OJMS – Online Journal Management System

Sandeep L. Yadav, Mahesh T. Chaubey, Ritesh R. Gupta, Dinesh G. Kumavat ,PrathameshTugaonkar

Computer Department, Terna Engineering College, Mumbai University

***Abstract*—Online journal submission systems help lecturers with time management concerns and student evaluations processes. The aim of this paper is to explore available tools, design and develop an interactive web based application which can be used by computer science students to submit programming journals and get real-time feedback. The online application which has been designed and developed within the scope of this paper consists of an authorization, submission and analysis component.**

Keywords—Online submission; programming languages; automated assessment; e-learning

I. INTRODUCTION

The learning curve is the most important step at the early stages of using a programming language. Among the most popular ones taught at universities level [1, 2, 3] are C/C++ and Java. While many of the programming languages follow the Clike syntax, making it easier for programmers and developers to become familiar with the programming environment and syntax, it implies that educational institutes and universities need to emphasize on full-featured programming language with an easy learning curve and support from different communities.

Universities offer programming courses to most of the science and engineering students. The focus of this paper is on the authors' affiliated university, namely the University of Mumbai, for the Department of Computer Science in which Programming Principles I & II are the required first year courses for computer science and computer engineer students. The nature of the Computer Science Program and its courses requires that students spend a considerable amount of their studying time in computer laboratories in order to familiarize themselves to the programming environment, in our case C++ using Visual Studio.

Journals are given to students on a regular basis, and are mostly practical in nature, i.e. by demanding actual writing of source code. In order to evaluate these journals, each individual's programming code is checked and executed by lecturers. As it is well known, this takes valuable time for lecturers to evaluate and follow student's performance.

The nature of programming courses allows us to automate the task of compilation and running the program and retrieving the result of the software being run. This programming online journal submission and evaluation system (OJMS) was developed to be used by both students and faculty members in collaboration. Instructors are given the opportunity to create journals clustered in categories, share them with other instructors and allow the students to attempt to solve some of them. Students can get real-time feedback about their code – for example if it was successfully compiled, if it has some syntax errors or run-time errors, and different analytics on their performance. The submission system makes it easier for students to work on journals while it helps instructors to record students' results for evaluation online, everything in one place.

The aim of this project is to design and develop the journal submission system and use the submitted data to analyse difficulties a student may face, while in the same time allow lecturers to easily manage their journals and help them with the evaluation.

This paper is organized as follow: we present in Sec. II some existing tools for automated submission of programming exercises. Sec. III provides a detailed analysis for the design and development of our submission system, including the main tools used and its performance. Sec. IV describes the implementation according to the specification and functional requirements. Finally, the last section summarizes, concludes and suggests possible further improvements of this project.

II. EXISTING ONLINE SUBMISSION SYSTEMS

Several tools for both static and dynamic assessment of computer programs have been reported in the literature. Automatic assessment tools can be used to help teachers in grading tasks, as well as to support students' working process with automatic feedback – surveys can be found in [4, 5].

[6] showed that tools that support online programming lectures can be divided into:

- visualization tools [7],
- automated assessment tools [8,9] and
- support tools [10]

The focus here concerns automated assessment tools. Literature often focuses on source code analysis [11, 12] with developments on the semantic and conceptual analysis [13, 14] of code for a better evaluation. Indeed a weakness of existing automated assessment tool is their lack of granulation, i.e. journals are either evaluated correct or wrong. OJMS answer to a different kind of need, as lecturers of University of Mumbai need an in-house system, adapted and tailored to the computer science and computer engineer curriculum. Moreover, a motivation for developing OJMS was to allow for a peer to peer collaboration between lecturers at the university by sharing exercises across different sections while allowing individual lecturers to tailor them as they want.

III. OJMS Components

An online journal submission system allows students to submit securely journals online, and contains a set of tools to allow lecturers to mark journals online and give the opportunity to students to get real-time feedback including syntax error recognition and other compilation and run-time errors. The submission system was developed using web technologies for portability and security purposes. OJMS (**P**rogramming **Journal**s **S**ubmission **S**ystem) is used for submitting C++ programming journals and was developed mainly with PHP/MYSQL for the server side functionalities or the backend and HTML/JavaScript for the front-end, which is the interface interacting with the user.

A. Functional Requirements

A functional requirement defines the exact components needed for a system in order to perform an action and execute the instruction. Behavioral requirements describing all the cases where the system uses the functional requirements are captured in used cases. The overall components of the system are illustrated in the following use-case diagram of Fig. 1.

