International Journal of Serious Games

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

Article

Bayesian Knowledge Tracing Implemented in a Telecommunications Serious Game

Halatedzi Nedombeloni[1], Reolyn Heymann[2] and Japie Greeff[23]

[1]*Electrical and Electronic Engineering Science, University of Johannesburg, Johannesburg, South Africa;* [2]*Optentia Research Unit, North-West University, Vanderbijlpark, South Africa;* [3]*National Institute for Theoretical and Computational Sciences (NITheCS), Stellenbosch, South Africa,*
halatedzi100@gmail.com; Reolyn.Heymann@nwu.ac.za; Japie.Greeff@nwu.ac.za

Keywords:
Serious games
Knowledge tracing
Procedural generation
Synthetic data generation
3-Dimensional

Received: February 2024
Accepted: June 2024
Published: June 2024
DOI: 10.17083/ijsg.v11i2.738

Abstract

The University of Johannesburg has integrated serious games into its teaching, exemplified by Codebreakers, a 2D game teaching information theory. While successful, Codebreakers lacked personalisation and used a criticised assessment method based on answer streaks. Knowledge tracing algorithms, known for their effectiveness in intelligent tutoring systems, were considered to address these limitations. This led to the research question: "Can a new serious game be designed, incorporating knowledge tracing algorithms to deliver personalised learning experiences in telecommunications education?" In response, an escape-themed serious game was developed, integrating Bayesian Knowledge Tracing as a statistical student model for personalised learning. This innovative approach combines free-roam gameplay with tailored educational content, significantly advancing serious game design. While primarily aimed at enhancing Codebreakers, this new game contributes substantially to serious game theory by successfully implementing personalised learning within an engaging format. The project showcases the potential of knowledge tracing algorithms in creating adaptive, student-centered learning experiences within the context of educational games.

1. Introduction

Serious games are tools that add playfulness to the classroom to make the educational experience less daunting and more enjoyable. Researchers suggest that, on average, learners can learn more in situations where they are used than in cases where they are not [1], [2]. This is because the model of serious games is closer to what modern-day beliefs about education consider effective learning [1], such as being structured by specific objectives, providing immediate feedback, delivering the opportunity to apply lessons learned from past experiences, and allowing interaction [3].

Because of this, the University of Johannesburg (UJ) has increasingly incorporated serious games into its pedagogical approach, specifically within the telecommunications classroom. A notable example is the game Codebreakers, introduced by [4]. This game aimed to elucidate the fundamentals of information theory for a final-year telecommunications module at the Faculty of Engineering and the Built Environment (FEBE). Codebreakers uses 2D or tile-based graphics to illustrate concepts such as the Entropy of messages and Error Control Coding (ECC), with particular emphasis on Hamming Codes, a linear error correction method designed to detect up to two-bit errors and correct single-bit errors introduced in messages transmitted between nodes in a network [5], among others. The game was structured so that learners had to navigate through various game worlds, where they faced content-based puzzles and had to solve them to move on to the next level, which addressed a different topic in the learning content.

A significant advantage of the game is that it was developed in collaboration with students. This allows them to contribute features they believe would enhance their learning experience. This collaborative effort proved successful, with the game mainly receiving positive acclaim from students for several years after its 2016 release. Figure 1 shows the graphical style of the game world of Codebreakers.

Figure 1. Depiction of Codebreakers' game world and the player's avatar within the game [4].

Though the gameplay was well received, the model used to assess students' understanding was not optimal. For instance, Codebreakers' assessment model assigned scores based on the student's longest out of 10 streaks of correct answers; eight consecutive correct answers would equate to an 80% score for the topic. A setback to zero followed each incorrect answer, requiring students to rebuild their success streak anew.

This method faced criticism from some students who expressed dissatisfaction with the assessment approach, stating that receiving an incorrect answer didn't necessarily indicate a lack of understanding. Additionally, they argued that achieving a perfect or near-perfect score shouldn't be the sole indicator of comprehension. These qualitative insights highlight a perceived discrepancy between the evaluation method and actual comprehension. Despite lacking quantifiable data, these informally collected perspectives offered valuable insights into the limitations of the assessment approach from the student's viewpoint. Because of this, the students vocalised a requirement for personalisation in the game. The criticised assessment method, relying on streaks and setbacks, often fails to capture the nuances of individual learning styles and preferences. In contrast, personalised games adapt to each student's skill level, interests, and learning pace [6]. These games offer tailored challenges and feedback, allowing students to progress at their rate and explore concepts more engagingly. By catering to students' needs, personalised games provide a more dynamic and practical approach to assessment, fostering deeper learning and retention of material. This allows students to engage with content that matches their skill level and possibly improve the game's reception. Since personalisation means adapting to someone's skill level, a more refined technique of assessment that can be tuned to students' cognitive abilities and needs is essential to avoid imposing a generic, one-size-fits-all solution. One that predicts students' mastery state and adapts the game's instructional content to it. This would ensure a more accurate and personalised learning experience, addressing each student's unique learning needs and preferences.

Another game was suggested in [17] to address the limitations of Codebreakers. One that would use a better assessment and cognitive model to offer a more personalised learning experience. It was also recommended that this successor improve upon the simplistic graphics used in Codebreakers using more realistic 3D environments. The game should also dive into theoretical content from the outset, paralleling the sequence presented in classroom instruction.

Although many algorithms for personalisation have previously been demonstrated in the literature, knowledge tracing (KT) ones are more well-tested and standardised in intelligent tutoring systems (ITS) for this very purpose, and their simplicity makes them desirable for integration into this project. This led us to formulate the following research question for developing the new game: Can a new serious game be developed, incorporating KT algorithms, to provide personalised learning experiences in telecommunications education? Given the substantial effort required to create a game encompassing the full scope of information theory taught at the final-year electrical engineering level, our game will focus exclusively on Hamming Codes, a type of ECC to quickly demonstrate the intended features. A successful implementation would yield a game that adapts to learners' knowledge state and gives them plentiful practice problems until they reach mastery. Mastery, in this context, refers to achieving a deep understanding and proficiency in the subject matter, where learners can confidently apply their knowledge in various contexts and quickly solve complex problems. This would allow learners to engage better and grasp the content than Codebreakers, as the learning content would be attuned to their capabilities.

While the new game's primary objective was to improve Codebreakers, it's important to note that several aspects of its design represent novel contributions to the field of serious games. Therefore, a formal framework delineating these design decisions will be presented later in the discussion, providing insight for future endeavours in serious game development.

2. Research Design

At its core, this project embodies a research and development (R&D) initiative. The research component entails gathering data to extract insights into the determinants of success in serious games. We recognised the wealth of successful serious games from which we can draw valuable lessons.
The development aspect focuses on instantiating these insights into a working product.
To address the objectives of this project, we followed the typical R&D research design, which includes:

2.1 Defining the problem

Here, we noted the needs of the students and formally noted them, in this case, the need for a successor to Codebreakers that includes 3D environments, linearised learning and an engine for personalised experiences.

2.2 Specify the requirements

Here, we defined the requirements of the solution that will meet the defined needs of the students, such as considering KT algorithms for personalised learning experiences and so forth.
Background research

In this phase, we conducted extensive background research through a comprehensive literature review. Our goal was to understand the existing work in this domain and position our research within it. This informed the design of our serious game and enabled us to identify gaps in the literature. Our exploration included addressing questions such as:

- What is a serious game [7]?
- How does it differ from a recreational video game [8]?
- Does adding educational content to a video game make it a serious game [8]?
- How are serious games built [9]?

