A Dynamic Content Generation Tool for OOP Course

Ibrahim A. Cankaya, Asim S. Yuksel, Arif Koyun, and Tuncay Yigit

***Abstract*—Rapid growth of technology invalidates newly produced information and causes it to become old. This situation leads teachers, instructors and academicians to use the information duly and effectively. To achieve this goal, there should be some tools that will be used by wide range of educationists in a way that is fast and effective. These tools should also allow them to create the knowledge and share it. In this study, we have developed a web-based dynamic content generation tool for educationists who are teaching Object Oriented Programming (OOP) course. The software tool that we have developed is capable of creating course content and generating UML class diagrams dynamically related to basic OOP concepts according to the information provided by the course instructor. Our developed tool is easy to use, has user-friendly interface and can be adapted to any other programming courses easily.**

***Index Terms*—Dynamic courseware generation, dynamic content generation, learning management, course management.**

I. Introduction

The fast growth of Internet caused traditional education to naturally transit to online education for institutions of 21st century. Teachers, academicians, instructors perceive the online education as an innovation that enhances teaching, promotes lifetime learning experience and reaches everyone.

In the context of studying engineering, online education has become possible in recent years but is not yet widely adopted by other disciplines. To make online engineering education broadly accepted and widely used, online course quality must be better than the traditional courses [1], [2] and courses should be available anywhere.

However, static teaching materials that are available online do not fulfill the requirements of educationalists and cannot be easily modified according to specific needs [3]. Therefore, there is need for software tools that will generate course materials according to the requirements of educationalists, especially in teaching of OOP concepts.

OOP is the most popular programming paradigm today and widely used in education and industry [4]. Most of the universities have courses related to OOP in their curriculums. Furthermore, the software development community agrees that teaching OOP is necessary and must be mandatory in universities. However, teaching this course is difficult [5]. From our perspective, it is not the "object-orientation" principles that make it difficult to learn, but the tools that are used to teach it. Some of the tools are developed for professional software developers making them difficult to use for educational purposes and others are not customizable for specific needs. Thus, we have developed a web-based dynamic content generation tool that aids academicians who are teaching OOP concepts in creating course content and generating UML class diagrams dynamically according to given input. The tool is developed in a way that it supports the basic principles of OOP that are "Inheritance, Polymorphism, Abstraction, Encapsulation". The only data needed by the tool for generation of course content is the textual data provided by the course instructor. Furthermore, to generate an UML diagram for a specific OOP principle, the instructor only enters the names of classes, methods and the related UML class diagram is generated automatically.

The remainder of the paper is organized as follows: Section II describes related work; while Section III describes system architecture. In Section IV, we talk about our developed application and in Section V we demonstrate our results. Finally in Section VI, we conclude the paper.

II. Related Work

Course generation has been studied for a long time as a research topic. In this section, we present the related work by focusing on tools that generate course contents.

In Ref. [6], authors present a course generator tool called PAIGOS that generates courses according to different learning objectives based on pedagogical knowledge. System uses the pedagogical knowledge as a building block for course generation. Courses are generated for a wide range of learning goals such as discovering new content, rehearsing and practicing for exams. Specific content such as exercises are retrieved according to the pedagogical knowledge.

The Dynamic Courseware Generator (DCG) [7] is a tool that generates courses according to the user's specification. This study is the foundation of today's course generation utilities. Planner module of the tool searches for sub-graphs that connect the goal with user goal. New course plans are generated according to the user's success on tests on a certain topic. The main difference with our tool is the learning material. DCG makes uses of fixed HTML-pages for the course content. On the other hand, in our tool, we generate courses dynamically using XML technology. Moreover, DCG is a general authoring tool and therefore integration with service systems is not intended.

Ref. [8] describes their architecture for dynamic course generation. The courses are generated automatically according to teaching goals and can be changed dynamically based on the rules that instructors provide.

In Ref. [9], authors present an automatic course generation system that is developed at the Shanghai Jiao Tong

Manuscript received January 30, 2014; revised April 21, 2014.

The authors are with the Suleyman Demirel University Computer Engineering Department, Isparta, 32260 Turkey, (e-mail: ardacankaya@sdu.edu.tr, asimyuksel@sdu.edu.tr, arifkoyun@sdu.edu.tr, tuncayyigit@sdu.edu.tr).

