A Tool for Creating Exams Automatically from an Object-Oriented Knowledge Base Question Bank

Khaled N. Elsayed

***Abstract*—The way of creating exams can enhance education. Also, the exam should cover majority of course material and should include various levels of difficulties. This paper presents a tool designed for automatically creating exams by selecting questions from a bank of questions for several courses. This bank of question is designed as an object-oriented knowledge base. Its questions should emphasize on the structuring categories of all course domains. An intelligent agent will help in selecting questions. It will watch the examiners and learn from their experience, at editing questions or at creating exams. The presented tool could be applied in several courses and can be used at various education level and nature. It is designed to create online exam, as well as, offline one. Also, the tool will produce a key for the produced exam. So, this tool can extend the E-Learning and provide more success in distant education.**

***Index Terms*—E-learning, exams, intelligent agent, object oriented knowledge base.**

I. Introduction

With the advent of computer technology, educational software programs in science education have become increasingly complex in both concept and design. Thus, there is a paradigm shift in the role of computer from solely a transmitter of knowledge to a tool that aids in the construction of knowledge. This is due to the fact the science concepts are abstracts, as in [1].

In these years, Distance learning is the hot issue in computer science. Online learning through the web has become popular in the decade [2]. E-Learning is nowadays recognized as one of the efficient methods to respond to the requirements of open and distance learning. In the e-learning system, several traditional learning styles should be combined with the learner-centered approach. It needs a good notation to represent the requirements of the e-learning system [3].

In the time being, Artificial Intelligence algorithms, techniques, and applications have wide use in education and tutoring systems. One of the AI techniques is the intelligent agent, which can be used in education. Computers have become essential tools in developing systems that caters to the different needs of users. An intelligent agent is an autonomous calculated entity.

In the dynamic changes information environment without prior modeling, it can independently plan complex operation steps to solve practical problems, can independently discover and obtain the available resources the learners needed and then provide the corresponding services under the circumstance that the learners do not take part in [4].

The agent has to collect users' personal interests and give fast response according to the pre-specified demands of users. The personal agent can discover users' personal interests voluntarily without bothering the users. It is very suitable for personalized e-learning by voluntarily recommending learning materials [5].

An agent is something that perceives and acts in an environment. The agent function for an agent specifies the action taken by the agent in response to any percept sequence as in [6]. Intelligent agents are task-oriented software components that have the ability to act intelligently. They may contain more knowledge about the needs, preferences and pattern of the behaviors of a person or a process as in [7].

Currently, the state of intelligent is focused on one-to-one learning instruction. Some examples include ACT systems [1], DEBUGGY [8], and PIXIE [9]. The kind of learning modality used is centered on learning by being told [10].

Intelligent agents should have the ability of adaptive reasoning. They must have the capability to access information from other sources or agents and perform actions leading to the completion of some task. Also, they must control over their internal state and behavior and work together to perform useful and complex tasks. Thus, they should be able to examine the external environment and the success of previous actions taken under similar conditions and adapt their actions [11].

In most of the exist software developed for creating exams, the user selects questions himself or the question are selected randomly. Many of them provide automatic grading. As examples of these systems; ProPrfos [12], Classmaker [13], and Adit Sofware [14]. The presented tool, provides automatic selection of questions and building exams and gets its inelegance from learning from examiner to enhance selection of questions.

The presented tool is built to be able to build an object-oriented knowledge base (OOKB) as a bank of questions and acquire its knowledge. Also, it is able to, automatically, from exercises by selecting questions from that OOKB bank and build exams according to some specification done by the examiner and based on an intelligent agent that helps in selection of questions. It is designed to create exams, quizzes, assessment for education in academic institutions, training centers, human resources departments, and anyone who want to create his own exams.

That agent will get its experience by watching the examiners, in two different situations. First, at editing a question; when the examiner specifies that some question(s)

Manuscript received July 26, 2012; revised December 13, 2012.

