الجمهورية الجزائرية الديمقراطية الشعبية

PEOPLE'S DEMOCRATIC REPUBLIC OF ALGERIA

MINISTRY OF HIGHER EDUCATION AND SCIENTIFIC RESEARCH

Higher Normal School of Technological Education. SKIKDA

Department of Mathematics and Computer Science

DISSERTATION

Presented to obtain a degree in Computer Science as a teacher of middle school

Serious Games as Learning Tools: A Case Study on Active Pedagogy in Computer Science

Field of study: **Computer Science**

Presented by

Hatem Merabtine

Ouail Mellak

Atef Gasmi

Board of examiners

President	Dr. Safia Bekouche	MCB Enset-Skikda
Supervisor	Dr. Rida Mezghache	MCB Enset-Skikda
Examiners	Dr. Samir Sellami	MCA Enset Skikda
	Ms. Leila Abdoune	MAA Enset Skikda

Academic year: 2024/2025

بسم الله الرحمن الرحيم

Abstract

Serious games are increasingly recognized as powerful instruments for fostering active, learner-centered education in technical fields. This dissertation documents the design, development, and evaluation of Smart in the Dark, a serious game that teaches intermediate-level Arduino programming concepts through contextual,

challenge-based scenarios. Players tackle authentic tasks—such as configuring basic sensors, implementing control structures, and applying input/output logic—thereby developing intuitive understanding through guided exploration and immediate feedback.

A mixed-methods evaluation was conducted to gauge the game's educational impact. Data were gathered from pre- and post-instruction questionnaires, the System Usability Scale (SUS), and behavioral metrics extracted from Firestore gameplay logs. Findings indicate statistically significant gains in conceptual understanding and moderate levels of engagement across game tasks. Although remote testing conditions and a limited sample size constrained the depth of observation, the results underscore the game's promise as a supplementary learning tool for introductory courses in computer science and electronics.

By presenting a validated model for integrating game-based learning into intermediate programming instruction, this study contributes to the field of educational technology and identifies avenues for future work, including content expansion and deployment within blended-learning environments.

Keywords: *Serious games, Arduino programming, Game-based learning, Educational technology, Mixed-methods evaluation*

ملخص

يتزايد الاعتراف بالألعاب الجادة كأدوات فعّالة لدعم التعليم النشط المتمحور حول المتعلم، لا سيما في المجالات التقنية. تستعرض هذه الأطروحة عملية تصميم وتطوير وتقييم لعبةSmart in the Dark ، وهي لعبة جادة تهدف إلى تعليم مفاهيم برمجة الأردوينو للمتعلمين في المستوى المتوسط، وذلك من خلال سيناريوهات سياقية قائمة على التحدي. تُشرك اللعبة اللاعبين في مهام مستوحاة من العالم الواقعي، مثل استخدام أنظمة الاستشعار الأساسية، وهياكل التحكم، ومنطق الإدخال/الإخراج، مما يعزز الفهم البديهي من خلال الاستكشاف والتغذية الراجعة.

تم اعتماد منهجية تقييم متعددة الأساليب لقياس الأثر التعليمي للعبة، حيث جُمعت البيانات باستخدام استبيانات قبلية وبعدية، ومقياس قابلية استخدام النظام(SUS) ، إلى جانب مؤشرات سلوكية تم تحليلها انطلاقًا من سجلات Firestore التي وثّقت تفاعل المستخدمين داخل اللعبة. وقد أظهرت النتائج تحقيق المشاركين لمكاسب ملموسة في الفهم المفاهيمي، مع تسجيل مستويات متوسطة من التفاعل عبر مختلف المهام داخل اللعبة.

ورغم بعض القيود مثل بيئات الاختبار عن بُعد وصغر حجم العينة، تؤكد النتائج إمكانيات اللعبة كأداة تعليمية تكميلية ضمن مقررات تمهيدية في علوم الحاسوب والإلكترونيات. وتسهم هذه الدراسة في مجال تكنولوجيا التعليم من خلال تقديم نموذج موثوق لتوظيف التعلم القائم على الألعاب في تعليم مفاهيم البرمجة لمتعلمين في المستوى المتوسط، مع اقتراح آفاق تطوير مستقبلية تشمل توسيع المحتوى ودمج اللعبة ضمن بيئات التعلم المدمج.

الكلمات المفتاحية: الألعاب الجادة، برمجة الأردوينو، التعلم القائم على الألعاب، التكنولوجيا التعليمية، التقييم المختلط

Acknowledgements

We would like to express our heartfelt gratitude to all those who contributed to the successful completion of this research and supported us throughout its development.

First and foremost, we extend our sincere thanks to our supervisor, **Dr. Rida Mezghache**, whose invaluable guidance, insightful feedback, and unwavering support have been instrumental in shaping this dissertation. His expertise and commitment have greatly enriched our work, and we are deeply thankful for his mentorship.

