International Journal of Serious Games

ISSN: 2384-8766
https://journal.seriousgamessociety.org/

Article

Introducing the Visitor Design Pattern through a Digital Serious RPG Game

Adilson Vahldick[1] and José Vargas Nolli[1]

[1]*Department of Software Engineering, University of Santa Catarina State, Ibirama, Brazil*
adilson.vahldick@udesc.br; jvargasnolli@gmail.com

Keywords:

Case study
Design patterns
Role-playing game (RPG)
Serious games
Software engineering education
Visitor

Received: August 2025
Accepted: January 2026
Published: January 2026
DOI: 10.17083/pzmm0y41

Abstract

Learning design patterns empowers students to solve complex problems in a way that code is reusable and extensible with minimal maintenance. Due to the immersive and exciting environment, the RPG genre of games holds the potential for more meaningful learning. This article presents a serious RPG game that introduces the concepts of the Visitor design pattern. A class of 14 students tested the game, achieving statistically significant performance through a pre-test and post-test validation protocol. Moreover, the instructor noted that the students demonstrated a greater introductory mastery of the pattern compared to previous semesters. The students also evaluated the usability and experience of the game, pointing out gaps in the game's difficulty. The main contribution is that, given the lack of games for learning design patterns, the game presented in this article has demonstrated its utility as a pattern comprehension tool.

1. Introduction

A design pattern describes a problem and how to solve it, considering objects and interfaces, which can be used a million times without ever doing it the same way [1]. Design patterns help choose solutions that make a system reusable and, consequently, easier to maintain. Design patterns are rarely used in isolation and most often need to be modified and adapted to coexist with other patterns. Design patterns facilitate the transition to an object-oriented way of thinking by presenting techniques on how to use object orientation correctly and efficiently [2].

Students find it difficult to perceive the necessity of design patterns. Using appropriate case studies can enhance learning patterns [3]. The correct and continuous learning of design patterns should represent a traditional practice in any course that aims to offer a solid foundation in the object-oriented paradigm [4]. Furthermore, teaching design patterns is also not an easy task: first, reminding students that they have already used patterns in other subjects (e.g., repetition in introductory courses); presenting each pattern to them; providing exercises for practice; making it clear that patterns are not used in isolation, but usually, more than one

pattern is used simultaneously; and also designing assessments so that students can decide which pattern is best applicable and how to do it [5].

There are numerous alternatives to increase students' motivation by bringing the topics to be taught into their daily lives, such as using games in the teaching-learning process. Games offer an alternative learning mechanism that should be used appropriately by teachers as a powerful motivator for the beginning of the learning process, stimulating cognitive relationships, and providing an active, critical, and creative reaction from students, socializing knowledge [6]. Games are simulated environments that allow for testing, hitting, and making mistakes multiple times, respecting the student's cognitive time, aiding their learning and practical application [7].

There are two ways to apply games in design patterns learning [8]: developing the game and using a game. For the first approach, there are several works in the literature that involve students developing games by applying design patterns. However, there is a shortage of games developed for learning design patterns where the student assumes the role of a player and tries to solve the challenges proposed by the game. Only two games, both based in cards, were found: [9], [10]. It is noted that both games should be used in the final classes of the course, after students have learned the patterns. Nevertheless, no games were found that can help with the initial understanding of patterns.

Although Bloom's Taxonomy determines that educational objectives are organized into levels of cognitive complexity, and these need to be followed to operationalize learning [11]. In this direction, the first and low levels are knowledge and comprehension, which involve the memorization of facts and information and the ability to understand and interpret this information. The highest level is evaluation, which involves judging the value or effectiveness of materials or methods based on specific criteria. The two games found in the literature are applications of the highest cognitive levels. As the use of serious games provides an interactive environment where students memorize information and understand basic concepts in an engaging manner, games can be an easily accessible alternative for cognitive development in the levels of knowledge and comprehension [12].

The Visitor design pattern is used to represent operations that can be performed on the elements of an object structure [1]. This is useful in situations where the algorithm operates on objects in a data structure of different concrete types. The Visitor pattern involves abstract concepts, such as visitors and elements, which can be difficult to understand initially for students, especially those with little experience in object-oriented programming. Additionally, implementing the Visitor pattern requires a solid understanding of concepts like polymorphism and interfaces. The need to create additional classes to represent visitors and modify existing classes to accept visitors can also add a layer of complexity. Due to its abstraction and relative complexity, it is not easily understood and can be readily misused, leading to code structures that are difficult to maintain [13]. For this reason, it is considered unsuitable for use by novices. However, its use makes systems more extensible and maintainable, while also aiding decomposition and reducing coupling [14].

Figure 1 presents the basic structure of the Visitor design pattern, which includes the pattern's three class types: visitors, visitables (or elements), and the object structure [1]. The purpose of visitor classes is to encapsulate operations to be performed on an object structure while avoiding tight coupling with the original classes of those objects. Visitable classes serve as data containers that expose an acceptance interface through which visitor classes can access and process their contents. Finally, the object structure serves as the organizing core of the Visitor pattern, responsible for storing and managing all visitable elements while coordinating the visitation flow - centralizing iteration logic and ensuring each visitor can process elements without needing to understand the collection's internal complexity.

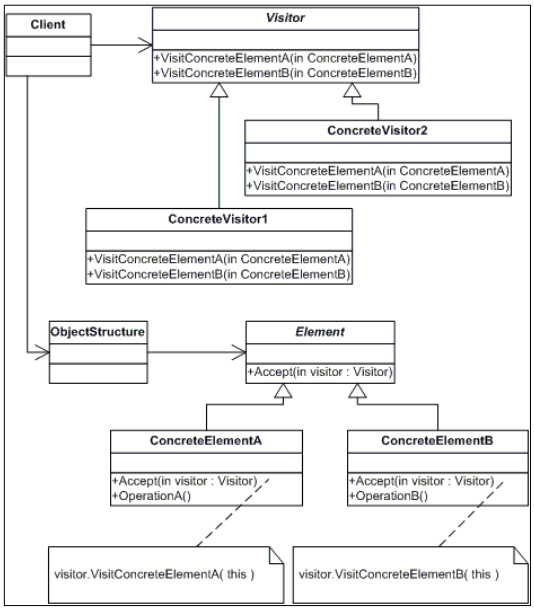


Figure 1. Basic UML Class Diagram of Visitor Design Pattern.

The Role-playing game (RPG) is a game genre where players assume a character class and increase their statistical skills in combat, exploration, and treasure hunting [15]. RPGs can be classified into analogue and digital [16]. For analogue examples are tabletop, live action (LARP), and gamebooks. For digital examples are solo adventure and multiplayer online role-playing game. RPG games allow students to become protagonists of their own stories, which can significantly increase their interest and commitment to learning [17]. By integrating curricular content into game narratives, students can learn in a more contextualized and meaningful way [18].

As mentioned, the motivations for this project are that (1) there are limited games in the literature, typically used to evaluate a set of patterns rather than to enhance understanding of a specific pattern, and (2) the Visitor pattern is complex for beginners to understand. Therefore, the general objective of this research was to develop an RPG game to introduce the initial and elementary concepts of the Visitor pattern. The game was named *Relics of Chaos*.

This work aims to demonstrate that simple tools with basic features - such as RPG Maker, which requires no programming language expertise nor 3D environment/character modeling skills - can effectively develop the initial cognitive levels of Bloom's Taxonomy.