K. N. Elsayed is with the Computer Science Department, Umm Al-Qura University,Makkah AlMokaramah, Saudi Arabia Kingdom (e-mail: knsayed@uqu.edu.sa, khnasser2000@yahoo.com).

DOI: 10.7763/IJIET.2013.V3.228

are preferred to be/ should prevented from being in the same exam with that question. Secondly, at exam creation; when the examiner asks to remove certain question(s) or add certain ones.

Questions to select from, should be distributed over the OOKB question bank. This OOKB is designed and organized in certain manner that enables the tool to select questions randomly according the examiner specifications and exam factors, which are supervised by the tool's agent.

At editing questions to be acquired in the OOKB bank, the editor will provide the OOKB bank with certain information to be used later as a basis for the tool-based agent to select a set of questions for certain exam. It should takes in consideration, that the exam should cover majority or even specified parts of course material and should include various levels of difficulties.

For online exams, the presented tool is a comprehensive solution for creating quizzes, assessments, or exams on the Internet and Intranet and grading those exams. The tool can easily select all exercises types and format the text, add graphics, formulas in offline and online exams. In the offline exams, the tool can produces an exam to be printed and copied to be distributed over students in a class. Also, it will produce a key for each exam, to help the examiner to check answers of the examined people, and grading their exam.

It is a quick and professional way to create and organize tests for employees, students, and people in training. The tool can check answers of the examined people and give the final grades for each of them. The presented tool is tested and applied in the Concepts of Programming Language course.

II. Structure of the Tool

Object-oriented analysis of application domains is used in defining domain as objects, like instances, attributes, categories, tests and procedures, and relations between these objects. This is the view of the presented tool on the presented domain. The components of the proposed tool are designed as general as possible to satisfy this requirement. This results in using the tool in multiple domains.

Applying object-oriented analysis to the application domain of exam is used in defining domain as objects, like courses, chapters, exams exercises, questions, attributes, answers and relations between these objects. The proposed tool consists of four parts, each part cooperates with the other to prove the generality of the tool. Also, the use of object-oriented techniques in analyzing of the domains of applications leads to that generality. The structure of this tool is shown in Fig. 1.

- **User Interface**; which is a menu-driven dialogs to help the user to interact with the tool directly in easy manner. The user could present question to the tool and/or create exam in his course through sessions.

- **Tool Engine**; which involves an intelligent agent to manages the process of questions selection to build exams. The agent uses random and mathematical functions in additions to the knowledge resides in the relations between different questions to pick the best choices from many enumerated possibilities. Agent manipulates questions and other objects and the in-between relations in a standard way with all presented courses. It watches the examiner when he requests to remove certain questions from the exam produced by the tool, or add certain question. It affects on the knowledge in the relation between questions for next exam creations.

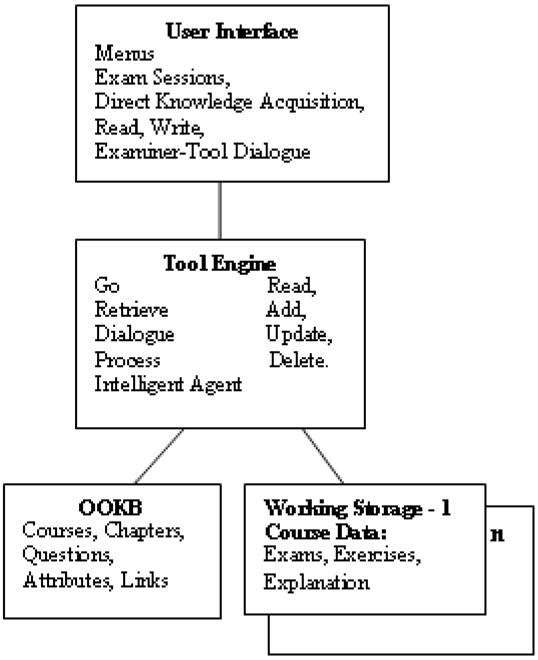

Fig. 1. Structure of the presented tool.

