Nor Farahwahida Mohd Noor
B.Eng (Computer and Information)
Department of Computing, Sultan Idris Education University, Tanjung Malim, Perak, Malaysia
norfarahwahida@gmail.com

Aslina Saad
PhD (Computer Science), Associate Professor, Lecturer
Department of Computing, Sultan Idris Education University, Tanjung Malim, Perak, Malaysia
Scopus ID: 37162127300
aslina@fskik.upsi.edu.my

Abu Bakar Ibrahim
PhD (Electronics), Associate Professor, Lecturer
Department of Computing, Sultan Idris Education University, Tanjung Malim, Perak, Malaysia
Scopus ID: 57189494415
abubakar.ibrahim@fskik.upsi.edu.my

Norashady Mohd Noor
PhD (Educational Measurement), Lecturer
Mechanical Engineering Department, Sultan Azlan Shah Polytechnic Institute, Behrang, Perak, Malaysia
norashady@gmail.com

THE ACCEPTANCE OF AN EDUCATIONAL INTEGRATED DEVELOPMENT ENVIRONMENT TO LEARN PROGRAMMING FUNDAMENTALS

Abstract. Programming is an important course for any IT or engineering-related course. However, previous research shows that students face difficulties in learning programming due to its abstract concepts. This study aims to evaluate the acceptance of a developed Integrated Development Environment (IDE), namely C-SOLVIS which is a web-based application that specifically intends to facilitate the teaching and learning of the C programming fundamentals in Malaysian tertiary education. The C-SOLVIS integrates problem-solving into a program development environment for the C language. The goal is to guide the users in problem-solving and help them write C programs based on problem-solving algorithms. The Rapid Application Development (RAD) Model was employed in the C-SOLVIS development process. Based on this model, the requirement planning phase was carried out through the triangulation technique by applying qualitative approaches comprising a literature review supported by semi-structured interviews, document reviews, and content validation by expert programming lecturers. Subsequently, the design of the application was accomplished through the iterative prototyping process which was then followed by the application construction. Then, the C-SOLVIS is deployed to be used by several programming lecturers to evaluate its usability by adopting a quantitative method using the System Usability Scale (SUS) questionnaire. The study has discovered several suitable techniques and designs for the problem-solving and program development environment. For the problem-solving environment, the Computational Thinking (CT) concepts were applied which were supported by the Input-Process-Output (IPO) Model through Scientific Instructions and Inquiries. Meanwhile, the program development environment was designed and developed based on frame-based programming using a set of developed Code Patterns. The C-SOLVIS evaluation using the SUS instrument has yielded a SUS mean score of 86.07. This score is interpreted by SUS as an A grade that indicates C-SOLVIS as a highly usable application and thus is accepted for C programming learning. Hence, the development process of the C-SOLVIS can be used as a guideline for educational software development, especially in the field of programming education.

Keywords: IDE; educational software; usability; programming.

1. INTRODUCTION

The Industrial Revolution 4.0 (IR4.0) has caused rapid development in computing and software technology. Consequently, the demand for expert programmers with high programming

© N.F.Mohd Noor, A. Saad, A.B.Ibrahim, N.Mohd Noor, 2023.

skills has increased to develop various software and solve various computing problems. Hence, there is an urge to produce computer science and engineering graduates with high programming and problem-solving skills. Moreover, to develop high programming skills, graduates need competency in both problem-solving and program writing [1], [2]. Therefore, these skills need to be inculcated among students in their programming courses at the tertiary level.

However, due to its high intrinsic cognitive load, programming has been a challenging lesson to learn among beginners [1]. Novice programmers often face difficulties in problem-solving due to their lack of programming experience [3]. They also face technical and conceptual challenges in writing a program that was reflected in their disfluency in developing a program syntactically, semantically, and pragmatically [4]. Therefore, to enhance the problem-solving and programming skills of novices, suitable educational programming software should be used.

To date, several applications with various approaches have been discovered to address these issues. However, most are targeting programming languages other than the C language [5]. Therefore, programming software i.e., IDE is often used in teaching and learning activities for C programming. Yet, the IDE is overwhelmed with complex functions which are believed to be intimidating for novices [6]. It also does not have facilities to help with problem-solving and programming learning.

Therefore, this research was carried out to verify the needs for an educational C programming application that could introduce novices to an IDE, at the same time guide them in problem-solving and programming. For that purpose, C-SOLVIS is designed as a simple introductory IDE to guide problem-solving and facilitate program development using computational thinking (CT) and frame-based programming. Then, this paper reported the acceptance of this introductory IDE, namely C-SOLVIS, as a tool to enhance the teaching and learning of C programming.

2. RESEARCH METHODS

The research is carried out in two major stages as shown in Figure 1.

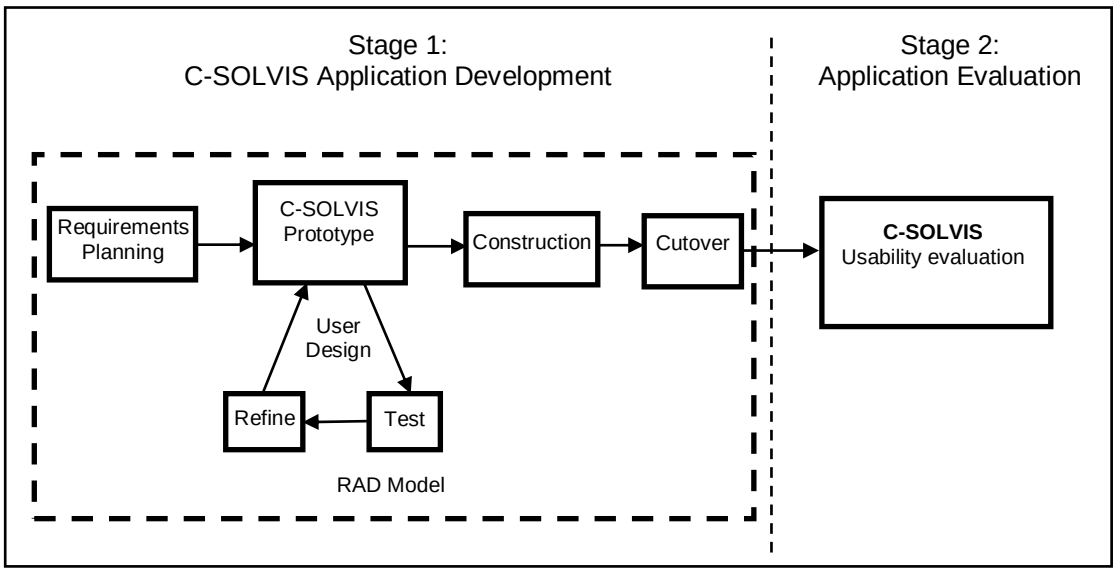

Figure 1. Two major stages of the research.

Table 1 summarizes the research design concerning its objectives and related methodologies that are implemented in these stages.