2. Learning and Teaching Design Patterns

Learning and teaching design patterns presents significant challenges, primarily due to their high level of abstraction and the difficulty students have in applying these patterns in large projects. Design patterns are high-level solutions to recurring design problems and require a deep understanding of object-oriented principles. Students often struggle to grasp how these patterns can be applied in practical scenarios without concrete examples and extensive practice [19]. Students might find it difficult to identify where and how to apply specific patterns effectively. The theoretical understanding of a pattern does not always translate into practical skills, which can lead to challenges in implementing these patterns in real-world projects [20]. Without a practical and interactive application, students may find it difficult to see the relevance of design patterns in real-world situations [21] .

To overcome these challenges, several strategies can be implemented, for instance, organization of exercises from the simplest to the most complex [22], peer code review [23], interactive programming [24], visual metaphors [25], game approach [26] and hackatons [27].

Regarding the game-based approach, several works in the literature refer to the development of small games. The motivation behind this approach, besides the fun factor [28], is grounded in the appropriate use of case studies that allow students to extract knowledge through real-world situations where the need for using patterns is evident [3]. Game development provides this opportunity for real-world applications, integrating patterns that enable incremental development. This approach allows students to recognize the advantages of using patterns to facilitate software maintenance and reusability [26]. The introduction should be written for a general audience.

3. Related Works

Despite there being many articles reporting on the task of students developing games to learn patterns, only two games suitable for student use were found, both of which were card games.

A commercial game called Dixit[1] was used in [9]. It is a card game with abstract figures. Each player has six cards in hand, and in each round, one of them is the narrator responsible for choosing one of their cards and associating it with a story that characterizes a design pattern. The other players choose one of their cards that can be characterized in the pattern and deliver it to the narrator. In the end, players need to vote on which card belonged to the narrator. The game was used to evaluate learning at the end of the course. Each player had to exercise their knowledge of the patterns learned in that course to use their creativity and launch the cards.

There are two sets of cards in [10]: one set with the 23 design patterns from [1] and another set with the requirements of a web-based coffee sales system, one side of the card contains an explanatory sentence, contains the answer, i.e., the pattern to be applied along with the UML diagram of the pattern application. The students' task is to select a card from the first set of applicable patterns that matches the drawn requirement card.

Both games are suitable for student use at the end of the semester, after they have mastered the patterns. However, one of the beginners' mistakes in learning is selecting the design pattern, that is, the ability to identify the pattern to be applied, due to the lack of demonstrating simple examples with diagrams and codes [29]. In other words, a possible flaw for students to have the ability to apply and create new solutions using a Design Pattern is disregarding the first cognitive levels of Bloom's Taxonomy [11], making students have to create new solutions (higher cognitive domains) without first understanding the essence of a pattern.

4. Methodology

For the development of this research, a professor of the Design Patterns course in the Bachelor's Degree in Software Engineering at University of Santa Catarina State in Brazil was supported, who recommended the pattern to be used in the game. The Design Patterns course is taught in the fifth term. Prerequisites include completion of CS1 (Introduction to Programming), CS2 (Object-Oriented Programming I & II), and CS3 (Data Structures). The professor also provided and recommended bibliographic material on the Visitor pattern. In conjunction, he already

[1] https://boardgamegeek.com/boardgame/39856/dixit

organized the teaching plan to use as an introductory strategy for the subject before the practical classes. The professor refrained from presenting the initial theoretical concepts of the design pattern. However, before beginning the practical sessions, he conducted an interactive discussion class to assess whether the knowledge gained from the game was sufficient to proceed to practice.

Next, information was collected on the development of serious games concerning their pedagogical particularities and some similar games were studied. Then, a technical study was conducted on the development of games regarding tools and methods. Based on this information, it was decided to develop an RPG game, and the storyline, characters, and instructional objectives, and consequently, the game's challenges were defined.

The game was developed with RPG Maker MZ scripts in JavaScript. A feature of the tool called Character Generator was used for the character art. Before the experimentation with the students, the game was tested for usability and quality by the GameLab Laboratory[2] team.

The game's validation protocol was determined as follows: first, a pre-test with 10 multiple-choice questions with a 10-minute limit was applied to determine the students' initial knowledge; then, students could experiment with the game for 50 minutes; right after, the post-test with the same questions was applied, but in a different order of questions and alternatives and also with a 10-minute limit; finally, the last task of the students was to evaluate the usability and experience with the game using a questionnaire adapted from MEEGA+ [30].

Nine questions were added to characterize the sample. The 26 items used from MEEGA+ are on a *likert* scale with five levels of agreement from strongly disagree to strongly agree. This instrument scale aims to classify educational games regarding their quality level, based on students' perceptions, allowing identifying which requirements correspond to the lowest or highest quality level. The items are grouped into the categories of Usability, Confidence, Challenge, Relevance, Satisfaction, Fun, Focused Attention, and Perceived Learning. It is noteworthy that the social interaction category was suppressed due to the game not having collaboration features.

5. Relics of Chaos

As mentioned, the RPG game's narrative helps in immersion and consequently meaningful learning. The story tells of Edivaldo's (the protagonist) quest for the Relics of Chaos, which are artifacts that bestow immense power on their holders. The game takes place in three maps where the student completes instructional tasks. During the game, the protagonist meets other NPCs who assign them tasks, participate in instructional tasks, choose to accompany the protagonist on their journey, as well as enemies and some supporting characters.

5.1 Instructional design

As previously described, a potential flaw in students' ability to apply and create new solutions using a Design Pattern is neglecting their mastery of it at the initial levels of Bloom's Taxonomy. Therefore, the game was developed to meet the following instructional objectives:

[2] https://www.udesc.br/ceavi/gamelab

1) Identify the intent and use of the Visitor pattern.
2) Understand the structure and components of the Visitor pattern.
3) Explain the interaction and functioning of the classes in the Visitor pattern.
4) Demonstrate the recognition of the elements of the Visitor pattern in simple code fragments.

The first three objectives fall under the category of knowing and understanding, and thus, the technique of explaining the subject and concluding with multiple-choice questions was used. The fourth objective pertains to the category of applying, where the student must use the acquired knowledge in new situations. Error correction tasks were used. However, due to the limitation of the tool, this task was adapted to multiple-choice questions.

The first two objectives were implemented with an NPC (a robot) explaining the components of the pattern through class diagrams, concluding four multiple-choice questions. In Figure 2 is illustrated an explanation screen. Essentially, this type of screen presents a diagram in its central part, an image of the NPC at the bottom with an explanation of the content in the central part and two buttons (<Prev and Next>) to navigate between the explanations. In Figure 3 is presented a multiple-choice question screen. At the bottom, there is a question based on the diagram in the central part, and on the right side, four alternatives are presented for the student to select one of them. The question is re-displayed, with the alternatives reshuffled, until the student answers correctly. This approach is not intended as a knowledge test, but rather as a guided learning tool—allowing the student to actively engage with the content, reinforce understanding through repetition, and internalize the correct association through feedback. The questions are based on a case study: a hunter spread various baits to capture different types of animals. The same bait has a different effect on each type of animal. The bear becomes enraged when it eats meat and sleepy when it eats honey. On the other hand, the bee doesn't care about meat at all but becomes happy with honey. This case study serves both a metaphor for the Visitor pattern and as a hint for the player to later confront the Dark Bear.