In this phase, we also investigated the different algorithms previously used for personalisation, noted their drawbacks and investigated how they could be improved to address literature gaps.

2.3 Brainstorm, evaluate, and choose solution

In this phase, information obtained from the background research was considered and filtered to select the best solution to the problem.

2.4 Develop and prototype solution

In this phase, the selected solution was implemented and prototyped using Unity as the game engine of choice.

2.5 Test solution

In this phase, the prototype was rigorously tested for correctness. This included manual playtesting to ensure the correct behaviour of the game objects and unit testing to ensure that the written scripts behaved as intended,

Because the R&D research design is iterative, the developed solution had to be further analysed to ensure it met all the requirements and behaved as expected. When it didn't, backtracking to previous steps was necessary to ensure the highest quality. Once the game was analysed to meet all the requirements, the results are communicated as in this paper. Overall, the structure of this research design is visualised in Figure 2.

Figure 2. R&D research design followed to meet the requirements of this project [10].

3. Similar Work

3.1 Personalization in Serious Games

Personalising the content of serious games or game-based learning systems to adapt to learners' individual needs and capabilities is a long-standing requirement that has spurred numerous solutions. For instance, some studies have taken a machine learning (ML) and artificial intelligence (AI) approach to tackle this issue, like the authors in [11] who designed an adaptive game for cognitive and psychosocial rehabilitation. The game named CogniChallenge adapts to the learner's capabilities by using agents, which are AI models trained to play the game. These models are trained to cooperate with or compete against the learner. The amount of cooperation decreases as the player improves; similarly, the degree of competition increases. This ensures the player is always engaged and obtains maximum rehabilitation.

Other studies have preferred a more novel approach, such as the study in [12], where the authors demonstrate a framework that dynamically adapts the game objects and activities to personal learning objectives. This is done to avoid a linear learning sequence, as the framework can infer the next best activity the learner should complete.

Although many studies have demonstrated extensive methods for solving the personalisation problem in serious games, the limitation is the kind of games to which these methods have been applied. For example, most, if not all, of the serious games that incorporate personalisation apply this method to static-style serious games, where the player faces a static environment and is given learning material and questions, such as in [11]-[13]. There is a lack of serious games incorporating personalisation to free-to-roam progressive games like the Codebreakers above, let alone 3D ones. Therefore, successfully implementing a personalization-capable successor to Codebreakers has the potential to contribute significantly to serious game design.

3.2 Knowledge tracing in serious games

KT algorithms have also previously been implemented in serious games. However, their capability for personalisation has rarely, if ever, been applied in this context. Most studies have focused instead on their ability to build student models or predict future student performance. For instance, the study in [14] demonstrates a deep learning enhanced knowledge tracing method known as deep knowledge tracing (DKT) for predicting students' future performance in programming serious games. These predictions are then used to provide proactive recommendations to complete the missions successfully. Similarly, the authors in [15] and [16] use Bayesian knowledge tracing (BKT), a knowledge tracing algorithm based on Bayes theorem, to build student models, which are mathematical relationships that seek to explain how students understand the learning material presented in the serious game.

3.3 Telecommunications education in serious games

Apart from Codebreakers, there has been a lot of previous work done in developing serious games for telecommunications and computer science education; for instance, the authors in [15] demonstrate three games for teaching website cryptography, search engines, and ECC, as in this paper. While initially designed for high school students, these games possess the potential to captivate any audience with an interest in computers and technology, thereby underscoring the broad applicability and effectiveness of serious games in educational contexts. Similarly, the authors in [4] and [17] demonstrate a series of 2D serious games for teaching information theory at the university level.

3.4 Contribution of study

While we have delved into related research about our project's objectives, as previously noted, there remains a notable gap in the literature regarding the demonstration of personalised serious games within a free-to-roam environment, particularly those in 3D as this lack can be seen in [11]-[13]. Furthermore, using KT algorithms as the inference engine in such games has been sparingly explored [14]-[16]. Therefore, attaining our study's objectives would represent a significant advancement in the field, showcasing a free-to-roam serious game implementing personalisation through a KT algorithm. Additionally, it holds the potential to serve as a comprehensive framework for the development of similar games, thereby laying the groundwork for future innovations. Expanding upon our background research, the following chapter provides a more thorough exploration of research fields that have endeavoured to address the challenge of personalisation. It delves into the intricacies of various approaches to solve this problem and offers a detailed examination of our selected solution.

4. Technical Background

4.1 Latent Knowledge Estimation

Modelling a student’s mastery state of a skill, a problem known as latent knowledge estimation, is a long, actively researched problem within educational data mining (EDM) [18]. EDM is a research field that applies data mining, machine learning (ML), and statistics to student data gathered from learning settings such as ITSs [19] and universities [20]. At its core, the field seeks to develop and improve upon methods for exploring this data to discover new insights about how people learn in the context of such settings [20]. The field also aims to build student models from this data that incorporate the learners' characteristics, such as their knowledge or mastery state, behaviours and motivation to learn. These models are then used to predict students’ future learning behaviour so that learning content that matches their knowledge level can be sequenced to avoid overwhelming them. This is precisely what we want to incorporate within the envisioned successor of Codebreakers.

Over the years, several researchers from the field have suggested models that can accurately predict students' knowledge states. Some of the recent works range from models that use deep-learning approaches [21]-[23], to ones that use simple statistical models [24]. The deep learning approaches are trained from data sourced from massive open online courses (MOOC) datasets such as Khan Academy and ASSISTments. [23], which makes them infeasible for this use case as our application is a moderately sized classroom. This means we wouldn't be able to collect enough data to train accurate models as these platforms have a global reach, and the datasets can encompass many student interactions.

A model favourable to our application is BKT, developed and elaborated on by [24]. BKT models use a probabilistic graphical model such as a Hidden Markov Model (HMM) and a Bayesian Belief Network (BBN). The idea behind BKT is to estimate the probability that the student has learned the learning content as they interact with the practice problems. The model increments and decreases this probability depending on the correctness of the student's response to a problem. Specifically, the BKT model forms a cognitive model from four parameters. These are:

- $P(L_0)$ is the probability that the student has mastered the learning content before the opportunity to apply the skill in a problem-solving setting. Which then becomes $P(L_n)$ after the first step.
- $P(T)$ is the probability that the learning content will be learned at each practice opportunity.
- $P(G)$ is the probability that a student guesses if they do not know the learning content.
- $P(S)$ is the probability that students will make mistakes if they know the learning content.

As the student engages with the practice problems, at steps $n>= 1$, the model updates $P(L_n)$ through its BBN, which comprises the following equations:

$$\mathrm{P}(\mathrm{L_n}) = \mathrm{P}(\mathrm{L_{n-1}}|\mathrm{Action_n}) + \left(1 - \mathrm{P}(\mathrm{L_{n-1}}|\mathrm{Action_n})\right)\mathrm{P}(\mathrm{T})$$

Where $P(L_{n-1})$ is the posterior probability of being in a learned state given the observation of the $\mathrm{n^{th}}$attempt by the student calculated as:

$$\mathrm{P}(\mathrm{L_{n-1}}|\mathrm{Correct_n}) = \frac{\mathrm{P}(\mathrm{L_{n-1}})\left(1 - \mathrm{P(S)}\right)}{\mathrm{P}(L_{n-1}(1 - \mathrm{P(S)})\ +\ (1\ -\ \mathrm{P}(L_{n-1})\ \mathrm{P(G)}}$$

If the attempt is correct, if the students' attempt is incorrect, the same is calculated as:

$$\mathrm{P}(\mathrm{L_{n-1}}|\mathrm{Incorrect_n}) = \frac{\mathrm{P}(\mathrm{L_{n-1}})\left(\mathrm{P(G)}\right)}{\mathrm{P}(L_{n-1}\ (\mathrm{P(S)})\ +\ (1\ -\ \mathrm{P}(L_{n-1})\ (1 - \mathrm{P(G)})}$$

A student is said to have mastered a skill once they obtain a $P(L_n)$ of 95%, as specified by the originators of the model [24].

