INTERFACE BASED PROGRAMMING ASSIGNMENTS AND AUTOMATIC ASSESSMENT AND GRADING AS A SERVICE

Ranjeet Sukhadeo Jagtap
Computer Science and Engineering, SVERI's College of Engineering, Pandharpur, India
Prof. Y. R. Kalshetty
Computer Science and Engineering, SVERI's College of Engineering, Pandharpur, India
D.V. Jadhav
Computer Science and Engineering, BMIT, Solapur, Maharashtra, India

ABSTRACT

As the knowledge base in the Computer Science is growing, many new programming languages are being introduced. On one hand the software industry, while hiring a computer programmer, expects him to be a proficient in computer programming. But on other hand there is deficiency of computer programmers who are skilled in programming. So to bridge this gap it's crucial to assess the programming abilities of students. Testing and grading programming assignments manually requires more time and are less efficient as far as its correctness is concerned.

Today, many software systems are available for conducting choice based assessment that are capable of assessing a student's knowledge but not his skills. We have implemented system for assessment which is based on programming assignments that are submitted by students. The system in turn compiles these programs online. After compiling it successfully the teacher of the corresponding subject can test it using this system, and grades it based on how many tasks have been accomplished. This paper reports about the design and implementation of automatic assessment of programming assignments. Report of the sample usage of this software system is included.

Keywords – ***Programming, Computer Science, assessment, e-learning, learning systems, assessment as a service.***

INTRODUCTION

Today, there are many online compilers to tools available which allow users to write, edit, compile and run programs online. Such systems are proving to be crucial for developing programming skills. But many of them just check for the syntactical errors or they just execute a program and provide its output. They don't test or evaluate the performance based on the results of test cases.

Another issue is that many new programming languages are being introduced in the field of computer science education. A teacher can't learn all of them, so there has to be system which will test the programs written by students and evaluate these programs and will provide the teacher its analysis

Assessment is an essential element in learning processes. It is therefore not unsurprising that almost all learning management systems (LMSs) offer support for assessment, e.g., for the creation, execution, and evaluation of multiple choice tests. We have designed and implemented generic support for assessment that is based on assignments that students submit as electronic documents. In addition to assignments that are graded by teachers, we also support assignments that can be automatically tested and evaluated, e.g., assignments in programming languages, or other formal notations. In this paper, we report about the design and implementation of a service-oriented approach for automatic assessment of programming assignments. The most relevant aspects of our "assessment as a service" solution are that on the one hand the advantages of automatic assessment can be used with a multitude of programming languages, as well as other formal notations (as so-called back ends); on the other hand, the features of these types of assessment can be easily interfaced with different existing learning management systems (as so called frontends). We also report about the practical use of the implemented software components at our university and other educational institutions.

Background and Motivation

The advantages for the learning processes are even more substantial when the online submission of students' programming assignments is combined with automatic testing and grading. Automatic assessment allows timely, nearly instant result for students, which is an added advantage of the system. Content Management Systems are far more efficient for managing the learning process.

The website used by the workgroup has been built on Plone [16]. In past too, e-learning or online learning systems have been used in learning process. With increasing interest in e-learning and in the ways in which e-learning technologies can be integrated into existing structures and technologies (organizational, as well as technical), the idea of enhancing Plone with additional components, in order to convert Plone into an LMS, arose. As they were unwilling to administer and maintain a second native LMS, which provides a good portion of the same functions as a CMS, they have designed, implemented, and deployed a number of modules for Plone which extend the CMS with specific e-learning functionality. These modules—collectively called eduComponents [17] —provide specialized content types offering the following main functions ([6] and [7]).

- ECLecture is a product for managing lectures, seminars, and other courses. It also serves as “portal” to all course-related materials and handles registration for courses.
- ECQuiz supports the creation and delivery of interactive multiple-choice tests. It can be used for formative tests to quickly assess the performance of a class without the need for extra grading work.
- ECAssignmentBox allows the creation, submission, and grading of essay-like assignments. The assessment process is semi automated, i.e., the assessment is done by the instructor, who is aided by the tool during the process of grading students’ work and giving feedback.

The eduComponents modules can be used separately, or in combination and, since many basic functions are already provided by the CMS. Deploying the eduComponents turns Plone into a full-fledged, tailor-made LMS. However, they wished to offer their students more timely feedback and more detailed discussion on their programming assignments too. In addition to the more conceptual aspects, programming includes practical aspects as well, e.g., techniques of testing and debugging programs and the use of a programming environment. Therefore, students should be given frequent programming assignments, but assessing a large number of such assignments is a time consuming and labor-intensive task.

