Web-based educational ecosystem for automatization of teaching process and assessment of students

Boban Vesin
University of Southeast Norway
Vestfold, Norway
boban.vesin@usn.no

Aleksandra Klašnja-Milićević
Faculty of Sciences
University of Novi Sad
Novi Sad, Serbia
akm@dmi.uns.ac.rs

Katerina Mangaroska
Norwegian University of Science and Technology
Trondheim, Norway
mangaroska@ntnu.no

Mirjana Ivanović
Faculty of Sciences
University of Novi Sad
Novi Sad, Serbia
mira@dmi.uns.ac.rs

Rodi Jolak
Chalmers University of Technology and Gothenburg University
Gothenburg, Sweden
jolak@chalmers.se

Dave Stikkolorum
Leiden Institute of Advanced Computer Science
Leiden, The Netherlands
d.r.stikkolorum@liacs.leidenuniv.nl

Michel Chaudron
Chalmers University of Technology and Gothenburg University
Gothenburg, Sweden
chaudron@chalmers.se

ABSTRACT

The complexity of the teaching process at universities creates many challenges. It becomes much harder for teachers to observe, control and adjust the learning process. Teaching process can be enhanced with use of different educational systems that not only help students construct their knowledge, but also make this process the most effective and efficient. One of the processes that could be automated and supported is the assessment of students' assignments. Three e-learning systems are currently used at different universities for teaching software design basics. The goal of this paper is to propose new integrated tool that can be used in university courses to support different stages of learning and evaluation of students' assignments. Such integrated system will be used to simplify the correction process of software design assignments.

CCS CONCEPTS

•**Applied computing** → E-learning •**Software and its engineering** → Software design engineering

WIMS '18, June 25–27, 2018, Novi Sad, Serbia

ACM ISBN 978-1-4503-5489-9/18/06…$15.00
https://doi.org/10.1145/3227609.3227662

KEYWORDS

Students' assessment, e-learning, software design

1 INTRODUCTION

Among the multiple concerns and goals of technology-enhanced learning (TEL), many are oriented to make teaching processes more efficient. Improvement of learning processes at universities around the world can be enhanced by different tools and learning environments. To achieve this, systems need to collect data about students, automate their assessment and provide actionable feedback on their progress and performance. Furthermore, even if the analysis of the students' data generates useful insights, those insights need to be properly interpreted [24] and presented to students, as well as used for further improvement of course design and teaching practices.

In previous years and as a part of the different projects, several WEB-based tools have been developed and used in different university courses at different universities. These tools have been used to support teachers in lecturing computer science university courses, especially in the area of software design and development:

- ProTuS - personalized tutoring system [10,15].
- WebUML - simplistic online UML editor.
- OctoUML – Software design environment [12].

The focus of this study is to investigate possibilities for integration of the existing systems and creating an online educational ecosystem for teaching and automated assessment of students in software design courses. In order teachers to use these tools in their lecturing processes, the

tools need to be integrated in efficient and practical way; therefore, the integration process of these systems is shifting from a research problem into an important practical task [29].

Currently, ProTuS offers programming courses with several personalization features. It automatically grades learners' programming assignments and recommends coding exercises that match their current knowledge. As part of the integration process, presented in this paper, ProTuS will offer software design courses where the evaluation process of student's assignments will be automated. This way students can develop self-directed learning (SDL) skills while engaged with content and resources in ProTuS to correctly represent the system requirements in UML model.

Several studies were performed to determine the feasibility for development of such personalized ecosystem and the potential benefits it could emerge [11,12]. Moreover, teachers and instructional designers from different universities used these tools separately and provided feedback on the developed features. The idea for integrating different tools is part of the effort to expand existing learning platform by automated and semi-automated review of student's assignments.

In the next section, we provided a brief literature review on the area of automated assessment in e-learning. The learning processes necessary for implementation of automated assessment will be presented in section three. After that, we will present the existing tools and the details about the architecture that will be used to integrate these tools into unique educational ecosystem for automatic and manual student assessment. Conclusions and the future directions of research will be presented in the final section.

2 RELATED WORK

The distributed and flexible nature of the learning process creates new challenges. It becomes much harder for teachers to observe, control and adjust the learning process [38]. Educational systems should not only support transfer of knowledge to students, but should make this process effective and efficient [20]. In the academic context, e-learning tools should provide not only basic learning functionalities for which the tools have been designed (e.g. presentation of learning material) but also additional functionalities related to other academic goals, such as continuous assessment, tracking student work, personalization, etc. [28].

Software design is a complex process and different modelling web tools have been proposed for creating software models [28]. Most of these tools have been designed with advanced users in mind and with many options that won't be used by novices and less experienced students during their learning process. The design tools intended for software design courses should offer some specific functionalities, such as automatic correction of the assignments and feedback about detected errors [28].

