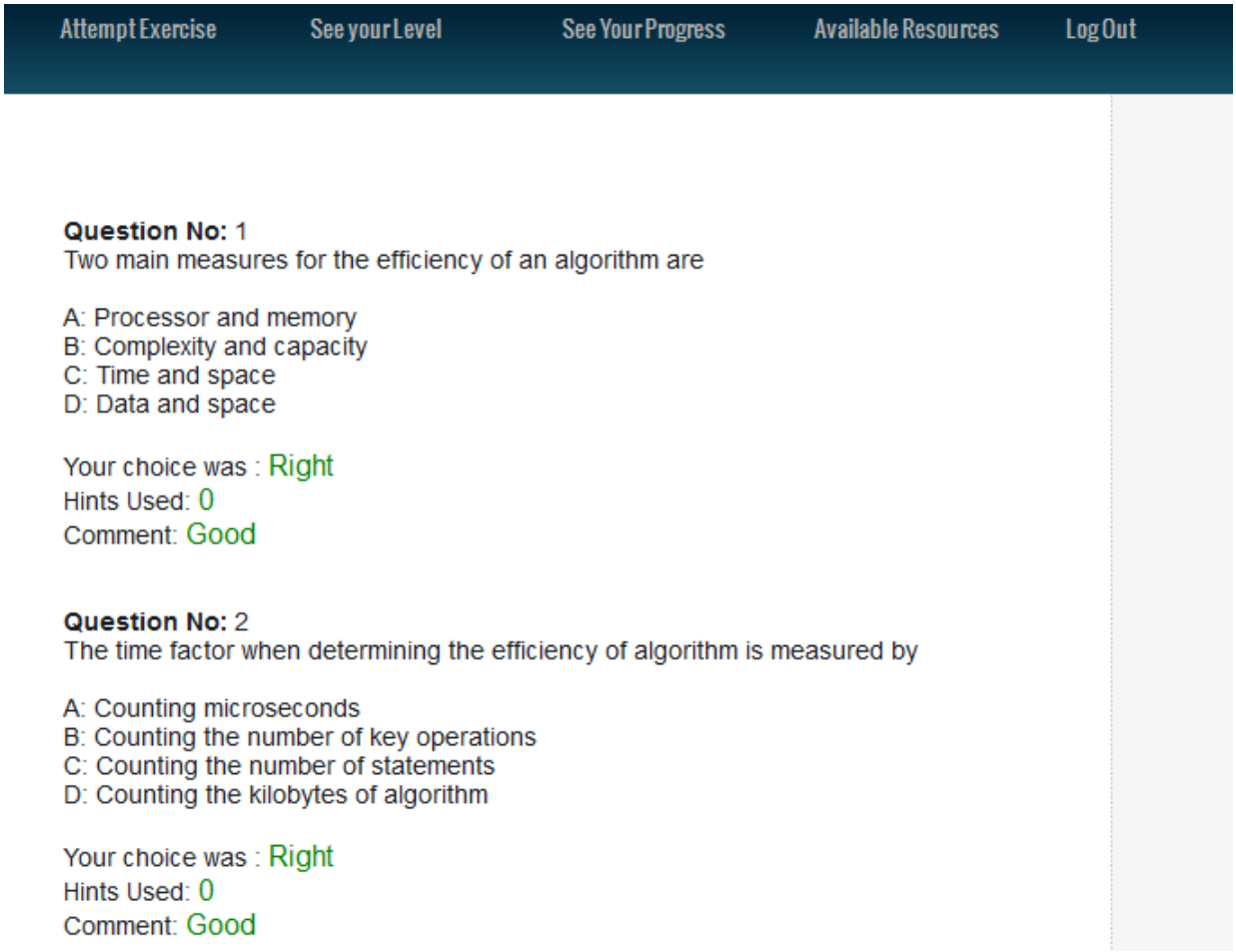
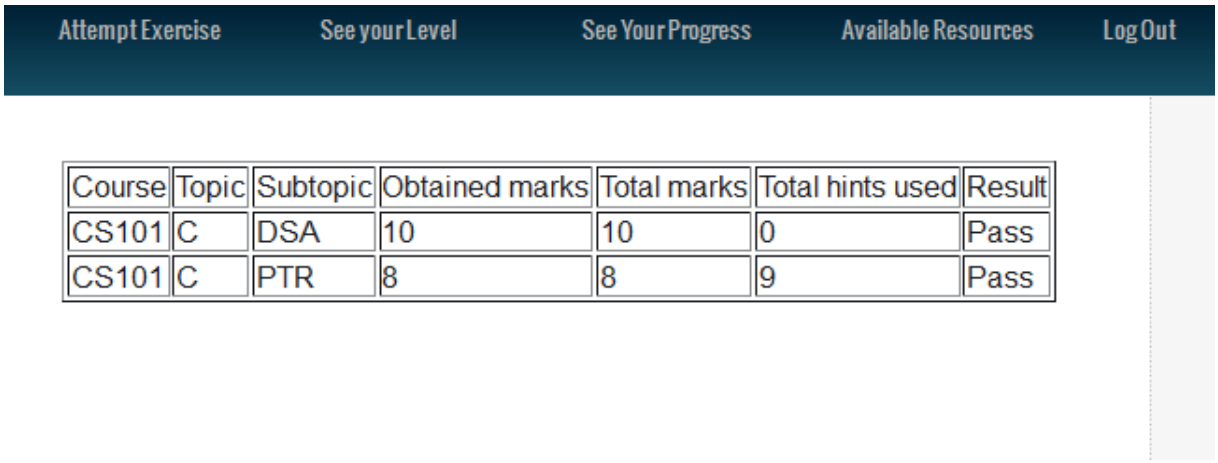


Figure 7.6: Showing Result of Subtopic.

Figure 7.7 shows the students' progress report. It shows the learning history of student. Whatever subtopic he has completed will be shown here.



Course	Topic	Subtopic	Obtained marks	Total marks	Total hints used	Result
CS101	C	DSA	10	10	0	Pass
CS101	C	PTR	8	8	9	Pass

Figure 7.7: Showing Report of Student's Progress.

Figure 7.8 shows the level of student. It shows what is the rank of current user among existing users. It will inspire student to do well and to improve his rank.