University. With this system, teachers are able to create courses according to their own curriculum. Additionally, courses can be changed to reflect the changes in objectives.

Approaches in [10]-[13] use the pedagogical knowledge to generate course content. As a result, they are not capable of generating content with respect to different learning objectives.

In Ref. [14], two different research groups designed two different adaptive E-Learning frameworks to address the problems of current E-Learning architectures. First one is called Adaptive Personalized E-Learning Service (APeLS). This framework delivers educational courses based on a multi-model, metadata driven approach and serves as a service. Second framework is called Knowledge Tree that replaces current course management systems such as Blackboard. The framework has three kinds of servers that are activity servers, learning portals, and student model servers. The learning portal is similar to the modern CMS. Teachers have ability to design their course and manage students.

.

III. System Architecture

Our dynamic content generation tool is comprised of four main modules that are Teacher, Student, Admin, Course Generation modules. The architecture of the system is shown in Fig. 1. It will be discussed in following sections.

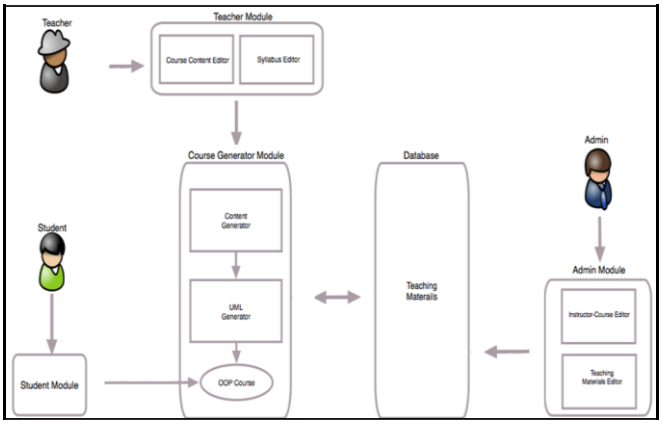

Fig. 1. System architecture.

A. Admin Module

Administrators use this module of the system. It has two sub modules that are Instructor-Course Editor (ICE) and Teaching Materials Editor (TME). They can perform all C.R.U.D (Create, Retrieve, Update, Delete) operations such as adding new instructor, updating instructor information, assigning courses to instructors, deleting course/instructor information. They can also manage teaching materials by adding new teaching materials or deleting them.

When adding a course in ICE module, a course form is presented to the administrator. Top part of the form contains course number and title and Course ID. The middle part of the course form has instructor related information and the description of course. There is also a semester drop down list that indicates when an instructor will teach the course. The bottom of the form contains the course description section. Administrators can enter or edit the description or add text from Microsoft Word.

B. The Database

The teaching materials are stored in our secure database. It contains animations, textual content that establishes the communication with the students and teachers. We used a combination of NoSQL and relational database for storage purposes. NoSQL that is schema-free, document-based database is chosen to store the teaching materials. This kind of database is to be known as scalable and provides high performance, allows storing large volumes of structured, semi-structured, and unstructured data. Other information is stored in regular relational database.

C. Student Module

In the student module, students login to the system with their username and password. Once they login, they can see the list of OOP courses in a monthly view, go through the content by clicking them and watch the UML animations prepared by the course instructor. Additionally, students are able to see and manage all the details associated with them such as address, personal information. The course materials that are seen by the students are pulled from our database.

D. Teacher Module

Teacher module features two modules for teachers to prepare and manage their online course contents. It is comprised of Course Content Editor (CCE) and Syllabus Editor (SE).

SE handles the management of OOP course syllabus. It helps teachers to prepare general information about their class, such as required learning materials, prerequisites, course credits or schedule of classes. Additional information such as course description can also be provided.

CCE provides a user interface for the teacher to prepare OOP course materials that includes textual content and UML animations. It enables teachers to start building a new course with a few clicks. A course dashboard is also presented for the teacher to see the list of courses assigned to him/her. After creating a course, it is instantly available for students.