Several researches presented overviews of the recent trends in automatic assessment tools [9,30] and demonstrated or proposed methods to automatic assessment of programming students' assignments [1,17,22,23,25–27,35,39] and assessment of software design diagrams [2,4,5,7,28,34]. These web tools present problems, grade student's assignments, and provide corrective feedback. However, in the case of a software design course, they mainly focus on class diagrams [2,6,28,33] since the evaluation process can be easily automated due to the simplicity of its representations within the files. Rare examples incorporate automatic assessment of other UML diagrams such as: use cases [6,7], sequence [7], activity diagrams [34], or deployment diagrams [7]. None of the previous systems offer a complete support for UML assignments. The first prototype of the integrated environment presented in this paper includes evaluation of Class and Activity diagrams. The environment will be later extended with a support for the remaining types (primarily use-cases, sequence, state, component, and deployment diagrams).

Nonetheless, the previously presented approaches do not consider the personal specificities of students to carry out a truly adaptive teaching–learning process, nor compliment with specific learning objectives for individuals [3]. Thus, the presented tools are limited only to certain domains, and it would be difficult to adapt them for student's assessment in different domains.

2.1 Personalized learning

For personalizing e-learning, several different strategies and characteristics can be used and considered by teachers and course designers [3,8,18,21]. The decision which personalization strategy to use depends on many factors, varying from the learning goals to the topics being thought. In any case, personalized learning requires personalized and automated assessment, and it is the responsibility of teachers and institutions to develop flexible courses [40]. The integration of several e-learning tools proposed in this paper aims to make assessment practices more flexible and personalized.

The goal of our research is to develop an intelligent personalized web-based learning ecosystem that will consider the learning objectives and the personal profiles of students, to offer tutorials and evaluation of assignments that fit user's requirements. The system will offer personalized courses with learning analytics reports for both, students and teachers. Students will be more involved in decision-making about the learning process, and will have flexibility while working on their assessments [40].

The presented integrated learning environment is a versatile tool that offers more content interactivity and automatic assessment of different types of students' assignments. The environment incorporates the features for handling the assessment differently depending on the course

goals. Learners attending programming course will benefit from the customized Junit tests, while software design students will be automatically evaluated based on the parsing of XML submission files.

3 LEARNING PROCESS

The proposed integrated system should offer online courses, propose exercises based on the level of knowledge of learners, and provide appropriate feedback. In addition, the system should automatically correct student's assignments, and as such automate the teaching process. In order to provide these features, the learning process in the integrated environment should include the following phases:

- Students take online tutorials supported by video material within the learning environment.
- The system presents a domain description that needs to be modelled.
- The students submit solutions via modelling environment.
- The system evaluates the submitted XMI (XML Metadata Interchange) solution and provides automated assessment.
- Teachers can overview the submitted solution, generate feedback, and provide additional comments.
- The actions of all users are monitored and recorded.
- The system generates reports and visualize the results of the learning process and students' evaluation.

The first milestone of our research is to provide efficient ways to evaluate UML diagrams submitted by students. Thus, students can get feedback for their assignments in two steps:

- **Automatic assessment**. In this step, the list of comments on the students' diagram in a text format is automatically generated. This guidance will be used by students to reflect on their assignments and possibly modify the submitted diagrams if they contain any errors. The system can produce grades that could be used by students to evaluate their level of understanding in designing system requirements using UML diagrams. As there exist many different solutions to a specific problem, the feedback is presented to the student only as guidance.

- **Feedback provided by teachers**. By reviewing automatically generated feedback given by this system, teachers should be able to provide additional comments and corrections.

This double assessment process can provide students with useful feedback based on which they can correct their submission and learn from their mistakes. Besides that, students can submit their answers for multiple assessments and receive immediate feedback on the answers.

4 SOFTWARE REQUIREMENTS FOR INTEGRATED ENVIRONMENT

The goal of the integrated system is to help students to comprehend design modelling and learn how to represent the elements of a software system using Unified Modeling Language (UML). It should support teachers to track and analyze learning progress of students in real-time, during the online learning sessions. The idea is to expand existing learning platform by automated and semi-automated review of student's assignments.

We created a preliminary list of functionalities that could be supported with the proposed integration:

- overview, access and browsing of teaching material;
- online UML editor for entering the solution (graphical interface that allows drawing UML class diagrams);
- automatic assigning of exercises to students;
- save and load options;
- submission of solutions to teachers;
- automatic correction of exercises solved by the students;
- immediate feedback and reports about errors for each students' solution;
- interface for teachers to perform the online correction of assignments;
- history of students' trial and errors, and tracking of student progress (monitoring and recording of students' actions);
- communication channel between students and teachers.

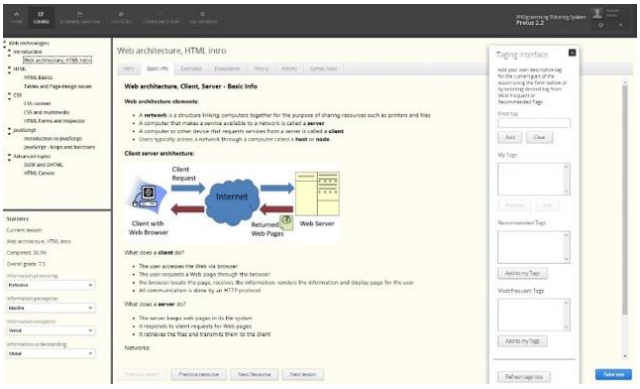

Figure 1. ProTuS – Programming tutoring system

As some of these features are already provided by the existing tools, our task is to develop new features. Additional modules that need to be developed are:

- correction module that will grade the assignments,
- analytics module for the collection of student's data, and
- communication channel between the system's components.