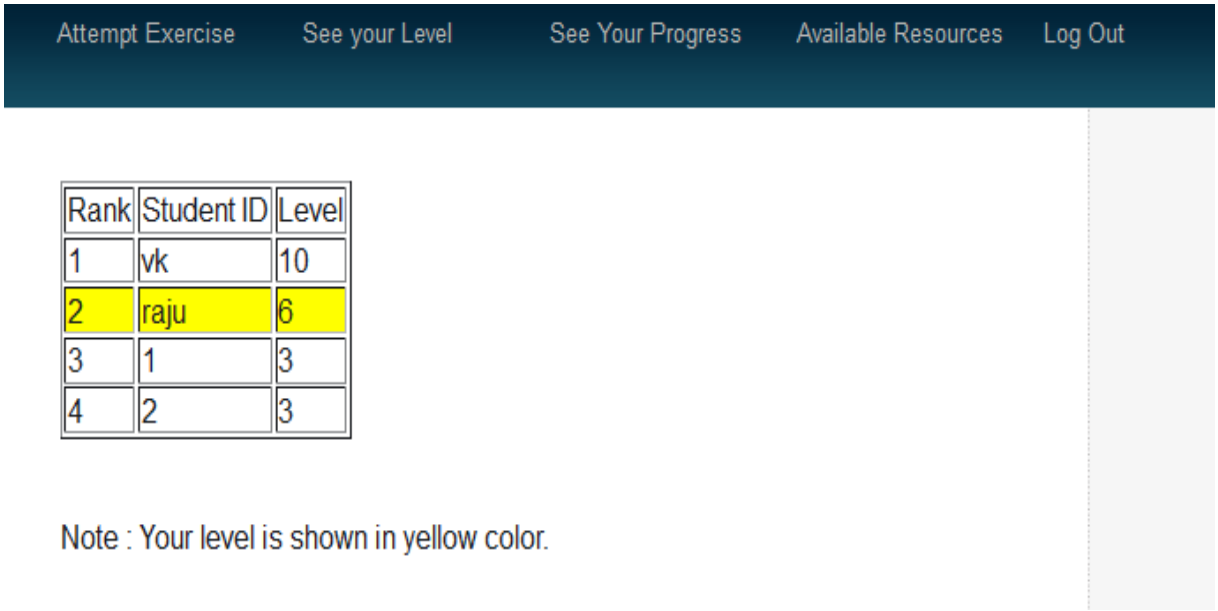


Figure 7.8: Showing Level of Student.

Chapter 8

Limitations & Conclusions

Our ITS has some limitations as follows.

- This ITS teach students with multiple choice questions (MCQs) only.
- This ITS will not provide the support to collaborative activities like forum, chat etc.
- Scaffolding does not consider the take time taken by student to answer the question.
- This ITS does not have the support for multimedia content like video lectures etc.

Conclusions:

- This ITS is limited to teach any subject with the help of multiple choice questions (MCQs) only. But the scope of our ITS can be extended in future.
- This ITS will provide the flexibility to the instructor to teach with the strategy of his interest.
- This ITS provide the functionality to the instructor to track the progress of each student.
- This ITS will teach the student with strategy which is best suitable for him. This decision is taken by the ITS according to the learning history of students.
- This ITS provide students the functionality by wich they can also see level number of other students.

Chapter 9

Future work

At this time we have a basic framework for developing ITS. There is lot more remains to be done. One can develop an ITS according to one's requirement. In future following thing can be done.

- **Collecting material for teaching :** The very first work is to prepare the material to student for learning like question with hints and some short tricks etc.
- **Response Time Theory :** One can implement a module for time response. So that time taken by student in attempting a question or subtopic can be stored. This stored time then can be used to take important decisions. This time will be helpful to find out the interest of student in topic or learning.
- **More teaching strategies :** Currently system is working for four strategy only. In future more teaching strategy can easily added.
- **Collaborative learning :** Some module for providing collaborative learning also can be added to the system. So that student's can interact with each other and learn more with each others experience. The level of student can be used as a competitive factor by showing top level students to each student.

Appendix A

Screenshots of Tables

localhost ▸ its ▸ login_table

Browse | Structure | SQL | Search | Insert

#	Column	Type	Collation	Attributes	Null	Default
1	**name**	varchar(50)	latin1_swedish_ci		No	*None*
2	**password**	varchar(20)	latin1_swedish_ci		No	*None*
3	**login_id**	varchar(50)	latin1_swedish_ci		No	*None*
4	**type**	varchar(20)	latin1_swedish_ci		No	*None*

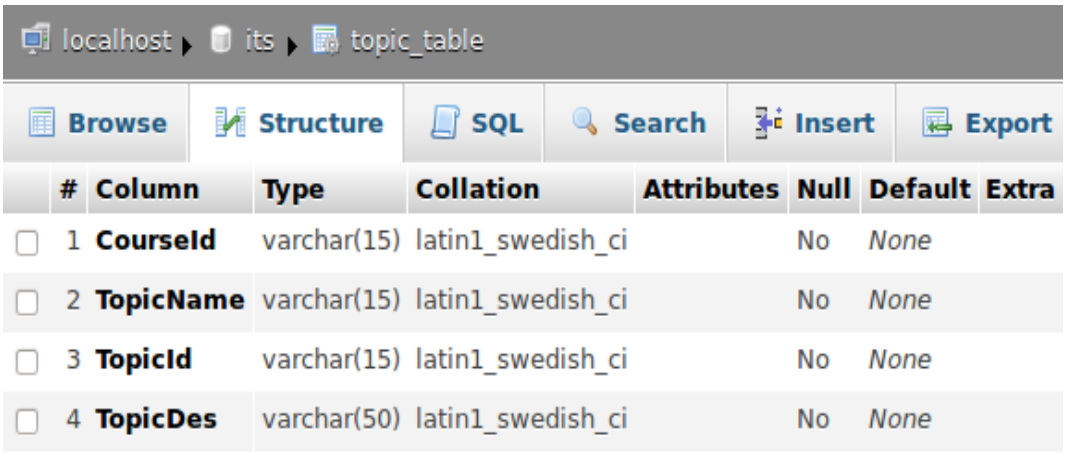
Figure A.1: Login Table for Teacher and Student.