The parameters of this model are learned through ML methods. One is the expectation maximisation (EM) algorithm, which iteratively updates the parameters of a statistical model such as the BKT model.

This model is favourable for our application because its HMM architecture means we can train it from minimal data, as HMMs do not need much data. It also uses simple equations to update the probability that the student has mastered the skill. This means that students and educators can understand how the model works. Its developers have also proved the model effective in educational settings [24]. However, a caveat of this model is that it is designed to operate in environments where only single skills are assessed. In the context of this model, these individual skills are known as knowledge components (KC). Because of how favourable the model is to our application, we will use it to realise the requirement of personalising students' learning experiences within the game we seek to develop.

4.2 Intelligent tutoring system

The BKT model is primarily used in the domain of ITSs; in fact, the originators of the model initially evaluated it in the context of an ITS. It makes sense to draw design inspiration from these systems as both our game and they would be using the same inference backend.

ITSs are programs that aim to provide sophisticated instructional advice on a one-on-one basis comparable to that of an excellent human tutor [25]. The most salient property of ITSs, which sets them apart from other artificial intelligence (AI) programs, is the diagnosis of the current student knowledge. [25]. Today's most popular ITS are the Cognitive Tutor developed by Carnegie Mellon University and Assessment and Learning in Knowledge Spaces (ALEKS) [26], widely used in mathematics and other subjects. ITSs have, especially recently, different architectures; however, the most popular consists of three modules that contain domain knowledge, a model of the learner's current state and pedagogical knowledge [25]. Some authors may refer to these modules using different names.

4.2.1 Knowledge Base

The knowledge base contains the domain knowledge of the field to which the ITS is applied. This is both the theoretical content and practical (practice problems) to assess students understanding.

4.2.2 Student Model

The student model represents the student's current knowledge state, usually modelled through the BKT model. However, depending on the reach of the tutor, more advanced models, such as Deep Knowledge Tracing (DKT) [21] may be used.

4.2.3 Pedagogical Module

The Pedagogical module is how the tutor delivers the learning content to the students. The author in [25] expresses that teaching can be considered a knowledge-based skill, guided by strategies and techniques selected and combined dynamically in reaction to the student's actions. And that the pedagogical module incorporates just this skill. The tutor uses this to determine the presentation method, the balance of the tutor and the student's control. [25].

4.2.4 User Interfaces

These form the core of ITSs; however, computers have recently used rich and dynamic user interfaces (UI) to improve the experience of interacting with them. UIs have become an essential part of the success of ITSs [25]. This part forms the front end or the student-facing portion of the ITS.

4.2.5 Overview of Operation

Although this doesn't apply to all ITSs, they work by drawing the practice question from the knowledge base that is best fitting to the student model knowledge state and using the pedagogical module to deliver instruction on how to best solve it through the UI, which also serves as a point of interaction for the student. This approach is known as the expert-centred approach, but some tutors use teacher and student-centred instruction approaches. Figure 3 visualises the architecture of a generalised ITS that follows the expert-centred approach.

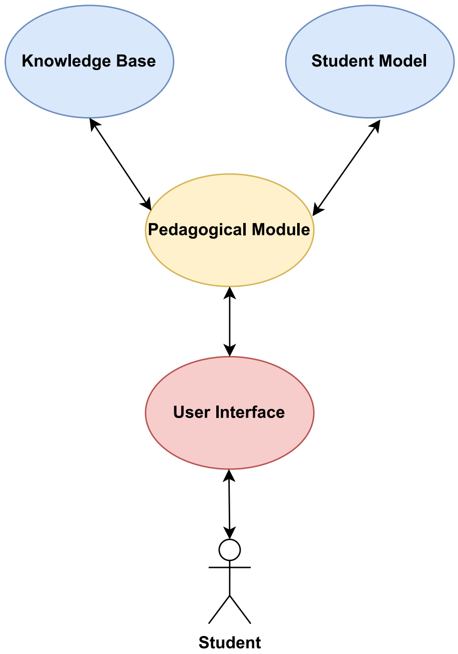

Figure 3. The architecture of a generalised ITS following the expert-centred approach [27]. The figure shows the different components within an ITS, how they connect, and the students' interaction with the ITS.

Given the simple and effective model of ITSs [25], we draw inspiration from them and seek to use a similar architecture for the game.

5. Proposed Approach

We mentioned that the game we intend to develop should offer a more personalised experience than Codebreakers by using a cognitive model such as the BKT to predict and adapt to students' mastery state so that they get practice questions that match their skill level until they reach mastery. The students also mentioned that they would prefer a game world with 3D environments as they felt this would improve the realism of the game and possibly their immersion in it. They also mentioned the game should dive into theoretical content from the onset, all in the context of Hamming Codes learning content. These requirements are similar to the architecture of an ITS; therefore, we propose the following architecture for the game's design.

5.1 Knowledge Base

We know that some students will require a significant number of practice problems before they grasp the learning content; therefore, there has to be a substantial number of them. Since we are building a game rather than an ITS, the knowledge base acts as a pool for practice problems for the students. Codebreakers had a limited number of these; hence, the streak model was used as an assessment model. However, in this case, a limited number of questions may prevent the student from reaching mastery because they would not have enough unseen questions to confirm their knowledge. We therefore propose that the knowledge base contain procedurally generated questions. Procedural generation involves creating the practice problems algorithmically rather than manually. However, in the context of Hamming Codes, an infinite number of questions cannot be created because of the difficulty. This means there has to be a finite number of questions, but there should be so many that the students cannot exhaust this question pool.

5.2 Student Model

We mentioned earlier that we seek to use the BKT model to represent student's understanding of the content within the game. However, one of the difficulties of using this model is that it is best suited to model student understanding when they attempt fine-grained KCs. The game must not try to teach Hamming Codes as a whole but instead break them down into their steps. This is not problematic, seeing that the computation of Hamming Codes requires many steps either way. A Hamming Code is a block ECC that involves adding parity bits to a binary message stream. These are added so that if an error occurs, the receiver can perform computations around these bits and determine which bit in the original message stream has an error and thus correct it. To compute a Hamming Code, students must be able to:

- Determine the number of parity bits required.
- Determine where they should be positioned in the modified message.
- Be able to compute the values of the parity bits and thus encode the Hamming Code.
- Be able to decode the Hamming Code by separating the message bits from the parity bits and determining the position of the erroneous bit, if any.

We, therefore, propose our game to teach and assess the knowledge of Hamming Codes through its KCs in the order mentioned above. Another difficulty is that the model needs to be trained from data from student interactions with a learning system. We don't have data ourselves. However, we will elaborate on a synthetic data generation method to overcome this limitation. This is important so we can test how good the correlation is after it says the student has mastered the skill and how they perform. Once a model is built, it will sequence the questions the student needs to interact with to reach mastery, giving them questions until they achieve mastery of that KC.

