# A UML approach for designing a VR-based smart learning environment for programming education

Friday Joseph Agbo
*School of Computing*
*University of Eastern Finland*
Joensuu, FInland
fridaya@uef.fi

Solomon Sunday Oyelere
*School of Computing*
*University of Eastern Finland*
Joenssu, Finland
solomon.oyelere@uef.fi

Nacir Bouali
*School of Computing*
*University of Eastern Finland*
Joenssu, Finland
nacirb@uef.fi

*Abstract*— **This study is a work in progress that aims to design and implement a smart learning environment based on virtual reality technology to aid the teaching and learning of programming concepts. The paper followed the approach of designing and modelling of requirement specification for the intended smart platform. This modelling approach is desirable in satisfying activities that engender the design and prototyping of the smart learning environment based on the design science research method. The study discusses the proposed architecture of the system, modelled the system with UML, presents a scenario-based model for teaching and learning of programming concepts, and connect the outcome to the future research.**

***Keywords—Smart learning environment, UML, programming, computing education, virtual reality, VR***

## I. INTRODUCTION

Programming education has become a mainstream subject taught in schools and higher education institutions [1] [2] [3] [4]. Teaching and learning of programming concepts have attracted scholars' interest in the recent past [5]. Many studies try to investigate students' performance in programming courses [6] [7]. These studies show that understanding programming concepts such as functions, loops, and recursion have become difficult for students of computer science [8] [9]. It is even more difficult for novices with little or no programming background because of the abstract nature of these programming concepts [10]. Anyango & Suleman [11] investigated programming topics that lecturers consider most difficult to comprehend by the students. Their study revealed that recursion is the most perceived difficult topic for novices pursuing computer science degree. Efforts to teach these programming concepts for better understanding have been made. For example, Henderson & Romero [12] studied how to teach recursion through a functional programming language. However, these challenges persist, as revealed by Piteira & Costa [13]. Nowadays, the proliferation of new technology presents the opportunity to provide educational tools that can enhance teaching/learning of these programming concepts. To this end, this study is focused on designing a smart learning environment (SLE) to provide better ways for teaching/learning of programming concepts by leveraging the latest technologies such as virtual reality (VR); mobile technology—smartphones, wearables; and ambient intelligence—GPS sensors. Smart learning environment is a technology-enhanced teaching and learning approach [14] [15], introduced as a higher level of the digital environment to support teaching and learning [16]. Although there is no consensus definition of a smart learning environment [17], scholars discussed the characteristic features that make SLE "smart" and what differentiate it from traditional learning environments [18]. These characteristics include adaptivity, context-awareness, ubiquity, personalization, automatic assessment, personalized feedback, motivation, and social-awareness. According to Agbo et al. [19], a smart learning environment is "an enhanced context-aware ubiquitous learning system that leverages social technologies, sensors and wireless communication of mobile devices to engage learners in hands-on experiences and present contents in a stimulating form; capable of connecting the learning community, increasing awareness of the physical environment, tracking and providing learning support". One of the emerging technologies that support the development of SLE is the VR, which creates an environment that seems realistic but not normally experienced [23]. VR environment can provide truly personalized learning by adapting to sensed data from the users' senses and physical environment. Nowadays, smartphones, VR headset (e.g., Gear VR), MOGA pro controller, and many other emerging hardware supports the deployment of VR systems for user's interaction and visualization experience.