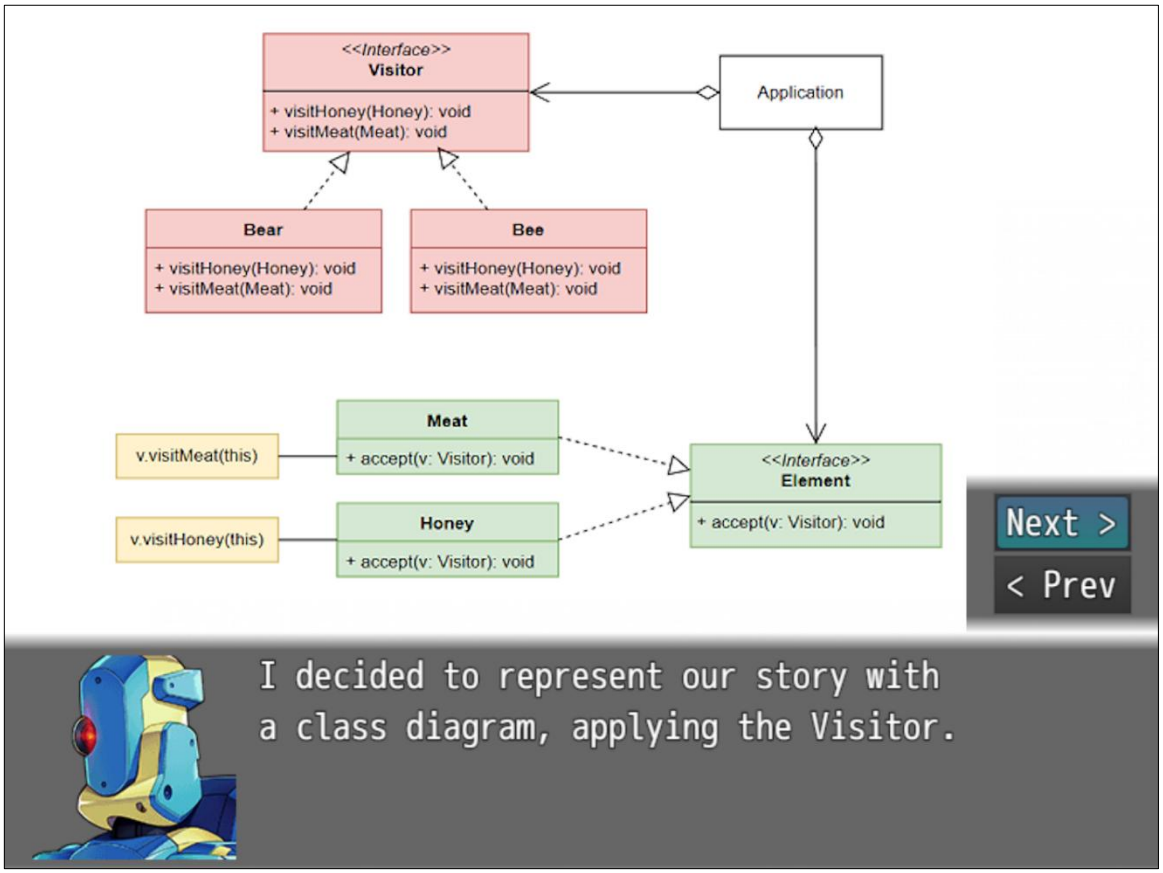


Figure 2. Explanation screen.

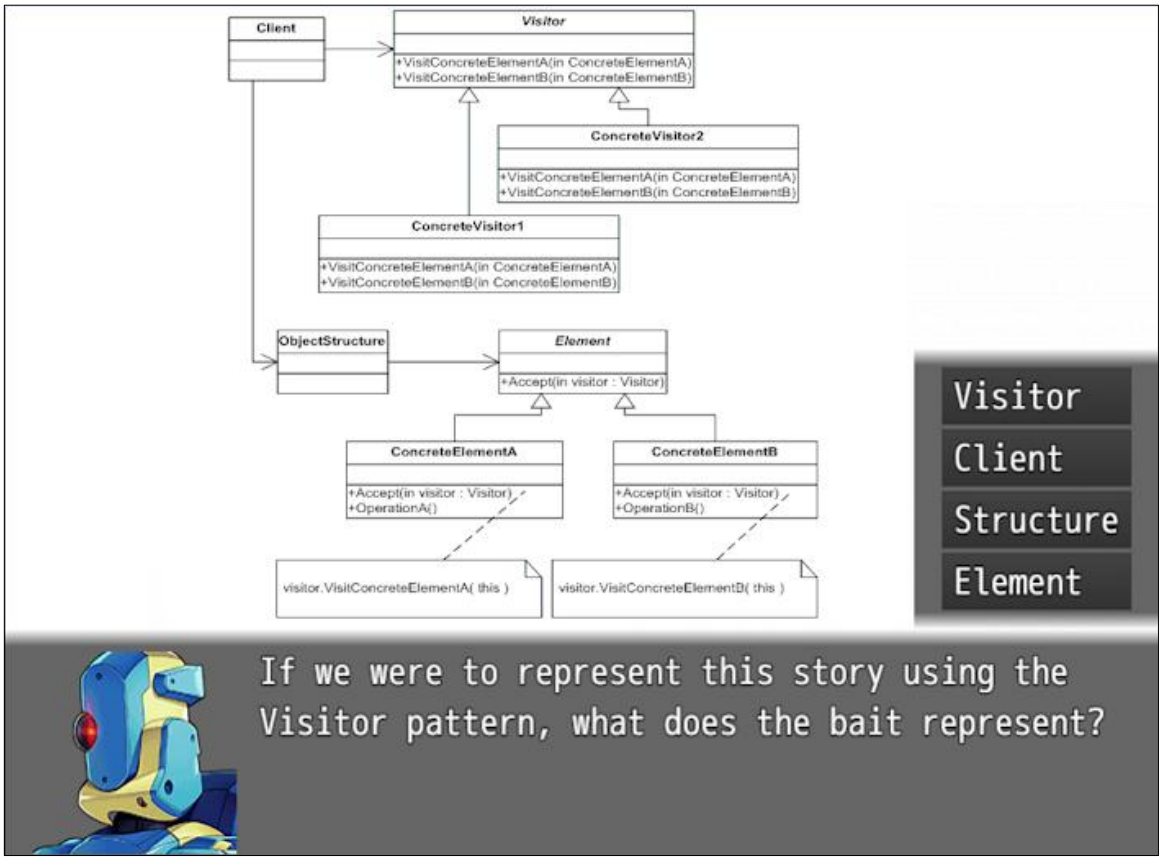


Figure 3. Multiple-choice question screen.

Another NPC (a mechanical) aims to help the student meet the third instructional objective, "Explain the interaction and functioning of the classes" presented by the previous NPC. This other NPC explains and presents four multiple-choice questions, but with the difference that the explanation uses Java source code instead of class diagrams.

The last objective is addressed at three points in the game. These moments are represented by battles with enemies. However, for the student to enter the battle, they first need to correctly answer a multiple-choice question that presents code fragments of five classes, with two of them being incorrect. Each alternative uses two class options. At this stage, it is expected that the student has understood the intent, structure, and functioning of the pattern and can identify which ones do not follow the pattern. In Figure 4 one of these three questions is illustrated.

Figure 4. One of the questions about the last instructional objective.

5.2 Game narrative

The main city of this history is Miriadin as illustrated in Figure 5. The game begins in the Lame Bull Tavern (1) where the protagonist receives the mission to bring the Dark Bear's Hide to Archimaus, an NPC who acts as a mentor. To do this, Edivaldo needs to pass through the gates (3) to reach the Enchanted Forest. However, he is stopped by two guards who direct the protagonist to the City Depot (2) to obtain authorization from the quartermaster. The instructional tasks to achieve the first three objectives take place in the City Depot. After completing the tasks, the protagonist receives authorization to enter the Enchanted Forest. Additionally, one of the NPCs involved in the learning task will accompany the protagonist.

Figure 5. City map of Miriadin.