LITERATURE REVIEW

The edu Components Approach

E-learning platforms are systems that support six activities in an educational context: Creation, organization, delivery, communication, collaboration, and assessment. Furthermore, we have concluded that modern Web content management systems support five of these activities. This means that few of these activities are educational—the one exception is assessment.
So, one could say

E-learning = Web + assessment.

To allow system administrators to adapt the platform to the needs of their users, the CMS should not just be “modular” in a general sense, but individual modules should be deployable individually. This requirement implies a *component-based architecture* [24] or possibly a *service-oriented architecture* [25].

Design and Implementation

Design Considerations

The eduComponents are a new approach to the architecture of an e-learning platform. The design of the eduComponents tries to overcome the technical problems with conventional e-learning platforms, as described in the previous chapters, and to enable new approaches for deployment.
Instead of re-implementing CMS functionality—as conventional e-learning platforms do—, the eduComponents rely on a general-purpose CMS to provide (1) a rich development environment and a basis for deploying components, and (2) a reliable implementation of basic document management functions. Furthermore, the eduComponents approach relies on a component-based architecture, which makes it possible

to install only the required components and to combine components from different sources. The third point is to use a uniform content representation, in our case; documents are the central objects, and tests are documents just like lecture notes.

Zope and Plone

For the actual implementation of the eduComponents they have selected the Plone [16] content management system as the basis for our components. Plone is built on top of the Zope [26] Web application framework. Zope provides a platform for building Web applications such as content management systems, intranets, portals, or other custom applications. Zope is based on a transactional object database (ZODB); it also includes an integratedWeb server (ZServer), a search engine (ZCatalog), and a workflow engine. It can connect to all major relational database management systems, interoperate with other Web servers (such as Apache), and supports further network protocols such as FTP, WebDAV, and XML-RPC.

PROPOSED METHODOLOGY

System Architecture:

Architecture of the proposed system is given in the figure 3.1. As the figure shows Users of the system will interact with it via a Graphical User Interface (GUI), which is built in ASP.NET on .NET platform. Users will have pages to sign up to the system, pages for logging in, for adding their respective information.

Environment:

Environment is the programming environment required to run a program written in a programming language, e.g. program written Python requires its own interpreter. Environment will be selected for respective programming languages.

Test Case Provider:

Test Case Provider (TCP) unit is the part where test cases for different programming assignments are stored. After creation of a new programming assignment, teacher will write test case for this assignments, for testing this assignment. These test cases will be stored in database, which will be called as Test Case Provider.

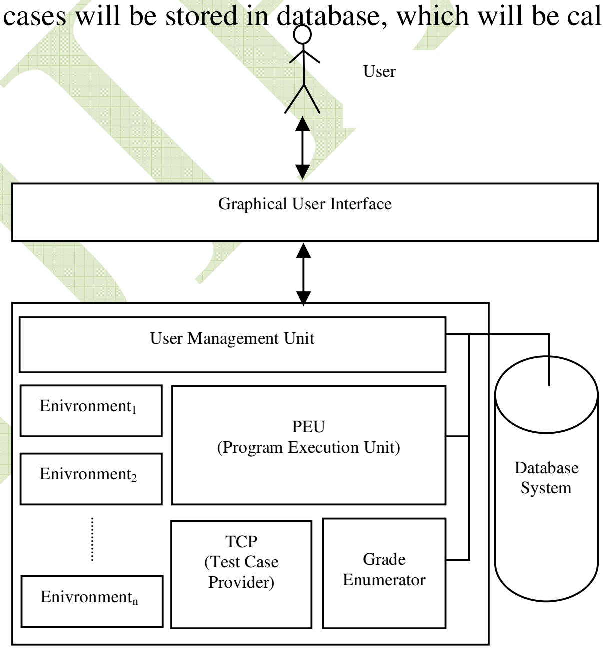

Figure 3.1: System Architecture

User Management Unit:

User Management Unit is responsible for managing the users. This unit is specifically used by administrator of the system. This unit provides features such as adding new user, authenticating access to the user, approving the user after sign up etc. Grade Enumerator unit grades the programming assignments based on test results.

Program Execution Unit:

Program Execution Unit (PEU) collects all necessary data to run a program from all the other units and generates the results which are then stored in database. Further, users can access these test results based on their access rights. Assignments created by the teacher are stored in database. Students will read these assignments from database, and try to solve it. When the user is done with assignment he will submit it, which will be stored in database. After the deadline for the submission of passes the teacher will auto test the assignment and the test results will be stored in database.

Grade Enumerator:

Grade Enumerator unit is responsible for assigning grades to assignments submitted by students. Assignments submitted by students are tested against the test cases written by corresponding teacher. If the tests successfully then this unit assigns the grade based on following formula. Grade enumerator first calculates grade points for every programming assignment, i.e. $P_{Grade,}$ submitted by student, as given in equation (1).

$$P_{Grade} = \sum_{k=1}^{n} \mathrm{Outcome}(t_k) \qquad (1)$$

Where,

t_k = the outcome of P for t_k^{th} test.
n = number of test cases for P.
P_{Grade} = Grade of Program P.

