

# Using Neoconstructivist-Based Serious Games to Teach Programming Fundamentals

Roberto B. Feitosa, José F. M. Netto

Programa de Pós-graduação em Informática (PPGI) - Universidade Federal do Amazonas (UFAM)

roberto.feitosa@icomp.ufam.edu.br, jnetto@icomp.ufam.edu.br

**Abstract.** *Programming education is a fundamental component of computer science courses, with the knowledge gained in introductory programming subjects playing a critical role throughout a student's academic journey.*

**Introduction:** *Programming skills have become increasingly important today, yet many students struggle with their first contact with programming due to the lack of prior foundational knowledge. One promising solution is the use of serious games. However, these games must be carefully designed with clear educational objectives. In this context, the neoconstructivist theory is adopted, as it emphasizes active learning while also considering social and neurological aspects of knowledge construction.* **Objective:** *This study explores the use of Serious Games as a tool for teaching Programming Fundamentals, grounded in Neoconstructivist theory, to support the teaching and learning process in programming education.* **Methodology:** *The methodology of this study was shaped by the Interad approach for organizing the development of a game-based Learning Object, including the creation of a Game Design Document (GDD) and data collection for evaluation using the MEEGA+ model, an assessment method tailored for educational games.* **Expected Results:** *It is expected that neoconstructivist-based serious games will prove to be effective tools for helping students learn complex and abstract programming concepts in a more engaging and motivating way compared to traditional teaching methods.*

**Keywords:** *Serious Games, Educational Games, Programming Education, Neoconstructivism.*

## 1. Introduction

### 1.1. Context

Programming skills have become increasingly important in today's context. If we observe carefully, we will notice that we are surrounded by computers, mobile phones, watches, televisions, all of which can be considered computing devices, but not everyone understands the logical functioning of these devices.

The first contact with programming can be very challenging for students, as they usually lack the prior knowledge necessary to support the acquisition of this new skill knowledge that is not typically developed during basic education. When students face significant difficulty in learning specific topics, it often leads to frustration and contributes to higher failure and dropout rates.

A study conducted at the Federal University of Ouro Preto by Dainese and Garbin (2024) shows that between 2015 and 2022, only 46.59% of students enrolled in

the course Computer Programming I passed, while 53.41% either failed or withdrew. Although these results refer to a single institution, they may reflect challenges encountered in introductory programming courses.

According to Silva et al. (2023), it is possible to identify students who need special attention through certain indicators, but it is difficult for instructors to do so, especially when teaching large classes. In such situations, instructors often lack the time to provide the necessary support to struggling students, making it essential to find tools that can assist in this process.

According to Severgnini (2018), Serious games are games designed primarily for purposes other than entertainment, such as educational and training. They integrate game mechanics with instructional content to promote engagement, active learning and skill development.

## **1.2 Justification**

Serious games can enhance student engagement by using elements such as narrative, challenges, and progression systems to present programming concepts in a gradual and structured way. However, they must be carefully designed with clear educational objectives. Neoconstructivism emphasizes active knowledge construction through meaningful experiences and social interaction. It was selected as the pedagogical foundation of this work because its principles align with the game's design, which promotes exploration, experimentation, feedback, and progressive learning of programming concepts.

## **1.3 General and Specific Objectives**

The objective of this study is to evaluate the application of neoconstructivist theory through serious games developed by incorporating its concepts into game elements aimed at teaching programming.

To achieve the general objective, the following specific objectives must be accomplished:

- To prototype a game that applies neoconstructivist theoretical concepts experienced by students during gameplay;
- To design a pedagogical intervention for the use of the game and data collection for evaluation;
- To analyze the collected data in order to understand the impact of using a serious game with these characteristics in the teaching of programming.

The last two specific objectives belong to the broader research project and are outside the scope of the present paper.