E. Course Generator Module

This module is the heart of whole system where the dynamic content generation takes place. It has three sub modules that are Content Generator (CG), UML Generator (UG), and OOP Course View (OCV). CG is responsible for the creation of textual course content. It takes the necessary information from Teacher Module where the course instructor enters the textual information and produces presentation style course materials.

In the same way, UG has the functionality to produce UML diagrams based on the input that is entered by the instructor in Teacher Module. OCV contains the final view of the produced course that will be displayed to the student. Whenever the instructor updates the course materials, the view is updated. The module acts as a bridge between the Teacher, Student modules and the database. Generated course materials are stored in the database and retrieved whenever they are needed.

IV. Application

Developed application is written in PHP language. After successful login, instructor, student or admin is redirected to his or her own page. When student logins to system, instructors and courses are listed. Student can choose any courses listed in their screen and go to related content by clicking on them. In the course page, student selects month and week information then he views the course that is prepared by the course instructor.

In admin panel, instructors are listed and weekly course lists are shown. Instructor list includes course name that is assigned to an instructor, course code, instructor id, instructor name and surname, instructor user name. Courses that are assigned to an instructor can be changed by admin through this panel. Administrator can also delete instructor or add new instructor.

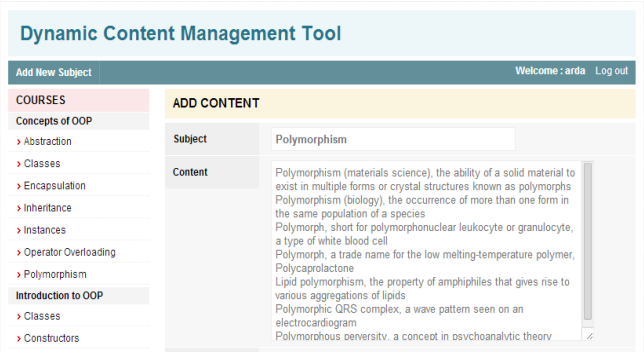

Fig. 2. Adding content section.

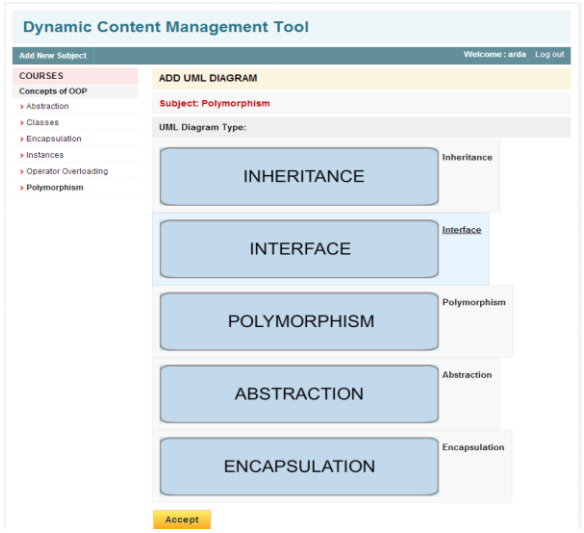

Fig. 3. UML diagram creation.

Instructor who logins via instructor panel can list courses, sort by month and week, can see the course he or she prepared, add or delete course, add content to the course. Fig. 2 and Fig. 3 show the process of adding new content and UML diagram creation. To able to add new course, instructor must enter the following information:

- Subject Name
- Content Information
- Week and Month Information
- Related UML Diagram