Table 1

Research Methodologies

Stage	Phase	Research Objective	Activity	Methods/ Techniques	Tools/ Models	Deliverable
Stage I: Software Development	Requirements Planning	To identify suitable techniques to overcome students' difficulties in learning programming.	To identify Functional Requirements Non-functional Requirements	Qualitative methods: Requirements Elicitation with a triangulation technique using: Literature review, Semi-structured interview, Document review, and Content validation	Tools: UMLet Models: Use-case diagrams Activity Diagram	Software Requirement Specification (SRS)
	User Design	To design an introductory IDE that integrates guided problem-solving and facilitating program development.	To design the architecture, component, data, and user interface of the application.	Prototyping Testing Refining	Tools: Draw.io Marvel Models: Architecture design Component design Sequence Diagram Data design User interface design	Software Design Document (SDD)
	Construction & Cutover	To develop an introductory IDE that integrates guided problem-solving and facilitating program development.	To construct the application and test for both front-end and back-end.	Coding System Integration Application testing	Tools: WebStorm 2021.1 IDE React JS framework Node.js server JavaScript programming language SQLite	C-SOLVIS application prototype Software Test Document (STD)
Stage II: Software Evaluation	Usability Evaluation	To evaluate the usability of the introductory IDE to be used in introductory programming courses.	To evaluate the usability of the application among programming lecturers.	Quantitative method: Software evaluation using the SUS questionnaire	ISO 9241-11: usability standard System Usability Scale (SUS) questionnaire SPSS	Usability result

Based on this table, the first stage is the application development stage which is employing the RAD Model. The RAD is one of the software process models that is widely used as an approach to developing software. The main reason for adopting the RAD model is

based on its ability to reduce the application development period, as we can get quick initial reviews from the stakeholders [7]. In brief, by adopting the RAD model, the application is developed through four development phases: Requirements Planning, User Design, Construction, and Cutover phase, as shown in Figure 2.

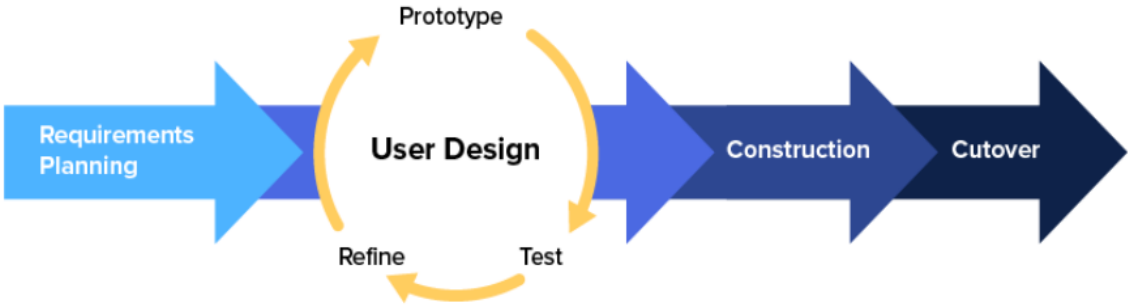

Figure 2. RAD Model

During the Requirements Planning phase, several qualitative approaches were employed in a triangulation technique to determine the functional and non-functional requirements of the application. This technique involves literature reviews supported by semi-structured interviews, document review and content validation. The implementation of these methods complements each other to support the multidimensionality of user requirements investigations [8]. Moreover, these methods could help to get a detailed description and intensive analysis of the related issues in this research [9], [10]. Therefore, the triangulation strategy helps in developing high-quality requirements for the application.

The Requirements Planning phase involves the participation of experts in programming education especially in the semi-structured interview sessions, document review and content validity process. These expert programming lecturers are chosen among Malaysian Polytechnic lecturers and other Malaysian higher learning institutions through purposive sampling. These respondents are also involved in the C-SOLVIS usability evaluation.

The participants are identified as experts for having a minimum of 10 years of teaching experience and possessing qualifications in the subject area [11]. With these years of experience, they are considered experts for being involved in the development of teaching materials, instruction, assessment, evaluation, intervention, and addressing questions from learners [12]. Moreover, some of them are also having experience in coordinating the course. Based on their teaching experience, the participants have experience and skills in using different types of IDEs and applications in teaching. They are also selected based on their experience in dealing with various types of students learning capabilities.

After the first stage is done, this study moves to the second stage which is the application evaluation stage where the usability of the C-SOLVIS is evaluated by seven samples of target users among the programming lecturers. Therefore, the evaluation involves the same seven expert programming lecturers who were involved in the Requirement Elicitation phase. They are chosen so that the usability evaluation can be done based on the subject matter purpose which is problem-solving and programming learning. Moreover, the involvement of the same participants could improve the accuracy of the study and the reliability of the data and results [13]. In addition, their various learning institutions' backgrounds also could avoid bias and broaden the evaluation perspectives to achieve a convincing result.

The usability evaluation is done by employing a quantitative method by using an existing established and reliable questionnaire for measuring software usability which is the

SUS [14]. The analysis is done to determine the mean SUS score. SUS instrument was used as it is an easily used instrument with uncomplicated calculation. SUS items can be referred to in Table 2. It also has been proven valid and reliable even though implemented for small sample size [15]. This is supported by the study of [14] who reported that SUS has been used for as low as three respondents. Moreover, SUS is also a robust instrument that has been widely used in evaluating usability [16].

Table 2

Items in the SUS Questionnaire

	Items
Q1	I think I would like to use this application frequently
Q2	I found the application unnecessarily complex
Q3	I thought the application was easy to use
Q4	I think I would need the support of a technical person to be able to use this application
Q5	I found that the various functions in this application were well-integrated
Q6	I thought there was too much inconsistency in this application
Q7	I would imagine that most people would learn to use this application very quickly
Q8	I found the application very cumbersome to use
Q9	I felt very confident using the application
Q10	I needed to learn many things before I could work with this application

The questionnaire items are organized alternately between positive and negative statements to prevent habitual bias from the respondent [17]. The oddly numbered items are positive statements, while the evenly numbered items are negative statements. These items measure the usability of an application that covers the aspects of effectiveness, efficiency, and satisfaction [18].

Since the SUS items comprise negative and positive statements, the data needs to be normalized before being analysed. To normalize the data of all odd-numbered items (Q1, Q3, Q5, Q7 and Q9), the scale position is subtracted by 1. Meanwhile, to normalize the data of all even-numbered items (Q2, Q4, Q6, Q8, and Q10), 5 is subtracted by the scale position on these items. Then, the sum of all items is multiplied by 2.5 to get an overall SUS score ranging between 0-100. In [19], a calculation formula has been summarized for an individual overall SUS score as shown in equation 1 below.

$$\text{SUS score} = [(Q1 - 1) + (5 - Q2) + (Q3 - 1) + (5 - Q4) + (Q5 - 1) + (5 - Q6) + (Q7 - 1) + (5 - Q8) + (Q9 - 1) + (5 - Q10)] * 2.5 \quad (1)$$

After the total SUS score for each respondent is calculated, the usability of the C-SOLVIS is then determined by the mean value, which is the average of the total SUS scores. The mean score is the summation of all SUS scores obtained from all respondents divided by the number of respondents. A formula was derived in [15] for the SUS mean score calculation as stated in equation 2 below.

$$\text{SUS mean score} = \frac{\sum \text{SUS score}}{N} \quad (2)$$

where N is the total number of respondents.

The SUS scores can be categorized into specific acceptability grades rating according to SUS score ranges as shown in Table 3 below.

The application should achieve a mean score of 68 and above to be considered to have good usability [17]. However, it was stated in [15] that the SUS score should be higher than 70 to be recognized as acceptable. Therefore, an application should score a SUS mean score higher than 70 to be within a comfortable acceptable range.