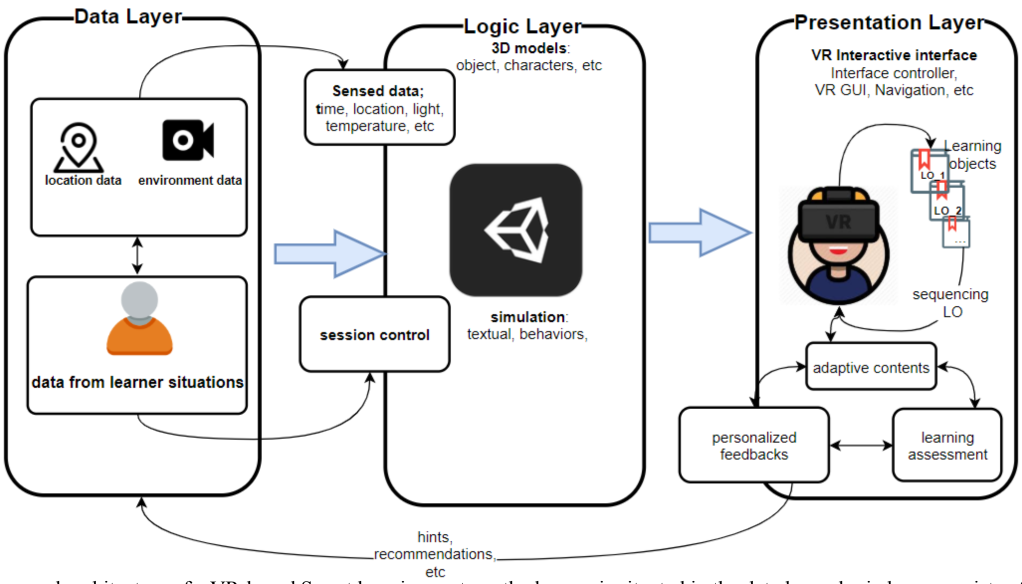
In the bid to develop the SLE prototype to support the teaching of basic programming concepts, this study focuses on eliciting the requirements and modelling the proposed system using the Unified Modelling Language (UML). The requirement definition phase is a key component in the design science research (DSR) method [20]. The DSR method, according to Johannesson & Perjons [20], involves five phases, which include problem explication, requirements definition, prototype design, prototype demonstration, and artifact evaluation. These DSR phases allow for iterative refinement of problems, solutions, and approaches until the desired outcome is achieved. After the requirement specification, being the focus of this study, we intend to conduct a co-creation process of the prototype involving the researcher and expected users.

In requirement definition, scholars propose different notations [21]. One of these notations is natural language, which is too ambiguous, imprecise, and lack visualization that aid the understanding of complex concepts. The UML is one of the formal notations that is commonly used as a standard for modelling requirement specification [22]. UML is a standard object-oriented modelling language which allows for visualising, specifying, and documenting software systems. This paper employs UML to design a scenario-based requirement definition of a smart learning environment for learning programming concepts within a VR environment. Similar studies that use UML modelling tools for engineering software systems exist, for example, Iordan and Panoiu [23].

However, the use of UML for designing a smart learning environment that is based on VR technology needs to be explored; hence, the focus of this paper.
This paper is organized as follows: section two presents the proposed VR-based smart learning environment architecture; section three presents the UML models of the intended system; section four describes a scenario of puzzle game for learning programming concepts in a VR-based SLE; and section five briefly discusses the study contributions, conclusions, and the future research.

## II. Proposed Architecture of Smart Learning Environment

This section introduces the proposed smart learning environment architecture (see Figure 1). In the system architecture, the input data from the learner environment are acquired automatically or specified by the learner at every learning session. These input data, for example, learner's geolocation, temperature, noise level, motion state, etc., are used by the system to support the learner intelligently.



Data Layer
location data
environment data
data from learner situations
Logic Layer
3D models: object, characters, etc
Sensed data; time, location, light, temperature, etc
session control
simulation: textual, behaviors,
Presentation Layer
VR Interactive interface
Interface controller, VR GUI, Navigation, etc
Learning objects
LO_1
LO_2
...
VR
sequencing LO
adaptive contents
personalized feedbacks
learning assessment
hints, recommendations, etc

Fig. 1. The proposed architecture of a VR-based Smart learning system: the learner is situated in the data layer; logic layer consists of tools for modelling the VR learning environment; presentation layer immerses the learner in a VR environment for a personalized learning experience

For instance, providing recommendations on how to learn, when to learn, and what to learn at a given time can be helpful to the learner based on the contextual information the system is able to receive. Besides, input data from the user's environment, his/her prior knowledge, if it exists, is also determined and is used to control the internal behavior of the system to achieve a highly adaptive and personalized learning experience.