Details about these modules will be given in the following sections.

4.1 ProTuS[1]

ProTuS is a tutoring system designed to provide learners with personalized courses from various domains [16]. It is a web-based interactive system that allows learners to use teaching content prepared for different courses and examine their acquired knowledge (Figure 1) [13,36].

ProTuS has been built upon architecture that integrates and interconnects cross-platform analytics capabilities. The system automatically evaluates coding exercises based on Junit tests and Elo-rating algorithm, and generates personalized recommendations of educational content in alignment with the collected students' data [16].

4.2 WebUML

WebUML is a simplistic online UML editor which has been designed and implemented for a minimum subset of the unified modelling language - UML [31]. The goal of the tool is to provide a possibility to embed it in different online education software environments, such as serious games or gamified courses. WebUML was designed to address a large number of students, independent of location or time (Figure 2).

Currently, WebUML targets only UML class diagrams equipped with the most used relationships, attributes, operations, labelling, and multiplicity elements. The editor supports import and export of XMI and creation of .png files. In addition, it is also capable of logging students' activities, such as creating elements, movements, deletion etc. [32].

Together with the LogViz component (visual log analyzer), it is possible to visualize WebUML's logs. As presented in [31], by visualizing the students' activities, it is possible to identify typical strategies students use to make class designs. LogViz is capable of displaying the designers' activities in WebUML over time and auto-classify the log files by the students' design strategies [32].

4.3 OctoUML

OctoUML, is a software design environment that supports collaborative software design and enables different input methods for creation of software models at various levels of formality [37]. It provides means to allow the creation of both, sketchy hand-drawn elements and formal notations simultaneously, using different input devices ranging from desktop computers, over touch screens, to large electronic whiteboards. Moreover, it is equipped with tools for multi-touch and sketch recognition that allows the transformation of sketchy designs into formal notations (Figure 3).

[1] protus.idi.ntnu.no

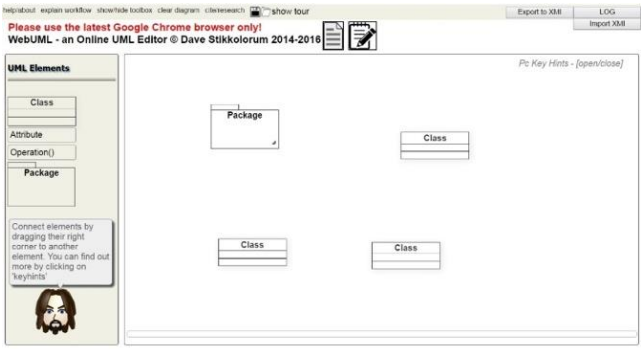

Figure 2. User interface of WebUML

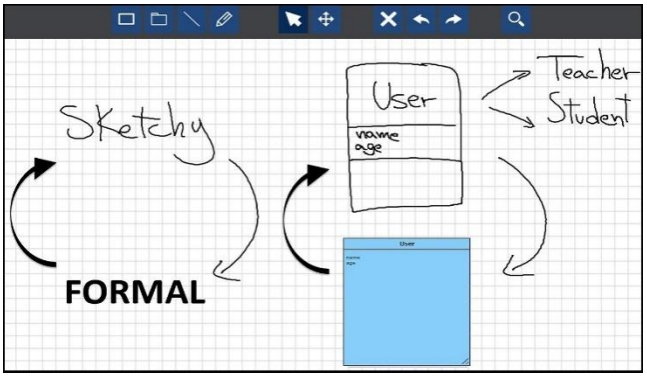

Figure 3. OctoUML – Software design environment

5. ARCHITECTURE

The following section describes the design and implementation of integrated learning environment into unique ecosystem. The proposed architecture integrates previously presented tools and creates an adequate infrastructure for communication between them. Special attention is given to the modules for automatic assessment of students' assignments. Furthermore, we explain how assignments are presented to students, how they submit their solutions, and how the automatic feedback is generated.

The integrated environment provides the functionality of an adaptive learning environment that helps students navigate through available learning resources, keeps track of student's actions, provides automatic assessment and updates the learner model (with the specific learners' knowledge of course topics) accordingly [14].

The four main modules in this architecture are (Figure 4):

- **User Interface (UI) components**. It is represented with two separated web interfaces: one for overview of courses and teaching material, and the other for editing and submitting of solutions (UML editor).
- **Data storage.** All educational material, assignments, solutions and data about students will be stored within this module.
- **Services.** This module includes services for performing different tasks within the environment

(assessments, collaboration, generation of reports, update of learner models, etc.).

- **Communication module**. This module will include different communication channels for delivering educational material to students and submissions of their solutions for automatic and manual assessment.