## **2. Methodology**

In this project, we chose to use the Interad method, which was designed for the development of Learning Objects. This method was selected because it provides development stages focused on interface design and user interaction, both of which are

fundamental elements in the design of educational games. A series of stages from the Interad method must be followed, in which each stage will have the following objective:

**Table 1. Project Stages According to the Interad Method**

| Stage           | Objective                                                                                                            |
|-----------------|----------------------------------------------------------------------------------------------------------------------|
| Comprehension   | Identify and gather requirements to ensure the game is designed with mechanics that reflect neoconstructivist theory |
| Preparation     | Begin the development of a Game Design Document (GDD) based on the initial requirements                              |
| Experimentation | Refine the initial aspects of the GDD by adding more detail and making necessary adjustments to the project scope    |
| Elaboration     | Develop the prototype based on the GDD, continuously making adjustments and documenting them                         |
| Presentation    | Finalize the prototype, conduct testing, collect data, and analyze the results                                       |

## 2.1. Comprehension

The project aims to define the content and presentation of a game focused on teaching basic programming concepts, such as variables, conditional structures, loops, and decision-making. It follows active learning principles, incorporating meaningful context, progressively increasing challenges, immediate feedback, and attention to emotional aspects to enhance motivation. The target audience consists of early-year undergraduate students in computing, information technology, or related fields.

Based on the identified needs, it is possible to classify the project's characteristics in order to clearly define its requirements, which are as follows:

- Theme: Develop a serious game applying neoconstructivist concepts to support the teaching of basic programming concepts, ensuring that students feel motivated to achieve the objectives.
- Target Audience: Undergraduate students in their early years enrolled in courses that include programming disciplines.
- Pedagogical Objectives: Promote the teaching of programming concepts such as variables, conditional structures, decision structures, and loops through games, aiming to facilitate the learning of abstract concepts. The goal is to create an environment where students engage in active and motivating learning with immediate feedback, reducing dropout rates.
- Educational Context: The serious game should be focused on the programming domain and can be used during classes in both classroom and laboratory settings, as well as outside the classroom, either in person or remotely.
- Student Needs: Programming courses require logical reasoning and computational thinking skills, which are not always developed during basic

education. Therefore, concepts must be presented clearly and didactically. Careful attention should be given to the progression of challenge difficulty, and upon completing each task, feedback should be immediate so that students remain motivated and can reflect on their solutions.

## **2.2. Preparation**

At this stage of the project, according to Passos (2011), with all data obtained in the previous stage we can start to analyze and synthesize these data. The focus shifts to the game's conception. To manage ideas and development progress, a document called a Game Design Document (GDD) was chosen for record-keeping. There are several ways to create a GDD, but for this project, an incremental model with version history tracking will be used. With this type of GDD, in addition to organizational benefits, it is expected that it will also make it possible to understand the project's origin and how it evolved until the final product delivery. The GDD can be accessed at the following link:

 SBGames Extra.pdf

In this project, it is essential that the student takes an active role and constructs their own knowledge. Therefore, the student needs something they can manipulate and shape according to their preferences. Initially, providing an in-game notebook appears to be an interesting way to meet this requirement. As the game progresses, learned concepts can be recorded in this notebook along with the student's own notes, which can later be consulted to solve challenges. In this way, this editable notebook becomes the main game mechanic.

## **2.3. Experimentation**

One of the needs of this project is to establish a meaningful context for the game. We require a setting and a narrative that capture students' interest while aligning with the proposed mechanics.

Another concern is how to establish a progressive level of difficulty. The game must remain challenging to prevent students from becoming bored, while also being fair by not requiring knowledge that has not yet been introduced.

Upon completing a challenge, students should receive immediate feedback: congratulating them when they perform excellently, encouraging reflection if their solution is not optimal, and providing motivation and guidance on where to find the necessary knowledge if they are unable to complete the task.

## **2.4. Elaboration**

This stage will be used to begin the development of the prototype, therefore, it will be necessary to create concept art and user interfaces to guide the development process.

## **2.5. Presentation**

At this stage, the visual identity is defined, interface development activities are completed, and the project itself is finalized. It is now necessary to determine how testing and data collection will be conducted. For data collection, the evaluation model developed by Petri et al. (2019), known as MEEGA+, will be used after testing the prototype. The questionnaire contains 35 questions, however for this project, 20 were

selected as they were considered sufficient and relevant within the project's context. The questionnaire can be accessed at the following link:  SBGames Extra.pdf

The 20 selected questions are relevant to the project because they assess how students feel about the game, whether they found it interesting and motivating, and whether they learned something relevant to the course. With this data, it will be possible to better understand how the game impacted students' learning.