The VR-based architecture for the smart learning environment aims to immerse learners in a VR environment in terms of visualization and interactions. VR is a technology that creates an environment that seems realistic but not normally experienced [24]. The application of VR has been studied by researchers in different scenarios [25]. This study draws motivation from Bouali et al. [26] to explore VR for teaching and learning of programming concepts. Thus, the system is designed based on Hwang [27] recommendation for designing a smart learning environment, and a brief explanation follows:

a. The data layer provides different types of input data required to manipulate the system behavior. For example, a first-time learner is required to provide information regarding his/her learning objectives, previous knowledge, other related profile data that the system may need to provide a more personalized learning experience. Additionally, data from learners' environments could come from their devices' sensors to adapt learning to suit learners' contextual needs. Besides, during the learning session, learners will interact with the virtual learning environment through interface controllers and other input devices.

b. The logic layer contains modules that collectively provide a smart learning experience. These modules will include instructions and program logics. Further, a 3D modeling of the game is expected to take place in this layer. Additionally, the use of C# programming language and the Unity game engine will be employed for the scripting of the game. The components, objects, and event-driven scripting that the Unity game engine provides are useful in creating simple games for learning.

c. The presentation layer is a 3-dimensional virtual environment with a head-mounted sensory display. With the smartphone placed in front of a headset, learners can control the behavior of the game object within the virtual environment by using a hand controller/joystick. While learning is taking place, learning objects (LO) automatically adapt to suit the learner based on a set of rules for sequencing LO and outcome of learning assessment carried out during the session. Similarly, sensed data from the device could also influence the behavior of the system in real-time at the presentation layer.

## III. UML Design of Smart Learning Environment for Teaching Programming Concepts

This section presents a UML design of an underway smart learning environment to teach programming concepts (i.e., variables, functions, and recursion). The use of UML to model educational tools is significant in order to achieve the design phase of the DSR, as endorsed by other authors [22] [28].

The use case diagram in Figure 2 shows the functionalities of the system (system requirements) where a learner is able to view and take lessons that are designed to teach basic programming concepts, practice quizzes at different stages of learning, and automatic assessment is done to evaluate learning outcome. Learners receive personalized feedback and recommendations based on the learning outcome. The lessons are game-based, which are modelled to contain LO. The outcome from the learner's assessment controls the sequencing of LO. If a learner is evaluated to perform the below satisfactory score, the SLE automatically recommend LOs that simplify the lesson before advancing to the next higher level LO.

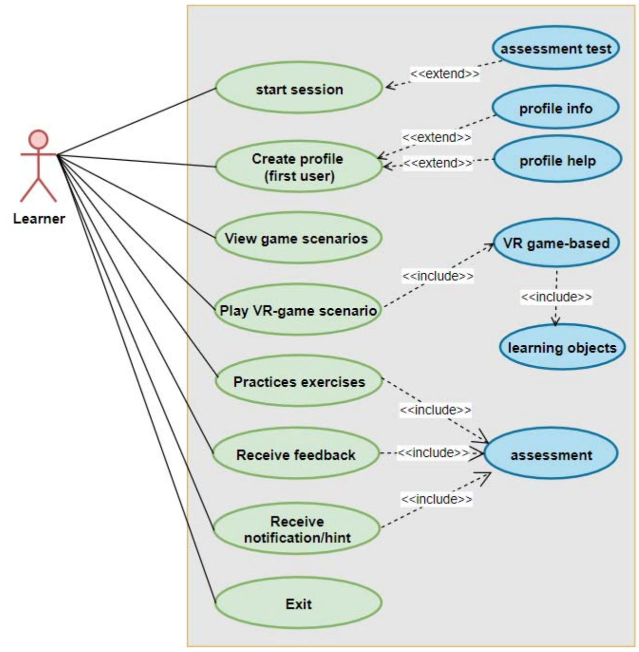


Fig. 2. Use Case Diagram of the smart learning environment: showing the system requirements that are either include a sub-system or extends from a sub-system

Figure 3 is a UML activity model that illustrates how the SLE platform immerses a learner within a VR environment by transforming the learner from the basic concept of programming represented as learning object into a more difficult concept. The diagram shows activities within the system and the flow of control that takes place from one activity to another. For example, a new learner is identified and prompted to provide some profiling data that may be useful to the system to make adaptive and personalized learning possible.