localhost ▸ its ▸ course_table

Browse | Structure | SQL | Search | Insert | Export

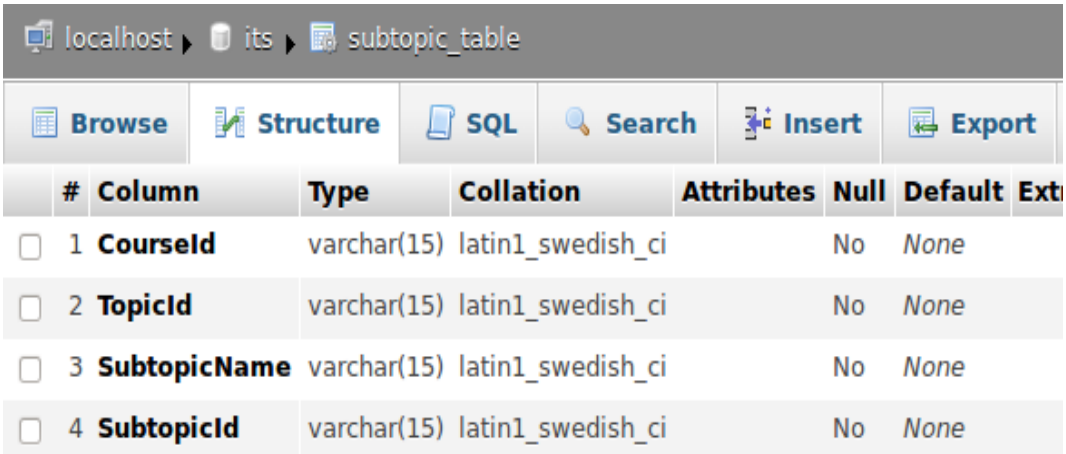
#	Column	Type	Collation	Attributes	Null	Default	Extra
1	**CourseName**	varchar(15)	latin1_swedish_ci		No	*None*	
2	**CourseId**	varchar(15)	latin1_swedish_ci		No	*None*	
3	**CourseDes**	varchar(40)	latin1_swedish_ci		No	*None*	

Figure A.2: Course Table.



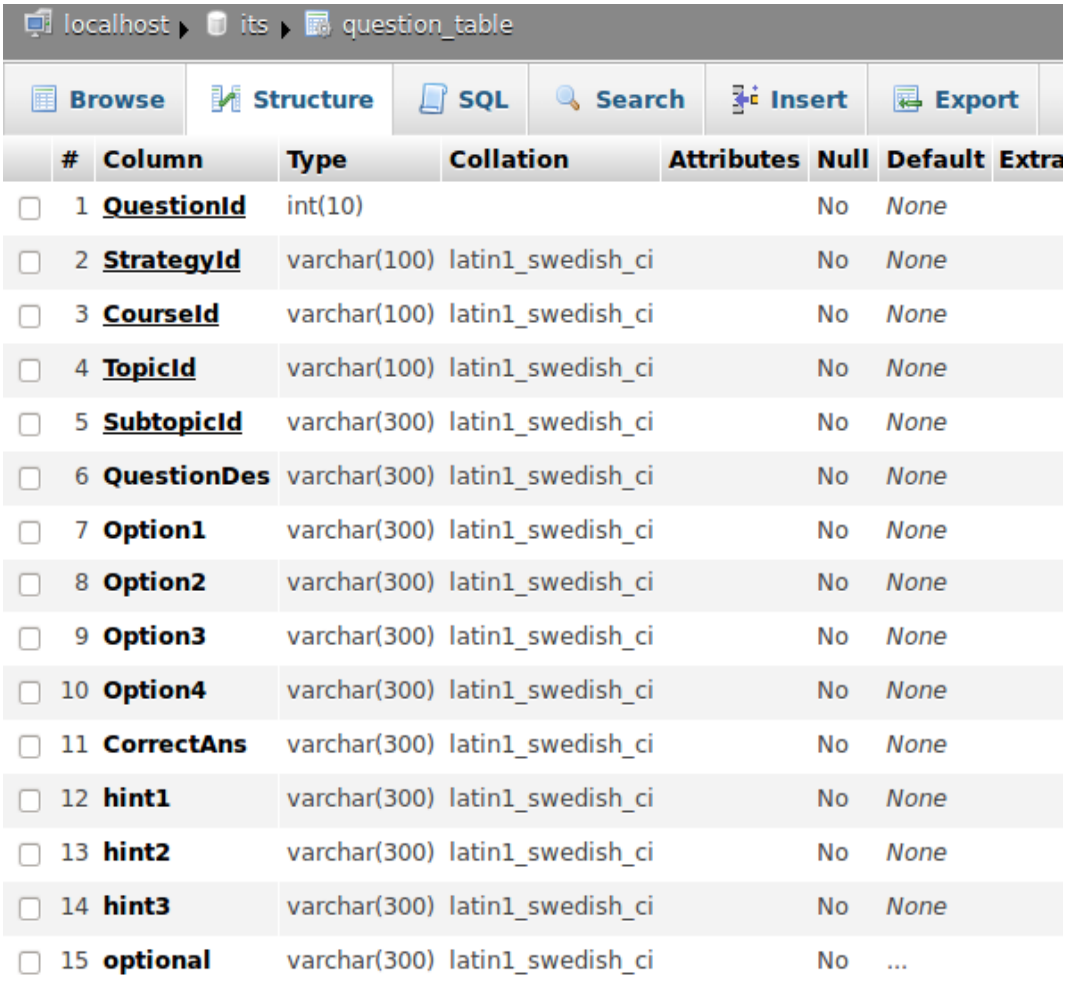
#	Column	Type	Collation	Attributes	Null	Default	Extra
1	**CourseId**	varchar(15)	latin1_swedish_ci		No	*None*	
2	**TopicName**	varchar(15)	latin1_swedish_ci		No	*None*	
3	**TopicId**	varchar(15)	latin1_swedish_ci		No	*None*	
4	**TopicDes**	varchar(50)	latin1_swedish_ci		No	*None*	

Figure A.3: Topic Table.



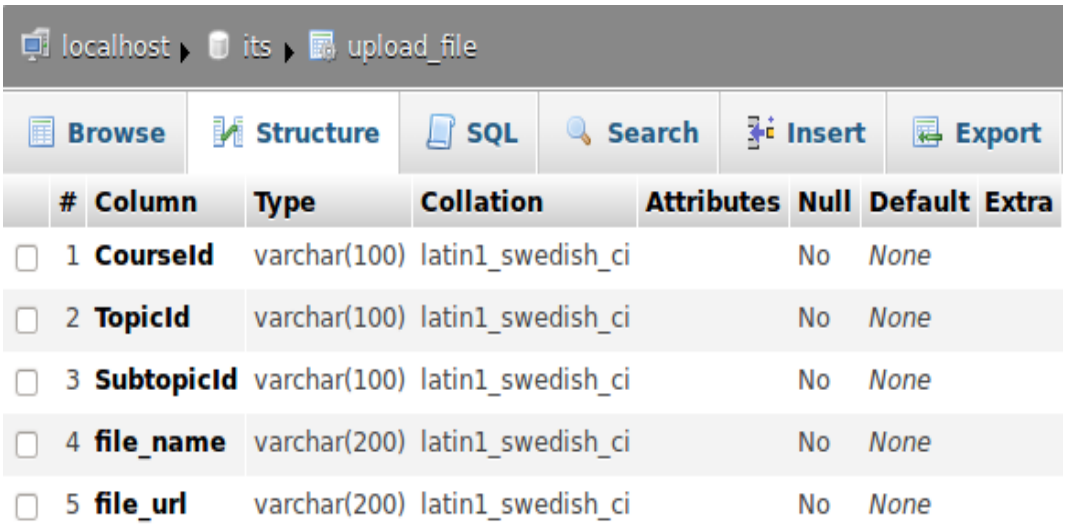
#	Column	Type	Collation	Attributes	Null	Default	Ext
1	**CourseId**	varchar(15)	latin1_swedish_ci		No	*None*	
2	**TopicId**	varchar(15)	latin1_swedish_ci		No	*None*	
3	**SubtopicName**	varchar(15)	latin1_swedish_ci		No	*None*	
4	**SubtopicId**	varchar(15)	latin1_swedish_ci		No	*None*	
5	**SubtopicDes**	varchar(50)	latin1_swedish_ci		No	*None*	

Figure A.4: Subtopic Table.