Accumulated grades points of set of programming assignments constitute student's grade points, i.e. S_{Grade}. S_{Grade} is calculated using equation (2).

$$S_{Grade} = \left(\sum_{j=1}^{m} P_j\right) \Big/ m \qquad (2)$$

Where,

P_j = Grade of P_j^{th} program.
m = Number of programming assignments.
S_{Grade} = Grade of Student.

The final stage of grade enumerator is to enumerate final grades based on student's grade points as given in equation (3).

$$S_{Enum} = \begin{cases} n & - \ Best \\ 0 < S_i < n & - \ Avg. \\ 0 & - \ Worst \end{cases} \qquad (3)$$

RESULTS AND DISCUSSION

The experiments are carried out on various samples and we found that grading every single programming assignment for large number students manually becomes a tedious task and it is sometimes inaccurate. Whereas doing same task using our system gives the better and accurate results and that too in very less amount of time compared to manual grading. The three samples from results generated by grade enumerator are as shown in table 4.1.

T_k →		T_1	T_2	T_3	P_{Grade}	S_{Grade}	S_{Enum}
S_i	P_j				Sum(T_k)	Sum(P_j)/j	
S_1	P_1	1	0	1	2	1.67	Avg
	P_2	0	1	1	2		
	P_3	0	0	1	1		
S_2	P_1	1	1	1	3	2.67	Avg
	P_2	1	1	1	3		
	P_3	0	1	1	2		
S_3	P_1	0	0	0	0	0	Worst
	P_2	0	0	0	0		
	P_3	0	0	0	0		
S_3	P_1	1	1	1	3	3	Best
	P_2	1	1	1	3		
	P_3	1	1	1	3		

Table 4.1 Sample output of Grade Enumerator

First of all, admin will come into existence in the system. Teachers will sign up for the system and admin will approve their accounts. Teacher will create assignments by giving its details such as assignment title, subject to which this assignment belongs to, deadline for submitting this assignment, inputs, outputs, method names etc. After creation of this assignment, he will publish this assignment. Once an assignment is published by a teacher, it will be visible to respective students. Students will read this assignment and will start implementing it. Once they are done with implementation they will submit this assignment. After the deadline for submission of this assignment passes the corresponding teacher will start testing the assignments submitted by his students. Results of the testing of the assignments will be stored in database and same will be used for grading of these assignments.

CONCLUSIONS

We have developed a software system for C# and Python programming language—for automatic assessment of programming assignment in Computer Science education. We showed how this system can enable a frontend, i.e., learning management system, to automatically test and grade programming assignments. We also have reported about grading an individual programming assignment. Then using this grade, we can grade the performance of a student. This grading technique may become crucial in assessing the performance of students in programming subject, which may in turn help the education system to provide the productive engineers to industry.

Support for automatic testing and grading of assignments in natural language can be developed, but this surely requires much more research to be done in this area.

REFERENCES

[1] Amelung, K. Krieger, and D. Ro¨sner, "E-Assessment as a Service" IEEE Transactions on Learning Technologies, Vol. 4, NO. 2, April-June 2011.

[2] Michael Striewe, Moritz Balz, Michael Goedicke, "A Flexible And Modular Software Architecture For Computer Aided Assessments And Automated Marking".

[3] M. Amelung, P. Forbrig, and D. Ro¨sner, "Towards Generic and Flexible Web Services for E-Assessment," Proc. 13th Ann. Conf. Innovation and Technology in Computer Science Education (ITiCSE '08), pp. 219-224, 2008.

[4] Carla Limongelli, Filippo Sciarrone, Marco Temperini, and Giulia Vaste, "Adaptive Learning with the LS-Plan System: A Field Evaluation".

[5] Glenford J. Myers, "The Art of Software Testing" Second Edition.

[6] M. Amelung, M. Piotrowski, and D. Ro¨sner, “Educomponents: A Component-Based E-Learning Environment,” Proc. 12th Ann. SIGCSE Conf. Innovation and Technology in Computer Science Education (ITiCSE ’07), p. 352, 2007.

[7] M. Piotrowski, “Document-Oriented E-Learning Components,” PhD thesis, Dept. of Computer Science, Otto von Guericke Univ., 2009.

[8] D. Ro¨sner, M. Piotrowski, and M. Amelung, “A Sustainable Learning Environment Based on an Open Source Content Management System,” Proc. German E-Science Conf. (GES ’07), 2007.

[9] M. Piotrowski, M. Amelung, and D. Ro¨sner, “Tactical, Document- Oriented E-Learning Components,” Proc. IADIS Int’l Conf. E-Learning, pp. 171-177, 2007.