As part of these four basic modules, the system contains separated components that will be responsible for specific tasks within the learning environment:

- **Learning portal** – user-friendly personalized Web interface for students.
- **Knowledge repository** - represents the domain knowledge as a collection of learning resources (tutorials, videos and tasks).
- **Learner model** - represents the collection of learners' profiles with detailed logs of their interaction and performances.
- **Personalization module** - performs the actions that direct the automated adaptation of the system towards the needs of specific learners.
- **Learner analytics module** - intelligently analyses and visually presents the data from the student model to both, students and teachers.
- **Design environment**–is an online UML editor.
- **Management console**–is the tool for automated evaluation of students' assignments.
- **Analysis component** - performs software model analysis. This tool will be used to automatically evaluate the created software models to detect general design flaws, security flaws, and performance bottlenecks.
- **Assessment module** - involves the verification of the submitted assignments (including class object and relationship notation).

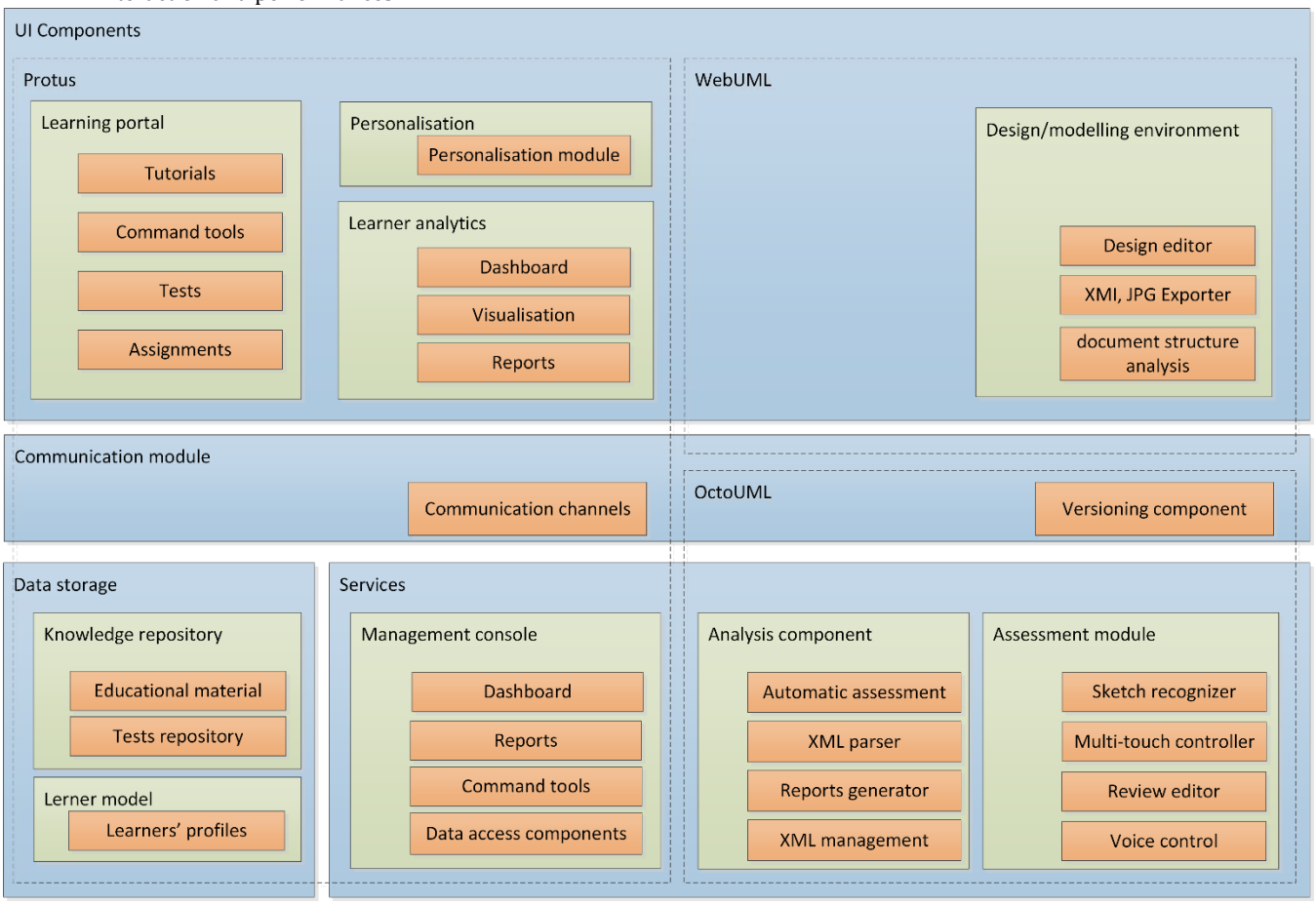

Figure 4. Proposed architecture of the integrated educational ecosystem

The proposed architecture presents the foundation for the development of integrated learning environment that can be used for personalized learning, automated assessment of learners' knowledge, and visualized data of teaching practices.