Table 3

SUS Ranges, Grade and Ratings [15]

SUS Score Ranges	Grade Scale	Adjective Ratings	Acceptability ratings
80.3 <= score <=100	A	Best imaginable	Acceptable
74.0 <= score < 80.3	B	Excellent	
68.0 <= score < 74.0	C	Good	
51.0 <= score < 68.0	D	Ok	Not acceptable
Score < 51	F		

3. THE RESULTS

This section reports the educational IDE development processes and the result of the usability evaluation.

3.1. C-SOLVIS Design and Development

After an intensive Requirement Planning phase, the C-SOLVIS was successfully designed to perform these tasks as depicted in the use-case diagram shown in Figure 3 below.

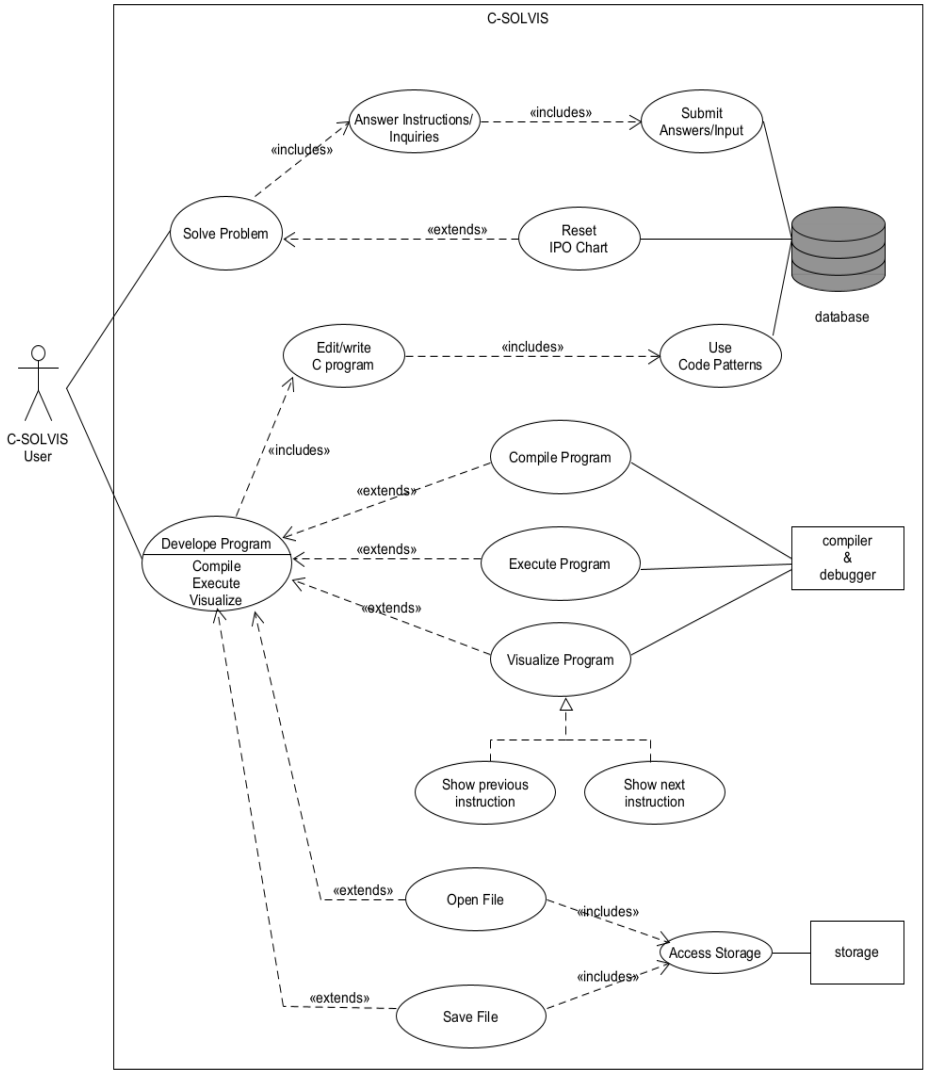

Figure 3. C-SOLVIS Use-Case Diagram

The specific workflows and activities within each use-case could be illustrated using the activity diagram as shown in Figure 4.

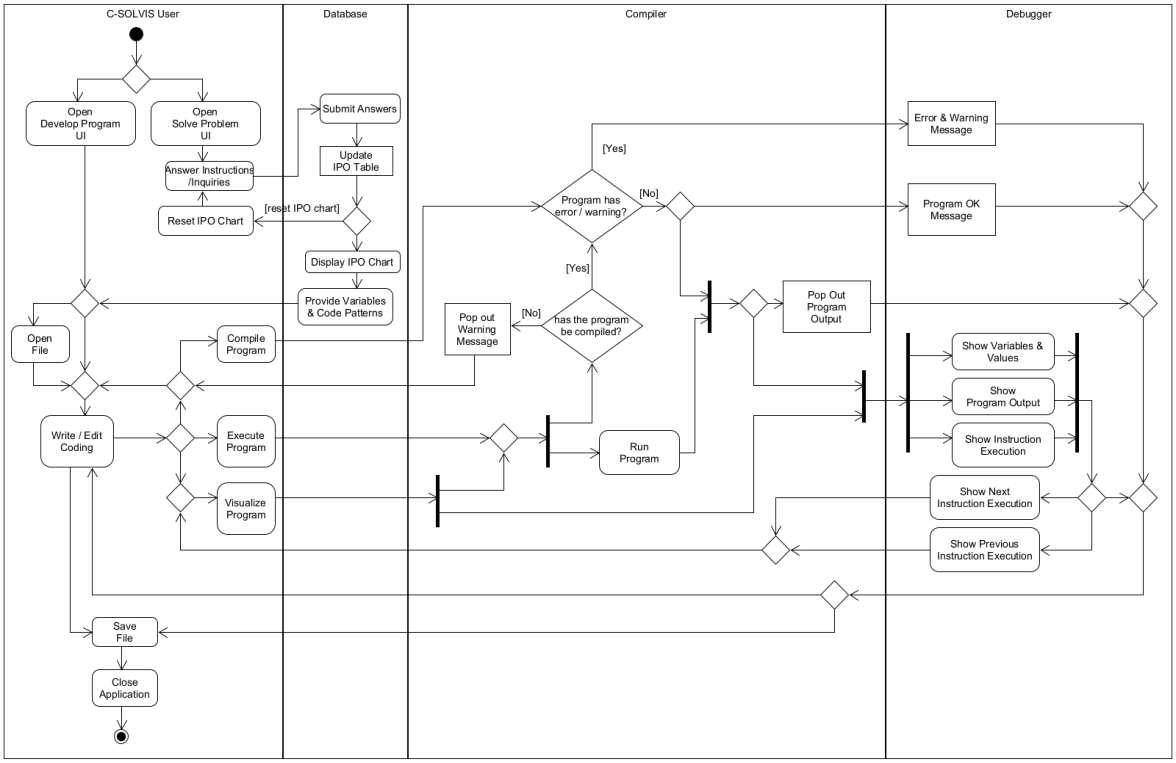

Figure 4. C-SOLVIS Activity Diagram

Then, the design of this application is drafted which is dependent on the application requirements produced in the Requirement Planning phase. The design process begins with designing the architecture of the application. The architectural design identifies all properties or subsystems making up the C-SOLVIS application which includes all hardware and software components with their interfacing elements. It represents all the interrelationships among them, as well as the data structure required to build this application as illustrated in Figure 5.