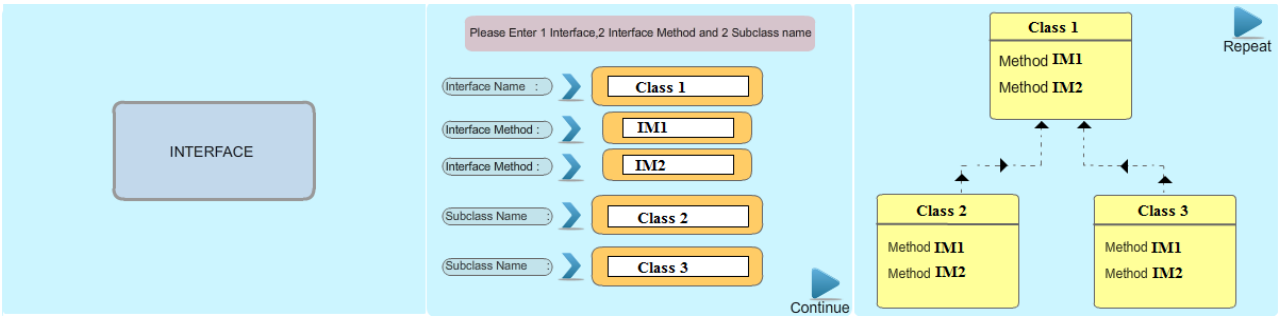

Fig. 4. Creating UML diagram.

UML Generation

UML stands for Unified Modeling Language that has become a de facto standard as an object modeling language. It reduces the overall time to develop software programs. As a diagram, it contains notations for understanding OOP concepts and easy communication with other programmers. UML diagram includes class, collaboration, sequence, state and use case diagram. In this study we only focused on the generation of class diagrams.

Action Script 3.0 language was used for preparing animated UML diagrams. To teach OOP programming, animation templates were created for the four basic principles of OOP that are Inheritance, Polymorphism, Abstraction, and Encapsulation. An instructor who wants to create an UML diagram first enters the class and method names. These class and method names are transferred in real time to animation template via XML. After the instructor finishes the creation of UML diagrams and submit to the system, the student can login to system and see the generated UML diagram. Students also have power to change the class and method names according to their wishes. In Fig. 4 shows the process of UML diagram creation.

V. Results

TABLE I: Test Results

	OOP Principles **True/False**	UML Diagrams **True/False**
Group A	2.06/0.94	1.43/1.57
Group B	2.16/0.84	2.33/0.67

To test the effectiveness of our tool, 60 students were selected who took OOP course for the first time. 30 students (Group A) used our tool to learn the creation of UML diagrams. Other 30 students (Group B) used instructor's lecture notes. After two weeks of training, we tested students with an exam by asking them to create UML diagrams in OOP course. In the exam, we asked 3 questions related to

OOP principles and 3 questions related to UML diagrams. Two groups' success in the exam was listed in Table I.

These results were obtained by averaging total number of correct answers for each group. Our results show that correct answer rate for the questions related to OOP principles did not increase by using our tool compared to group B. On the other hand, correct answer rate in questions related to UML diagrams were higher and promising in Group A who learned the UML diagrams via our tool. The animated content helped students understand and remember OOP principles.

VI. Conclusion

OOP languages are the most popular type of languages that are being taught to students. Teaching methods for programming courses have been the cause of debate for years. One perspective that is widely accepted is the use of graphical environments. Many environments have been developed and many are effective to their use. However there is a need for online environments to teach OOP languages in an effective way.

In this study, we developed a software tool that is capable of creating dynamic course content and generating UML class diagrams for the teaching of OOP course. Our developed tool is easy to use, has user-friendly interface and can be adapted to any other programming courses easily. Since our tool is web-based, it can be accessed from anywhere.

Traditional approaches to create animations require labor-intensive techniques that are expensive and time-consuming. However, our software tool makes it possible for teachers, instructors to prepare their own animations without the need for an expert.

Although the tool helps instructors to prepare OOP course materials, and generate UML diagrams, it has limitations. It supports only two levels of class hierarchy and limited method definition.

References

[1] J. Bourne, D. Harris, and F. Mayadas, "Online engineering education: Learning anywhere, anytime," *Journal of Engineering Education*, vol. 94, no. 1, pp. 131-146, 2005.

[2] R. E. Gomery, "Internet Learning: Is it real and what does it mean for universities," *Journal of Asynchronous Learning Networks*, Sheffield Lecture-Yale University, vol. 5, no. 1, January 2001.

[3] J. Bennedsen and M. E. Caspersen, "Revealing the programming process," *ACM SIGCSE Bulletin*, vol. 37, no. 1, pp. 186-190, February 2005.

[4] M. Kölling, "The problem of teaching object-oriented programming, Part 1: Languages," *Journal of Object-oriented Programming*, vol. 11, no. 8, pp. 8-15, 1999.