The first version of the Forest is represented in Figure 6A. The protagonist and his companion are greeted by an NPC (Victim is his name) who explains that the forest is inhospitable due to the Dark Bear's spell. The entrance to the cave where he is hiding will be revealed after defeating two guardians (1) and (2), who are also related to the tasks to achieve the last instructional objective.

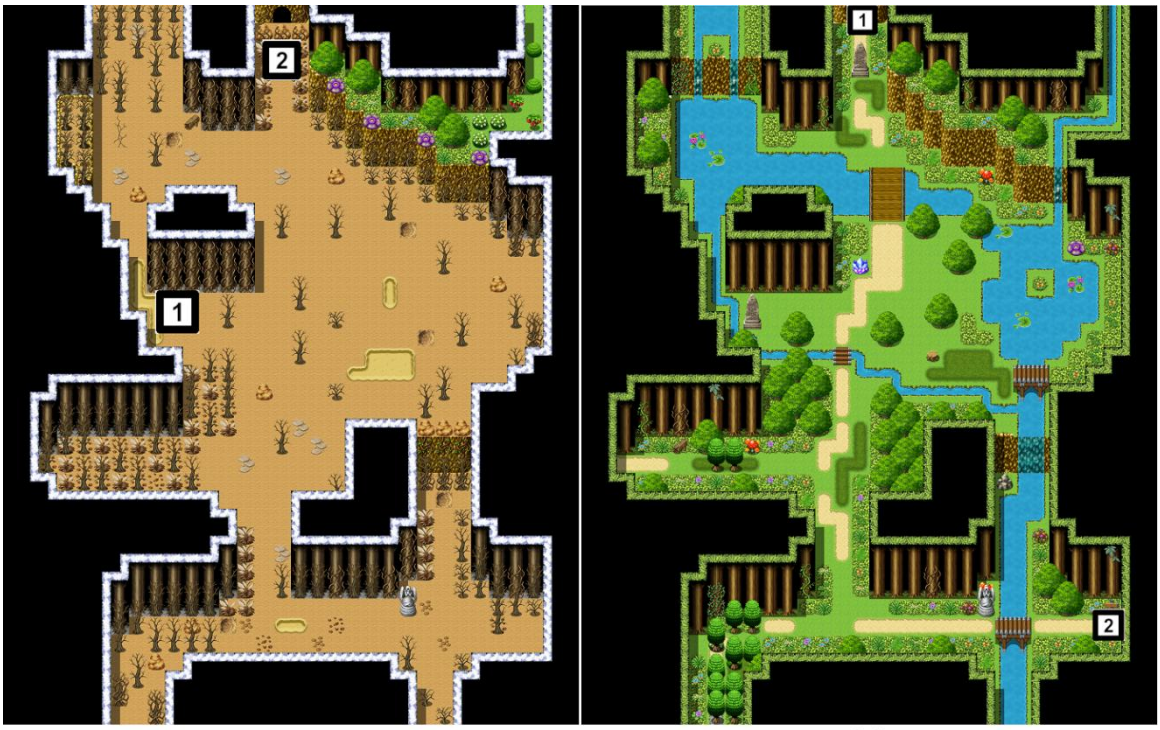


Figure 6. A) Enchanted Dark Forest B) Enchanted Forest of Life.

To enter and face the guardians, the student needs to complete the challenge of the last instructional objective (4). Once the student gains access, they must use the RPG Maker's turn-based battle system to defeat the guardian. An example of these battles is exemplified in Figure 7. This is an element of challenge and fun, ensuring that the game, even though educational, still retains the characteristics of the RPG style.

Figure 7. Turn-Based Battle System.

After defeating the two guardians, the forest is restored (Figure 6B), and the NPC Victim reveals herself to be a fairy who will accompany the duo. The entrance of the cave is revealed and in Figure 6B it is located at (1). The case study presented in the questions for the first two instructional objectives provides a hint for defeating the bear: it calms down when it eats honey and gets upset when eating meat.

When the bear is defeated, they return to Archimaus at the Lame Bull Tavern (through 6B (2)), carrying the bear's hide. In the tavern, Archimaus mentions the latest quest in pursuit of the *Relics of Chaos*, where they will need to defeat the deadliest enemy: the Herald of Infinity. So much so that in this battle, Archimaus joins the team, which at this point consists of three characters: Edivaldo, the NPC from City Depot, and the fairy from the Enchanted Forest. To create a tense moment, the first strike comes from the Herald, defeating the entire team at once. In the second round, the standard battle turns continue.

After defeating the Herald, the characters are transported to Ram Village. The game ends with the appearance of a new character: a fox. With this mystery, the game allows for the continuation of the story, so future works can explore other patterns.

Each turn, the player has four types of attacks (water, earth, fire, and lightning), which may or may not be effective depending on the enemy they are facing. For example, if a water spell is cast against a marine enemy, the guardian will recover health.

The game's narrative is carefully designed to scaffold learning by integrating instructional objectives into the storyline and gameplay. Table 1 maps the instructional objectives outlined in Section 5.1 to specific points on the game maps, the NPCs involved, Bloom's cognitive level and emphasizes the activities students needed to perform at each location. This scaffolding ensures that players gradually build their comprehension of the Visitor pattern while advancing through the game's narrative.

Table 1. Mapping of instructional objectives to in-game activities

Instructional Objective	Location	NPC	Bloom's cognitive level	Activity
1	Miriadin (Figure 4, [2])	Robot	Knowing and Understanding	Explanation screens and multiple-choice questions.
2		Robot		
3		Mechanical		
4	Enchanted Dark Forest (Figure 5, [1])	Victim	Applying	To identify two incorrect code fragments via multiple-choice questions.
4	Enchanted Dark Forest (Figure 5, [2])	Victim		
4	Miriadin (Figure 4, [1])	Archimaus		

6. Validation and the results

The study aimed to evaluate the knowledge acquired between the two validation moments, according to the statistical hypothesis below.

H_0: $\mu pre = \mu post$ – There is no significant difference in the means between the pre-test and post-test scores.

Fourteen students participated in the experiment (n=14), with an average age of 23 years, including 2 females. The game was experienced individually, each on their computer in the university lab. To better characterize the sample, the they were asked about the frequency of playing (Figure 8), and we can notice that the majority of these students responded that they do not usually play. These numbers align with reports [31] showing that college students have less free time available for gaming. However, 10 (71%) students answered that they enjoy playing RPG games.

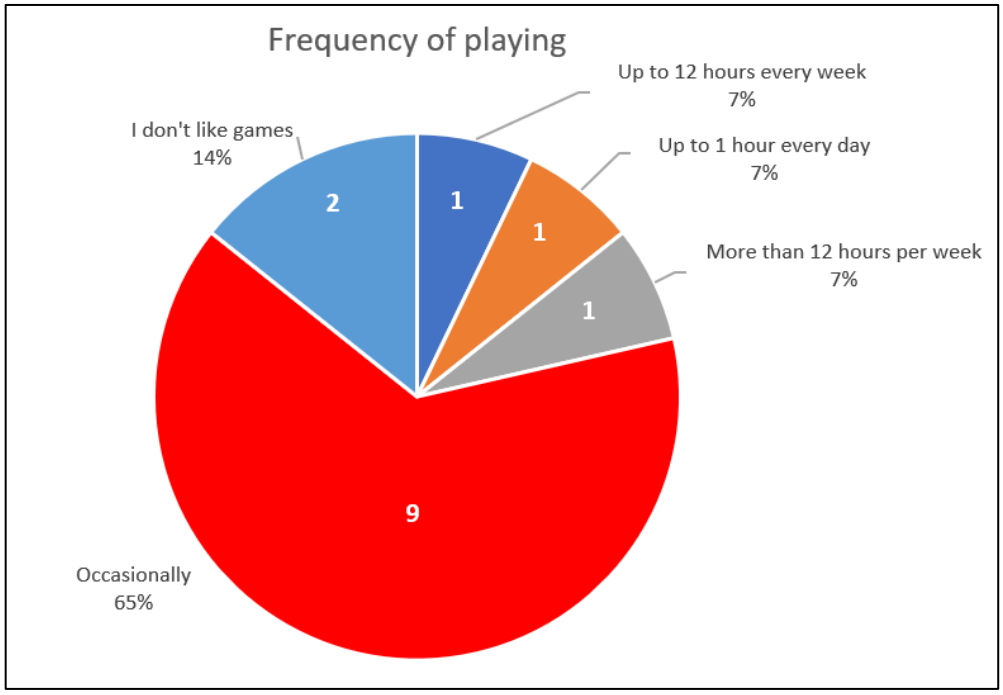


Figure 8. Frequency at which the sample usually plays.