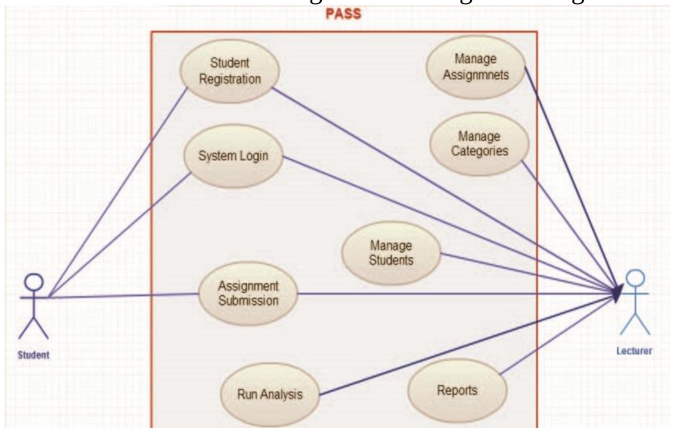


Figure 1: The use-case diagram for OJMS

B. IDEONE Introduction

Student's submission contains a source code which is compiled and ran against constraints set by the instructor, as explained in the next section. Compiling source programs requires a compiler and developing an automated compiler was out of the scope of this project. In order to process and compile the student's code, services like IDEONE.com offer online compilation and debugging services and let users compile more than 60 programming languages including C/C++ and JAVA.

IDEONE is accessed through ideone.com. During the registration phase of a new student in OJMS, users (students) are required to sign up with IDEONE in order to get an API credentials which will be used to send the queries, following their terms and conditions of use. IDEONE is used to supply standard input and can be handled like real interactive input. The API includes methods of data input, submission, validation with errors and finally the result. OJMS uses IDEONE API only for the compilation process using SOAP protocol. [15, 16]

Simple Object Access protocol (SOAP) is a protocol specification for exchanging structured information in the implementation of web services. It uses XML Information Set for its message format, and relies on other application layer protocols mostly by HTTP. IDEONE uses SOAP specification for accessing the methods as it is neutral - it can operate independently and over any transport protocol (e.g. HTTP and SMTP), allows for flexibility on the programming paradigm used and makes use of open standards. Although other middleware techniques could have been employed, SOAP was used for its simplicity.

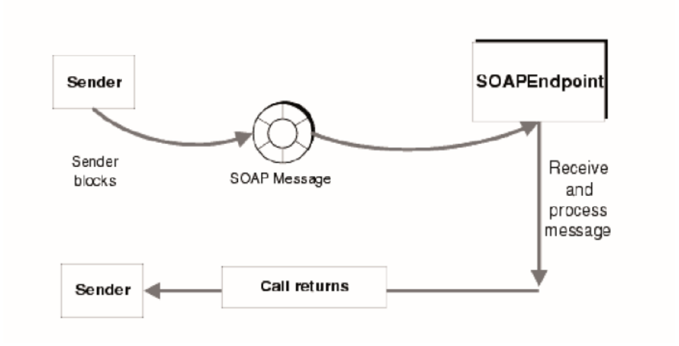


Figure 2: SOAP layer representation [17]

C. Student Registration

OJMS requires each student to login with their "Student Intranet" credentials – the University Intranet provides students with on-line access to all the University's rules and regulations regarding admission, academics and finance, information on student life, as well as personal student information (timetable, exam schedule, academic path, transcript, progress reports, financial statements, etc.). These credentials are given to the student upon registration for their program of study. In addition, as mentioned before, in order to use IDEONE compiler, students are required to register on IDEONE to obtain API credential access which later will be used in OJMS to call the API compilation/debugging methods.

D. System Login/Authentication

There is an API developed for authenticating student intranet users by supplying their intranet username and ojms word. In the journal submission system, this API is necessary to authenticate the students and allow them to access the platform. Similarly, lecturers are required to login to get access to the administration area.

E. Journal Submission

Students and lecturers can submit journals without any limitations for both of the users. Lecturer's submission data is not saved for data analysis but student's data is saved for later processing. The user interface (UI) was built to be as simple as possible and to allow users to code online, see the compilation results and review their submission.

Submission data should be complete enough to contain the required data for the analysis part of this project. These data include:

- Student/Lecturer ID
- Timestamps
- Source Code
- Time used from writing to submitting the data
- Errors which include of a set of constraints

F. Management Area (Administration Area)

This administration area is used only by the lecturers to perform different tasks, e.g., creating categories and journal, modifying them, view students data submissions and performing analysis reports.