[10] P. Black and D. Wiliam, “Inside the Black Box: Raising Standards through Classroom Assessment,” Phi Delta Kappan, vol. 80, no. 2, pp. 139-148, 1998.

[11] K.M. Ala-Mutka, “A Survey of Automated Assessment Approaches for Programming Assignments,” J. Computer Science Education, vol. 15, no. 2, pp. 83-102, June 2005.

[12] G.E. Forsythe and N. Wirth, “Automatic Grading Programs,” Comm. ACM, vol. 8, no. 5, pp. 275-278, 1965.

[13] M. Laakso, T. Salakoski, A. Korhonen, and L. Malmi, “Automatic Assessment of Exercises for Algorithms and Data Structures—A Case Study with TRAKLA2,” Proc. Fourth Finnish/Baltic Sea Conf. Computer Science Education, pp. 28-36, Oct. 2004.

[14] R. Saikkonen, L. Malmi, and A. Korhonen, “Fully Automatic Assessment of Programming Exercises,” Proc. Sixth Ann. Conf. Innovation and Technology in Computer Science Education (ITiCSE ’01).

[15] Michael T. Helmik, “Interface-based Programming Assignments and Automatic Grading of Java Programs.

[16] http://plone.org/ .

[17] http://wdok.cs.ovgu.de/eduComponents/ .

[18] Attila Nagy. e-Learning. E-Content Report 6, ACTeN, 2004. URL http://www.acten.net/uploads/images/423/e-learning.pdf

[19] Gavin Busuttil-Reynaud and John Winkley. e-Assessment Glossary (Extended). Version 1.1, Joint Information Systems Committee (JISC) and Qualifications and Curriculum Authority (QCA), January 2006. URL http://www.jisc.ac.uk/assessment .

[20] http://astd.org/lc/glossary.htm

[21] James G. Keramas. The impact of new technologies on our changing global environment. In OCEANS ’95. MTS/IEEE. Challenges of Our Changing Global Environment. Conference Proceedings, volume 1, pages 242-246, 1995. Doi: 10.1109/OCEANS.1995.526779.

[22] Tony Bates. *National strategies for e-learning in post-secondary education and training*. United Nations Educational, Scientific and Cultural Organization (UNESCO), Paris, 2001. URL http://unesdoc.unesco.org/ulis/cgi-bin/ulis.pl?catno=126230.

[23] William D. Graziadei. VICE in REST. In Teresa M. Harrison and Timothy D. Stephen, editors, *Computer Networking and Scholarly Communication in Twenty-First-Century University*, SUNY series in Computer-Mediated Communication, pages 257–276. SUNY Press, Albany, NY, 1996.

[24] Clemens Szyperski, Dominik Gruntz, and Stephan Murer. *Component Software: Beyond Object-Oriented Programming*. Addison- Wesley, Harlow, UK, 2002.

[25] Mike P. Papazoglou. Service-oriented computing: Concepts, characteristics and directions. In *Proceedings of the Fourth Internationa Conference on Web Information Systems Engineering (WISE’03)*, pages 3–12. IEEE, 2003. Doi.: 10.1109/WISE.2003.1254461.

[26] http://zope.org/

[27] Benjamin S. Bloom, editor. *Taxonomy of Educational Objectives, Handbook 1: Cognitive Domain*. Longman, White Plains, NY, USA, 1956.

[28] Susan M. Case and David B. Swanson. *Constructing Written Test Questions For the Basic and Clinical Sciences*. National Board of Medical Examiners, Philadelphia, PA, USA, 3rd edition, 2002. URL http://www.nbme.org/PDF/ItemWriting_ 2003/2003IWGwhole.pdf

[29] Stephen Petrina. Sidney Pressey and the automation of education, 1924–1934. *Technology and Culture*, 45(2):305–330, April 2004. URL http://muse.jhu.edu/journals/technology_and_culture/v045/45.2petrina.html

[30] The IBM 850 Test Scoring Machine, see http://ibm.com/ibm/history/exhibits/specialprod1/specialprod1_9.html

[31] Eugene Judson and Daiyo Sawada. Learning from past and present: Electronic response systems in college lecture halls. *Journal of Computers in Mathematics and Science Teaching*, 21(2):167–181, 2002.

[32] V. K. Kumar and James L. Rogers. Instructional uses of the Olin experimental classroom. In *Proceedings of the ACM SIGCSESIGCUE technical symposium on Computer science and education*, pages 189–191, New York, NY, USA, 1976. ACM. doi: 10.1145/800107.803472.

[33] Robert E. Silverman. Auto-instructional devices: Some theoretical and practical considerations. *The Journal of Higher Education*, 31(9):481–486, 1960. doi: 10.2307/1978637.