The ten questions in the tests were created primarily to assess knowledge and understanding of the Visitor pattern, meaning they aim to evaluate the lower cognitive levels of Bloom's Taxonomy. They were reviewed for content validity by the course professor where this game was validated. The complete questionnaire is presented in Appendix A.

In Table 2 are presented the results of the knowledge tests conducted with the students. The total points refer to the number of questions in the tests multiplied by the sample size (10 questions X 14 students = 140 points). Additionally, Figure 9 shows the grades obtained by each of the 14 students in each test. Only one student performed worse on the post-test. These tests had a duration of 10 minutes, and he was the only one who answered few questions within that time on the post-test. Therefore, this result does not refer to errors made by the student but was mainly influenced by unanswered questions. It can be observed that the mean of the post-test is higher than that of the pre-test.

Table 2. Descriptive statistics of the tests

	Total points	Points obtained	Mean	Median	Standard deviation	Variance
Pre-test	140	33	2.4	2	1.78	3.17
Post-test	140	83	5.9	6	2.20	4.85

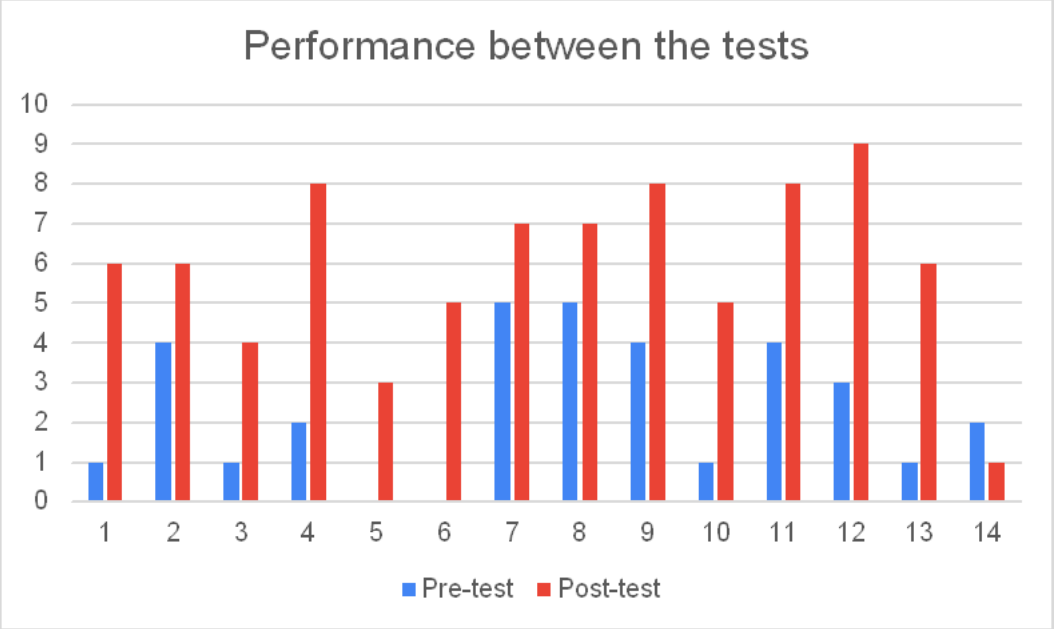


Figure 9. Comparison of performance between pre-test and post-test.

However, to verify if the difference is statistically significant, the statistical hypothesis was evaluated using a paired t-test. The paired t-test revealed a statistically significant difference ($p = 9{,}395 \times 10^{-6}$, power = 99.9%, d = 1.76), rejecting the null hypothesis with 99% confidence that the performance was the same between the two tests. Since the mean of the post-test results exceeded the mean of the pre-test results, an improvement in students' knowledge after using the game can be noted. It is also worth noting that the next day after validation, a second class of this pattern was held, where students had contact with more code and examples. The teacher reported that at the beginning of the class, he asked students to assist in modelling the class diagram of a new problem and emphasizes that he was surprised by the proficiency with which the students responded. To further complement the analysis of the game's effectiveness, it is also worth mentioning that an individual programming assessment (exam) on the Visitor design pattern was conducted. In the previous semester, the class average had been 5.0, whereas for this class where the experiment was applied, it was 8.0.

To conclude the tests, the MEEGA+ instrument was applied to evaluate gaming experience. According to the results presented in Figure 10, it can be observed that almost all questions had high percentages of agree and strongly agree. Two items require special attention due to the number of disagreements. The item "Q14-It was not necessary to have any prior knowledge to understand the game mechanics" can be explained by the maximum time given to them to play, which was 50 minutes. The GameLab Laboratory team had already identified the need for a tutorial to take the first steps in the turn-based battle system. This was implemented in the game. However, with the concern of finishing the game within 50 minutes, it was noted that the students ended up not using the tutorial. Next, items "Q12-The game mechanics were easy to understand" and "Q13-It was easy to identify the tasks in the game" also had some responses expressing disagreement. And because of that, they took more time to learn the battle system, which ended up frustrating the experience, which may explain the dissatisfaction level of the item "Q28-The time spent playing the game was enjoyable". Nevertheless, it is worth noting that there were no disagreements regarding the item "Q27-You have fun playing". Interestingly, item "Q19-The game appeared to be enjoyable upon first viewing." reflects the students' initial expectations.

Despite the absence of "Strongly Disagree" options, the third item with the lowest satisfaction level is item "Q22-The game is challenging for you". Challenging games are those where difficulties are progressively promoted [32], and they are offered in various forms [33]. Due to the native challenge types in the RPG Maker tool, it was possible to develop multiple-choice questions and use the battle system. New types of challenges can be incorporated through the acquisition of plugins. In terms of usability, the top-rated items were Q10 and Q11 (covering interface, colors, and text), followed by Q15 (characters' dialogues) and Q18 (content presentation structure in the game).

An open-ended question was asked, soliciting their suggestions and opinions, both positive and negative. The most evident response was that they started playing wondering "how will the game teach design patterns?". This is reflected in the items "Q34-The game contributed to your learning about Visitor" and "Q35-The game facilitated understanding of the subject matter", where there are no responses disagreeing. Additionally, they commented that (1) the instructions on the pattern did not break immersion in the game, as they are well integrated into the history, and (2) they felt the need to take notes because the explanations provide shortcuts for the game. However, there were suggestions for a deeper exploration of the pattern (despite not being the game's purpose), with more examples (that could increase gameplay time), as well as more questions and other types of challenges, such as completing the code. As negatives, there were complaints about the quality of some images used in explanations and questions, which stretched disproportionately horizontally or vertically, the lack of feedback on incorrectly answered alternatives (for example, sound or visual alert), and of course, about the battle system. Even though 71% of the students admit to liking RPG games, they may naturally lack experience with games developed using RPG Maker.