1) Category Management: In this section the instructor is able to create and modify categories – here called chapters. The important attributes needed in this section are title, description of the chapter, timestamps and finally a keywords array which will be used by the analyser to search for different patterns in the source code by using these keywords and relate the data together for a better and more accurate result.

2) Journal Management: Journal management is also carried out by the instructor. Each journal belongs to a category or chapter in order to categorize them. It includes a title, a chapter id, keywords, the lecturer who created the journal, the output of the journal, difficulty indicator and standard input and output. Each specific journal is supplied with a source code solution together with a series of inputs and expected outputs. The outputs are used in order to verify the correctness of the journal, and values that will be bounded to the journal by the instructor and is expected to be seen in the result of the journal submitted by the student in order to be flagged as solved. OJMS uses a series of regular expressions to extract the data and carry out the comparison. Moreover an optional source code template can be set by the instructor if the exercise requires specific formatting or approach.

3) Students Management: This component is read-only in which lecturers can view the solved journal by each specific student and see the total solved exercises. It is used for evaluating the student's performance.

IV. OJMS Implementation

The application developed for this project is written in PHP for the server side programming and for the front end uses HTML, CSS for styling and JavaScript for client-side interaction.

A. PHP/MYSQL

PHP is used for server side scripting, and MySQL as the database management system. There is also a web-server needed to communicate with the client's web browser. The most widely used web-server is Apache which is open-source. WampServer is a Windows web development environment, allowing users to create web applications with Apache2, PHP and a MySQL database. Alongside, PhpMyAdmin allows you to manage easily your databases. WampServer includes all the required tools for developing the application and is available under Windows and Linux distributions.

Although MySQL is known for its relatively poor performance scaling, the initial size of our application is small enough for it not to be an issue, while its strong data protection features was essential for an online grading tool. MySQL provides powerful mechanisms for ensuring only authorized users have entry to the database server, and powerful data encryption and decryption functions ensure that sensitive data is protected from unauthorized viewing [18].

B. Software Design

OJMS has been developed based using Object Oriented Programming (OOP) approach. Thus it consists of a collection of classes with their data members and methods interacting with each other as directed by the application functional requirements. The application's entity relationship is shown in Fig. 3 with the different classes and fields.

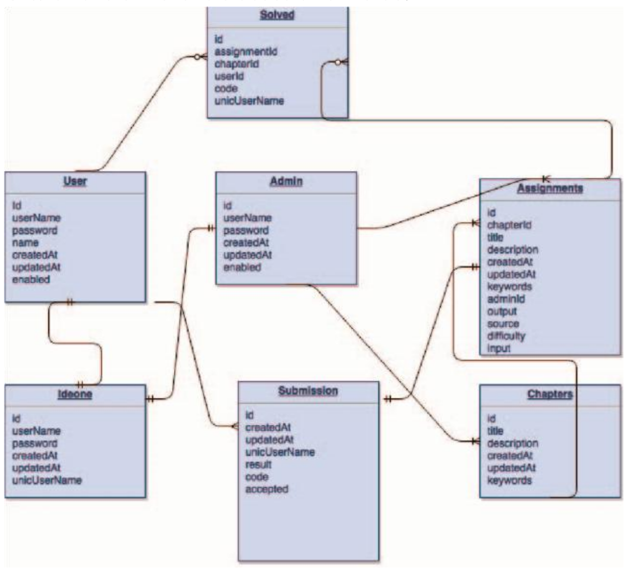


Figure 3: OJMS entity relationship diagram

Each entity in Fig. 3 represents a table in the MYSQL database and is used to relate the component of the system, as well as the data used throughout the application.

C. Software Components Implementations

In this subsection a detailed description of the major components of OJMS, together with their functionalities, is provided. During the functional requirements analysis phase, which was done in the previous section, each component was briefly analyzed and the data members needed were specified.

1) Authentication

The user interface of the authentication components is simple and user oriented. It asks for a username and a ojmsword as illustrated in Fig. 4. Students' username and ojmswords are their intranet credentials as previously mentioned. This was done to simplify OJMS use and reinforce its role as a teaching tool, alleviating the need of new username and ojmsword for the students. Security being a concern, in order to secure users' ojmswords, only the SHA hash is stored in the database. Upon login by users, only the hashes are transmitted and compared.