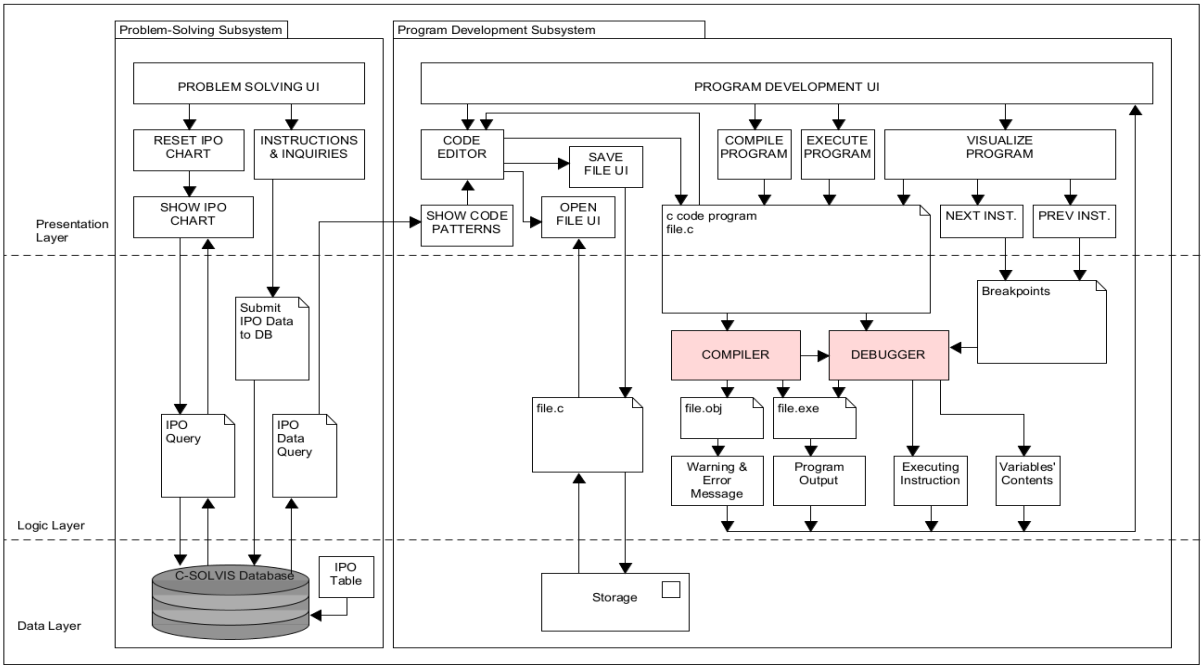

Figure 5. C-SOLVIS Architectural Diagram

Whilst the architectural design identified the main components of the application, a component design specifically composes these components to implement a certain subsystem. Each component is designed to satisfy relevant aspects of the application requirements and all design structures of the application architecture. A component interacts with other components to perform a certain set of functionalities with an explicit adaptation specification. The component design of the C-SOLVIS application is shown in Figure 6.

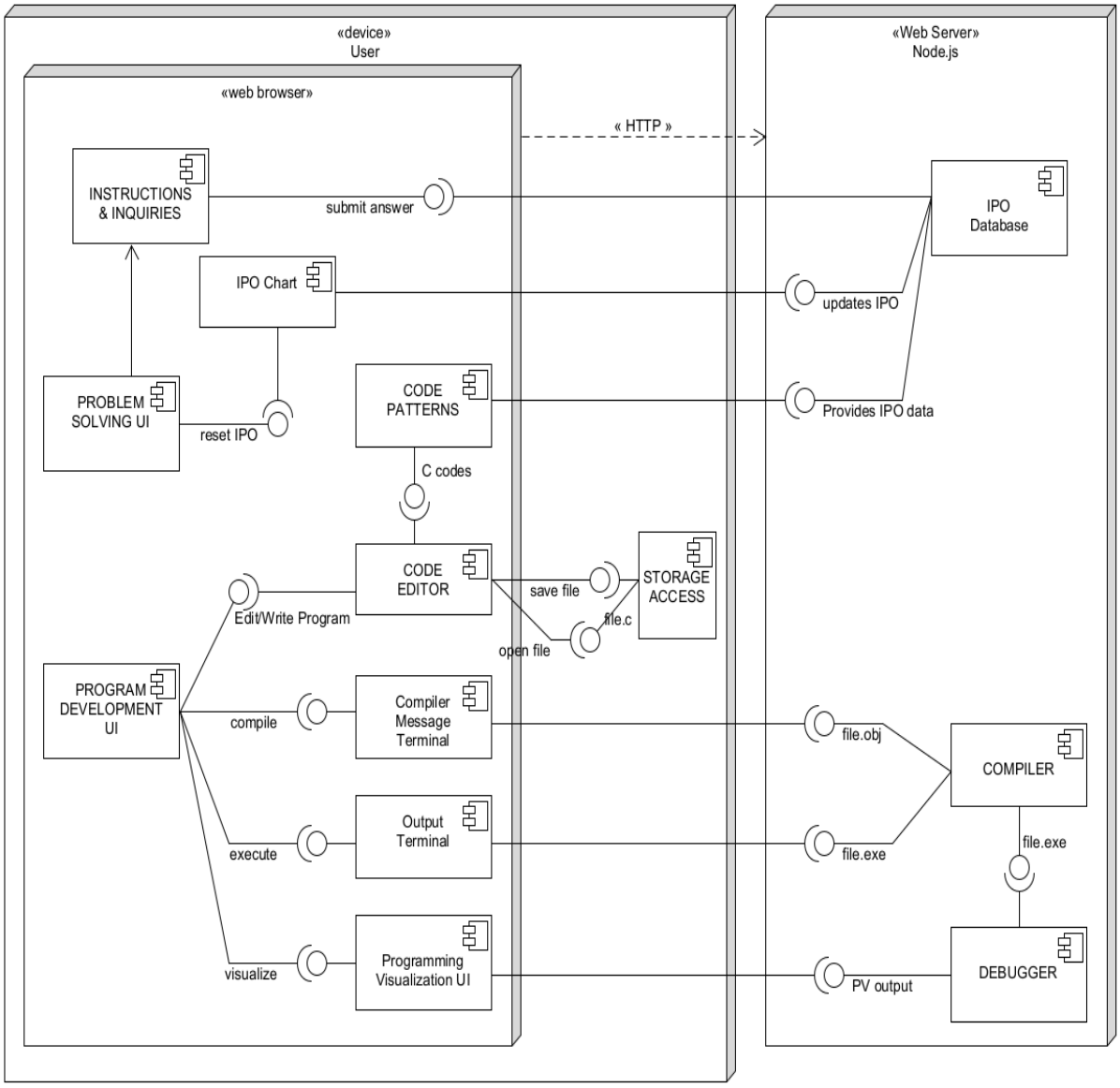

Figure 6. C-SOLVIS Component Design

Then, the study proceeds to the user interface design of the C-SOLVIS which follows the Semantic User Interface Guideline (SUIG) that focuses on the usability aspect to produce an application with simple, familiar, and consistent user interface (UI) to help the user to navigate intuitively throughout the application as shown in Figure 7.