- **OOKB**; which is a semantic network of nodes like: exams, exercise, questions, answers and links (relations between nodes). Its structure is based on object oriented analysis of application domains, to distribute the retained questions from the examiners. It holds an indexing structure used by the inference engine to locate questions. Each node or link is represented by an object of a suitable class.

- **Working Storages**; which could be created and manipulated isolated from the OOKB of the tool. Each working storage is assigned to certain course, and while one is active, the tool is specialized in certain course, and can be switched from one to other.

III. Exercises, Questions, and Answers

The major important part of the proposed tool is the OOKB that holds a huge number of questions and their answers of all courses defined in the OOKB. It is the infrastructure of the tool. In this section, the questions and their answers, that could be edited, are presented. The tool allows four types of exercises, each include number of questions of that type.

- **True/False Exercises**; which include questions, each of them is written as a text (sentence), while the answer should be one of two options (true or false), it is a logical choice. Answer of this type of question might be checked and graded automatically or manually, by the examiner. An example of this type is shown in Fig. 2 (a).

- **Multiple Choice Exercises**; which include questions, each of them is written as a text (sentence), while the answer is one correct choice from the available multiple choices (4 choices are available). Answer of this type of question might be checked and graded automatically or manually, by the examiner. An example of this type is shown in Fig. 2 (b).

- **Fill in the Blanks Exercises**; which include questions, each of them is written as a text (sentence), while the answer is one selected word form a pool of words. So, the process of filling in the spaces looks like matching certain question with a suitable answer. The answer of this type of question might be checked and graded automatically or manually, by the examiner. An example of this type is shown in Fig. 2 (c).

a. **The first compiler of FORTRAN I was written in *Cobol* language.**

Answer: □ **True** ☑**False**

b.

	Which of the following is not a problem with dynamic scoping?
☑	❑Array element access is slower
	❑Static type checking is not possible
	❑Readability
	❑Reliability

c. **A ………… is the lowest level syntactic unit of a language (e.g., *, sum, begin).**

d. **1. Consider the following XX grammar :**

XX • XX a | XX b | c

2. Show the start symbol, terminals and non terminals of that grammar.

3. Give a string belongs to that grammar has 4 characters.

Fig. 2. Examples of Question Types stored in OOKB Bank.

- **Non-Standard Exercises**; which include questions, each of them is written in a general form rather than each of the above 3 forms. It could be a long or short text, including graphs, equations, or numbers. Also, it could be imported from an external file. While the answer could be written in free text including graphs, equation, or whatever the examined person want to write. Answer of this type of question should be checked and graded manually, by the examiner. An example of this type is shown in Fig 2-d.

IV. Building the OOKB Question Bank

Questions and their answers are arranged in an object oriented OOKB as a bank of questions. This OOKB bank is designed to be instantiated for several course, one course at a time. In the first use of the tool, name of college, department, and courses could be identified. After that questions and their answered could be edited and acquired to the OOKB Bank. The tool manipulates knowledge as general objects with no care of its contents. It uses an intelligent indexing mechanism.

A. Structure of the OOKB

The OOKB is designed as normal OOKBs to deal with the types of question listed in the previous section. Its entity classes are fulfilled with attributes that cause the OOKB is suitable to acquire the knowledge learned from the examiner at building exams. The OOKB has one Super Class and 4 Sub Classes, which has to be suitable to hold all types of question listed in the previous section.

The Super Class question is the base for all types of questions. It has some important attributes that are inherited by the other four sub classes. These attributes include a string attribute for the question text, pointer to the chapter related to, and pointers to two lists of links (relations).

The first type of links are to the reminded questions –questions that are suggested to be with that a certain question in the same exam-. While, the second type of links are to the rejected questions –questions that are not allowed to be with that a certain question in the same exam-. The other four sub classes are listed below.

- **Sub Class of True/False Questions**: True/False questions should have only one special attribute, which is a binary attribute. This attribute will hold the answer, which is 1 (for True question text) or 0 (for False Question text), in addition to the attributes of the super class question.