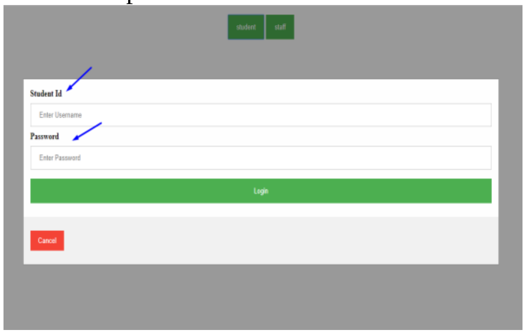
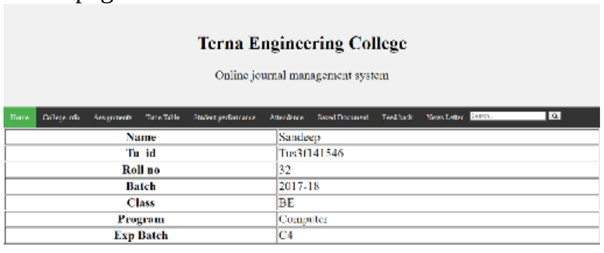


Figure 4: Screenshot of Login

2) *Student Panel – Journal List*

Students upon a successful login are directed to the panel where each chapter is listed with the corresponding exercises. The UI of our toy example is shown in Fig. 5. The level of completion is indicated next to each chapter title. Upon selecting a chapter, students are redirected to the list of exercises, including the ones already solved. Students have access to those in order to be able to review them as OJMS provides both their solution and the lecturer's solution. When students click on a new exercise, they are sent to the journals submission page.



Terna Engineering College

Online journal management system

Name	Sandeep
Tu_id	Tu31141546
Roll no	32
Batch	2017-18
Class	BE
Program	Computer
Exp Batch	C4

Figure 5: Screenshot of Student Panel

3) *Journal Submission*

Journal submission page contains a text area where the source code will be written. The screen is divided into two areas, with the coding area on the left, containing the optional source code template. The output box in the right side of the page is placed to show the compilation result, either the list of syntax errors, logical errors or indicated the successful completion of the exercise.

In this page the software makes use of the IDEONE API methods. The function creates a new SOAP client, and asks for submission permission. If the result is successful, the results from IDEONE are retrieved and parsed.

4) *Administration & Management*

The administration panel as specified in the previous chapter includes all the required elements. Each exercise has a title, a chapter in which it belongs, a difficulty level and the lecturer who created it. The chapters and difficulty levels are used to analyze students' performance and, in a future stage of development of OJMS, for recommendation.

5) *Database Connector & Configuration*

Almost all the data used by OJMS are stored within a relational database. This data might be needed at any point of time so a configuration file and a database class has been created and used throughout all the components.

6) *CSS Styling*

Cascading Style Sheets (CSS) is a style sheet language used for describing the look and formatting of a document written in a mark-up language such as HTML. All the pages share a main style sheet and make use of the style properties which uses the University of Mumbai styles and colour in order to provide consistency and reinforce the importance of this tool as learning interface.

V. Discussion, Implication and Conclusion

"Practice makes perfect", then practical journals make programming perfect. Programming journals help students to gain experience in the programming field and overcome the learning curve issue. OJMS can be easily integrated in the traditional universities curriculum of assigning programming journal by simply allowing students to write the program and get immediate and accurate feedback for personal evaluation. The aim of this project was to design and develop a tool where students and instructors could both benefit. For instructors evaluation and time management problems were tackled, while students are now able to spot their difficulty domains and practice even more on these domains.

A limitation of our approach is its recent implementation, rendering it difficult to compare with existing work on automated submission systems [19]. Once OJMS reaches some maturity, with significant users' data, we are considering using a technology acceptance model approach to evaluate it. Although a slight overall improvement of students' grades by the end of the semester was observed, this empirical evidence needs to be thoroughly and systematically tested once enough students will have used OJMS. A common feedback received from students is the lack of intelligent code completion, which means that some students implement exercises using tools like Visual Studio and copy their answer on the web page. Novel approaches are also investigated to simplify another bottleneck for lecturers to convert their exercises into a format fitting for OJMS, through sharing of exercises and solutions. Another weakness of OJMS is that it is currently supporting only one programming language, thus limiting its use [21].

This project will be further expanded by the development of a specialized plagiarism tool [20] for detecting code which is identical during submission process. Current essay based plagiarism tools are not adapted to the special case of programming journals. Further expansions would also include the implementation of an interface to design and draw analytics in the web environment for the lecturer to visualize students' progress. The next implementation phase is to transform our web implementation into an app, to allow students to submit their journals on the go. Finally, implementing an algorithm for a recommender system which basically recommends journal to students based on their submitted data and other factors is under consideration.

ACKNOWLEDGMENT

The authors thank Prof.PrathameshTugaonkar for his collaboration and participation on the development on this project, and are grateful to the anonymous referees for their constructive input.