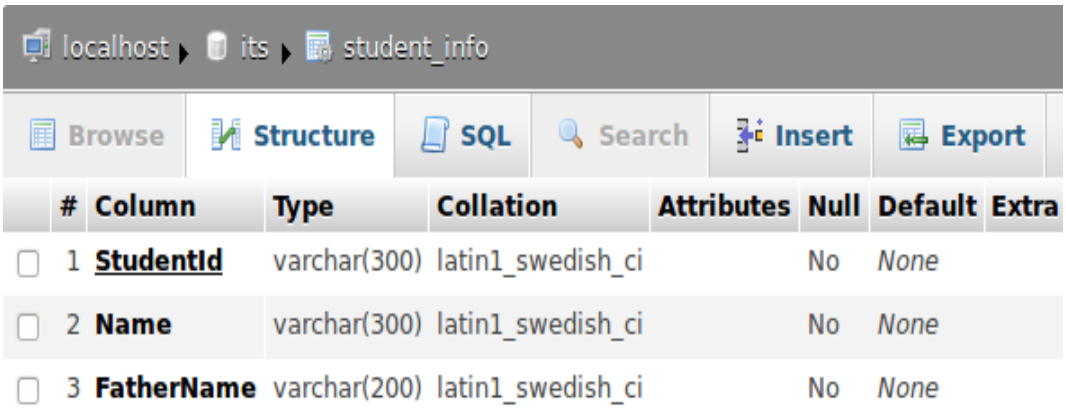
#	Column	Type	Collation	Attributes	Null	Default	Extra
1	**QuestionId**	int(10)			No	*None*	
2	**StrategyId**	varchar(100)	latin1_swedish_ci		No	*None*	
3	**CourseId**	varchar(100)	latin1_swedish_ci		No	*None*	
4	**TopicId**	varchar(100)	latin1_swedish_ci		No	*None*	
5	**SubtopicId**	varchar(300)	latin1_swedish_ci		No	*None*	
6	**QuestionDes**	varchar(300)	latin1_swedish_ci		No	*None*	
7	**Option1**	varchar(300)	latin1_swedish_ci		No	*None*	
8	**Option2**	varchar(300)	latin1_swedish_ci		No	*None*	
9	**Option3**	varchar(300)	latin1_swedish_ci		No	*None*	
10	**Option4**	varchar(300)	latin1_swedish_ci		No	*None*	
11	**CorrectAns**	varchar(300)	latin1_swedish_ci		No	*None*	
12	**hint1**	varchar(300)	latin1_swedish_ci		No	*None*	
13	**hint2**	varchar(300)	latin1_swedish_ci		No	*None*	
14	**hint3**	varchar(300)	latin1_swedish_ci		No	*None*	
15	**optional**	varchar(300)	latin1_swedish_ci		No	...	

Figure A.5: Question Table.



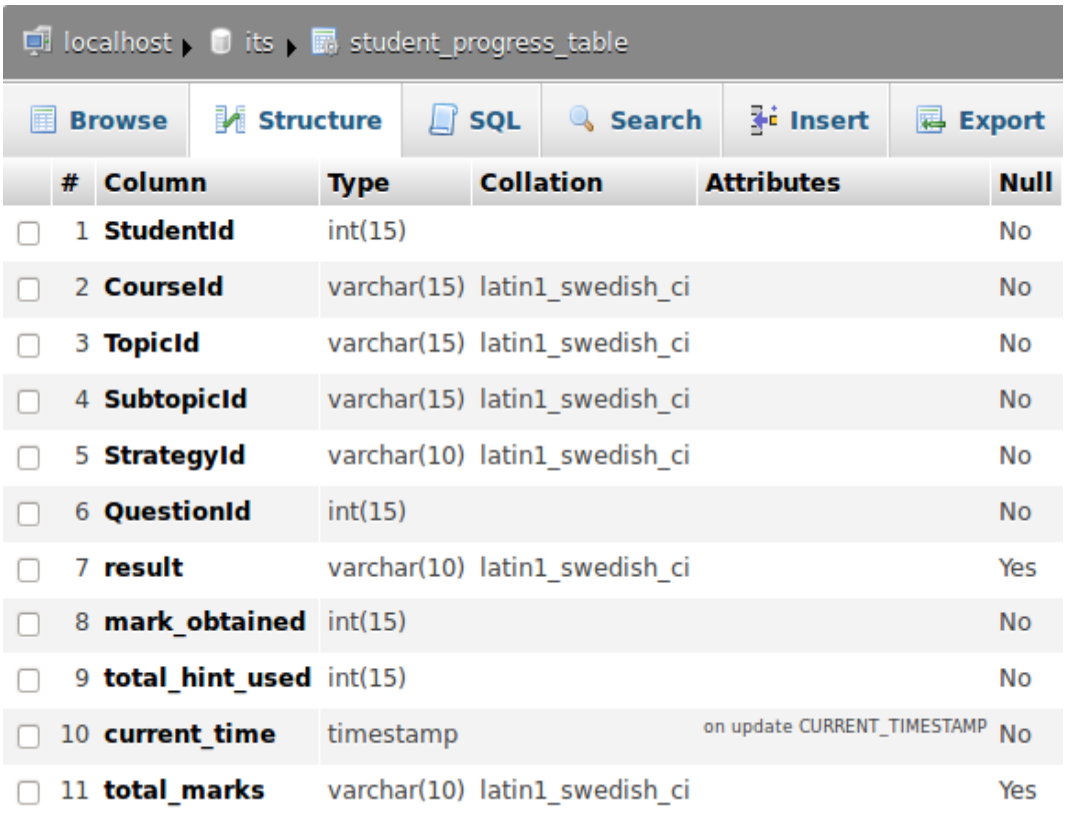
#	Column	Type	Collation	Attributes	Null	Default	Extra
1	**CourseId**	varchar(100)	latin1_swedish_ci		No	*None*	
2	**TopicId**	varchar(100)	latin1_swedish_ci		No	*None*	
3	**SubtopicId**	varchar(100)	latin1_swedish_ci		No	*None*	
4	**file_name**	varchar(200)	latin1_swedish_ci		No	*None*	
5	**file_url**	varchar(200)	latin1_swedish_ci		No	*None*	

Figure A.6: Upload Resource Table.