- **Sub Class of Multiple Choice Questions**: Multiple choice questions should have many special attributes, which are four string attributes for four predicted choices, and character attribute to denote the correct choice, in addition to the attributes of the super question.

- **Sub Class of Fill in the Blanks Questions**: Fill in the Blanks (spaces) questions should have only one special attribute, which is a string attribute to hold the correct text to be filled in the spaces, in addition to the attributes of the super question.

- **Non-Standard Exercises**: General exercise could have only one special attribute, to hold the complete answer of the generic question. Contents of this attributes could be imported from an external file. This answer could be edited or imported from a file.

B. Filling the OOKB Bank

OOKB question bank should be initiated for each course individually by certain information, like course code and name, department offering that course; number of predicted chapter (could be modified latter by increasing or decreasing certain chapter). For each chapter, questions of the four previous types could be provided. Also answers of those questions should be provided.

The examiner starts his work by supplying questions to the tool. First, he selects the course, chapter, and one question type of those listed in section II. Then, he supplies the question text, accompanied with all needed information as described in section III-B. The tool will fill the OOKB tables with questions and their related knowledge.

Knowledge within the two lists of links could be acquired at entering a new question or be learnt from the examiner when he/she rejects suggested question or choose certain question directly to be added.(to be discussed later).

V. Agent Creates an Exam

The process of editing an exam should emphasize that each course include a number of chapters. Some or all of these chapters could be covered by the exam. This will help the tool in selecting questions from a specified chapter. Also, dividing a course into chapters help us to divide test into logical parts and "accumulate" test questions base (for example, you have 500 questions, but single test consists of 20 questions that are randomly selected from the OOKB question bank);

A. *Criteria in Creating an Exam*

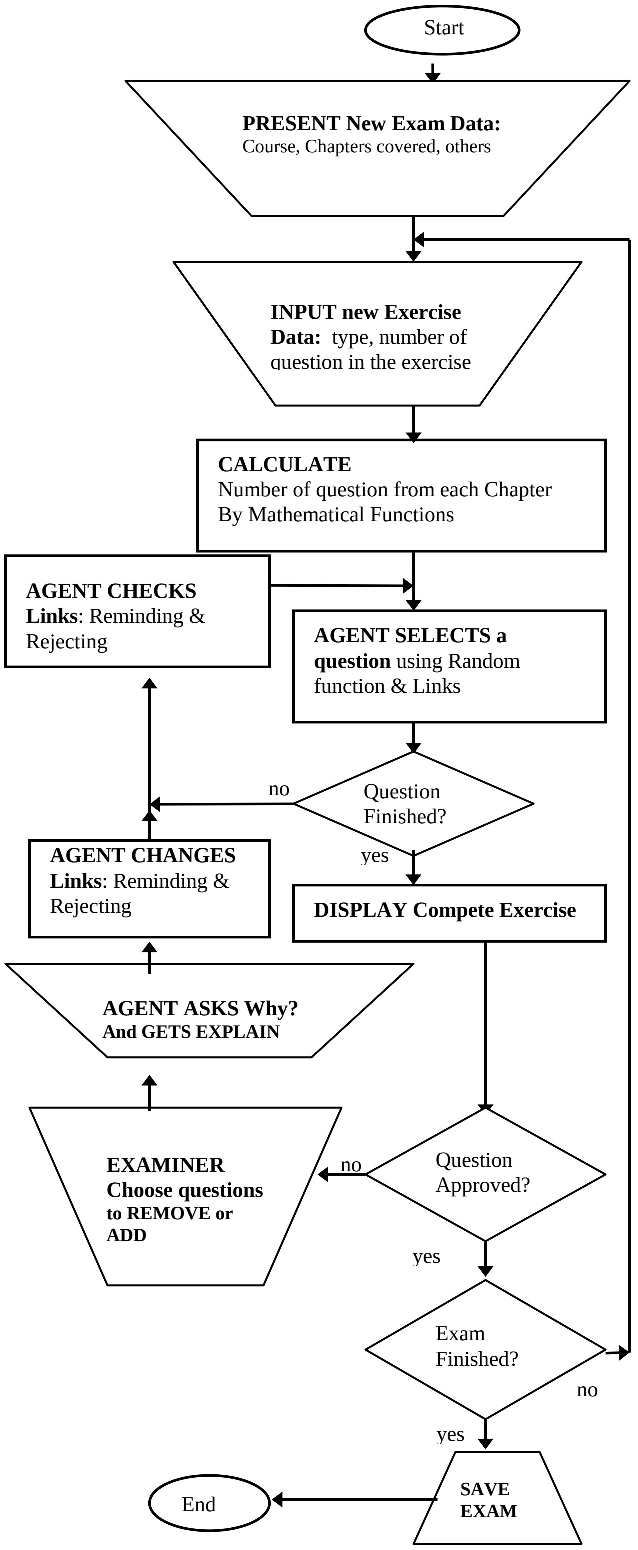
Start
PRESENT New Exam Data:
Course, Chapters covered, others
INPUT new Exercise Data: type, number of question in the exercise
CALCULATE
Number of question from each Chapter By Mathematical Functions
AGENT CHECKS
Links: Reminding & Rejecting
AGENT SELECTS a question using Random function & Links
no
Question Finished?
yes
AGENT CHANGES
Links: Reminding & Rejecting
DISPLAY Compete Exercise
AGENT ASKS Why?
And GETS EXPLAIN
EXAMINER
Choose questions to REMOVE or ADD
no
Question Approved?
yes
Exam Finished?
no
yes
SAVE EXAM
End