A learner can decide to continuously learn by progressing from one level to another or pause to continue at a later time. The system records the learning progress and evaluates the learner in order to provide personalized feedback, hints, and supports. Similarly, we have presented the sequence diagram of the SLE platform to further show detailed interaction that happens between the actors (users) and the objects (characters/assets). Figure 4 depicts the different sequences of interactions that take place during a particular use case instance. The actor triggers the app after connecting various devices such as Google Cardboard and smartphone. The system initiates a session and responds to the actor with a welcome message and identifies the learner's status, environment, and context. Actors are then redirected to proceed to the virtual learning environment and can control the environment with an input device. As learning progresses at each level, the system evaluates the learner's score. If the score is satisfactory, the system automatically redirects the learner to the next stage of the lesson; otherwise, the system provides hints and personalized support that allows the learner to gain better knowledge.

## IV. A SCENARIO-BASED SMART LEARNING ENVIRONMENT FOR TEACHING PROGRAMMING CONCEPTS

Generally, the intended VR-based SLE is expected to hold a few puzzle games that depict real-world problems. While trying to provide solutions to these problems, the player visualizes programming concepts of function and recursion in an interactive manner with the game element for optimal engagement. In other words, during the gameplay, we try to demonstrate these programming concepts to aide player's understanding. For example, we have presented a scenario in this paper called the river crossing puzzle (a computational thinking concept). Aside from teaching students how to acquire problem-solving skills and algorithmic thinking, the river crossing puzzle has been identified to teach programming concepts such as functions and recursion [29]. River crossing puzzle presents a problem where a farmer has got a number of items to be taken across the river. The available boat is able to carry the farmer and only one item at a time. Some combination of these items cannot mutually co-habit in the absence of the farmer. Hence, the farmer ensures that all the items cross the river unharmed.

To solve this problem, the farmer would need to create and call some functions within the gameplay environment and pass parameters to them. For example, function to get an object within the game and place it inside the boat; function to move the object across the river; etcetera will be instantiated and called during the game session. Similarly, some repetition will occur that may require a function to recall itself (recursion). All of these processes happening in the gameplay will demonstrate the practicality of function and recursion and explanation of each event within the gameplay be given to the learner by a game agent.

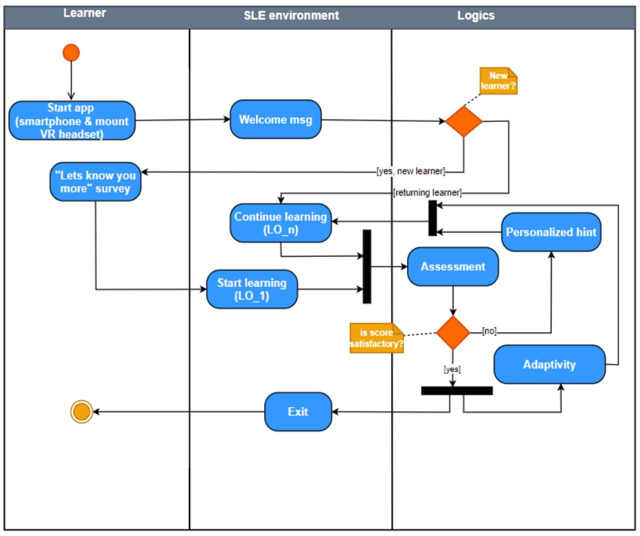


Fig. 3. Activity diagram of the smart learning environment: presenting interactions between each activity; the arrow shows the flow of communication from start to exit points