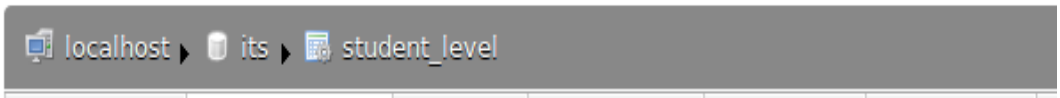
#	Column	Type	Collation	Attributes	Null	Default	Extra
1	**StudentId**	varchar(300)	latin1_swedish_ci		No	*None*	
2	**Name**	varchar(300)	latin1_swedish_ci		No	*None*	
3	**FatherName**	varchar(200)	latin1_swedish_ci		No	*None*	
4	**DOB**	date			No	*None*	
5	**Education**	varchar(300)	latin1_swedish_ci		No	*None*	
6	**Email**	varchar(300)	latin1_swedish_ci		No	*None*	

Figure A.7: Student Information Table.



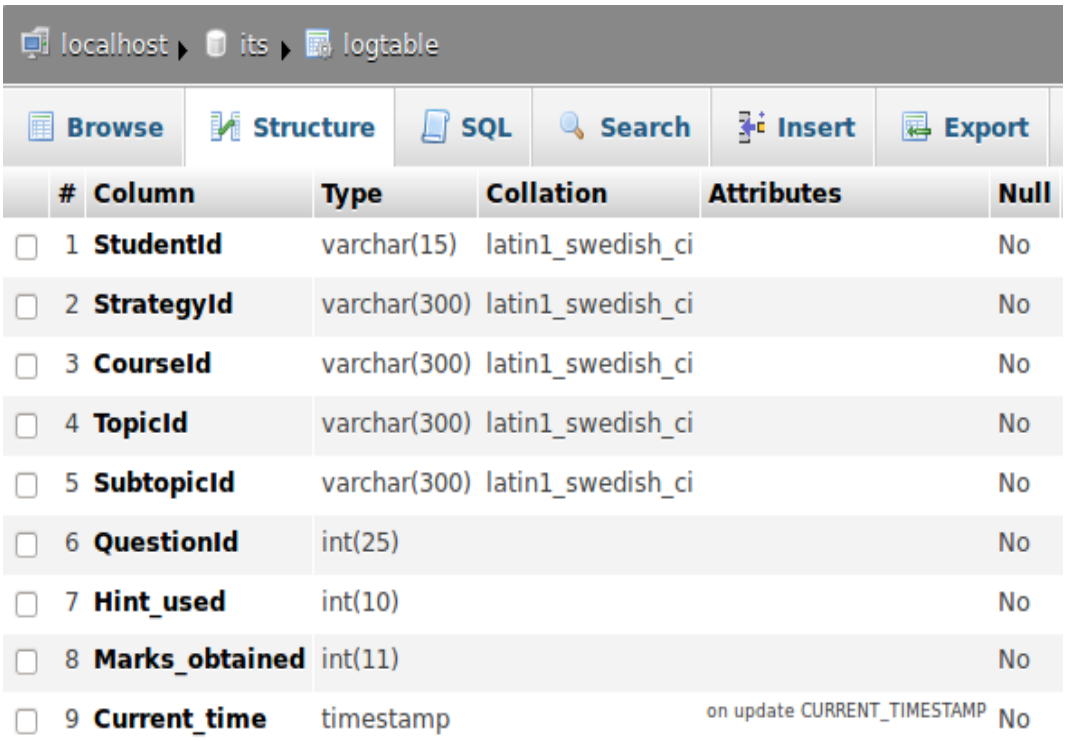
#	Column	Type	Collation	Attributes	Null
1	**StudentId**	int(15)			No
2	**CourseId**	varchar(15)	latin1_swedish_ci		No
3	**TopicId**	varchar(15)	latin1_swedish_ci		No
4	**SubtopicId**	varchar(15)	latin1_swedish_ci		No
5	**StrategyId**	varchar(10)	latin1_swedish_ci		No
6	**QuestionId**	int(15)			No
7	**result**	varchar(10)	latin1_swedish_ci		Yes
8	**mark_obtained**	int(15)			No
9	**total_hint_used**	int(15)			No
10	**current_time**	timestamp		on update CURRENT_TIMESTAMP	No
11	**total_marks**	varchar(10)	latin1_swedish_ci		Yes

Figure A.8: Student Progress Table.



#	Column	Type	Collation	Attributes	Null	Default	Extra
1	**StudentId**	varchar(20)	latin1_swedish_ci		No	*None*	
2	**overall_level**	int(10)			No	*None*	

Figure A.9: Student Level Table.



	#	Column	Type	Collation	Attributes	Null
☐	1	**StudentId**	varchar(15)	latin1_swedish_ci		No
☐	2	**StrategyId**	varchar(300)	latin1_swedish_ci		No
☐	3	**CourseId**	varchar(300)	latin1_swedish_ci		No
☐	4	**TopicId**	varchar(300)	latin1_swedish_ci		No
☐	5	**SubtopicId**	varchar(300)	latin1_swedish_ci		No
☐	6	**QuestionId**	int(25)			No
☐	7	**Hint_used**	int(10)			No
☐	8	**Marks_obtained**	int(11)			No
☐	9	**Current_time**	timestamp		on update CURRENT_TIMESTAMP	No

Figure A.10: Log Table.

Appendix B

Source Code

Figure B.1 shows the AJAX function for showing hint to the student in case student attempt wrong answer.

```
function showHint(str,str2,str3,str4,str5,str6)
{
    var xmlhttp;
    if (str.length==0)
    {
        document.getElementById("txtHint").innerHTML="";
        return;
    }
    if (window.XMLHttpRequest)
        {// code for IE7+, Firefox, Chrome, Opera, Safari
            xmlhttp=new XMLHttpRequest();
        }
    else
    {// code for IE6, IE5
    xmlhttp=new ActiveXObject("Microsoft.XMLHTTP");
    }
    xmlhttp.onreadystatechange=function()
    {
        if (xmlhttp.readyState==4 && xmlhttp.status==200)
        {
            if(xmlhttp.responseText)
                alert("Please see hint");
            document.getElementById("myDiv").innerHTML=xmlhttp.responseText;
        }
    }
    xmlhttp.open("GET","gethint.php?opt="+str+"&qid="+str2+"&cid="+str3+"&tid="+str4+"&stid="+str5+"&l="+str6,true);
    xmlhttp.send();
}
```

Figure B.1: AJAX code for showing hint.