5.3 Pedagogical Model

Unlike an ITS, which is explicitly an educational tool, serious games must blend play and education methods to be successful [28]. We mentioned the requirement for the game to linearise the content. Insights from our literature study will help us design the game to balance play and education.

5.4 User Interface

The user interface is the visual component of the game, the worlds, and the amenities.

5.5 Overview of the game architecture

We have detailed our chosen architecture for the game. We mentioned that it would contain a pool of procedurally generated practice problems for the learners. We then said that as the student interacts with the game, the student model, which is the BKT, will select which questions they should be asked, given their performance within the game. Thus adapting the game to their needs. A summary of the game's architecture can be seen in Figure 4.

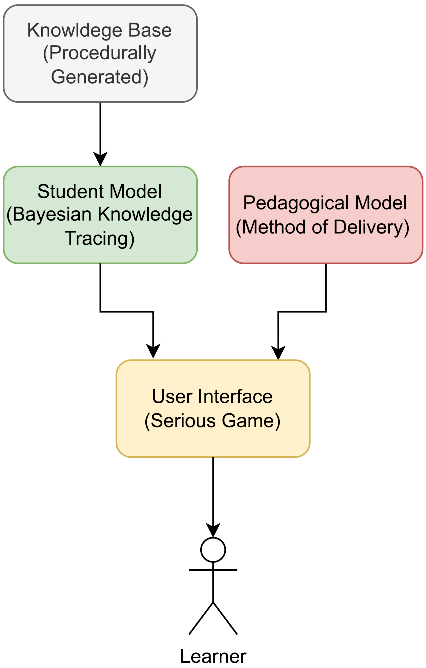

Figure 4. The architecture of our proposed serious game.

6. Implementation

From our literature study, we gleaned many insights about successful serious games, some beyond the scope of this paper. However, some key points that we learned are that serious games include design elements such as:

- Captivating game worlds [29].
- Entertaining story [30].
- Focus on the characterising goal [28].
- Offer feedback on progress [31].

In light of this, we decided to implement the game as a virtual escape game. This choice aligns with specific design principles outlined for educational escape games tailored to computer science education, as discussed by the author in [32] such as evaluating, analysing and understanding. These aspects and more will be elaborated on in the coming sections. The game does this but still retains its novelty by incorporating personalisation.

6.1 Story

The narrative centres around the player assuming the role of a personal assistant to a telecommunications professor. The professor tasks the player with retrieving a laptop from his house, which is urgently needed for an important meeting. Upon arrival at the premises, the player realised they had forgotten to ask the professor for his access card. However, the player remembers a previous conversation with the professor about his passion for code-breaking and how he established a security system based on Hamming Codes. The question now arises: can the player utilise their knowledge of Hamming Codes to bypass the security systems and retrieve the flash drive? This story hopes to immerse the players while testing their knowledge and skills in a fun and challenging way. Throughout many stages of the game, the player interacts with the professor, especially when seeking assistance in solving puzzles, as will be demonstrated in later sections.

6.2 Design of the Game World

We designed the game world in such a way that it fulfils the requirement for linearised learning content. During the design of the game world, we made sure not to divert too much from the well-received design of Codebreakers, making sure to keep the characteristics of content-based puzzles. The house was modelled using the Unity game engine because of its ease of use and the advantage of having many premade assets in the Unity asset store. Using assets sourced from Target Studios, we modelled the Professor's house. Separating it into different sections that assess different Hamming Code KCs. The model of the house can be seen in Figure 5

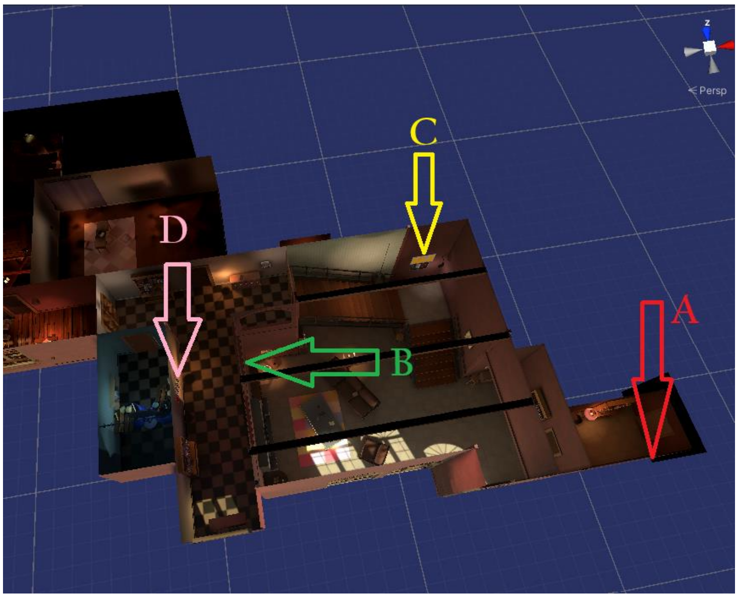

Figure 5. Top view of the professor's house, which acts as the game world. The demarcated areas represent the locations of the different puzzles within the house.

The different positions within the house contain the following puzzles:

- Position A: In position A, right after the entrance, we assess whether the students can compute the number of parity bits required given the number of message bits.
- Position B: In position B, we assess whether the students can compute the position of the parity bits given the number of message bits.
- Position C: In position C, we ask the students to compute the values of the parity bits given a binary message stream. They are then required to enter the entire Hamming Code, not just the values of the parity bits.
- Position D: In position D, we assess whether the students can find the corrupted message bit given a complete Hamming Code.

The core concept is to confine the student to a specific section of the house until they achieve mastery of a particular KC. Once mastery is attained, they can explore other areas of the house, as illustrated in Figure 5. This approach aligns with the principles of escape-type games, as discussed in [32]. After the students solve all the puzzles, they retrieve the laptop they were sent to fetch. We decided to name the game Volatile Systems, which we believe fully encapsulates the game's story and plot.

6.3 Pedagogical Module

As mentioned, the pedagogical model is how the game teaches the learning content. We identified in the literature review that quality serious games balance playful and serious content and focus on the characterising goal [28]. We, therefore, devised the following elements in the pedagogical model.

6.3.1 Teaching through Note Systems

To add a sense of exploration in the game, we scattered note systems throughout different areas within the house and students have to find these to help them remember how to compute the different Hamming Code KCs. An example of how the notes are hidden throughout the house and how they appear to the students once they see them can be seen in Figures 6 and 7. Figure 6 shows a tutorial note hidden within a drawer.

Figure 6. Example of how the tutorial notes are hidden within the game.

Figure 7 shows the tutorial message students see when interacting with one of the tutorial notes hidden throughout the game. This tutorial note contains a hint for how to compute the number of parity bits given the number of message bits for a Hamming Code.

Figure 7. An example of a note that shows the student how to determine the number of parity bits.

6.3.2 Questioning

To give the students new questions and feedback within the game, we devised a seven-segment display. This display shows the students the questions they are expected to solve and provides feedback should they get incorrect answers. To add fun to how the questions are asked, we made them a little cryptic so the students could figure out the message on their own. Figure 8 shows an example of the seven-segment display.

Figure 8. Example of the seven-segment display used within the game. The message 4BITS lets the student know the number of message bits the computation revolves around.

To implement the critical feedback feature, the same display shows the correct computation they should have taken to get to the correct answer when the student receives an incorrect answer, as demonstrated in Figure 9.