5.1 Evaluation of students' assignments

The process of student's solution assessment will be executed through the following stages (Figure 5):

- Assignments are presented to students (*ProTuS*)
- The student enters his/her solution via web interface (*WebUML*)
- The student submits the solution via web interface (*WebUML*)
- Automatic part of the assessment (*Analysis component*). The system evaluates the solution by running the verification algorithm that compares the XML solution with the provided solution. Automated feedback is generated and presented to the student.
- Manual part of the assessment (*OctoUML*). Teacher (if necessary) evaluates the feedback, adds his/her comments, and sends the combined feedback to the student.
- The visual presentation of the students' results (Learning analytics module). Reports are automatically generated and presented to students and teachers.

In general, tutorials, examples and challenges are presented to the students in ProTuS. WebUML is used as an online editor for creating students' solutions, while OctoUML can be used by teachers to import solutions and provide feedback to students.

5.2 Automated evaluation of students' assignments

We proposed the development of an assessment module for UML diagrams, integrated with existing software design tools and learning environments. This tool analyses students UML diagram solutions which are submitted by students in the form of XMI files.

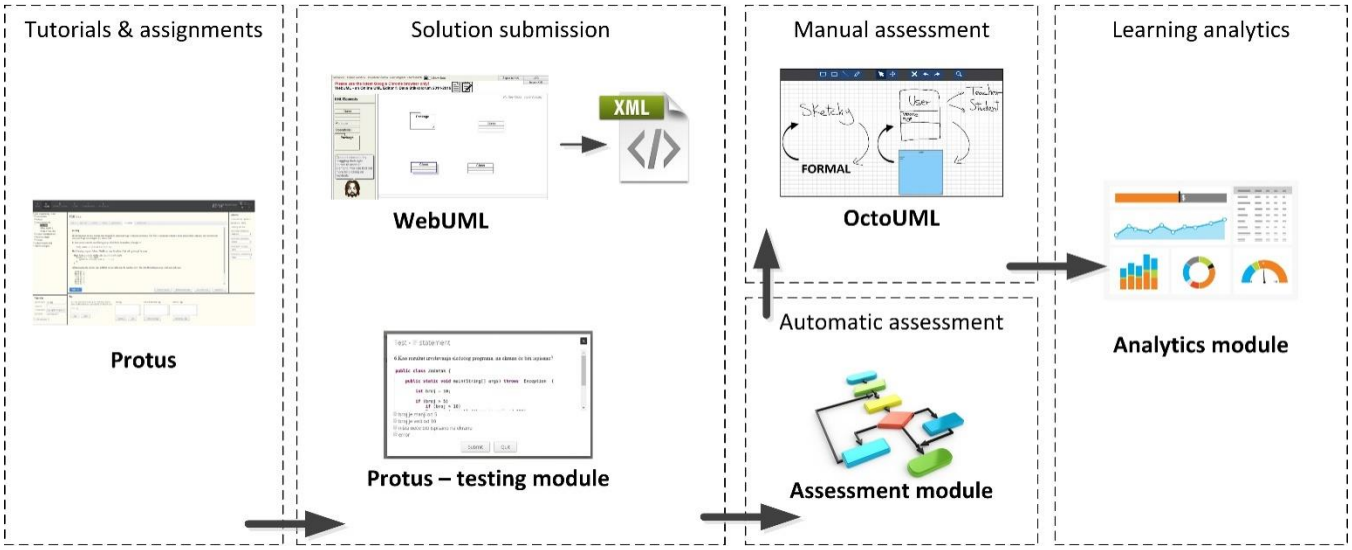

Figure 5. Automatic assessment in the integrated tool

The integrated system should be able to provide features for manual and automatic assessment. Moreover, it should provide verification mechanism that will examine student's solutions and investigate whether or not the solutions satisfy a given problem statement [2]. This will be determined by parsing the XML file of the solution and its comparison with the solution diagram created by the teacher (designer of the problem). The comparison will include (but should not be limited to):

- number and names of classes;
- number of attributes and their types;
- number of associations and their types.

The feedback is automatically generated by the system and presented to the students and teachers.

The comparison creates the graph (matrix) to represent an UML diagram. In the case of Class diagram, nodes of the graph represent classes, whilst connectors represent associations of class diagram. Similar graph can be created for other diagram types. For the initial integration, we will focus on class, sequence, component, and activity diagrams. Therefore, the system should be able to perform document structure analysis, check for the fundamental issues of diagram design, and decide what feedback to present to the student.

5.3 Manual evaluation of students' assignments

In the second phase of the evaluation, teacher (if necessary) evaluates the automatically generated feedback, adds his/her comments, and sends the combined feedback to the student. This process is done in OctoUML. This system offers formal and informal inputs. Therefore, students can create their solutions in the WebUML, and later this solution can be imported and edited in OctoUML in order to create feedback from the teacher.

6 EVALUATIONS

In order to evaluate the proposed concept, we execute several experiments on separated components to show that the presented features of web-based educational ecosystem can have important positive effects on learning and students' meta-cognitive skills. Besides, it is also relevant to detect the benefits this approach can bring, the preferred order of activities, and how one activity affects another.