Table B.1: Source code description

Folder/file Name	**Folder/file's work description**
course_create.php	This file is used to create the new course.
topic_create.php	This file is used to create the new topic.
subtopic_create.php	This file is used to create the subtopic.
select	this folder is used select the course, topic and subtopic for which resorses are uploading.
login_module/student	This folder contains files which used by student for login and register.
login_module/student/index.php	This file is used for authentication and validation of student.
login_module/student/register.php	This file is used for registering the new student.
login_module/teacher	This folder contains files which used by teacher for login and register.
login_module/teacher/index.php	This file is used for authentication and validation of teacher.
login_module/teacher/register.php	This file is used for registering the new teacher.
teacher_module/self_store_question.php	This file is used to store the questions in new course.
teacher_module/store_question.php	This file is used to store the new questions in existing course.
teacher_module/student progress.php	This file is used to show the progress of particular student.
teacher_module/student progress	This folder contains all files used for selecting the student whom progress report is required by teacher.
student_module/attemptAgain.php	This files is used to attempt the subtopic which is already has been done by the student.
student_module/my _progress.php	This file is used to see his/her progress report by student.
student_module/submitQuiz.php	This file is used to submit the answers of questions.
student_module/gethint.php	This file is used by the ITS to provide appropriate hint to the student.This file also update the level number of student according to answer of student.
student_module/files.php	This file shows the all available resources for requested subtopic.
student_module/my _level.php	This file is used to fetch the current level number of student.
student_module/startQuiz.php	This file is used to start attempting question from subtopic.

Appendix C

Example Questions

C.1 Question for subtopic "Data structure & Algorithm"

Course-Id: CS101
Topic-Id: C
Subtopic-Id: DSA (Data structure & Algorithm)

1. **Question Description:** Two main measures for the efficiency of an algorithm are

   Option1: Processor and memory
   Option2: Complexity and capacity
   Option3: Time and space
   Option4: Data and space
   Correct answer: Time and space

   Hint-1: Processor does not affect efficiency of algorithm
   Hint-2: Efficiency does not depend on Complexity of Algorithm
   Hint-3: Efficiency has no relation with Data
   Comment: ...

2. **Question Description:** The time factor when determining the efficiency of algorithm is measured by

   Option1: Counting microseconds
   Option2: Counting the number of key operations
   Option3: Counting the number of statements
   Option4: Counting the kilobytes of algorithm
   Correct answer: Counting the number of key operations

Hint-1: It is difficult to get run-time of key operations by dry run
Hint-2: Statement can be redundant
Hint-3: An algorithm having more size can be efficient
Comment: ...

3. **Question Description:** The space factor when determining the efficiency of algorithm is measured by

   Option1: Counting the maximum memory needed by the algorithm
   Option2: Counting the minimum memory needed by the algorithm
   Option3: Counting the average memory needed by the algorithm
   Option4: Counting the maximum disk space needed by the algorithm
   Correct answer: Counting the maximum memory needed by the algorithm

   Hint-1: We should count maximum space needed
   Hint-2: Disk space is not an option
   Hint-3: There is no case such as average space
   Comment: ...

4. **Question Description:** Which of the following case does not exist in complexity theory

   Option1: Best case
   Option2: Worst case
   Option3: Average case
   Option4: Null case
   Correct answer: Counting the maximum memory needed by the algorithm

   Hint-1: There is nothing called NULL case
   Hint-2: Most favorable conditions called Best case
   Hint-3: Most infavorable conditions called Worst case
   Comment: ...

5. **Question Description:** The Worst case occur in linear search algorithm when

   Option1: Item is somewhere in the middle of the array
   Option2: Item is not in the array at all
   Option3: Item is the last element in the array
   Option4: Item is the last element in the array or is not there at all
   Correct answer: Item is the last element in the array or is not there at all

   Hint-1: Worst case occur when all element of array is traversed

Hint-2: When element is not present in array and present at last then only all element will be traversed
Hint-3: If Item is in middle then only half element will be traversed
Comment: ...

6. **Question Description:** The Average case occur in linear search algorithm

   Option1: When Item is somewhere in the middle of the array
   Option2: When Item is not in the array at all
   Option3: When Item is the last element in the array
   Option4: When Item is the last element in the array or is not there at all
   Correct answer: When Item is somewhere in the middle of the array

   Hint-1: Worst case occur when all element of aray is traversed
   Hint-2: When element is not present in array and present at last then only all element will be traversed
   Hint-3: If Item is in middle then only half element will be traversed
   Comment: ...

7. **Question Description:** The complexity of linear search algorithm is

   Option1: O(n)
   Option2: O(log n)
   Option3: O(n2)
   Option4: O(n log n)
   Correct answer: O(n)

   Hint-1: In linear search algorithm each element is traversed till element is not found
   Hint-2: Binary search has O(log n) complexity
   Hint-3: Complexity of linear search algorithm is directly proportional to number of element
   Comment: ...

8. **Question Description:** The complexity of Bubble sort algorithm is

   Option1: O(n)
   Option2: O(log n)
   Option3: O(n2)
   Option4: O(n log n)
   Correct answer: O(n2)

Hint-1: O(n) can not be an option
Hint-2: If we are dividing array at each step then only O(n log n) complexity is possible
Hint-3: Bubble sort compare each element with all other element
Comment: ...

9. **Question Description:** The complexity of Binary search algorithm is

Option1: O(n)
Option2: O(log n)
Option3: O(n2)
Option4: O(n log n)
Correct answer: O(log n)

Hint-1: In this Algorithm array is divided in two parts at each step
Hint-2: O(n2) can not be an option because in searching we have to only traverse the elements
Hint-3: O(n) is a complexity of linear search
Comment: ...

10. **Question Description:** Which of the following data structure is not linear data structure?

Option1: Arrays
Option2: Linked lists
Option3: Both of above
Option4: None of above
Correct answer: None of above

Hint-1: In linear data structure all elements are traverse linearly
Hint-2: Arrays are traversed linearly
Hint-3: liked list are traversed linearly
Comment: ...

C.2 Question for subtopic "Pointers"

Course-Id: CS101
Topic-Id: C
Subtopic-Id: PTR (Pointers)

1. **Question Description:** What is (void*)0?