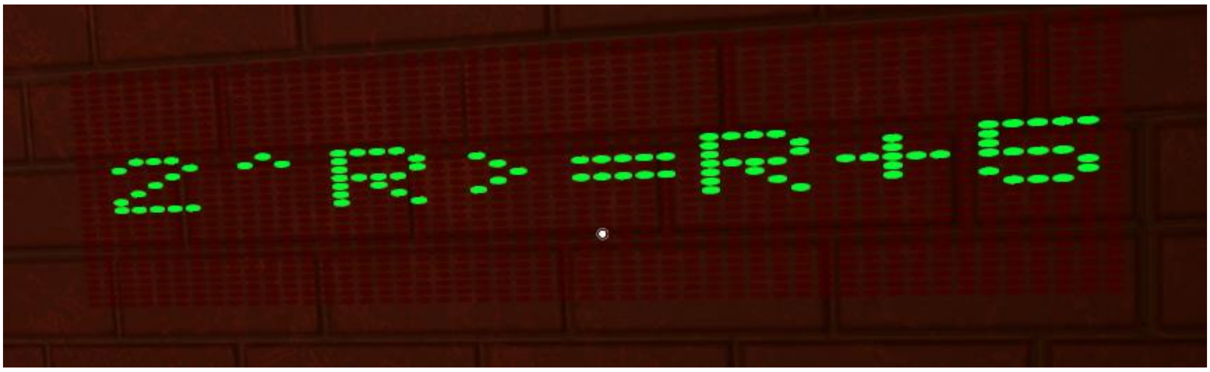

Figure 9. An example of the feedback mechanism in the game that shows students the correct computation they should have taken if they answered the Hamming Code KC incorrectly.

6.4 Puzzles

We designed puzzles, which are input mechanisms for how the students input their answers into the game and act as the UI. Inheriting this from Codebreakers and adding a 3D redesign to them, the puzzles are as follows:

6.4.1 Rotate Cylinder

This puzzle is used for the “Number of parity bits” and “Encode a Hamming Code KC”. The students are provided with a segmented cylinder, where each segment can be rotated independently a specific number of times, revealing different numbers. They must then correctly align these numbers to input the sequence representing the answer to the question they received. An example of this puzzle and the direction of rotation of these segments can be seen in Figure 10.

Figure 10. Example rotate cylinder used. The arrow shows the direction in which each cylinder can rotate independently.

6.4.2 Rotating Concentric Circles

This puzzle is used in the “calculate the position of the parity bits” KC. In this mechanism, students are presented with several concentric circles, each distinguished by a red line marking its starting position. Each ring is divided into segments, allowing the red line to assume various positions. The challenge for students is strategically positioning the red lines, starting from the outermost circle and moving inwards. These aligned red lines should represent the positions of the parity bits. For instance, if a student determines that the parity bit positions for an ‘n’ number Hamming Code are 1-3-5-7-9, the red line on the outermost circle would be set to position 1, followed by position 3 on the next circle, and so forth, An example of this puzzle can be seen in Figure 11.

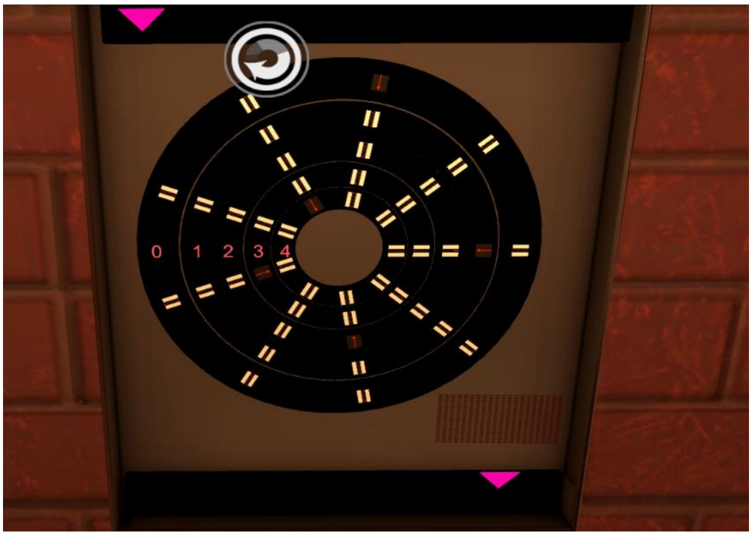

Figure 11. Example of the rotate cylinder puzzle used within the game. The puzzle is interacted with from the outer circle to the inner circle. The positions of the red markings on the puzzles are 1-3-5-7-9. Representing the positions of the parity bits for an ‘n’ bit Hamming Code.

6.4.3 Grouped Levers

Lastly, for the KC “Decoding a Hamming Code”, students are expected to solve a puzzle that uses grouped levers. This system presents students with a series of levers, each corresponding to a specific bit position. Upon identifying the erroneous bit in the given Hamming Code, students are tasked with pulling the lever representing this particular bit. For instance, if students discern the error in the 6th bit of the provided Hamming Code, they should pull the 6th lever, as illustrated in Figure 12.

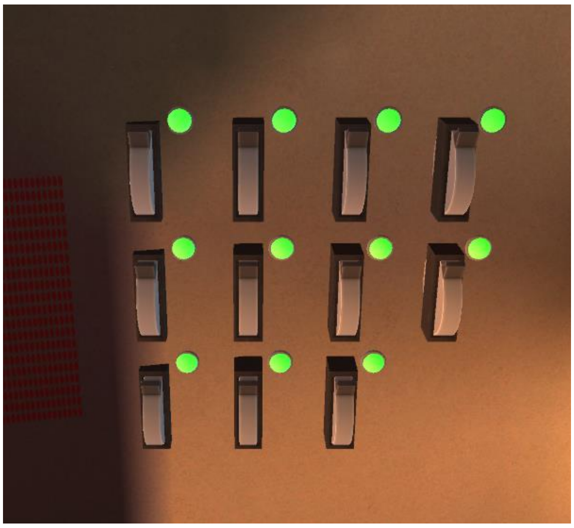

Figure 12. Example of the grouped levers used within the game.

6.5 Student Perspective

The student perspective is how the student sees and experiences the game. It is the view of their player model and a collection of the gauges present within the game. A view of this perspective can be seen in Figure 13.

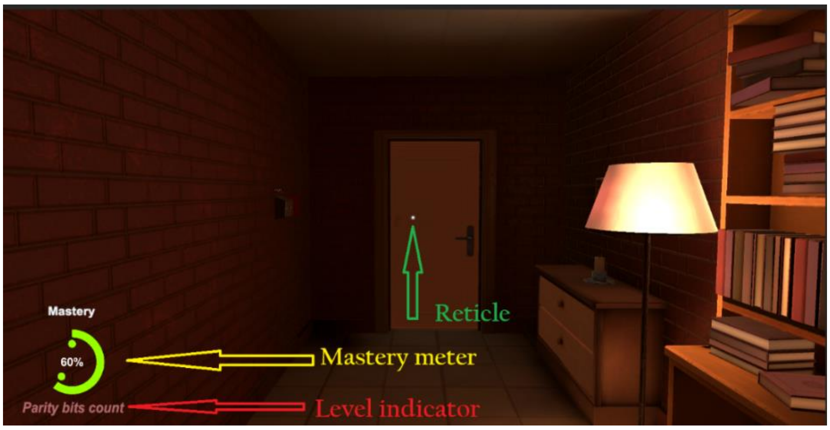

Figure 13. A view of the first-person experience of the game.