In one of the studies, we performed a data collection with the teaching assistants (TAs) that are assigned to the course Web technologies at Norwegian University of Science and Technology, in Trondheim, Norway [19]. Teaching assistants are one of the user groups that apply ProTuS to follow students' learning paths and call for interventions if they perceive that particular student fails to keep up with the expected progress. Subsequently, a focus group was organized with the TAs to gather the best practices that they have accumulated over the last few years, working with students in the particular course.

In order to investigate the usability of the OctoUML system and issues like ease of use, efficiency, and user satisfaction we organized user studies at the University of Gothenburg, Sweden [11]. Sixteen subjects were engaged in a design assignment. The assignment was to create a UML class diagram of a given scenario using our tool. To give a global overview of the subjective assessment of our tool's usability, we asked our subjects to answer the System Usability Scale (SUS) questionnaire. Furthermore, we planned semi-structured interviews using both closed and open questions. The evaluation shows that OctoUML provides a user-friendly environment and has the potential to effectively support the activities of the designers.

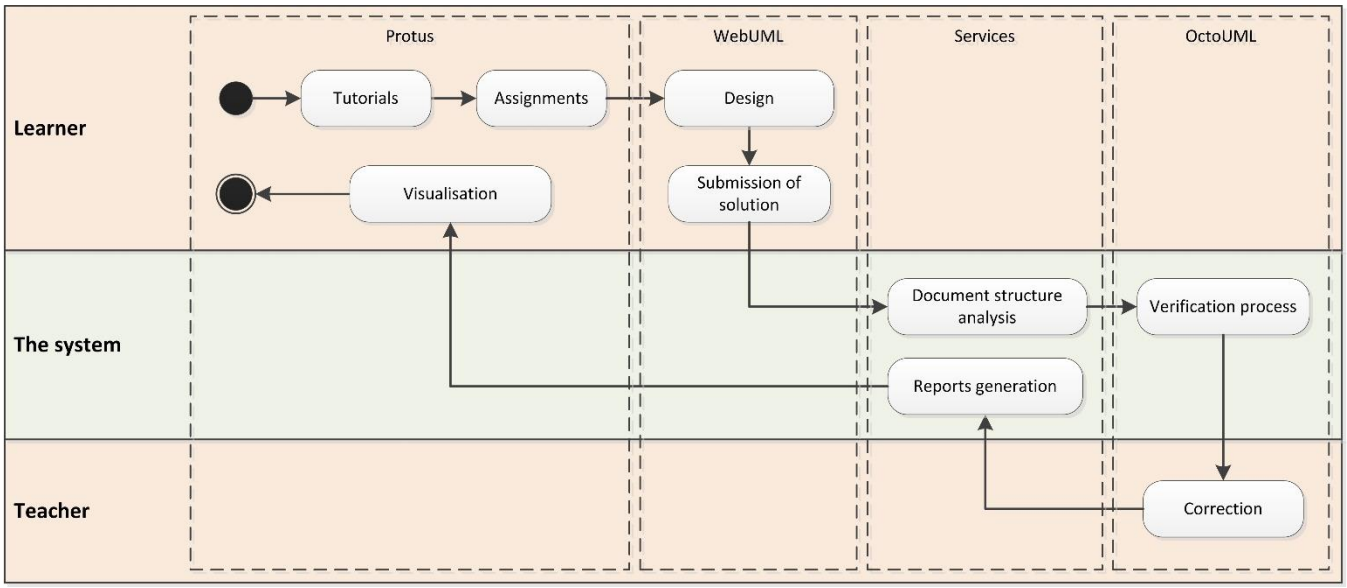

Figure 6. The assessment process in integrated environment

The results of the empirical experiments exemplify the benefits the component provides to students. The experiments showed that the students and the system and its component attractive and usable, which provided useful insights for further implementation of web-based educational ecosystem.

7 CONCLUSIONS

This paper presents an integrated learning environment that can be used for personalized learning, automated assessment of learners' knowledge, and visualization of teaching practices. We proposed architecture for adaptive e-learning environment that integrates the functionalities provided by three existing educational or software design tools.

The proposed architecture includes integration of existing tools (i.e. ProTuS, Web UML, and Octo UML) into full-scale web-based learning environment. This paper presents the first step that includes an implementation of automated and semi-automated assessment. Consequently, we expect that the integrated system will improve the efficiency and effectiveness of students' learning by providing them with personalized and automated learning process. We believe that this approach will significantly improve the teaching process in software design courses at university level.

However, further efforts are needed to implement the additional functionalities (i.e.) not already provided by the existing tools. For now, the integrated educational ecosystem can provide automated and semi-automated assessment of students' software design assignments.

In future, we aim to develop educational programs in the field of software design that fit students' profiles better. The

evaluation needs to focus on assessing the system's overall effectiveness and the accuracy of the students' modelling.