### 3. Expected Results

It is expected that this study will provide data demonstrating that serious games based on neoconstructivist methodology are effective in supporting the teaching of programming fundamentals in a motivating and engaging way, through playfulness and gamification, which are inherent elements of games.

### 4. Conclusions

This paper presented the design of a serious game based on Neoconstructivist principles for teaching programming fundamentals, including the adoption of the Interad methodology, the development of the Game Design Document (GDD), and the definition of the evaluation strategy using the MEEGA+ model.

The main contribution of this paper is the proposal of a structured approach for integrating neoconstructivist concepts into the design of educational games for programming education, providing a foundation for future implementations and empirical studies.

Future work includes completing the implementation of the game, conducting the planned pedagogical intervention with undergraduate students, collecting evaluation data through the MEEGA+ model, and analyzing the educational impact of the proposed approach.

### References

- Dainese, C. A. and Garbin, T. R. (2024) "Taxa de aprovação, reprovação e trancamento na disciplina Linguagem de Computadores I em cursos de Engenharia da UFOP".
- Dantas, D. O., Nobre, I. A. M. and Passos, M. L. S. (2015) "Teddy Racer: Lógica de Programação e Ludicidade".
- Holanda, W. D. and Coutinho, J. C. S. (2022) "Word Prog: Um jogo educacional para aprendizagem de conceitos básicos de programação".
- Kazimoglu, C. (2020) "Enhancing Confidence in Using Computational Thinking Skills via Playing a Serious Game: A Case Study to Increase Motivation in Learning Computer Programming".
- Krouska, A., Troussas, C., Voutos, Y., Mylonas, P. and Sgouropoulou, C. (2023) "Optimizing Player Engagement in an Educational Virtual Game through Fuzzy Logic-based Challenge Adaptation".

Madureira, J. S. and Schneider, H. N. (2021) “Gamificação no ensino de programação de computadores em turmas do ensino médio: uma experiência com o software Kahoot”.

Natal, M. E. C., Barbosa, B. A., Hernandez, J. C., Much, B. S., Bigolin, M., Silva, S. J. R., Silva, C. B. and Carvalho, L. F. B. (2018) “Tri-Logic: Um ambiente gamificado como ferramenta de auxílio ao ensino de aprendizagem de lógica de programação”.

Panegalli, F. S., Bernardi, G. and Cordenonsi, A. Z. (2019) “Super Mario Logic: um jogo sério para auxiliar no processo de ensino e aprendizagem de lógica de programação”.

Passos, P. C. S. J. (2011) “Interad: uma metodologia para design de interface de materiais educacionais digitais”.

Sajana, A., Bijlani, K. and Jayakrishnan, R. (2015) “An interactive serious game via visualization of real-life scenarios to learn programming concepts”.

Sakar, S. P., Saker, B. and Hossain, S. A. (2016) “Cross Platform Interactive Programming Learning Environment for Kids with Edutainment and Gamification”.

Severgnini, L. F. (2018) “Alice e o Mistério dos Algoritmos: um serious game como ferramenta de aprendizagem de lógica de programação para crianças”.

Silva, C. E. P., Solano, J. L. S., Santos, A. G. and Reis, J. C. S. (2023) “Previsão de reprovações em disciplinas introdutórias de programação: um estudo em um ambiente de correção automática de códigos”.

Silva, R. R., Rivero, L. and Santos, R. P. (2021) “ProgramSE: Um jogo para aprendizagem de conceitos de lógica de programação”.

Sirikong, N. and Visutsak, P. (2024) “Gamifying Python: Enhancing Learning Outcomes and Motivation through a Game-Based Approach”.

Tacouri, H. and Nagowah, L. (2021) “Code Saga – A Mobile Serious Game for Learning Programming”.

Tolfo, A. R. and Silveira, S. R. (2024) “PC Transit: Jogo Mobile para Estimular o Desenvolvimento do Pensamento Computacional”.

Wong, Y. S., Yatim, M. H. M. and Hoe, T. W. (2017) “A Proprietary Game-Based Learning Mobile Game to Learn Object-Oriented Programming – Odyssey of Phoenix”.