The level indicator shows which KC the player is currently attempting. The mastery meter shows the student's predicted level of mastery. This widget will be controlled by the BKT algorithm, which will be implemented later. The reticule is a UI element that shows the player in which direction they are facing. Again, the player practices the KC until they reach a level of mastery of 95%.

6.6 Learning objectives

To help ensure that the students master each learning objective, the house was designed so that each Hamming code KC must be mastered before moving on to the next one. This was implemented through locked doors that open once the player has demonstrated mastery of the current KC. As shown in Figure 14.

Figure 14. An example of the locked doors mechanic that prevents students from progressing before they demonstrate mastery of the current KC.

If students struggle to answer questions correctly multiple times, they may receive guidance via a phone call from the professor, as visualised in Figure 15. During these calls, students may be prompted to revisit the learning notes of the level depicted in Figure 7 or receive helpful hints for the puzzle they are attempting to solve. This feature strengthens the vital feedback mechanism essential for serious games and enhances the story.

Figure 15. An example of another feedback mechanism within the game is when the professor calls the player to assist them.

In addition to the features illustrated here, the game incorporates other elements that enrich its educational and narrative aspects. For instance, players encounter images of notable figures in information theory, such as Claude Shannon, and voice-overs highlighting their contributions to the field. The player also learns more about the professor and his passion for codebreaking. These additions deepen the educational experience by providing context and insight into the historical significance of key figures and their advancements in the field. These improvements also help the game have a more engaging experience than just quizzing students

6.7 Procedural Generation

We want students to have plentiful practice problems. So, they can slowly work with the learning content till they reach mastery. However, creating Hamming Code problems ourselves or sourcing them from elsewhere has the limitation that students may quickly run out of practice problems. To overcome this issue, we wrote algorithms that created these problems for us. This method is called procedural generation [33] [34]. For each Hamming Code KC, we wrote an algorithm that generates these questions for us and validates the student's answer. Since Unity uses the C# programming language, we implemented these programs using the language. Our technique used the randomisation method as a basis for generating these questions. To give an example of one of the algorithms, consider the following explanation of how we procedurally generated questions for encoding a binary message to a Hamming Code.

- First, begin by instantiating an empty set.
- For randomisation, a lower and upper threshold is set, in this case, 8 and 2048, respectively. 8 is the smallest 4-bit number, and 2048 is the most significant 11-bit number. We chose an upper threshold of 11 bits because that's the limit in our classroom. This allows for 2040 possible questions. Which is a significant question pool. For each question, a random number between these two extremes is generated and inserted into the set so that the same random number cannot be drawn twice.
- The random number is then converted to its binary representation.
- The students are asked to compute the Hamming Code for it.
- A C# function generated a Hamming Code and compared it to the student's answer.

In contrast to procedural generation techniques used in creating 3D models, where algorithms must be constrained to ensure the resultant models maintain intended shapes recognisable to humans, our approach in this context differs. Since Hamming Codes can be applied to any binary number greater than length 4, there is less emphasis on ensuring direct playability for students. Many of the algorithms we employed mirror this approach. Consequently, all generated questions should be playable, with unit testing also utilised to guarantee accuracy.

6.8 Latent Knowledge Estimation

The most sought-after upgrade from Codebreakers is the ability to estimate students' skill mastery, so that we can sequence problems attuned to their skill level to them till they achieve mastery. This method allows them to grasp the learning content fully and encourages understanding over just wanting to pass. This feature also forms part of the student model in the game.

To do this, we implemented the BKT model, which uses its four parameters to estimate students' comprehension of a skill. However, two problems are readily apparent. The first is that the BKT model does not sequence problems for students. It merely estimates the probability of skill mastery. Secondly, we cannot use this model if it cannot accurately predict students' skill mastery. The originators of the model in [24] discuss the accuracy of the model when given large datasets, as previously mentioned; however, in our case, within a class of 75 students, we have to evaluate ourselves if this model is the best method so that variations can be used should the model not be accurate enough. This is especially important since we do not want a situation where students who have demonstrated mastery of the skill are said to have not and vice versa.

6.8.1 Student Sequencing

As specified in [24], the BKT model estimates the students' mastery as a percentage. That is to say, it predicts that students have a 50%, 75% and 90% chance of mastering the skill. Therefore, as specified in [24], students in the lower ranges of predicted mastery can be taken as new or struggling with the content, and the inverse is true. We took advantage of this and divided this percentage into three ranges. 0-33%, 33-66% and 66 to 95%. The first-range students are still struggling with the skill. Thus, they are to be given simple questions. The second students are more familiar with the skill and should be given moderate questions. The last students demonstrate a good understanding and should be given tough questions. We partitioned the procedurally generated questions into these three categories using criteria such as how significant the random number generated is.

6.8.2 Synthetic Data Generation

We first had to create a synthetic dataset to gauge how well the BKT model predicts students' skill mastery. This is because we lacked real-world data. This dataset aimed to mimic the data we seek to collect from our natural world classroom. Gauging how well the BKT model predicts skill mastery means assessing how often the model correctly estimated the student getting the question correct. To create a synthetic dataset, we needed to replicate the diverse range of student's capabilities and interactions with questions of varying difficulty levels. By algorithmically deriving each student's proficiency from a normal distribution and difficulty level from a beta distribution, we ensure that the dataset encapsulates a realistic spectrum of abilities and learning outcomes. This methodological approach eliminates the possible bias of making the BKT model seem more accurate than it is. Moreover, simulating students with diverse capabilities engaging with questions across the difficulty spectrum enhances the dataset's robustness and generalizability, fostering a more comprehensive understanding of learning dynamics and facilitating more accurate analyses and model training. Subsequent paragraphs will delve deeper into the intricacies of this simulation process and its implications for dataset quality and research outcomes. We simulated 75 students, each practising 10 questions for all four Hamming Code KCs and leading to 750 questions for each KC. To simulate student responses, we used the Rasch model. The Rasch model is a one-parameter logistic model utilised primarily for assessing the probability of a given respondent (or test-taker) providing a correct answer to a particular item (or question) based on two main factors: the respondent's ability and the item's difficulty. [35]. Mathematically, the probability P that a person n with ability α_n will correctly answer an item i with difficulty βi is given by:

$$P(correct\ by\ n\ on\ i) = \frac{e^{(\alpha_n - \beta i)}}{1 + e^{(\alpha_n - \beta i)}}$$

We manually set each KC's difficulty, ranging from -1 to 2. We drew the initial latent abilities from a normal distribution. We then declined the student's knowledge state to simulate the effect of transience using a beta distribution for the knowledge decay and an exponential distribution for the time elapsed. We then simulated the students' responses using the computed values for latent ability, item difficulty, and the Rasch model. We made sure to increase the students' latent ability after answering the question by 10% if they got it correct and 5% if they got it incorrect since answering a question is a practice. To incorporate the natural unpredictability seen in educational contexts, a random variable between 0 and 1 is generated. If our predicted probability surpasses this random threshold, the student's response is classified as correct; otherwise, it's considered incorrect. This entire simulation process is reiterated ten times, representing ten consecutive practice attempts by the student for each KC. This method resulted in a robust dataset representing the data we aim to collect from our classroom.

6.8.3 Fitting Process

To fit the BKT model to our dataset, we used pyBKT [36]. This is a Python module for this particular purpose. An advantage of this module is that it natively uses the Expectation Maximization (EM) algorithm to search for these parameters. This method is an iterative approach to estimating the parameters more efficiently than brute force methods, which try every possible parameter set within a fine-grained grid.