Fig. 3. A Session for exam creation and tool-examiner modification

The process of creating an exam is done automatically, as shown in Fig. 3, by the presented tool. It emphasized the following criteria:

- Questions and answers could be shuffled randomly (for every particular question), to avoid any fraud by people who pass your tests.
- Multilanguage support: English, Arabic.
- The test interface is fully customizable.
- Unlimited number of questions in a single test and up to 4 answers in multiple choices question.

The whole testing process is automated and doesn't require assistance.

B. *Initiating an Exam*

Before creating exams in a certain institute using the presented tool, the examiner should initiate the exam by supplying some information like the name, top introduction, bottom conclusion, and logo of the examining institute. Then, he should provide course name and chapters to be covered by the exam should be selected from those available in the OOKB.

Also, some optional data -could be applied to the tool before creating the exam or even after producing it- like: Academic year, Semester, Student sections, exam duration, maximum score, exam type (midterm, quiz, final), sort of exam (A, B, ..) , the total mark of the exam.

Then the examiner should specify number of exercises, the tool will ask about the type of each exercise and number of question in each, and the mark specified for each exercise. Now, the tool is ready for creating the exam automatically, as described in the following section.

C. *Creating an Exam and Agent Role*

When beginning a session to create exam from OOKB question bank, the tool performs its task through several steps, which incorporates knowledge acquisition and learning with knowledge retrieval from its OOKB, based on an intelligent agent.

The tool asks the examiner about type of each exercise and number of its questions. It calculates number of questions per each chapter using mathematical functions.

Then, agent starts a process of questions selection, one by one, until finishing selecting. This selection is based on random function and according to the reminding and rejecting knowledge resides in the links between each question and other questions in the OOKB bank.

The tool shows selection result in a complete exercise to the examiner for approval. Sometimes, she/he wants to remove certain questions or to add another question. In this case, the agent asks her/him for explanation.

This explanation is acquired to a rejecting knowledge in case of refusing certain question, and to a reminding knowledge in case of adding certain questions. This knowledge will reside in the links between any requested question and other questions in the OOKB.

After performing the requested modification, the tool, finally, produces a file including the final form of the exam or the quiz. This exam version should be reviewed by the examiner (the expert). Also, the tool can produce key answers for the exam.