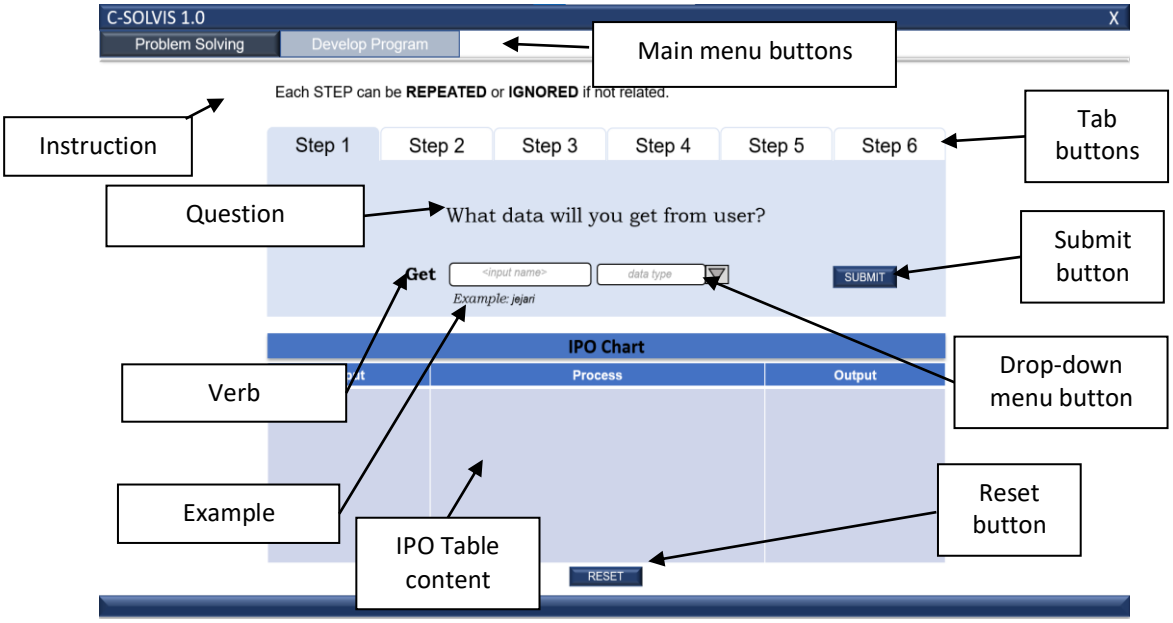

Figure 7. General Setting of the Problem-Solving UI

Then, after the design process of the C-SOLVIS is completed, the C-SOLVIS is constructed. The construction is divided into two parts: frontend development and backend development as shown in Figure 8. The construction is completed after the C-SOLVIS has successfully undergone several series of software testing which are unit testing, integration testing and system testing. Then, the application is deployed to be used for evaluation among the chosen respondents.

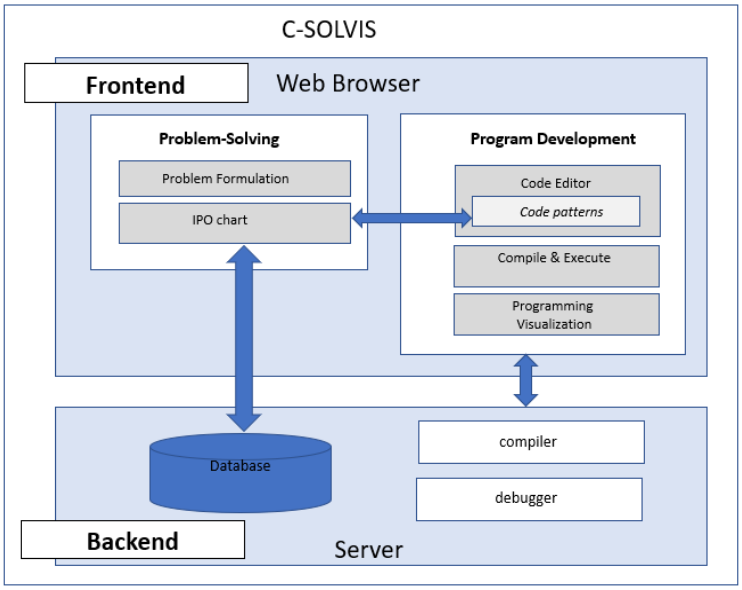

Figure 8. The C-SOLVIS Construction

3.2. C-SOLVIS Usability Evaluation

The usability evaluation of the C-SOLVIS was done among seven respondents denoted as P1-P7. The respondents' responses for each SUS item are recorded as Q1 – Q10 as shown in Table 4. The scale score is between 1 to 5.

Table 4

Original Data from SUS Scores

Participants	Original SUS Score									
	Q1	Q2	Q3	Q4	Q5	Q6	Q7	Q8	Q9	Q10
P1	4	2	4	2	4	2	4	2	4	2
P2	4	2	4	2	3	2	4	2	4	2
P3	4	2	4	2	4	2	4	2	4	2
P4	5	1	5	1	5	1	5	1	5	1
P5	5	1	4	2	4	2	5	2	4	1
P6	5	1	5	1	4	2	5	1	5	1
P7	5	1	5	1	5	1	5	1	5	1

Since the SUS items comprise negative and positive items, the SUS original scores need to be normalized as shown in Table 5. The original scores in Q1, Q3, Q5, Q7 and Q9 items are transformed to T1, T3, T5, T7 and T9 by subtracting 1 from the original scores. Meanwhile, the original scores in Q2, Q4, Q6, Q8 and Q10 are transformed to T2, T4, T6, T8 and T10 by subtracting the original scores from 5. The transformed score values and calculated SUS scores are recorded in Table 5.

Table 5

Calculated SUS Score

Participants	Normalized SUS Score										
	T1	T2	T3	T4	T5	T6	T7	T8	T9	T10	Total SUS Score (x 2.5)
P1	3	3	3	3	3	3	3	3	3	3	75.0
P2	3	3	3	3	2	3	3	3	3	3	72.5
P3	3	3	3	3	2	3	3	3	3	3	75.0
P4	4	4	4	4	4	4	4	4	4	4	100.0
P5	4	4	3	3	3	3	4	3	3	4	85.0
P6	4	4	4	4	3	3	4	4	4	4	95.0
P7	4	4	4	4	4	4	4	4	4	4	100.0

Based on this table, the normalization process produces a scale score between 0 to 4, with a total score value ranging from 0 to 40. Then, the Total SUS Score for each participant is calculated to produce a percentile ranking that ranges from 0 to 100. By referring to this table, the total SUS scores obtained from the participants ranged from a minimum score of 72.5 to a maximum score of 100. The Total SUS Score was calculated by using this formula in equation 3 below.

$$\text{Total SUS score} = (T1 + T2 + T3 + T4 + T5 + T6 + T7 + T8 + T9 + T10) * 2.5 \quad (3)$$

Then, the data is further analysed by determining its reliability. The reliability statistics are examined as shown in Table 6.

Table 6

Reliability Statistics

Cronbach's Alpha	Cronbach's Alpha Based on Standardized Items	Number of Items
0.972	0.974	10

Based on this table, the usability analysis using the SUS instrument has shown high reliability which is indicated by Cronbach's alpha of 0.972. The high value of Cronbach's Alpha shows that all items are internally consistent and reliable. This is supported by the SUS item statistics shown in Table 7.

Table 7

SUS Item Statistics

SUS Item	Scale Mean if Item Deleted	Scale Variance if Item Deleted	Corrected Item-Total Correlation	Cronbach's Alpha if Item Deleted
Q1	30.86	19.476	.888	.968
Q2	30.86	19.476	.888	.968
Q3	31.00	19.333	.922	.967
Q4	31.00	19.333	.922	.967
Q5	31.29	18.905	.762	.975
Q6	31.14	20.476	.733	.973
Q7	30.86	19.476	.888	.968
Q8	31.00	19.333	.922	.967
Q9	31.00	19.333	.922	.967
Q10	30.86	19.476	.888	.968

This table shows that Cronbach's alpha value of 0.972 which was obtained previously decreases if the item is deleted except for item Q5 and item Q6. The Cronbach's alpha value increases to 0.975 and 0.973 if Q5 and Q6 are deleted respectively. However, the slight increase of Cronbach's alpha value is not significant as they are still in the same range of excellent reliability indication. This means that all items are acceptable in the instrument for data analysis as shown in Table 8.