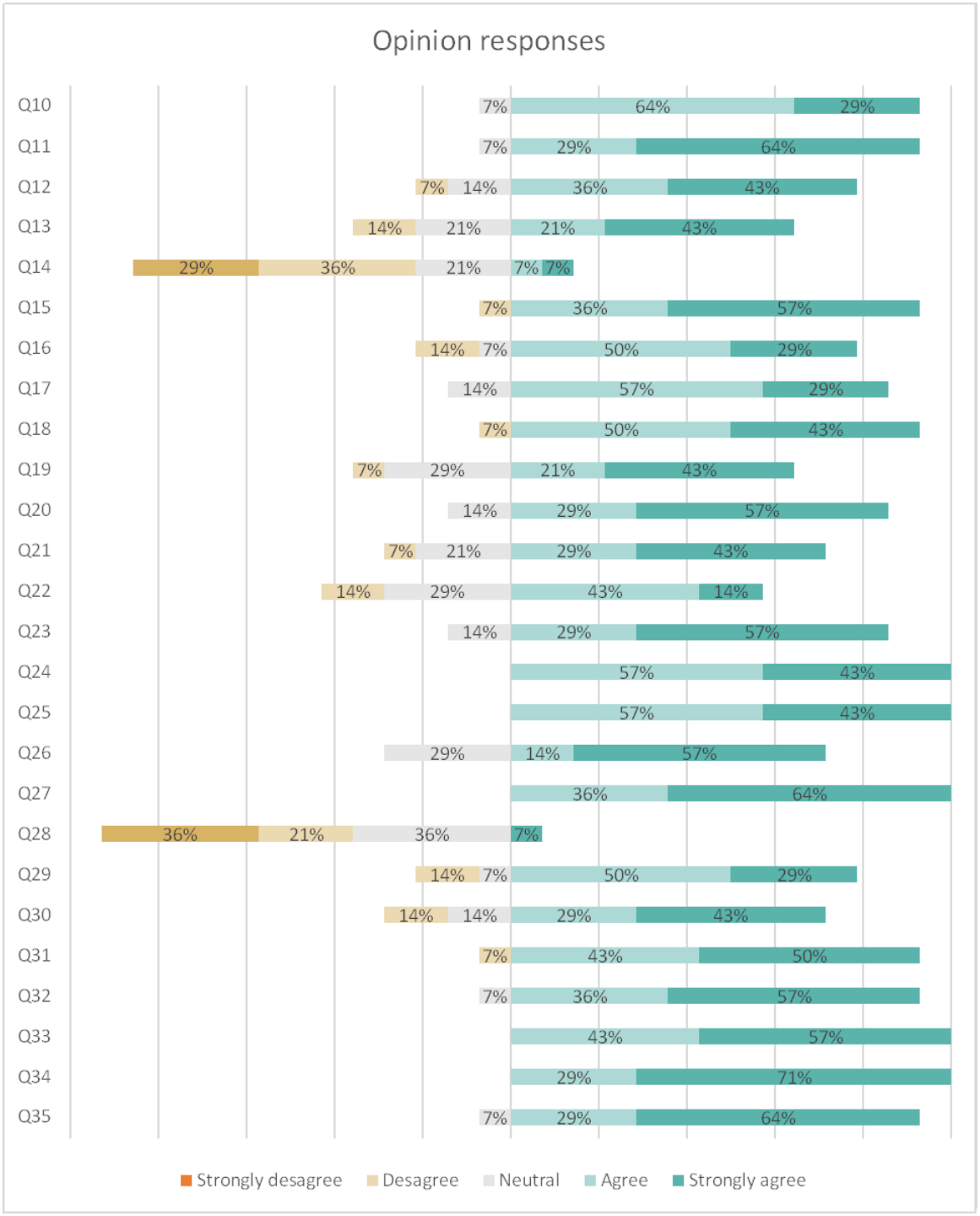


Figure 10. Opinion responses of gaming experience and usability.

6.1 Limitations and Validation Constraints

While the study successfully demonstrated a statistically significant improvement in knowledge acquisition immediately following the use of the serious game, it is important to acknowledge certain limitations and constraints that influence the generalizability and completeness of the validation.

Furthermore, the small sample size (n=14) limits the generalizability of the findings. While the statistically significant result is strong for this group, larger and more diverse participant groups are needed to validate the game's effectiveness across broader educational contexts.

While an increase in the class average for a subsequent programming assessment (from 5.0 to 8.0 in previous semesters) suggests a positive effect, a dedicated follow-up retention test

would be necessary to formally confirm that the knowledge acquired through the game was durable. The current validation measured immediate learning gains by comparing pre-test and post-test scores, demonstrating a clear knowledge improvement after playing the game. However, the study does not include data on the long-term retention of the learned concepts. Future work could address this by including a delayed post-test.

The experimental protocol allotted only 50 minutes for students to experience the game. The short duration may explain why students did not fully utilize the tutorial for the turn-based battle system, leading to disagreements regarding the ease of understanding game mechanics (Q12, Q13, Q14). This difficulty learning the battle system may have frustrated the experience, potentially explaining the dissatisfaction with the item "The time spent playing the game was enjoyable" (Q28).

The short time also limited the students' ability to explore the content fully, leading to suggestions for deeper exploration of the pattern and more examples. Future validation could allocate more time for gameplay, especially since the pre/post-tests and MEEGA+ application will not be necessary in subsequent uses by the instructor.

The development of the game, Relics of Chaos, relied on RPG Maker MZ. This tool, while accessible for creating a narrative-driven RPG without extensive programming, imposed limitations on the types of instructional tasks that could be incorporated. Despite the initial goal of using error correction tasks for the "applying" cognitive level (Objective 4), the tool's limitations required these tasks to be adapted back to multiple-choice questions. This constraint may have restricted the depth of practical application and problem-solving skills the students could demonstrate within the game environment. The study noted that incorporating new types of challenges, such as code completion (as suggested by students), would require the acquisition of plugins for RPG Maker. This highlights that the current version of the game's validation was conducted within the constraints of its default functionality.

7. Conclusions

This work demonstrated the development of an RPG game using RPG Maker to introduce students to the Visitor design pattern. While the results showed statistically significant learning gains ($p < 0.01$), we acknowledge that the small sample size ($n=14$) limits the generalizability of these findings. Future studies with larger and more diverse participant groups are needed to validate the game's effectiveness across broader educational contexts. In the literature, only two card games were found to have been used to assess the application of various patterns after teaching them. RPG Maker was chosen because instructional tasks did not require coding, despite being in the programming field. A classroom experiment was conducted to validate the game, using pre and post-tests to evaluate knowledge acquisition, as well as using the MEEGA+ instrument to assess game quality.

Knowledge tests showed statistical significance, demonstrating the game's relevance in pattern learning. Students highlighted gaps related to challenges in the game, suggesting the inclusion of other types of tasks, which can be achieved through the acquisition of plugins for RPG Maker. As the teacher will no longer need the pre and post-tests, as well as the use of MEEGA+, in the upcoming semesters, he can allocate more time to gameplay. This includes suggesting that students go through the tutorial that teaches how to use the turn-based battle system.

The game's storyline allows for continuity in the story. Therefore, future work aims to incorporate other patterns into the game, as well as new types of instructional tasks as suggested by the students.