REFERENCES

1. José Luis Fernández Alemán. 2011. Automated assessment in a programming tools course. *IEEE Transactions on Education* 54, 4: 576–581.
2. Noraida Haji Ali, Zarina Shukur, and Sufian Idris. 2007. A design of an assessment system for UML class diagram. In *Proceedings - The 2007 International Conference on Computational Science and its Applications, ICCSA 2007*, 539–544. https://doi.org/10.1109/ICCSA.2007.2
3. Helen Ashman, Tim Brailsford, Alexandra I Cristea, Quan Z Sheng, Craig Stewart, Elaine G Toms, and Vincent Wade. 2014. The ethical and social implications of personalization technologies for e-learning. *Information & Management* 51, 6: 819–832.
4. Mohammed Basheri, Malcolm Munro, and Liz Burd. 2013. Collaborative Learning Skills in Multi-touch Tables for UML Software Design. *International Journal of Advanced Computer Science and Applications* 4, 3: 60–66.
5. Julia Clemente, Jaime Ramírez, and Angélica De Antonio. 2011. A proposal for student modeling based on ontologies and diagnosis rules. *Expert Systems with Applications* 38, 7: 8066–8078. https://doi.org/10.1016/j.eswa.2010.12.146
6. Helder Correia, José Paulo Leal, and José Carlos Paiva. 2017. Enhancing feedback to students in automated diagram assessment. In *OASIcs-OpenAccess Series in Informatics*.
7. Vittorio Cortellessa, Harshinder Singh, and Bojan Cukic. 2002. Early reliability assessment of UML based software models. In *Proceedings of the 3rd international workshop on Software and performance*, 302–309.
8. Fathi Essalmi, Leila Jemni Ben Ayed, Mohamed Jemni, Sabine Graf, and others. 2015. Generalized metrics for the analysis of E-learning personalization strategies. *Computers in Human Behavior* 48: 310–322.
9. Petri Ihantola, Tuukka Ahoniemi, Ville Karavirta, and Otto Seppälä. 2010. Review of recent systems for automatic assessment of programming assignments. In *Proceedings of the 10th Koli calling international conference on computing education research*, 86–93.
10. Mirjana Ivanović, Dejan Mitrović, Zoran Budimac, Boban Vesin, and Lubomir Jerinić. 2014. *Different roles of agents in personalized programming learning environment*. https://doi.org/10.1007/978-3-662-43454-3_17
11. Rodi Jolak, Boban Vesin, and Michel R.V. Chaudron. 2017. Using voice commands for uml modelling support on interactive whiteboards: Insights & experiences. In *CIbSE 2017 - XX Ibero-American Conference on Software Engineering*.
12. Rodi Jolak, Boban Vesin, Marcus Isaksson, and Michel R.V. Chaudron. 2016. Towards a new generation of software design environments: Supporting the use of informal and formal notations with OctoUML. In *CEUR Workshop Proceedings*, 3–10.
13. Aleksandra Klašnja-Milićević, Mirjana Ivanović, Boban Vesin, and Zoran Budimac. 2017. Enhancing e-learning systems with personalized recommendation based on collaborative tagging techniques. *Applied Intelligence*. https://doi.org/10.1007/s10489-017-1051-8
14. Aleksandra Klašnja-Milićević, Boban Vesin, and Mirjana Ivanović. 2018. Social tagging strategy for enhancing e-learning experience. *Computers & Education* 118: 166–181.
15. Aleksandra Klašnja-Milićević, Boban Vesin, Mirjana Ivanović, Zoran Budimac, and Lakhmi C Jain. 2017. Design, Architecture and Interface of Protus 2.1 System. In *E-Learning Systems*. Springer International Publishing, 185–212.
16. Aleksandra Klašnja-Milićević, Boban Vesin, Mirjana Ivanović, Zoran Budimac, and Lakhmi C Jain. 2017. Personalization in Protus 2.1 System. In *E-Learning Systems*. Springer International Publishing, 213–257.
17. Amruth Kumar. 2006. A scalable solution for adaptive problem sequencing and its evaluation. In *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 161–171. https://doi.org/10.1007/11768012_18
18. Eugenijus Kurilovas, Inga Zilinskiene, and Valentina Dagiene. 2015. Recommending suitable learning paths according to learners' preferences: Experimental research results. *Computers in Human Behavior* 51: 945–951.
19. Katerina Mangaroska and Michail Giannakos. 2017. Learning Analytics for Learning Design: Towards Evidence-Driven Decisions to Enhance Learning. In *European Conference on Technology Enhanced Learning*, 428–433.
20. Antonija Mitrovic, Brent Martin, and Pramuditha Suraweera. 2007. Intelligent tutors for all: The constraint-based approach. *IEEE Intelligent Systems* 22, 4: 38–45. https://doi.org/10.1109/MIS.2007.74
21. Eileen O'Donnell, Séamus Lawless, Mary Sharp, and Vincent Wade. 2015. A review of personalised e-learning: Towards supporting learner diversity.
22. Vreda Pieterse. 2013. Automated Assessment of Programming Assignments. *3rd Computer Science Education Research Conference on Computer Science Education Research* 3, April: 45–56. https://doi.org/http://dx.doi.org/10.1145/1559755.1559763