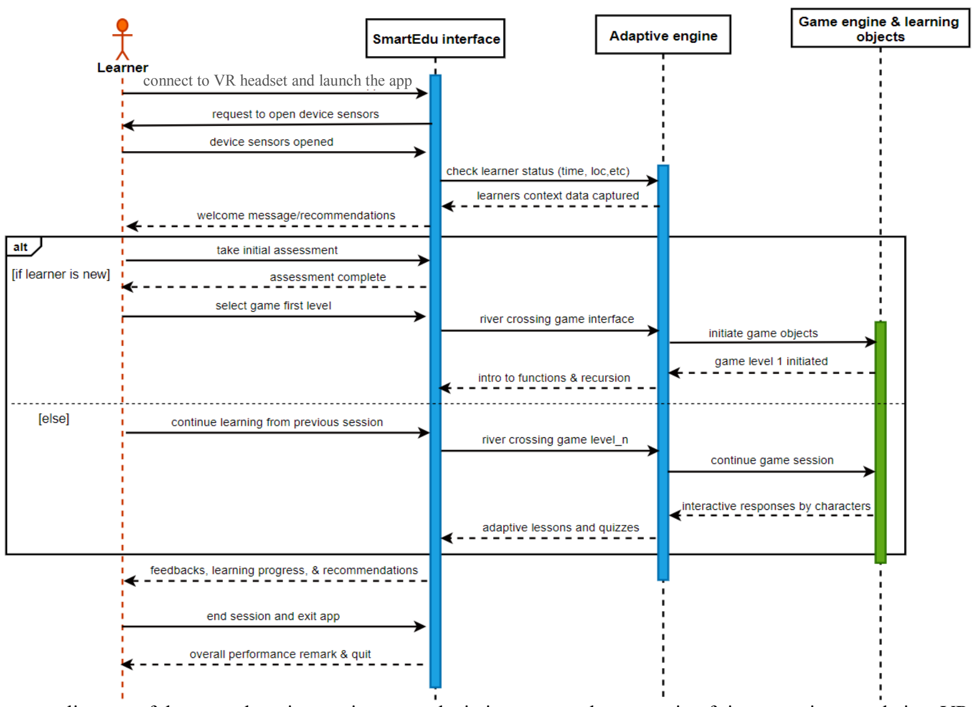

Learner
SmartEdu interface
Adaptive engine
Game engine & learning objects
connect to VR headset and launch the app
request to open device sensors
device sensors opened
check learner status (time, loc,etc)
learners context data captured
welcome message/recommendations
alt
[if learner is new]
take initial assessment
assessment complete
select game first level
river crossing game interface
initiate game objects
game level 1 initiated
intro to functions & recursion
[else]
continue learning from previous session
river crossing game level_n
continue game session
interactive responses by characters
adaptive lessons and quizzes
feedbacks, learning progress, & recommendations
end session and exit app
overall performance remark & quit

Fig. 4. Sequence diagram of the smart learning environment depicting a gameplay scenario of river crossing puzzle in a VR environment

TABLE I. CONCEPTUAL DESCRIPTION OF THE PUZZLE; GAME LEVELS AND BEHAVIOURS

**Stepwise description of the river crossing puzzle**
Step 1: the player selects a character (farmer) and at least three items as explained in each level of the game below.
Step 2: The player instantiates functions to collect each item into the boat from the departure terminal and dropoff at the destination terminal.
Step 3: The player calls the function to move each item across the river.
Step 4: move function is recursively called after a conditional statement is evaluated

**Two levels description of the puzzle game**
Our conceptual design of the game will be modelled using the 3D Unity game engine, where assets and objects are classified into three categories:
• first category is a universal set of animals containing subsets {goat, wolf}, {dog, cat}, & {lion, monkey}
• second category is a set of farm products {banana, watermelon}
• third category contains sets of farmers (river-crossers)

**Level 1 rules:** In no order, one grouped asset is picked from the first category, one asset is picked from the second, and one river-crosser is picked from the third category, respectively. Alternatively, the arrangement of the assets can be pre-defined within the game so that players do not need to select them. These options can be agreed upon during the co-design process of the game with prospective users.

**Gameplay for level 1:**
The river-crosser takes selected objects across the river with the function MOVE, which can be recursively called based on the following conditions.
i. if objects (animals) cannot co-habit on their own without causing harm to each other
ii. if an object (animal or farm product) remains to be moved across the river
The task is to move all the objects across the river with a limited number of calling the function MOVE.
After completing the level 1 task, the game rewards, give feedback/recommendations, and initiates the next level.