A Portuguese-language version of the game is available for download on Windows 64Bits at https://www.udesc.br/ceavi/gamelab/tccs/jvargasnolli. All screenshots and examples in this paper are presented in English for clarity.

Acknowledgments

The authors would like to thank the Design Patterns class of the Bachelor's in Software Engineering at the University of Santa Catarina State (UDESC) for evaluating the game. We are also grateful to the GameLab team for the pilot testing. This work was supported by **FAPESC** through the infrastructure grant for the Information Technology Engineering and Development Research Group (Grant No. 2023TR000246).

References

[1] E. Gamma, R. Helm, R. Johnson, and J. Vlissides, *Design Patterns: Elements of Reusable Object-Oriented Software*. Reading, Massachusetts: Addison-Wesley Professional, 1994.

[2] O. Astrachan, G. Berry, L. Cox, and G. Mitchener, "Design Patterns: An Essential Component of CS Curricula," *ACM SIGCSE Bull.*, vol. 30, no. 1, pp. 153–160, 1998, doi: https://doi.org/10.1145/274790.273182.

[3] P. V. Gestwicki, "Computer Games as Motivation for Design Patterns," in *Special Interest Group on Computer Science Education*, 2007, pp. 233–237, doi: https://doi.org/10.1145/1227504.1227391.

[4] P. N. Mustaro, L. Silva, and I. F. Silveira, "Using Games to Teach Design Patterns and Computer Graphics," in *Instructional Design: Concepts, Methodologies, Tools and Applications*, I. Global, Ed. 2011, pp. 173–191.

[5] N. Pillay, "Teaching Design Patterns," in *Annual Conference of the Southern African Computer Lecturers' Association*, 2010, pp. 1–4.

[6] A. De Gloria, F. Bellotti, and R. Berta, "Serious Games for education and training," *Int. J. Serious Games*, vol. 1, no. 1, 2014, doi: https://doi.org/10.17083/ijsg.v1i1.11.

[7] C. E. Catalano, A. M. Luccini, and M. Mortara, "Guidelines for an effective design of serious games," *Int. J. Serious Games*, vol. 1, no. 1, 2014, doi: 10.17083/ijsg.v1i1.8.

[8] H. P. Pontes, "Desenvolvimento de jogos no processo de aprendizado em algoritmos e programação de computadores," in *SBGames*, 2013, pp. 220–228.

[9] P. Piccolo and E. Guerra, "Using a card game for teaching design patterns," in *11th Latin-American Conference on Pattern Languages of Programs*, 2016, pp. 1–10.

[10] L. E. González-Castaño, S. V. Marroquín-Soto, G. V. Maturana-González, and R. A. Manjarrés-Betancur, "Coffee Challenge: Un juego para la enseñanza de patrones de diseño de software," *Rev. Politécnica*, vol. 17, no. 33, pp. 34–46, 2021, doi: http://dx.doi.org/10.33571/rpolitec.v17n33a3.

[11] L. W. Anderson, D. R. Krathwohl, and B. S. Bloom, *A taxonomy for learning, teaching, and assessing: A revision of Bloom's taxonomy of educational objectives*. New York: Addison Wesley Longman, Inc, 2001.

[12] C. C. Selby, "Relationships: Computational Thinking, Pedagogy of Programming, and Bloom's Taxonomy," in *10th Workshop in Primary and Secondary Computing Education*, 2015, pp. 1–8, doi: https://doi.org/10.1145/2818314.2818315.

[13] C. Zhang and D. Budgen, "A survey of experienced user perceptions about software design patterns," *Inf. Softw. Technol.*, vol. 55, no. 5, pp. 822–835, 2013, doi: https://doi.org/10.1016/j.infsof.2012.11.003.

[14] T. Pati and J. H. Hill, "A survey report of enhancements to the visitor software design pattern," *J. Softw. Pract. Exp.*, vol. 44, no. 6, pp. 699–733, 2014, doi: https://doi.org/10.1002/spe.2167.

[15] S. Rogers, *Level up: The guide of great video game design*. West Sussex: John Wiley & Sons, 2010.

[16] W. L. Schmit and Y. Moro, "Introduction to the study of analog role playing games," *Tsukuba Psychol. Res.*, vol. 52, pp. 47–58, 2016.

[17] C. Knorr and B. Zinn, "Role-Playing in Teacher Education with InCoLearn and its Qualitative Usability," *Int. J. Serious Games*, vol. 11, no. 1, pp. 25–44, 2024, doi: https://doi.org/10.17083/ijsg.v11i1.681.

[18] P. E. F. Pereira, "Construindo narrativas: o RPG 'role playing game' no ensino de história," in *II Congresso Nacional de Educação*, 2015, no. 1.

[19] M. van Bekkum, M. de Boer, F. van Harmelen, A. Meyer-Vitali, and A. ten Teije, "Modular design patterns for hybrid learning and reasoning systems: a taxonomy, patterns and use cases," *Appl. Intell.*, vol. 51, no. 9, pp. 6528–6546, 2021, doi: https://doi.org/10.1007/s10489-021-02394-3.

[20] R. Lartigue, Jonathan W. Chapman, "Comprehension and application of design patterns by novice software engineers: An empirical study of undergraduate software engineering and computer

science students," in *ACMSE 2018 Conference*, 2018, pp. 1–10, doi: https://doi.org/10.1145/3190645.3190686.
[21] M. Cho, Samuel Sungmin Caporusso, Nicholas Doyle, "P4: Principles, Patterns, Practices, and Projects for Effective Software Engineering Education," in *2024 47th MIPRO ICT and Electronics Convention (MIPRO)*, 2024, pp. 1289–1294, doi: https://doi.org/10.1109/MIPRO60963.2024.10569934.
[22] A. Vahldick, P. Schoeffel, and P. Moser, "Metodologia de Ensino de Padrões de Projeto Baseado no Modelo 4C/ID," in *VIII Congresso Brasileiro de Informática na Educação*, 2019, pp. 397–406, doi: https://doi.org/10.5753/cbie.wcbie.2019.397.
[23] M. A. Yurtsever and E. Tuzun, "Teaching Design Patterns Using Interactive Methods," in *55th Hawaii International Conference on System Sciences*, 2022, pp. 941–949, doi: https://doi.org/10.24251/HICSS.2022.116.
[24] T. Reischmann and H. Kuchen, "An interactive learning environment for software engineering design patterns," in *18th Koli Calling International Conference on Computing Education Research*, 2018, pp. 1–2, doi: https://doi.org/10.1145/3279720.3279896.
[25] Z. Azimullah, Y. S. An, and P. Denny, "Evaluating an interactive tool for teaching design patterns," in *22nd Australasian Computing Education Conference*, 2020, pp. 167–176, doi: https://doi.org/10.1145/3373165.3373184.
[26] M. A. Gómez-Martín, G. Jiménez-Díaz, and J. Arroyo, "Teaching design patterns using a family of games," *ACM SIGCSE Bull.*, vol. 41, no. 3, pp. 268–272, 2009, doi: https://doi.org/10.1145/1595496.1562960.
[27] R. Haque, A. Salmani, and M. Raessinejad, Niyousha Moshirpour, "Effectiveness of Hackathons in Software Engineering Education," in *2022 IEEE Frontiers in Education Conference (FIE)*, 2022, pp. 1–8, doi: https://doi.org/10.1109/FIE56618.2022.9962527.
[28] K. Claypool and M. Claypool, "Teaching software engineering through game design," *ACM SIGCSE Bull.*, vol. 37, no. 3, pp. 123–127, 2005, doi: https://doi.org/10.1145/1151954.1067482.
[29] M. A. Jalil and S. A. M. Noah, "The Difficulties of Using Design Patterns among Novices: An Exploratory Study," in *International Conference on Computational Science and its Applications*, 2007, pp. 245–250, doi: https://doi.org/10.1109/ICCSA.2007.32.
[30] G. Petri and C. G. von Wangenheim, "A Method for the Evaluation of the Quality of Games for Computing Education," 2019, doi: https://doi.org/10.5753/cbie.wcbie.2019.951.
[31] Newzoo, "How consumers engage with games today," Amsterdam, 2024.
[32] J. McGonigal, *Reality Is Broken: Why Games Make Us Better and How They Can Change the World*. New York: Penguin Books, 2011.
[33] J. Hamari, D. J. Shernoff, E. Rowe, B. Coller, J. Asbell-clarke, and T. Edwards, "Challenging games help students learn: An empirical study on engagement , flow and immersion in game-based learning," *Comput. Human Behav.*, vol. 54, pp. 170–179, 2016, doi: https://doi.org/10.1016/j.chb.2015.07.045.