6.8.4 Results

Since the BKT model is designed to infer student mastery of a skill, recognising knowledge as a latent and directly unobservable concept, no gold standard exists for direct comparison. There are two main approaches to assessing how well the BKT model can describe data. One involves a more intricate method, calculating a correlation coefficient that correlates student performance after the model predicts skill mastery. However, this method has a notable drawback—some students may never master a skill.

In contrast, others may do so only on their final attempt, resulting in the inability to compute this value for many students. A more conventional approach is to evaluate the model's predictive capability concerning student responses. By assessing how accurately the model predicts student responses, we can infer its likelihood of accurately predicting student mastery of a skill. We opted for this method.

Using metrics such as accuracy, how often the model correctly predicted a student response over the total number of predictions made and the root mean square error (RMSE), describing the average discrepancy between the actual value, in the case of a binary response 1, and the value predicted by the model.

We found that the BKT model achieved an average accuracy of 73.6% across all our KCs. And an average RMSE of 0.4325. Which is a good level of performance for an HMM. This figure suggests that, on average, the BKT model will predict student performance correctly 7 out of 10 times.

7. Discussion

This paper presented a novel approach to teaching telecommunications theory through a personalised escape-type serious game, drawing heavy inspiration from ITSs. The subsequent subsection provides a comprehensive overview of the game's design, serving as a foundational framework for creating similar serious games. Additionally, it elucidates the unique challenges encountered during the development process, offering valuable insights for future implementations.

7.1 Design Framework

Table 1 presents the design framework of the game.

Table 1. Design framework of the game

Design decision	Rationale	Implementation	Challenge
User-Centered Design Process	As demonstrated in Codebreakers, including the target audiences' opinions in the design process may improve the game's reception.	This new game, Volatile Systems, relied heavily on students' opinions for its implementation.	The target audience may have diverse preferences and opinions regarding game mechanics, aesthetics, and content. Balancing these differing preferences while maintaining coherence and appeal can be challenging.
Procedurally generated learning questions.	Some students will require many questions before they attain mastery of a skill.	Implemented a system that uses randomisation as a basis to create new questions.	Depending on the field to which this method is applied, it may not be easy to ensure that the generated questions are solvable.
Puzzle mechanisms for inputting student answers.	Using puzzles to input the students' answers may promote better engagement than traditional write-the-correct-answer responses.	Different puzzles were used as a mechanism to input the answers. These include different levers, rotating cylinders, and concentric circle combinations,	Designing an interesting puzzle that is not too difficult or takes away the student's attention from the current learning content may be challenging.

BKT for personalisation.	Students have different learning capabilities; therefore, a system is needed that understands their learning needs and guides them to better mastery,	The BKT algorithm, a novel approach to personalisation in serious games, was implemented to infer the students' mastery state as a probability. This probability was divided into three ranges, beginner, moderate, and advanced, based on the range that the student is at. Questions are drawn from the procedural generation engine that matches the students' capability.	Like many ML methods, BKT needs data to offer accurate predictions. Gathering this data may require manual data collection from tests or other learning systems, which might be a long process.
Scattered learning hints.	It may improve student engagement and make them more active in the game.	We placed hint notes throughout the game's environment to encourage students to explore the environment more.	If overdone, this may ruin the game's engagement.

8. Conclusions

The primary objective of this project was to enhance and build upon Codebreakers' success by answering the following research question: Can a new serious game be developed, incorporating KT algorithms, to provide personalised learning experiences in telecommunications education? The critical features sought for the successor included the ability to adapt the game to students' skill levels dynamically, providing questions tailored to their proficiency. Additionally, there was a desire for 3D environments and linearised teaching content. In response to this research question, we developed a game called Volatile Systems.

Volatile Systems unfolds within the confines of a professor's house, where students overcome puzzles based on the Hamming Code to retrieve a laptop. The inspiration for the game's design draws from ITSs. To gauge students' skill mastery, we employed the BKT statistical model. This model helped determine the appropriate level of questions for personalisation and concluded the process once mastery was achieved. Diverse puzzles were incorporated to enhance learner immersion and engagement. The Unity game engine was utilized to model the game's 3D environment.
We also used the design decisions of this game to propose a design framework for contributing to serious game design.

Our ongoing efforts include evaluating the effectiveness of the game with actual students. This evaluation aims to assess the game's usability, students' experiences, and whether there is a noticeable improvement in their understanding of the subject. This assessment is crucial for obtaining valuable feedback from students, enabling us to incorporate their suggestions into the subsequent iterations of the game.

Acknowledgements

We want to thank Target Studios for providing a myriad of premade assets that helped accelerate the development of this game.

Conflicts of Interest

No conflicts of interest are present.

References

[1] J. P. Gee, "What Video Games Have to Teach Us About Learning and Literacy.," Computers in Entertainment, p. 20, 2003, doi: https://doi.org/10.1145/950566.950595.

[2] D. Chandross and E. DeCourcy, "Serious games in online learning," International journal on innovations in online education, vol. 2, no. 3, 2018., doi: https://doi.org/10.1615/IntJInnovOnlineEdu.2019029871.

[3] D. B. Clark, E. E. Tanner-Smith and S. S. Killingsworth, "Digital Games, Design, and Learning: A Meta Analysis," Review of Educational Research, vol. 86, no. 1, pp. 79-122, 2016, doi: https://doi.org/10.3102/0034654315582065.

[4] J. Greeff, R. Heymann and M. H. a. C. Heymann, "Codebreakers: Designing and developing a serious game for the teaching of information theory," in Advances in Web-Based Learning-ICWL: 16th International Conferenece, Cape Town, South Africa, 2017, doi: https://doi.org/10.1007/978-3-319-66733-1_2.

[5] R. Y, "Hamming Codes," Electronic Desk, [Online]. Available: https://electronicsdesk.com/hamming-codes.html. [Accessed 29 March 2024].

[6] A.Streicher and J. D. Smeddinek, "Personalized and Adaptive Serious Games," Entertainment Computing and Serious Games, vol. 9970, 2016, https://doi.org/10.1007/978-3-319-46152-6_14

[7] F. C. Blumberg, D. E. Almonte, N. Hashimoto and J. S. Anthony, "Serious Games: What Are They? What Do They Do? Why Should We Play Them?," in The Oxford Handbook of Media Psychology, Oxford Academic, 2013, pp. 334-351, doi: https://doi.org/10.1093/oxfordhb/9780195398809.013.0019.

[8] J. Alvarez, Serious games, advergaming, edugaming, training and more, L. Michaud, Ed., IDATE, 2008.

[9] K. Balasubramanian, "Game Engines: All You Need to Know," GAMEOPEDIA, [Online]. Available: https://www.gameopedia.com/game-engines-all-you-need-to-know-about/. [Accessed 29 March 2024].

[10] D. R. Johnson, "Introduction to engineering design," IED, [Online]. Available: https://dariusrjohnson.weebly.com/ied.html. [Accessed 2023].

[11] E. Silva, R. Lopes and L. P. Reis, "CogniChallenge: Multiplayer serious games' platform for cognitive and psychosocial rehabilitation," International Journal of Serious Gamess (IJSG), 2023, doi: https://doi.org/10.17083/ijsg.v10i4.646.