We also wish to acknowledge the esteemed members of the examination committee **Dr. Safia Bekouche**, **Dr. Samir Sellami**, and **Ms. Leila Abdoune** for their time.

Our deepest appreciation goes to the **Higher Normal School of Technological Education in Skikda** and the **Department of Mathematics and Computer Science** for providing a stimulating academic environment and the resources necessary to carry out this study.

We are sincerely grateful to our families and friends for their continuous encouragement, patience, and support. Their understanding and motivation have been essential in helping us overcome challenges and remain committed to our goals.

Additionally, we acknowledge the open-source communities behind **GDevelop**, **BlocklyDuino**, and **Firebase**, whose tools and documentation played a crucial role in the development and implementation of *Smart in the Dark*.

This dissertation is the result of a collective effort, and we extend our heartfelt thanks to everyone who played a part in its realization. We hope that our work contributes to the advancement of interactive and effective learning methodologies in computer science education.

Table of Contents

Abstract ii

ملخص iii

Acknowledgements iv

List of Figures 6

List of Tables 7

Introduction 8

I. Chapter I. Introduction to Serious Games 10

I.1) Introduction 10

I.2) Serious Games 10

I.2.1) Definitions of Serious Games 10

I.3) History 15

I.4) Evolution of Serious Games 18

I.5) Classification of Serious Games 19

I.5.1) By Purpose 20

I.5.2) By sector 20

I.5.3) By gameplay 20

I.6) Benefits and Challenges of Serious Games 20

I.6.1) Challenges 20

I.6.2) Benefits 22

I.7) Examples of Serious Games 24

I.7.1) KangaZoo 24

I.7.1.1) Overview 24

I.7.1.2) Gameplay Mechanics 25

I.7.1.3) Objectives 25

I.7.1.4) Audience 25

I.7.1.5) Domain 25

I.7.2) The Great Flu: Pandemic Awareness Game 25

I.7.2.1) Overview 25

I.7.2.2) Objective 26

I.7.2.3) Gameplay 26
I.7.2.4) Audience 26
I.7.2.5) Domain 26

I.7.3) IBM CityOne: Urban Management Training 26
I.7.3.1) Overview 26
I.7.3.2) Objective 27
I.7.3.3) Gameplay 27
I.7.3.4) Audience 27
I.7.3.5) Domain 27

I.7.4) Duolingo: Gamified Language Learning 27
I.7.4.1) Overview 27
I.7.4.2) Objective 28
I.7.4.3) Gameplay 28
I.7.4.4) Audience 28
I.7.4.5) Domain 28

I.8) The difference between gamification, Serious Games and game-based learning 29

I.9) Conclusion 30

II. Chapter II. Serious Games and Learning 31

II.1) Introduction 31

II.2) Learning theory and Serious Games 31

II.3) Educational attribute of Serious Games 33

II.4) Core Elements of Serious Games 34

II.4.1) Serious Games Components 34
II.4.1.1) Game Mechanics and Dynamics 35
II.4.1.2) Narrative and Storytelling 35
II.4.1.3) Feedback Mechanisms 35
II.4.1.4) Assessment and Evaluation 35

II.4.2) Structural Components of Serious Games 35
II.4.2.1) Game World 35
II.4.2.2) Characters 36
II.4.2.3) Technological Infrastructure 36
II.4.2.4) User Interface (UI) and Experience (UX) Design 36
II.4.2.5) Levels and Missions 36
II.4.2.6) Rules System 36

II.5) Serious Game Underlies "A Pedagogical Scenario" 37

II.5.1) Defining the Pedagogical Scenario 38

II.5.2) Defining the "Pedagogical Objective" 38

II.6) Fundamental Qualities of a Good Serious Game 39

II.6.1) introduction 39

II.6.2) Balance Between Learning and Entertainment ... 39
II.6.3) Realism and Immersion ... 39
II.6.4) Interaction and Feedback ... 40
II.6.5) Adaptability and Artificial Intelligence ... 40
II.6.6) Choices and Challenges ... 40
II.6.7) Usability and Accessibility ... 40
II.7) Evaluation of a Serious Games ... 40
II.7.1) Quantitative tools ... 41
II.7.1.1) Pre/Post Game Questionnaires ... 41
II.7.1.2) Activity logs and tracking ... 42
II.7.2) Qualitative tools ... 42
II.7.2.1) Interviews ... 42
II.7.2.2) Live observation ... 43
II.7.3) Frameworks for Evaluation ... 43
II.7.3.1) The Four-Dimensional Framework ... 43
II.7.3.2) The Game-Based Learning Evaluation Model ... 44
II.8) Conclusion ... 44
III. Chapter III. Arduino ... 45
III.1) Introduction to the Arduino Platform ... 45
III.2) History and Evolution of Arduino ... 45
III.3) Arduino Board Types ... 46
III.4) Hardware Components ... 46
III.5) Software Environment ... 47
III.6) Teaching Electronics with Arduino ... 48
III.7) Applications and Integration ... 49
III.8) Advantages and Limitations ... 50
III.9) Serious Games in Arduino Education ... 50
III.10) Conclusion ... 50
IV. Chapter IV. Design and Implementation ... 52
IV.1) Design ... 52
IV.1.1) Introduction ... 52
IV.1.2) Project Overview and Objectives ... 52