Table 8

SUS Data Analysis

Number of participants	Min	Max	Mean	Standard Deviation
7	72.5	100	86.0714	12.23529

Based on this table, C-SOLVIS has obtained a minimum score of 72.5, while the maximum score is 100. The usability evaluation via the SUS instrument has yielded a mean SUS value of 86.0714 which is above the minimum acceptable usability score. This mean value indicates that the C-SOLVIS prototype is the "best imaginable" application according to the SUS score rating. Therefore, the C-SOLVIS application is acceptable and considered to have a good level of usability.

Then, the SUS scores for all respondents are used for further analysis to find out response frequencies. The frequencies of the responses are figured out based on the score ranges and grades as shown in Table 9.

Table 9

SUS Grade Frequency

Grade	Score Range	Adjective rating	Frequency	Percentage
A	80.3 <= score <=100	Best imaginable	4	57.1
B	74.0 <= score < 80.3	Excellent	2	28.6
C	68.0 <= score < 74.0	Good	1	14.3
		Total	7	100.0

From the analysis shown in this table, 57.1% of the participants rated the C-SOLVIS application in Grade A which indicates C-SOLVIS as the best imaginable application. 28.6% of the participants rated C-SOLVIS as an excellent application by giving scores in the Grade B range, while 14.3% of the participants rated it as a good application by giving scores in the Grade C range. This analysis is also shown in a bar chart form in Figure 9.

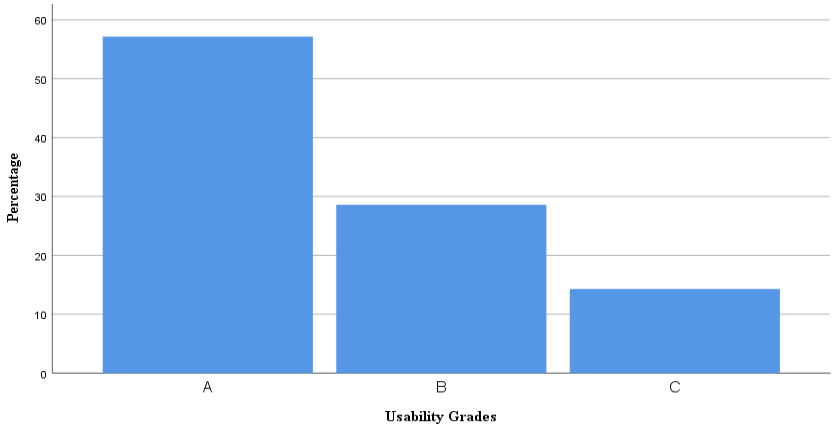

Figure 9. Frequency of SUS Score Based on Grade

From this bar chart, most of the participants evaluated C-SOLVIS as an excellent and best imaginable application for C programming by giving C-SOLVIS A and B grades. Only one participant who represents 14.3% of all participants gave a C grade for the C-SOLVIS. Nevertheless, all scores are within an acceptable range which gave a mean score of 86.07 which is within the A grade.

4. DISCUSSION

The C-SOLVIS has obtained a high mean score value for the usability evaluation which also indicates the acceptance of C-SOLVIS as an educational IDE to learn programming fundamentals. This introductory IDE is highly usable for its simple and usable design. The simple design is in line with the Cognitive Load Theory which is meant to reduce the extraneous cognitive load in learning programming [20]. This simple yet usable design is achieved as a result of strong requirements elicitation technique.

The triangulation technique used in eliciting the requirements also has produced strong functionalities of the application that involves both problem-solving and program development environments. Therefore, it provides a comprehensive application that allows the C-SOLVIS to cater from the beginning of problem-solving until program development processes, whereas the existing IDE only focuses on the programming facilities. Based on its elicited requirements, the integration of a problem-solving environment into the program

development environment in a single application is supporting the literature finding to enhance proficiency in both problem-solving and program writing tasks [21].

For the problem-solving task, the CT techniques which have been suggested by [22] were implemented in the Scientific Instructions and Inquiries as suggested by [23]. This set of Scientific Instructions and Inquiries which is the main component in the problem-solving environment applies the main pillars of CT techniques by decomposing a problem into steps of logical segments which is easier to abstract main information and variables guided by the IPO Model. They also help in the recognition of selection and repetition structure, thus helping in algorithm design [24]. Moreover, this decomposition technique could reduce the student's cognitive load in their working memory as described by the Cognitive Load Theory [25].

Meanwhile, for the program writing task, the frame-based programming technique was implemented in the program development environment of the C-SOLVIS which provides a better structure to the programming environment [26]. Moreover, it provides transparent coding where all codes automatically appear in the code editor according to the frame to accelerate coding and thus could transcend the conventional text-based programming technique [27]. Through this frame-based programming technique, C-SOLVIS helps novices to build a C program through the existence of the Code Patterns. Based on the Constructivism Learning Theory, these Code Patterns provide an active-learning environment in which the students build new knowledge upon the foundation of previous learning and acquire the skills and attributes in programming [24].

Being an educational software, the UI design is very important to ensure this application is easily used and benefits the teaching and learning process. Based on the Cognitive Load Theory, the C-SOLVIS UI design aims to reduce the cognitive load of the user by simplifying the application. To achieve this purpose, the UI is designed to be minimalist, consistent and flexible as suggested by the SUIG guidelines. Therefore, the problem-solving steps were separated into different pages to avoid crowding a single page with many buttons that could confuse the user. In addition, the C-SOLVIS is also flexible to corrections by having the IPO Chart which is designed to display the data submitted by the user so that the user can review and reset it if needed.

The C-SOLVIS is constructed using the WebStorm IDE utilizing JavaScript programming language for both client-side and also server-side programming. WebStorm offers coding assistance for full-stack development that allows code debugging for both client-side and server-side with its built-in debugger, which then integrates both of those sides. Although WebStrom is restricted to being run only on the Google Chrome web browser, it is the only Javascript IDE that practically supports code editing during debugging [28]. It is an advantage to use JavaScript in constructing this web-based IDE as it is the most popular programming language for the development of the web-based application by most developers and thus enables sharing and support from the developers' community [29].

Since the C-SOLVIS prototype is meant for educational purposes, evaluation was done among a group of the application users who are the programming lecturers which is only a small sample size. Therefore, the use of SUS for this evaluation is the most suitable instrument as it has been proven as valid and reliable even though implemented for as low as three respondents [14], [15]. Moreover, the high value of Cronbach's Alpha obtained in this study has justified the reliability of this evaluation.

The design and development of C-SOLVIS have contributes to significant implications in educational software design and development. It also contributes to programming education as it could enhance the teaching methods among the programming lecturers. Besides, its usable design also could help beginners to be familiar with C coding and problem-solving and adapt to the professional IDE confidently later.