   Option1: Representation of NULL pointer
   Option2: Representation of void pointer
   Option3: Error
   Option4: None of above
   Correct answer: Representation of NULL pointer

   Hint-1: void *a is a representation of void pointer
   Hint-2: NULL pointer is present in C
   Hint-3: there is no error in this representation
   Comment: ...

2. **Question Description:** In which header file is the NULL macro defined?

   Option1: stdio.h
   Option2: stddef.h
   Option3: stdio.h and stddef.h
   Option4: stdlib.h
   Correct answer: stdio.h and stddef.h

   Hint-1: stdlib.h does not contain NULL macro defination
   Hint-2: NULL macro is also defined in stdio.h
   Hint-3: The macro "NULL" is defined in locale.h, stddef.h, stdio.h, stdlib.h, string.h, time.h, and wchar.h.
   Comment: ...

3. **Question Description:** How many bytes are occupied by near, far and huge pointers (DOS)?

   Option1: near=2 far=4 huge=4
   Option2: near=4 far=8 huge=8
   Option3: near=2 far=4 huge=8
   Option4: near=4 far=4 huge=8
   Correct answer: near=2 far=4 huge=4

Hint-1: near occupied 2 bytes
Hint-2: far occupied 4 bytes
Hint-3: huge occupied 4 bytes
Comment: ...

4. **Question Description:** If a variable is a pointer to a structure, then which of the following operator is used to access data members of the structure through the pointer variable?

   Option1: .
   Option2: &
   Option3: *
   Option4: $->$
   Correct answer: $->$

   Hint-1: . is used to access data member of structure by simple object of that data structure
   Hint-2: * is used to dereference the pointer
   Hint-3: $->$ is used by pointer variables
   Comment: ...

5. **Question Description:** A pointer is

   Option1: A keyword used to create variables
   Option2: A variable that stores address of an instruction
   Option3: A variable that stores address of other variable
   Option4: All of the above
   Correct answer: A variable that stores address of other variable

   Hint-1: A pointer does not stores address of an instruction
   Hint-2: pointer is used to create pointer to variable not variable.
   Hint-3: pointer store the address of variable
   Comment: ...

6. **Question Description:** The operator used to get value at address stored in a pointer variable is

   Option1: *
   Option2: &
   Option3: &&
   Option4: ||
   Correct answer: *

Hint-1: & is a bitwise operator
Hint-2: && is used for ANDing two variables
Hint-3: |||| is used for ORing two variables
Comment: ...

7. **Question Description:**
#include

void main()
{
int a = 2;
switch(a)
{
case 1:
printf(”goodbye”);
break;
case 2:
continue;
case 3:
printf(”bye”);
} }

Option1: error
Option2: 10
Option3: 10 10
Option4: None of above
Correct answer: error

Hint-1: ??
Hint-2: ??
Hint-3: ??
Comment: ...

8. **Question Description:**
int y[4] = 6, 7, 8, 9;
int *ptr = y + 2; printf(”%d”, ptr[1]);

What is printed when the sample code above is executed?

Option1: 6

Option2: 7
Option3: 8
Option4: 9
Correct answer: 9

Hint-1: array name denotes the base address of array
Hint-2: pointers allowed addition and subtraction.
Hint-3: In this question ptr is pointing the address of third element e.i 8..
Comment: ...

C.3 Question for subtopic "Miscellaneous questions"

Course-Id: CS101
Topic-Id: C language
Subtopic-Id: MSL (Miscellaneous questions)

1. **Question Description:** What will be the output of the following arithmetic expression ?
   5+3*2%10-8*6

   Option1: -37
   Option2: -42
   Option3: -32
   Option4: -28
   Correct answer: -37

   Hint-1: *, / and % have same precedence level
   Hint-2: + and - has lower precedence level then multiplication
   Hint-3: All these operator (+, -, *, /, %) have left to right associativity
   Comment: ...

2. **Question Description:** What will be the output of the following statement ?
   int a = 4, b = 7,c;
   c = a == b;
   printf("%i",c);

   Option1: 0
   Option2: error
   Option3: garbage value
   Option4: 1
   Correct answer: 0

Hint-1: == is used for comparing values
Hint-2: = is an assignment operator
Hint-3: = operator has right to left associativity
Comment: ...

3. **Question Description:** What will be the output of the following statements ?
int a = 5, b = 2, c = 10, i = $a > b$;
void main()
{
printf(”hello”);
main();
}

Option1: 1
Option2: 2
Option3: infinite number of times
Option4: none of these
Correct answer: infinite number of times

Hint-1: Variable can be defined before void main()
Hint-2: main function can be called like an ordinary function
Hint-3: Recursive call is there
Comment: ...

4. **Question Description:** Which of the following case does not exist in complexity theory

Option1: Best case
Option2: Worst case
Option3: Average case
Option4: Null case
Correct answer: Counting the maximum memory needed by the algorithm

Hint-1: There is nothing called NULL case
Hint-2: Most favorable conditions called Best case
Hint-3: Most unfavorable conditions called Worst case
Comment: ...

5. **Question Description:** What will be the output of the following statements ?
int x[4] = 1,2,3; printf(”%d %d %D” ,x[3],x[2],x[1]);

Option1: 03D
Option2: 000
Option3: 032
Option4: 321
Correct answer: 032

Hint-1: Array index start with 0
Hint-2: D is permissible in place of d
Hint-3: default value of local array element is 0
Comment: ...

6. **Question Description:** What will be the output of following program ?
```
main()
{
int x,y = 10;
x = y * NULL;
printf("%d",x);
}
```

Option1: error
Option2: 0
Option3: garbage value
Option4: 10
Correct answer: 0

Hint-1: NULL is a macro defined in stdlib.h
Hint-2: NULL is defined as zero
Hint-3: Macro just replaced by their value at compile time
Comment: ...

7. **Question Description:** What will be the output of following statements ?
char x[ ] = "hello hi"; printf("%d%d", sizeof(*x), sizeof(x));

Option1: 88
Option2: 18
Option3: 29
Option4: 19
Correct answer: 19

Hint-1: size of character pointer is 1
Hint-2:

0 is added by default at the last in character arrays and in strings
Hint-3: x is an array of 9 character
Comment: ...

8. **Question Description:** What will be the output of the following program ?

```
void main()
{
int a = 36, b = 9;
printf("%d",a >> a/b – 2);
}
```

Option1: 9
Option2: 7
Option3: 5
Option4: none of these
Correct answer: 5

Hint-1: order of precedence is / , - and then >>
Hint-2: >> is a bitwise right shift operator
Hint-3: answer is (36 >> 2) or (100100 >> 2) in binary
Comment: ...