[12] L. C. Shum, Y. Rosunally, S. Scarle and K. Munir1, "Personalised Learning through Context-Based Adaptation," Education and Information Technologies, pp. 1-32, 2023, doi: https://doi.org/13077–13108

[13] O. Ku, C.-C. Hou and S. Y. Chen, "Incorporating customisation and personalization into game-based learning: A cognitive style perspective," Computers in Human Behavior, vol. 65, pp. 359-368, 2016, doi: https://doi.org/10.1016/j.chb.2016.08.040.

[14] B. Thabet and F. Zanichelli, "Towards Intelligent Serious Games: Deep Knowledge Tracing with Hybrid Prediction," in Computer Science and Education, Singapore, 2022, doi: https://doi.org/10.1007/978-981-99-2443-1_25.

[15] I. Grabska-Gradzinska and J. K. Argasinski, "Graph-Based Method for the interpretation of user activities in serious games," in Human-Computer-Interaction, Bari, Italy, 2021, doi: https://doi.org/10.1007/978-3-030-85613-7_6.

[16] E. Lotfi, "Towards a new platform based on learning outcomes analysis for mobile serious games," International Journal of Emerging Technologies in Learning(iJET), vol. 15, no. 2, pp. 42-57, 2020, doi: https://doi.org/10.3991/ijet.v15i02.11637

[17] J. Greeff and R. Heymann, "Designing and developing a narrative driven serious game for teaching," in IEEE Global Engineering Education Conference (EDUCON), 2018, doi: https://doi.org/10.1109/EDUCON.2018.8363271

[18] C. Romero and S. Ventura, "Educational Data Mining and Learning Analytics: An updated survey," arxiv, Cordoba, doi: https://doi.org/10.1002/widm.1355

[19] M. Eagle and T. Barnes, "Intelligent Tutoring Systems, Educational Data Mining,," in International Conference on Intelligent Tutoring Systems, Charlote, 2010, doi: https://doi.org/10.1007/978-3-642-13437-1_23.

[20] S. Kovalev, A. Kolodenkova and E. Muntyan, "Educational Data Mining: Current Problems and Solutions," in International Conference on Information Technologies in Engineering Education (Inforino), Moscow, Russia, 2020, doi: https://doi.org/10.1109/Inforino48376.2020.9111699.

[21] C. Piech, J. Spencer, J. Huang, S. Ganguli, M. Sahami, L. Guibas and J. Sohl-Dickstein, "Deep Knowledge Tracing," arXiv, New York, 2015, doi: https://doi.org/10.48550/arXiv.1506.05908

[22] Y. Zhang, R. An, W. Zhang and L. Shuhui, "Deep Knowledge Tracing with Concept Trees," in Advanced Data Mining and Applications, Sydney, 2023, doi: https://doi.org/10.1007/978-3-031-46664-9_26.

[23] S. Pu, M. Yudelson, L. Ou and Y. Huang, "Deep Knowledge Tracing with Transformers," Artificial Intelligence in Education, pp. 252-256, 2022, doi: https://doi.org/10.1007/978-3-030-52240-7_46.

[24] A. Corbett and J. Anderson, "Knowledge Tracing: Modeling the Acquisition of Procedural Knowledge," Human Computer Interaction Institute, School of Computer Science, Carnegie Mellon University, Pittsburgh, 1995, doi: https://doi.org/10.1007/BF01099821

[25] P. Sedlmeier, "Intelligent Tutoring Systems," International Encyclopedia of the Social & Behavioral Sciences, pp. 7674-7677, 2001, doi: https://doi.org/10.1016/B0-08-043076-7/01618-1.

[26] ALEKS, "Knowledge Space Theory," ALEKS, [Online]. Available: https://www.aleks.com/about_aleks/knowledge_space_theory. [Accessed 13 12 2023].

[27] V. Devedzic, D. Radovic and L. Jerinic, "Innovative Modeling Techniques On Intelligent Tutoring Systems," Innovative Teaching and Learning: Knowledge-Based Paradigms, vol. 36, pp. 189-234, 2000, doi: https://doi.org/10.1007/978-3-7908-1868-0_6.

[28] P. Caserman, K. Hoffman and P. Muller, "Quality criteria for serious games: Serious part, game part, and balance," 2020, doi: https://doi.org/10.2196/preprints.19037.

[29] E. Adams, Fundamentals of game design, New Riders Pub, 2009. Available: https://api.semanticscholar.org/CorpusID:54104725

[30] M. W. Travis, "What is a story, and where does it?," [Online]. Available: https://www.thewrap.com/.

[31] W. S. Ravyse, A. S. Blignaut, V. Leendertz and H. Gelderblom, Success factors for serious games to enhance learning: A systematic review, 2016, doi: https://doi.org/10.1007/s10055-016-0298-4.

[32] D. Menon and M. Romero, "Game Mechanics Supporting a Learning and Playful Experience in Educational Escape Games," in In Global perspectives on gameful and playful teaching and learning, IGI Global, 2019, pp. 143-162, doi: https://doi.org/10.4018/978-1-7998-2015-4.ch007.

[33] Zenva, "What is Procedural Generation – Complete Guide," Zenva, 23 December 2023. [Online]. Available: https://gamedevacademy.org/what-is-procedural-generation/#What_is_procedural_generation. [Accessed 7 April 2024].

[34] S. Risi and J. Togelius, "Increasing Generality in Machine Learning through Procedural Content Generation," arxiv, 2019, doi: https://doi.org/10.1038/s42256-020-0208-z.

[35] G. Rasch, Probabilistic models for some intelligence and attainment tests, Chicago: Chicago Press, 1980, doi: https://doi.org/10.2307/2287805.

[36] A. Badrinath, F. Wang and Z. Pardos, "pyBKT: An Accessible Python Library of Bayesian Knowledge Tracing Models," in Proceedings of the 14th International Conference on Educational Data Mining, 2021.

[37] P. Caserman, K. Hoffmann, P. Müller, M. Schaub, K. Straßburg, J. Wiemeyer, J. Wiemeyer, R. Bruder and S. Göbel, "Quality Criteria for Serious Games: Serious Part, Game Part, and Balance," JMIR Serious Games, vol. 8, no. 3, 2020, doi: https://doi.org/10.2196/19037.

[38] C. Georgovassilopoulos, Moisiadis and Papaioannou, "SIMPLE GAMES TO EXPLAIN COMPLEX ALGORITHMS USED IN EVERYDAY LIFE," in EDULEARN21, Online Conference, 2021, doi: https://doi.org/10.21125/edulearn.2021.0206

[39] P. Neale, S. Thapa and C. Boyce, "PREPARING A CASE STUDY: A Guide for Designing and Conducting a Case Study for Evaluation Input," Pathfinder international, May 2006. [Online]. Available: https://resources.peopleinneed.net/documents/102-m-e-tool-series-case-study.pdf. [Accessed 7 April 2024].

[40] Purdue Global, "Check Your Sources: A Checklist for Validating Academic Information," Purdue Global, 24 January 2024. [Online]. Available: https://www.purdueglobal.edu/blog/online-learning/credible-academic-sources/. [Accessed 7 April 2024].

[41] N. Kara, "A Systematic Review of the Use of Serious Games in Science Education," CONTEMPORARY EDUCATIONAL TECHNOLOGY, vol. 13, no. 2, 2020, doi: https://doi.org/10.30935/cedtech/9608.

[42] C. Georgovassilopoulos, G. Moisiadis and E.Papaioannou, "Simple games to explain complex algorithms," EDULEARN21 proceedings, pp. 747-756, 5-6 July 2021, doi: https://doi.org/10.21125/edulearn.2021.0206.