5. CONCLUSIONS AND PROSPECTS FOR FURTHER RESEARCH

This study has been carried out according to the software engineering discipline and has successfully produced educational software to enhance the teaching and learning of C programming. From past literature and recent research, it was found that teaching and learning programming could be enhanced by certain learning theories. The implementation of suitable problem-solving models and programming techniques also could assist in the problem-solving process and program writing among beginners.

Problem-solving using Scientific Instructions and Inquiries and frame-based programming using Code Patterns have been figured out in this research as techniques to overcome programming difficulties among the students. These techniques were implemented by following the CT approach and IPO Model based on Constructivism Learning Theory and Cognitive Load Theory.

This study has focused on developing an application by following the RAD model which begins with the requirements planning to elicit the requirements of the application. From the application's requirements, the application was designed which focuses on its architecture, components, data and user interface. These designs have been the blueprints for developing the C-SOLVIS which was constructed as a web-based application.

The usability evaluation which was using the SUS measurement and interpretation shows that C-SOLVIS is highly usable by obtaining a mean value of 86.07. According to the SUS interpretation, this value indicates that C-SOLVIS is "the best imaginable" application. Therefore, C-SOLVIS is suggested to be used for the teaching and learning of introductory programming for the C language at the tertiary education level to familiarize the students with the problem-solving process and programming interface in the professional IDE. The high usability score of the C-SOLVIS also indicates that this educational software design and development framework can be used as a guide and followed by other researchers and software developers in the future.

Since the C-SOLVIS is developed for the introductory programming fundamentals, thus it only provides simple facilities for simple problem-solving and programming tasks. Therefore, it is recommended that this application could be expanded further to cater for modular and advanced programming. It is also suggested that further evaluation is done to measure learning achievement among the students after using this application by evaluating the effectiveness of C-SOLVIS in enhancing the understanding of students in programming and their problem-solving skills. Future research is also recommended to compare novices' achievement in learning programming by using C-SOLVIS and using programming IDE in a suitable research design and compatible research environment.

ACKNOWLEDGEMENT

The authors would like to acknowledge the Ministry of Higher Education of Malaysia for its support in conducting this research. This research is supported by Universiti Pendidikan Sultan Idris, Tanjung Malim, Perak, Malaysia. The authors also would like to express appreciation to the Malaysian Polytechnic and all the involved lecturers for supporting this research.

REFERENCES (TRANSLATED AND TRANSLITERATED)

[1] C. S. Cheah, "Factors Contributing to the Difficulties in Teaching and Learning of Computer Programming: A Literature Review," *Contemp. Educ. Technol.*, vol. 12, no. 2, p. ep272, 2020, doi: 10.30935/cedtech/8247.

[2] S. Y. Choi, "Development of an instructional model based on constructivism for fostering computational thinking," *Int. J. Innov. Technol. Explor. Eng.*, vol. 8, no. 3C, pp. 381–385, 2019.
[3] A. S. Hashim, R. Ahmad, and M. S. Shahrul Amar, "Difficulties in Learning Structured Programming: A Case Study in UTP," *Proc. - 2017 7th World Eng. Educ. Forum, WEEF 2017- Conjunction with 7th Reg. Conf. Eng. Educ. Res. High. Educ. 2017, RCEE RHEd 2017, 1st Int. STEAM Educ. Conf. STEAMEC 201*, pp. 210–215, 2017, doi: 10.1109/WEEF.2017.8467151.
[4] J. Henry and B. Dumas, "Developing an Assessment to Profile Students based on their Understanding of the Variable Programming Concept," *Annu. Conf. Innov. Technol. Comput. Sci. Educ. ITiCSE*, no. June, pp. 33–39, 2020, doi: 10.1145/3341525.3387400.
[5] M. H. Egan and C. Mcdonald, "An Evaluation of SeeC : A Tool Designed to Assist Novice C Programmers with Program Understanding and Debugging," *Comput. Sci. Educ.*, vol. 00, no. 00, pp. 1–34, 2020, doi: 10.1080/08993408.2020.1777034.
[6] J. Warner and P. J. Guo, "CodePilot : Scaffolding End-to-End Collaborative Software Development for Novice Programmers," vol. 9, pp. 1136–1141, 2017.
[7] F. Q. Khan, S. Rasheed, M. Alsheshtawi, T. M. Ahmed, and S. Jan, "A Comparative Analysis of RAD and Agile Technique for Management of Computing Graduation Projects," *Comput. Mater. Contin.*, no. June, 2020, doi: 10.32604/cmc.2020.010959.
[8] A. Saad and C. Dawson, "Requirement elicitation techniques for an improved case based lesson planning system," *J. Syst. Inf. Technol.*, vol. 20, no. 1, pp. 19–32, 2018, doi: 10.1108/JSIT-12-2016-0080.
[9] S. B. Merriam, *Introduction to Qualitative Research*. 2002.
[10] K. Schoch, "Chapter 16 Case Study Research," in *The Scholar-Practitioner's Guide to Research Design*, 2020, pp. 245–256.
[11] C. E. Wolff, H. Jarodzka, and H. P. A. Boshuizen, "Classroom Management Scripts: a Theoretical Model Contrasting Expert and Novice Teachers' Knowledge and Awareness of Classroom Events," *Educ. Psychol. Rev.*, vol. 33, no. 1, pp. 131–148, 2021, doi: 10.1007/s10648-020-09542-0.
[12] R. D. G. D. Reyes and V. A. G. Torio, "The Relationship of Expert Teacher–Learner Rapport and Learner Autonomy in the CVIF-Dynamic Learning Program," *Asia-Pacific Educ. Res.*, vol. 30, no. 5, pp. 471–481, 2021, doi: 10.1007/s40299-020-00532-y.
[13] S. Campbell *et al.*, "Purposive sampling: complex or simple? Research case examples," *J. Res. Nurs.*, vol. 25, no. 8, pp. 652–661, 2020, doi: 10.1177/1744987120927206.
[14] N. Clark, M. Dabkowski, P. J. Driscoll, D. Kennedy, I. Kloo, and H. Shi, "Empirical Decision Rules for Improving the Uncertainty Reporting of Small Sample System Usability Scale Scores," *Int. J. Hum. Comput. Interact.*, vol. 37, no. 13, pp. 1191–1206, 2021, doi: 10.1080/10447318.2020.1870831.
[15] Derisma, "The usability analysis online learning site for supporting computer programming course using System Usability Scale (SUS) in a university," *Int. J. Interact. Mob. Technol.*, vol. 14, no. 9, pp. 182–195, 2020, doi: 10.3991/ijim.v14i09.13123.
[16] J. Brooke, "SUS - A quick and dirty usability scale," *Iron Steel Technol.*, vol. 15, no. 8, pp. 41–47, 2018, doi: 10.5948/upo9781614440260.011.
[17] M. F. Mohamad Marzuki, N. A. Yaacob, and N. M. Yaacob, "Translation, cross-cultural adaptation, and validation of the Malay version of the system usability scale questionnaire for the assessment of mobile apps," *JMIR Hum. Factors*, vol. 5, no. 2, pp. 1–7, 2018, doi: 10.2196/10308.
[18] J. Morales, F. Botella, C. Rusu, and D. Qui, "How ' Friendly ' Integrated Development Environments Are?," *G. Meiselwitz HCII 2019, LNCS 11578, pp. 80–91, 2019*, vol. 3, pp. 80–91, 2019, doi: 10.1007/978-3-030-21902-4.
[19] D. Supriyadi, S. Thya Safitri, and D. Y. Kristiyanto, "Higher Education e-Learning Usability Analysis Using System Usability Scale," *Int. J. Inf. Syst. Technol. Akreditasi*, vol. 4, no. 1, pp. 436–446, 2020.
[20] M. A. Bakar, M. Mukhtar, and F. Khalid, "The development of a visual output approach for programming via the application of cognitive load theory and constructivism," *Int. J. Adv. Comput. Sci. Appl.*, vol. 10, no. 11, pp. 305–312, 2019, doi: 10.14569/IJACSA.2019.0101142.
[21] C. Chaka, "Skills, Competencies and Literacies Attributed to 4IR/Industry 4.0: Scoping Review," *Int. Fed. Libr. Assoc. Institutions*, vol. 46, no. 4, pp. 369–399, 2020, doi: 10.1177/0340035219896376.
[22] K. Mohd Yusoff, N. S. Ashaari, T. S. M. Tengku Wook, and N. Mohd Ali, "Analysis on the Requirements of Computational Thinking Skills to Overcome the Difficulties in Learning Programming," *Int. J. Adv. Comput. Sci. Appl.*, vol. 11, no. 3, pp. 244–253, 2020, doi: 10.14569/ijacsa.2020.0110329.
[23] A. A. Tawfik, A. Graesser, J. Gatewood, and J. Gishbaugher, "Role of questions in inquiry-based instruction: towards a design taxonomy for question-asking and implications for design," *Educ. Technol. Res. Dev.*, vol. 68, no. 2, pp. 653–678, 2020, doi: 10.1007/s11423-020-09738-9.
[24] G. Chen, "Programming Language Teaching Model Based on Computational Thinking and Problem-based Learning," *Adv. Soc. Sci. Educ. Humanit. Res. (ASSEHR), 2nd Int. Semin. Educ. Innov. Econ. Manag. (SEIEM 2017)*, vol. 156, no. Seiem, pp. 128–131, 2017, doi: 10.2991/seiem-17.2018.31.