**Level 2 rules:** three objects (animals), two objects (farm produce), and one river-crosser are selected.

**Gameplay for level 2:**
Rules of the gameplay in level 1 apply. However, the number of objects is increased; therefore, the game becomes harder, and the task remains to limit as much as possible, the number of the recursive call to the function MOVE in order to complete the task with the best score

## V. CONCLUSION AND FUTURE WORK

The goal of this ongoing study is to design a game prototype of a smart learning environment based on the proposed architecture. Currently, the study elicits requirements for the intended SLE, which are modelled using the UML. However, future work will involve a co-design process of the prototype with prospective users. After completion, evaluation of the prototype to find out the usefulness will be next. Despite the significant role that the smart learning environment plays in transforming teaching and learning at a higher education institution, not too many studies regarding the design of a smart learning environment with UML have been explored [27]. This paper tries to fill the gap by using the UML to design a game-based educational tool. The paper proposes the architecture of a VR-based SLE for teaching and learning of programming concepts and further designed the requirement for the system. A scenario of a puzzle game (River Closing) that teaches the concept of functions and recursion was also presented. This study contributes to the literature by presenting formal notations (UML) for designing a SLE, which allows for the representation that aids the understanding of researchers and other experts in this field. Besides, it also provides a guide for our ongoing research towards the implementation of a SLE platform for programming education, which is underway. Programming remains a challenging topic to teach in games, given the abstract nature of the topics it encompasses. We, however, believe that the visualization quality that VR technology provides will greatly help in communicating these concepts to learners. Via the game puzzle, we do not teach recursion and functions as abstract concepts, but we demonstrate and illustrate how these two pillars of programming are used to solve a practical (although imaginative) problem. We deviate from traditional learning games, where learners are usually asked questions throughout their gameplay to play further levels. Our approach relies on exploiting the qualities inherent to functions and recursion to play the game.

## ACKNOWLEDGMENT

Work by the author, Solomon Sunday Oyelere was supported by the Jenny and Antti Wihuri Foundation.

## REFERENCES

[1] S. S. Oyelere, J. Suhonen, G. M. Wajiga, and E. Sutinen. "Design, development, and evaluation of a mobile learning application for computing education," Education and Information Technologies, vol. 23, no. 1, pp. 467-495, 2018.

[2] C. Yongqiang, W. Xiaojun and Q. Chengbin, "Computer Programming Education for Primary School Students," In 2018 13th International Conference on Computer Science & Education (ICCSE), 2018.

[3] M. Ebert, "Increase active learning in programming courses," In 2017 IEEE Global Engineering Education Conference (EDUCON), 2017.

[4] S. S. Oyelere, F. J. Agbo, I. T. Sanusi, A. A. Yunusa and K. Sunday, "Impact of Puzzle-Based Learning Technique for Programming Education in Nigeria Context," In 2019 IEEE 19th International Conference on Advanced Learning Technologies (ICALT), 2019.

[5] W. Cazzola and D. M. Olivares, "Gradually learning programming supported by a growable programming language," IEEE Transactions on Emerging Topics in Computing, vol. 4, no. 3, pp. 404-415, 2015.

[6] O. O. Ortiz, J. A. P. Franco, P. M. A. Garau and R. H. Martin, "Innovative mobile robot method: improving the learning of programming languages in engineering degrees," IEEE Transactions on Education, vol. 60, no. 2, pp. 143-148, 2016.

[7] K. Sunday, P. Ocheja, S. Hussain, S. S. Oyelere, B. O. Samson and F. J. Agbo, "Analyzing Student Performance in Programming Education Using Classification Techniques," International Journal of Emerging Technologies in Learning (iJET), vol. 15, no. 2, pp. 127-144, 2020.

[8] A. Pears, S. Seidman, L. Malmi, L. Mannila, E. Adams, J. Bennedsen, M. Devlin and J. Paterson, "A survey of literature on the teaching of introductory programming," SIGCSE Bulletin, vol. 39, no. 4, pp. 204-223, 2007.

[9] B. Law, " Introducing novice programmers to functions and recursion using computer games," In Academic Conferences and Publishing International Limited, ECGBL 2018 12th European Conference on Game-Based Learning, 2018, pp. 325-334.