23. Juan C Rodríguez-del-Pino, Enrique Rubio-Royo, and Zenón J Hernández-Figueroa. 2012. A Virtual Programming Lab for Moodle with automatic assessment and anti-plagiarism features. In *Proceedings of the International Conference on e-Learning, e-Business, Enterprise Information Systems, and e-Government (EEE)*, 1.
24. María Jesús Rodríguez-Triana, Alejandra Martínez-Monés, Juan I. Asensio-Pérez, and Yannis Dimitriadis. 2015. Scripting and monitoring meet each other: Aligning learning analytics and learning design to support teachers in orchestrating CSCL situations. *British Journal of Educational Technology* 46, 2: 330–343. https://doi.org/10.1111/bjet.12198
25. Rohaida Romli, Shahida Sulaiman, and Kamal Zuhairi Zamli. 2010. Automatic programming assessment and test data generation a review on its approaches. In *Information Technology (ITSim), 2010 International Symposium in*, 1186–1192.
26. Manuel Rubio-Sánchez, Päivi Kinnunen, Cristóbal Pareja-Flores, and Ángel Velázquez-Iturbide. 2014. Student perception and usage of an automated programming assessment tool. *Computers in Human Behavior* 31: 453–460.
27. Rishabh Singh, Sumit Gulwani, and Armando Solar-Lezama. 2013. Automated feedback generation for introductory programming assignments. *ACM SIGPLAN Notices* 48, 6: 15–26.
28. Josep Soler, Imma Boada, Ferran Prados, Jordi Poch, and Ramon Fabregat. 2010. A web-based e-learning tool for UML class diagrams. In *Education Engineering (EDUCON), 2010 IEEE*, 973–979.
29. Sergey Sosnovsky, Peter Brusilovsky, Michael Yudelson, Antonija Mitrovic, Moffat Mathews, and Amruth Kumar. 2009. Semantic Integration of Adaptive Educational Systems. *Advances in Ubiquitous User Modelling*: 134–158. https://doi.org/10.1007/978-3-642-05039-8_8
30. Thomas Staubitz, Hauke Klement, Jan Renz, Ralf Teusner, and Christoph Meinel. 2015. Towards practical programming exercises and automated assessment in Massive Open Online Courses. In *Teaching, Assessment, and Learning for Engineering (TALE), 2015 IEEE International Conference on*, 23–30.
31. Dave R. Stikkolorum, Truong Ho-Quang, and Michel R V Chaudron. 2015. Revealing Students' UML Class Diagram Modelling Strategies with WebUML and LogViz. In *Proceedings - 41st Euromicro Conference on Software Engineering and Advanced Applications, SEAA 2015*, 275–279. https://doi.org/10.1109/SEAA.2015.77
32. Dave R Stikkolorum, Truong Ho-Quang, Bilal Karasneh, and Michel R V Chaudron. 2015. Uncovering Students' Common Difficulties and Strategies During a Class Diagram Design Process: an Online Experiment. In *Proceedings of the {MODELS} Educators Symposium co-located with the {ACM/IEEE} 18th International Conference on Model Driven Engineering Languages and Systems {(MODELS} 2015), Ottawa, Canada, September 29, 2015.* ({CEUR} Workshop Proceedings), 29–42. Retrieved from http://ceur-ws.org/Vol-1555/4.pdf
33. Michael Striewe and Michael Goedicke. 2011. Automated checks on UML diagrams. In *Proceedings of the 16th annual joint conference on Innovation and technology in computer science education*, 38–42.
34. Michael Striewe and Michael Goedicke. 2014. Automated assessment of UML activity diagrams. In *Proceedings of the 2014 conference on Innovation & technology in computer science education*, 336.
35. Hallvard Trætteberg and Trond Aalberg. 2006. JExercise: a specification-based and test-driven exercise support plugin for Eclipse. In *Proceedings of the 2006 OOPSLA workshop on Eclipse technology eXchange*, 70–74.
36. Boban Vesin, Mirjana Ivanović, Aleksandra Klašnja-Milićević, and Zoran Budimac. 2011. Rule-based reasoning for building learner model in programming tutoring system. In *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 154–163. https://doi.org/10.1007/978-3-642-25813-8_17
37. Boban Vesin, Rodi Jolak, and Michel R V M.R.V. Chaudron. 2017. OctoUML: An Environment for Exploratory and Collaborative Software Design. In *Proceedings of the 39th International Conference on Software Engineering Companion* (ICSE-C '17), 7–10. https://doi.org/10.1109/ICSE-C.2017.19
38. Andrii Vozniuk, Sten Govaerts, and Denis Gillet. 2013. Towards portable learning analytics dashboards. In *Proceedings - 2013 IEEE 13th International Conference on Advanced Learning Technologies, ICALT 2013*, 412–416. https://doi.org/10.1109/ICALT.2013.126
39. Tiantian Wang, Xiaohong Su, Peijun Ma, Yuying Wang, and Kuanquan Wang. 2011. Ability-training-oriented automated assessment in introductory programming course. *Computers & Education* 56, 1: 220–226.
40. Thomas Wanner and Edward Palmer. 2015. Personalising learning: Exploring student and teacher perceptions about flexible learning and assessment in a flipped university course. *Computers & Education* 51, 6: 1.