Appendix A – Complete questionnaire

Below are listed the ten multiple-choice questions used in the tests. The correct answer is indicated in bold for each question.

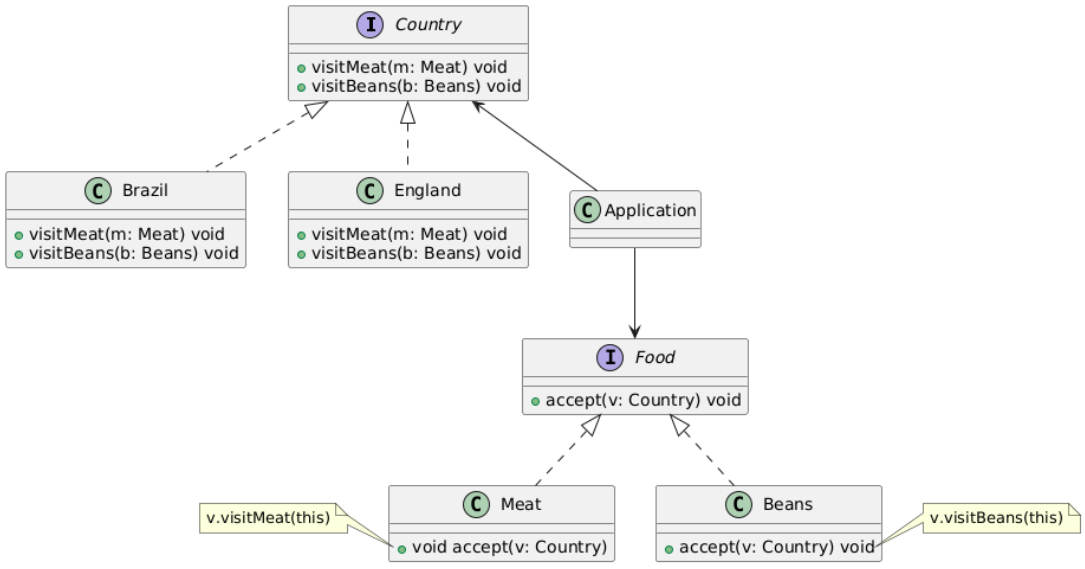
1) Which of the following is an appropriate example of using the Visitor design pattern?
a) **An inventory management system that needs to calculate the total price of all products over a specific time period.**
b) A login system that verifies user credentials and grants access to specific resources.
c) A network packet routing system that forwards packets between different nodes.
d) A text editor that allows users to apply different formatting styles (bold, italic, underline) to a document.

---

2) What is the role of Visitor objects in this pattern?
a) Define the structure of elements to be visited.
b) **Perform specific operations on each visited element.**
c) Encapsulate each element to be visited.
d) Establish communication between elements and the Visitor pattern.

---

3) What are the two main phases, and their sequence, to implement the Visitor design pattern?
a) Create a class hierarchy to represent visitor classes, and in the visited elements classes define methods responsible for relating each concrete visitor.
b) **Create a class hierarchy to represent visited elements, and in the visitor classes define methods responsible for executing operations for each visited object.** *(correct answer)*
c) Create a class hierarchy to represent visited elements, and in the visitor classes define how the method responsible for executing each visited object will be implemented.
d) Create a class hierarchy to represent visitor classes, and in the visited elements classes define methods responsible for relating each visitor.



4) Based on the diagram above, select the correct statement:
a) The Food class represents a visitor class.
b) The Brazil class represents an element class.
c) **The England class represents a visitor class.**
d) The Meat and Beans classes implement a visitor interface.

---

5) Which of the following is a disadvantage of implementing the Visitor design pattern?
a) It breaks the visibility of the elements' states for the visitor to access them.
b) When adding new visitors to implement additional operations, existing elements or previous visitors are not modified.
c) The same visitor class has multiple versions of the same behaviour.
d) **Every time a class is added or removed from the element hierarchy, all visitors must be updated.**

---

6) What is the advantage of the Visitor design pattern when adding new operations to a class hierarchy?
a) Avoids the need to create specific visitor classes for each new operation.
b) Simplifies adding new visitable element classes to the hierarchy.
c) **Makes it easier to modify existing visitable element classes without affecting current visitors.**
d) Reduces the complexity of operations to be implemented in each visitable element.

---

7) Which of the following is an advantage of the Visitor design pattern?
a) It breaks the visibility of the elements' states for the visitor to access them.
b) **When adding new visitors to implement additional operations, existing elements or previous visitors are not modified.**
c) The same visitor class has multiple versions of the same behaviour.
d) Every time a class is added or removed from the element hierarchy, all visitors must be updated.

---

8) What is the role of Element objects in the Visitor design pattern?
a) Define the structure of elements to be visited.
b) Perform specific operations on each visited element.
c) **Encapsulate each element to be visited.**
d) Establish communication between elements and the Visitor pattern.

---

9) What are the required steps to implement a new Element in the Visitor pattern?
a) **Implement the Element interface and add a new method to all Visitor classes.**
b) Implement the Visitor interface and add a new method to all Element classes.
c) Implement the Element interface and all Visitor classes remain unchanged.
d) Implement the Visitor interface and all Element classes remain unchanged.

---

10) Mark (X) the Java code segment that correctly implements the Visitor design pattern:

```
public interface Element {
    void accept(Visitor visitor);
}

public interface Visitor {
    void visit(ConcreteElement element);
}

public class ConcreteElement implements Element {
    public void accept(Visitor visitor) {
        visitor.visit(this);
    }
}

public class ConcreteVisitor implements Visitor {
    public void visit(ConcreteElement element) {
    }
```

```
}
```

```
public interface Element {
    void accept(ConcreteVisitor visitor);
}

public interface Visitor {
    void visit(Element element);
}

public class ConcreteElement implements Element {
    public void accept(ConcreteVisitor visitor) {
        visitor.visit(this);
    }
}

public class ConcreteVisitor implements Visitor {
    public void visit(Element element) {
    }
}
```

```
public interface Element {
    void accept(Visitor visitor);
}

public interface Visitor {
    void visit(ConcreteElement element);
}

public class ConcreteElement implements Element {
    public void accept(Visitor visitor) {
        this.visit(visitor);
    }
}

public class ConcreteVisitor implements Visitor {
    public void visit(ConcreteElement element) {
    }
}
```

```
public interface Element {
    void accept(Visitor visitor);
}

public interface Visitor {
    void visit(Element element);
}

public class ConcreteElement implements Element {
    public void accept(Visitor visitor) {
        visitor.visit(this);
    }
}

public class ConcreteVisitor implements Visitor {
    public void visit(Element element) {
    }
}
```