[25] A. H. Hasan, M. F. Hilmi, F. Ibrahim, and H. Haron, “INPUT PROCESS OUTPUT (IPO) AI CHATBOT AS PERSONAL LEARNING ASSISTANT FOR PROGRAMMING COURSEWORK,” in *Proceedings of International Conference on The Future of Education IConFEd) 2020,* 2020, no. November 2020, pp. 17–18.
[26] T. Y. Sim and S. L. Lau, “Online Tools to Support Novice Programming: A Systematic Review,” *2018 IEEE Conf. e-Learning, e-Management e-Services, IC3e 2018*, pp. 91–96, 2018, doi: 10.1109/IC3e.2018.8632649.
[27] P. Perera, G. Tennakoon, S. Ahangama, R. Panditharathna, and B. Chathuranga, “A Systematic Mapping of Introductory Programming Languages for Novice Learners,” *IEEE Access*, vol. 9, pp. 88121–88136, 2021, doi: 10.1109/ACCESS.2021.3089560.
[28] N. Bouraqadi and D. Mason, “Test-driven development for generated portable Javascript apps,” *Sci. Comput. Program.*, vol. 161, no. February, pp. 2–17, 2018, doi: 10.1016/j.scico.2018.02.003.
[29] S. Alimadadi, A. Mesbah, and K. Pattabiraman, “Understanding asynchronous interactions in full-stack JavaScript,” *Proc. - Int. Conf. Softw. Eng.*, vol. 14-22-May-, pp. 1169–1180, 2016, doi: 10.1145/2884781.2884864.

Text of the article was accepted by Editorial Team 05.10.2022

ВИКОРИСТАННЯ ОСВІТНЬОГО ІНТЕГРОВАНОГО СЕРЕДОВИЩА РОЗРОБКИ ДЛЯ ВИВЧЕННЯ ОСНОВ ПРОГРАМУВАННЯ

Нор Фарахвахіда Мохд Нур
бакалавр з інженерії (комп'ютер та інформація),
кафедра обчислювальної техніки, Педагогічний університет Султана Ідріса
Танджунг Малім, Перак, Малайзія
norfarahwahida@gmail.com

Асліна Саад
PhD (інформатика), доцент, викладач
кафедра обчислювальної техніки, Педагогічний університет Султана Ідріса
Танджунг Малім, Перак, Малайзія
Ідентифікатор Scopus: 37162127300
aslina@fskik.upsi.edu.my

Абу Бакар Ібрагім
PhD (електроніка), доцент, викладач
кафедра обчислювальної техніки, Педагогічний університет Султана Ідріса
Танджунг Малім, Перак, Малайзія
Ідентифікатор Scopus: 57189494415
abubakar.ibrahim@fskik.upsi.edu.my

Норашаді Мохд Нур
PhD (освітні вимірювання), викладач
кафедра машинобудування, Політехнічний інститут Султана Азлан Шаха,
Беранг, Перак, Малайзія
norashady@gmail.com

Анотація. Програмування є важливим предметом для будь-якого навчального курсу, пов’язаного з ІТ або інженерією. Попередні дослідження демонструють, що через абстрактні поняття в учнів виникають труднощі під час навчання програмуванню. Метою представленого дослідження є оцінювання прийнятності розробленого Інтегрованого Середовища Розробки (ІСР) (Integrated Development Environment – IDE), а саме C-SOLVIS – вебдодатку, який має полегшити процес викладання та вивчення основ програмування мовою С у закладах вищої освіти Малайзії. C-SOLVIS інтегрує прийняття рішень у середовище розвитку програм мовою С, маючи на меті спрямовувати користувачів під час прийняття рішень і написання програми мовою С на основі алгоритмів розв’язування задач. Під час розробки C-SOLVIS використовувалась Модель Швидкої Розробки Додатків (ШРД) (Rapid Application Development – RAD). На основі цієї моделі фаза планування вимог була здійснена за допомогою техніки тріангуляції із застосуванням якісних підходів, які передбачають огляд літератури, підкріплений напівструктурованими інтерв’ю, оглядами

документів і перевіркою змісту лекторами-експертами з програмування. Згодом дизайн додатка здійснювався через ітеративний процес прототипування з подальшим створенням програми. Після цього C-SOLVIS було використано декількома викладачами з програмування, які оцінили зручність його використання за допомогою кількісного методу шляхом опитування за Шкалою Зручності Використання Системи (ШЗВС) (System Usability Scale – (SUS). Під час дослідження було виявлено декілька зручних методів і схем, відповідних середовищу програмування і розробки програм. Для середовища програмування було застосовано концепції Обчислювального Мислення (ОМ) (Computational Thinking – (CT), які підтримувалися моделлю «Вхід-Процес-Вихід» (Input-Process-Output (IPO) Model) з використанням наукових інструкцій та запитів. Середовище розробки програм було створено на основі фреймової моделі програмування з використанням набору розроблених шаблонів коду. C-SOLVIS був оцінений за допомогою інструменту ШЗВС (SUS) з середнім балом ШЗВС (SUS) 86,07, що інтерпретується ШЗВС (SUS) як оцінка A і вказує на те, що C-SOLVIS є дуже зручним додатком, прийнятним для вивчення програмування на C. Тож процес розробки C-SOLVIS може бути використано як керівництво для розробки освітнього програмного забезпечення, особливо в навчанні програмуванню.

Ключові слова: Інтегроване Середовище Розробки (ICP); освітнє програмне забезпечення; зручність використання; програмування.

(cc) BY-NC-SA

This work is licensed under Creative Commons Attribution-NonCommercial-ShareAlike 4.0 International License.