9. **Question Description:** void main()
{
int a=10,b=20;
char x=1,y=0;
if(a,b,x,y)
{
printf("EXAM");
}
What is the output?

Option1: XAM is printed
Option2: exam is printed
Option3: Compiler Error
Option4: Nothing is printed
Correct answer: Nothing is printed

Hint-1: comma operator is allowed in if condition.
Hint-2: comma operator is processed from right to left

Hint-3: Right element is 0 so programm will not go in if condition.
Comment: ...

10. **Question Description:** What will be the value of ‘a‘ after the following code is executed

```
# define square(x) x*x
a = square(2+3)
```

Option1: 25
Option2: 13
Option3: 11
Option4: 10
Correct answer: 11

Hint-1: #define is a macro and macro is replaced by the corresponding value at the pre-proccessing time.
Hint-2: so square(x) will be replaced by x*x.
Hint-3: After macro expansion code will become a = 2+3*2+3
Comment: ...

11. **Question Description:**

```
void func()
{
int x = 0;
static int y = 0;
x++; y++;
printf( ”%d – %d ”, x, y );
}

int main()
{
func();
func();
return 0;
}
```

What will the code above print when it is executed?

Option1: 1 – 1 1 – 1
Option2: 1 – 1 2 – 1
Option3: 1 – 1 2 – 2
Option4: 1 – 1 1 – 2

Correct answer: 1 – 1 1 – 2

Hint-1: static variables share their values between function calls
Hint-2: Loacl variables lost their value when program exit from the function
Hint-3: In this case y is static variable and x is local variable.
Comment: ...

12. **Question Description:** long factorial (long x)
{
???? return x * factorial(x - 1);
}

With what do you replace the ???? to make the function shown above return the correct answer?

Option1: if (x == 0) return 0;
Option2: return 1;
Option3: if (x ¡= 1) return 1;
Option4: if (x ¿= 2) return 2;
Correct answer: if (x ¡= 1) return 1;

Hint-1: the factorial of a non-negative integer n, denoted by n!, is the product of all positive integers less than or equal to n.
Hint-2: This is recursion function of factorial. factorial function is called until x is equal to base value
Hint-3: factorial of 0 is 1 and factorial of 1 is also 1 and base value is 0 and 1.
Comment: ...

Bibliography

[1] *ITSs:* http://aaai.org/AITopics/IntelligentTutoringSystems

[2] *Moodle:* http://dev.moodle.org/mod/page/view.php?id=34.

[3] McKenzie, J. (2000). Scaffolding for Success. [Electronic version] Beyond Technology, *Questioning, Research and the Information Literate School Community. Retrieved October 12, 2002 from* http://fno.org/dec99/scaffold.html

[4] Jamie McKenzie, *Beyond Technology: Questioning, Research and the Information Literate School Community* http://serc.carleton.edu/NAGTWorkshops/webdesign/Scaffolding/

[5] MERLOT Pedagogy Portal *Teaching Strategies :* http://pedagogy.merlot.org/TeachingStrategies.html

[6] *MIT:* http://info.scratch.mit.edu/

[7] Sawyer, R. Keith. (2006). *The Cambridge Handbook of the Learning Sciences. New York: Cambridge University Press*

[8] Van Der Stuyf, R. Rachel, *Scaffolding as a Teaching Strategy, Adolescent Learning and Development Section 0500A - Fall 2002 November 17, 2002*

[9] *Razzaq, L., Heffernan, N. T. (Submitted) What level of tutor feedback is best? In Luckin and Koedinger (Eds) Proceedings of the 13th Conference on Artificial Intelligence in Education. IOS Press.*

[10] Ivon Arroyo, Rena Walles, Carole R. Beal, Beverly P. Woolf, *Tutoring for SAT-Math with Wayang Outpost,* University of Massachusetts, Amherst

[11] B. Cheung, L. Hui, J. Zhang, S.M. Yiu, *SmartTutor: An intelligent tutoring system in web-based adult education, Journal of Systems and Software (2003),* http://www.sciencedirect.com/science/article/pii/S0164121202001334

[12] *L.S. Vygotsky: Mind in Society: Development of Higher Psychological Processes, p. 86*

[13] Bransford, J. D., Brown, A. L., and Cocking, R. R. (2000), *How People Learn: Brain, Mind, Experience, and School: Expanded Edition. Washington, D. C*

[14] Beth Lewis, *Scaffolding Instruction,* http://k6educators.about.com/od/helpfornewteachers/a/scaffoldingtech.htm

[15] Ivon Arroyo, Rena Walles, Carole R. Beal, Beverly P. Woolf, *Tutoring for SAT-Math with Wayang Outpost,* University of Massachusetts, Amherst

[16] Yogendra Pal's APS report: *INTELLIGENT TUTORING SYSTEM TO TEACH PROGRAMMING TO BILINGUAL STUDENTS, IIT Bombay*

[17] Ayturk Keles, Rahim Ocak, Ali Keles, *ZOSMAT: Web-based intelligent tutoring system for teaching–learning process,* Expert Systems with Applications 36 (2009)

[18] Beverly Park Woolf, *Building Intelligent Interactive Tutors Student-centered strategies for revolutionizing e-learning*

[19] Ivon Arroyo, Carole Beal, Tom Murray, Rena Walles, Beverly Woolf, *Wayang Outpost: Intelligent Tutoring for High Stakes Achievement Tests , University of Massachusetts, Amherst*

[20] Linda Lawson, *Scaffolding Works as a Teaching Strategy* City College, EDUC 0500, November 2002 from http://condor.admin.ccny.cuny.edu/~group4/Lawson/Lawson%20Paper.doc

[21] Riehle, Dirk (2000), *Framework Design: A Role Modeling Approach,* Swiss Federal Institute of Technology

[22] Arroyo, Beck, Schultz & Woolf, 1999; Beal & Arroyo, in press www.carnegielearning.com

[23] Vikash Kumar, *Development of an Intelligent Tutoring System Framework for Socratic Questioning,* M-Tech Thesis, IIT Bombay 2012

[24] M. Rajashekhar, *Development of an Intelligent Tutoring System Framework for Guided Discovery,* M-Tech Thesis, IIT Bombay 2012

[25] Praveen Dhanala, *Development of an Intelligent Tutoring System Framework for Game Based Learning,* M-Tech Thesis, IIT Bombay 2012

Acknowledgment

I express my deepest gratitude for my supervisor ***Prof. Sridhar Iyer***, who has always been making things simple to understand. Without his deep insight into this domain and his valuable time for this report, it would not have been possible for me to move ahead properly. He has been remarkable in his attempt to keep me motivated in this project and has always tried to improve me with proper feedback.

Signature :

Name : Chandra Pal Singh

Date :