REFERENCES

[1] M. de Raadt, R. Watson, and M. Toleman, "Language trends in introductory programming courses", in Proceedings of the 2002 Informing Science+ Information Technology Education Joint Conference (InSITE 2002), pp. 229-337, 19-21 June 2002

[2] M.S. Farooq, S.A. Khan, F. Ahmad, S. Islam, and A. Abid, "An evaluation framework and comparative analysis of the widely used first programming languages", PloS one, 9(2), 02., 2014.

[3] P. Guo, "Python Is Now The Most Popular Introductory Teaching Language At Top U.S. Universities", Cacm.acm.org, N.p., Web. 15 June 2015.

[4] K.M. Ala-Mutka, "A survey of automated assessment approaches for programming journals.", Computer science education 15.2, pp. 83102, 2005.

[5] P. Ihantola, T. Ahoniemi, V. Karavirta, and O. Seppälä, "Review of recent systems for automatic assessment of programming journals." Proceedings of the 10th Koli Calling International Conference on Computing Education Research. ACM, pp. 86-93, 2010.

[6] A. Pears, S. Seidman, C. Eney, P. Kinnunen, and L. Malmi, "Constructing a core literature for computing education research", SIGCSE Bulletin, 37(4), pp. 152–161, 2005.

[7] A. Moreno, N. Myller, E. Sutinen, and M. Ben-Ari, "Visualizing programs with Jeliot 3", In Proceedings of the International Working Conf. on Advanced Visual Interfaces, Gallipoli (Lecce), Italy, pp. 373 – 376, May 2004.

[8] S. H. Edwards and M . A. Ṕerez-Qui nones . ̃" Web -cat: automatically grading programming journals", In ITiCSE'08: Proceedings of the 13th annual Conf. on Innovation and technology in computer science education, New York, NY, USA, pp. 328–328, 2008.

[9] M. Joy, N. Griffiths, and R. Boyatt, "The BOSS online submission and assessment system", Journal on Educational Resources in Computing (JERIC), 5(3), 2, September 2005.

[10] D. J. Barnes and M . K̈ olling , "Objects first with Java – a practical introduction using BlueJ", Third edition, Prentice Hall / Pearson Education, 2006

[11] M. Novak, M. Binas, M. Michalko, and F. Jakab, "Student's progress tracking on programming journals", IEEE 10th International Conference on Emerging eLearning Technologies & Applications (ICETA), pp. 279-282, 2012.

[12] D. Binkley, "Source Code Analysis: A Road Map", Future of Software Engineering,FOSE '07, pp. 104-119, 2007.

[13] U. Priss, P. Riegler, N. Jensen, "Using FCA for modelling conceptual difficulties in learning processes", In: Domenach; Ignatov; Poelmans (eds.), Contributions to the 10th International Conference on Formal Concept Analysis (ICFCA 2012), 161-173, 2012.

[14] U. Priss, "Using FCA to analyse how students learn to program." In: Cellier&Distel (eds.), Proceedings of ICFCA'2013, Springer Verlag, LNAI 7880, p. 216-227, 2013.

[15] K. Scribner, K. Scribner, and M. C. Stiver, "Understanding Soap: simple object access protocol", Sams, 2000.

[16] D. Box, D. Ehnebuske, G. Kakivaya, A. Layman, N. Mendelsohn, H. F. Nielsen, T. Satish, and D. Winer. "Simple object access protocol (SOAP) 1.1." (2000).

[17] Sun Java System, Message Queue 3.5 Java Client Developer's Guide, https://docs.oracle.com/cd/E19263-01/817-6026/SOAP.html, retrieved Sept. 2015.

[18] MySQL website, http://www.mysql.com/, retrieved Sept. 2015.

[19] S. Jain, M. Singhal, and A. Shah. "Exploring the Usage of Existing Plagiarism Tools for Automated Student Assessment for Java Program." International Journal of Information and Education Technology 6, no. 3, pp. 219-223, 2016

[20] M. Kaya, and S. A. Özel. "Integrating an online compiler and a plagiarism detection tool into the moodle distance education system for easy assessment of programming journals.", Computer Applications in Engineering Education 23, no. 3, pp. 363-373, 2015.

[21] A. Patel, D. Panchal, and M. Shah. "Towards Improving Automated Evaluation of Java Program." In Emerging ICT for Bridging the FutureProceedings of the 49th Annual Convention of the Computer Society of India (CSI) Volume 1, pp. 489-496. Springer International Publishing,
2015