[10] S. S. Oyelere, J. Suhonen and, T.H. Laine, "Integrating parson's programming puzzles into a game-based mobile learning application," In Proceedings of the 17th Koli Calling International Conference on Computing Education Research, pp. 158-162, 2017.

[11] J.T. Anyango, and H. Suleman, "Teaching Programming in Kenya and South Africa: What is difficult and is it universal?," In Proceedings of the 18th Koli Calling International Conference on Computing Education Research, pp. 1-2. 2018.

[12] P. B. Henderson and F. J. Romero, "eaching recursion as a problem-solving tool using standard ML," ACM SIGCSE Bulletin, vol. 21, no. 1, pp. 27-31, 1989.

[13] M. Piteira and C. Costa, "Learning computer programming: study of difficulties in learning programming," In Proceedings of the 2013 International Conference on Information Systems and Design of Communication, 2013.

[14] L. D., H. R. and W. M., "Contexts of Smart Learning Environments," In Smart Learning in Smart Cities, Singapore, Springer, 2017, pp. 15-29.

[15] H. Peng, S. Ma and J. M. Spector, "Personalized adaptive learning: an emerging pedagogical approach enabled by a smart learning environment," Smart Learning Environments, vol. 6, no. 1, pp. 9, 2019.

[16] R. Huang, J. Yang and Y. Hu, "From digital to smart: The evolution and trends of learning environment," Open Education Research, vol. 1, no. 1, pp. 75-84, 2012.

[17] F. J. Agbo, S. S. Oyelere, J. Suhonen and M. Tukiainen, "Smart learning environment for computing education: readiness for implementation in Nigeria," In EdMedia+ Innovate Learning, Amsterdam, 2019.

[18] J. M. Spector, "Conceptualizing the emerging field of smart learning environments," Smart learning environments, vol. 1, no. 1, p. 2, 2014.

[19] F. J. Agbo, S. S. Oyelere, J. Suhonen and T. M., "Identifying potential design features of a smart learning environment for programming education in Nigeria," International Journal of Learning Technology, vol. 14, no. 4, pp. 331-354, 2019.

[20] P. Johannesson and E. Perjons, A design science primer, CreateSpace Independent Publishing Platform, 2012.

[21] D. Silingas and B. Rimantas, "UML-intensive framework for modeling software requirements," In Proceedings of the 14th International Conference on Information and Software Technologies, 2008.

[22] K. Wiegers and J. Beatty, Software requirements, Pearson Education, 2013.

[23] A. Iordan and M. Panoiu, "Design using UML diagrams of an educational informatics system for the study of computational geometry elements," WSEAS Transactions on Computers, vol. 9, no. 9, pp. 960-970, 2010.

[24] J. N. Latta and D. J. Oberg, "A conceptual virtual reality model," IEEE Computer Graphics and Applications, vol. 14, no. 1, pp. 23-29, 1994.

[25] J. Radianti, T. A. Majchrzak, J. Fromm and I. Wohlgenannt, "A systematic review of immersive virtual reality applications for higher education: Design elements, lessons learned, and research agenda," Computers & Education, vol. 147, no. 103778, 2020.

[26] N. Bouali, E, Nygren, S. S. Oyelere, J. Suhonen and V. Cavalli-Sforza, "Imikode: A VR Game to Introduce OOP Concepts," In Proceedings of the 19th Koli Calling International Conference on Computing Education Research, pp. 1-2. 2019.

[27] G.-J. Hwang, "Definition, framework and research issues of smart learning environments-a context-aware ubiquitous learning perspective," Smart Learning Environments, vol. 1, no. 1, p. 8, 2014.

[28] R. P. De Lope and N. Medina-Medina, "Using UML to Model Educational Games," in 8th International Conference on Games and Virtual Worlds for Serious Applications (VS-GAMES), 2016.

[29] F. J. Agbo, S. S. Oyelere, J. Suhonen and S. Adewumi, "A Systematic Review of Computational Thinking Approach for Programming Education in Higher Education Institutions," In Proceedings of the 19th Koli Calling International Conference on Computing Education Research, Koli, 2019.