[5] M. Kölling and J. Rosenberg, "An object-oriented program development environment for the first programming course," *ACM SIGCSE Bulletin*, vol. 28, no. 1, pp. 83-87, March 1996.

[6] C. Ullrich and E. Melis, "Pedagogically founded courseware generation based on HTN-planning," *Expert Systems with Applications*, vol. 36, no. 5, pp. 9319-9332, 2009.

[7] J. Vassileva, "Dynamic course generation on the www," in *Proc. AI-ED'97, 8th World Conference on Artificial Intelligence in Education*, Amsterdam, August 1997, pp. 498-505.

[8] J. Vassileva, "Dynamic courseware generation: at the cross point of CAL, ITS and authoring," in *Proc. International Conference on Computers in Education*, December 1995, vol. 95, pp. 290-297.

[9] X. Tan, C. Ullrich, Y. Wang, and R. Shen, "The design and application of an automatic course generation system for large-scale education," in *Proc. 2010 IEEE 10th International Conference on Advanced Learning Technologies ICALT*, July 2010, pp. 607-609.

[10] A. I. Cristea and A. De Mooij, "Adaptive Course Authoring: My Online Teacher," in *Proc. ICT*, Papeete, French Polynesia, 2003.

[11] N. D. D. Méndez, C. J. Ramírez, and J. A. G. Luna, "IA planning for automatic generation of customized virtual courses," *Frontiers of Artificial Intelligence and Applications*, Valencia Spain: IOS Press, vol. 117, pp. 138–147.

[12] P. Karampiperis, and D. Sampson, "Adaptive Instructional Planning using Ontologies," in *Proc. the IEEE International of Conference on Advanced Learning Technologies*, 2004.

[13] Y. Huang, J. Chen, T. Huang, Y. Jeng, and Y. Kuo, "Standardized course generation process using Dynamic Fuzzy Petri Nets," *Expert Systems with Applications*, vol. 34, pp. 72-86, 2008.

[14] P. Brusilovsky, V. Wade, and O. Conlan, "From learning objects to adaptive content services for e-learning," *Architecture Solutions for E-Learning Systems*, pp. 243-261, 2007.

Ibrahim Arda Cankaya received his B.S. degree in 2012 from the Suleyman Demirel University Computer Engineering Department, Turkey. He continues his M.S. degree education at Suleyman Demirel University Computer Engineering Department, Turkey. He has been working as a research assistant in Suleyman Demirel University Computer Engineering Department since 2012. His research interest includes algorithm and programming, database management, mobile programming.

Asim Sinan Yuksel received his B.S. degree in 2006 from the Ege University Computer Engineering Department, Turkey. He worked as a software developer at Social Security Institution between 2006 and 2007 in Ankara. He received his M.S. degree in Indiana University School of Informatics and Computer, USA in 2010 and he pursues his Ph.D. education in the Istanbul University Computer Engineering Department, Turkey. He has been working as a research assistant in Suleyman Demirel University Computer Engineering Department since 2012. His research interests include privacy and security of social networks, mobile security and privacy, human computer interaction, object oriented programming and design.

Arif Koyun received the B.S. degree in 1988 from Mechanical Engineering of Akdeniz University, Turkey and in 1995 from computer engineering in Germany. He received M.S. degree in 1991 from Akdeniz University, Turkey and Ph.D. in 2006 from Suleyman Demirel University, Turkey. Between 2001 and 2007, he has worked as a lecturer in Suleyman Demirel University. Currently, he is an assist. Prof. Dr. in Suleyman Demirel University, Turkey. His research interest includes artificial intelligence, data mining, database management systems, distance education, e-learning, computer education and computer science.

Tuncay Yigit received his MS and PhD degrees in electrical engineering from the Institute of Natural and Applied Sciences of Gazi University, in 2000 and 2005, respectively. He worked as a lecturer at Computer Education University of Gazi University until 2007. He is currently an associate professor of computer engineering at the Faculty of Engineering in Suleyman Demirel University. He has extensive experience in teaching, object-oriented programming, software engineering, artificial intelligence and e-learning. His research interests include object-oriented software development with the UML, computer science and software engineering education development of e-learning and Web-based learning systems. He has published over 20 papers in international journals and conference proceedings.