VI. Conclusion

The presented paper provided an agent-based tool that could build an object-oriented knowledge base (OOKB) as a bank of questions and acquire its knowledge. It could form exams automatically, by selecting questions from its OOKB bank according to examiner's specification and based on an intelligent agent in selection of questions.

The tool’s intelligent agent could get its experience by watching the examiners, in two different situations. First, at editing a question; when the examiner specifies that some question(s) are preferred to be/ should prevented from being in the same exam with that question. Secondly, at exam creation; when the examiner asks to remove certain question(s) or add certain ones.

References

[1] J. Anderson, F. Boyle, A. Corbett, and M. Lewis, “Cognitive Modeling and Intelligent Tutoring,” Advanced Computer Project, Carnegie-Mellon University, Pittsburgh, PA, USA, *Artificial Intelligence*, vol. 42, pp. 7-49, Elsevier Science Publishers, B. V. (North Holland), 1990.

[2] H. W. Tsang, L. M. Hung, and S. C. Ng, “A multimedia distance learning system on the Internet,” in *Proceedings of IEEE International Conference on Systems, Man, and Cybernetics,* vol. 2, pp. 243 –246, 1999.

[3] Z. Liu and B. Chen, “Model and Implement an Agent Oriented E-Learning System,” in *Proceedings of the International Conference on Computational Intelligence for Modelling, Control and Automation, and International Conference Intelligent Agents, Web Technologies and Internet Commerce,* 0-7695-2504-0/05,2005.

[4] Y.-H. Fang and L. Shao, “Agent-Based E-Learning System Research and Implementation,” in *Proceedings of the 7th International Conference on Machine Learning and Cybernetics*, Kunming, pp. 4080-4084, 12-15 July, 2008.

[5] J.-L. Lin and M.-H. Chen, “An Intelligent Agent for Personalized E-Learning,” in *Proc. of 8th Inter. Conference on Intelligent Systems Design and Applications*, vol. 1, pp. 27-31, 2008.

[6] S. J. Russell and P. Norvig, *Artificial Intelligence – A Modern Approach*, 3rd Ed., Prentice Hall, 2009.

[7] H. K. Mohammed, “An Intelligent Agent to Teach C-Language,” in *Proceedings of ICECS*, Cairo, Egypt, pp. 483-488, December 15-18, 1997.

[8] R. Burton, “Diagnosing Bugs in a Simple Procedural Skills,” Xerox Palo Alto Research Center, Cognitive and Instructional Sciences Group, Palo, Alto, CA 94304, 1986.

[9] D. Sleeman, “Frameworks for Cooperation in Distributed Problem Solving,” *Readings in Distributed Artificial Intelligence*, Morgan Kaufmann Publishers, Inc. San Mateo, California, 1982, pp. 61-69.

[10] A. M. Florea, “An Agent-Based Collaborative Learning System,” *Advanced Research in Computers and Communications in Education*, G. Cumming *et al.* (eds), IOS Press, International Conference on Computers in Education, pp. 161-164, 1999.

[11] M. Nissen, “Intelligent Agents: A Technology and Business Application Analysis,” *Telecommunications and Distributed Processing*, November 1994.

[12] ProfProf. Create Online Tests & Quizzes Easily. [Online]. Available: http://www.proprofs.com/quiz-school/

[13] ClassMaker. The Quiz Maker for Professionals. Easily Create Web Based Exams. [Online]. Available: http://www.classmarker.com/

[14] AditSofware. Exams, Tests and Quizzes Made Easy. [Online] Available: http://www.aditsoftware.com/

Khaled N. ElSayed was born on 9 Oct. 1963, in Cairo, Egypt. He have got his PhD of from Faculty of Engineering, Ain Shams University, Cairo, Egypt, 1996.

He worked as an associate professor of computer science, in Umm-AlQura Uni. in Makkah, Saudi Arabia since 2006. Artificial Intelligence is his major. His interest research is Distant Education, E-Learning, and Intelligent Agent. He translated “Fundamentals of Database Systems,” R. Elmasei and S. Navathe, Addison Wesley, 4th Ed.