IV.1.2.1) Primary Educational Objectives 52
IV.1.2.2) Technical Objectives 53
IV.1.2.3) Success Criteria 53

IV.1.3) Scenario and Pedagogy 53
IV.1.3.1) Narrative and Flow 53
IV.1.3.2) Arduino Concepts Integration 54

IV.1.4) System Architecture 55
IV.1.4.1) GDevelop Game Engine version 5.5.231 (Front-End) 56
IV.1.4.2) BlocklyDuino Editor version 9.4.0 (Embedded via iframe) 56
IV.1.4.3) Firebase Backend 10.7.0 (Authentication + Firestore) 56
IV.1.4.4) Integration Strategy 58

IV.1.5) User Interface 58
IV.1.5.1) Visual and Accessibility Considerations 60

IV.2) Implementation 60

IV.2.1) Tools and Technologies 60
IV.2.1.1) GDevelop Engine 60
IV.2.1.2) BlocklyDuino Editor 61
IV.2.1.3) Firebase (Authentication & Firestore): 61
IV.2.1.4) Web Technologies: 61
IV.2.1.5) Additional Libraries: 61

IV.2.2) Implementation Highlights 62
IV.2.2.1) Core Mechanics 62
IV.2.2.2) Scenario Execution 64
IV.2.2.3) User Interaction 66

IV.2.3) Screenshots 67

IV.3) Conclusion 69

V. Chapter V. Evaluation and Validation 70

V.1) Introduction 70

V.2) Evaluation Objectives and Tools 70

V.3) Description of the Tests (Pre/Post, SUS, In-Game Metrics) 71

V.4) Analysis of results 73

V.4.1) Learning Outcomes: Pre- and Post-Game Questionnaire Analysis 73

V.4.2) System Usability Scale (SUS) Evaluation of “Smart in the Dark” 75

V.4.3) In-Game Metrics 76

V.5) Learner feedback 78

V.6) Limitations and Future Perspectives 78

V.7) Conclusion 79

Conclusion **80**

References **82**

List of Figures

Figure 1 *Venn diagram of Serious Games and other related disciplines.* 12
Figure 2 *Relationship between video game, serious game and serious gaming (Djaouti, Alvarez & Jessel, 2011, p. 4)* 13
Figure 3 *Military Computer Wargaming Evolution* 15
Figure 4 *The game « la troncule »* 16
Figure 5 *First serious game «Army Battlezone»* 16
Figure 6 *1985 Modification of Flight program* 17
Figure 8 *The Great Flu: Pandemic Awareness Game* 26
Figure 9 *IBM CityOne: Urban Management Training* 27
Figure 10 *dualingo game logo* 28
Figure 11 *The (ideal) difficulty and learning curves according to S. Natkin* 31
Figure 12 *Diagram showing the link between serious gaming and the pedagogical component of video game development (diagram reproduced by kind permission of M.Zyda)* 38
Figure 13 *Arduino Boards: Uno, Mega, Nano* 46
Figure 14 *Arduino Uno Components* 47
Figure 15 *LED Circuit Setup* 48
Figure 16 *Smart Lighting System with PIR Sensor* 50
Figure 17 *UML Sequence Diagram of the Smart in the Dark System Architecture* 57
Figure 18 *Game View Panel Mockup* 59
Figure 19 *A forked version of the open-source BlocklyDuino Editor Panel Mockup* 60
Figure 20 *The Menu of Smart in the Dark* 67
Figure 21 *Level 4 Restoring the Security System* 68
Figure 22 *Level 4 arduino circuit simulator* 68
Figure 23 *Smart in the Dark" User Study* 71
Figure 24 *Screenshot from Google Form pre-test survey* 72
Figure 25 *Screenshot from Google Form post-test survey* 72
Figure 26 *Screenshot from Google Form SUS survey* 73
Figure 27 *Pre-Game vs Post-Game Mean accuracy comparison* 74
Figure 28 *Mean Scores for Pre-Game and Post-Game Arduino Questionnaires* 75
Figure 29 *Mean time spent and clicks per level* 77

List of Tables

Table 1 *Differences between entertainment games and Serious Games* *14*
Table 2 *Milestones in the history of Serious Games* *18*
Table 3 *Percentage of responses to questions on attitudinal competencies* *22*
Table 4 *Percentage of responses to questions on self-efficacy* *23*
Table 5 *Percentage of responses to questions on academic performance* *24*
Table 6 *The difference between gamification, Serious Games and game-based learning* *29*
Table 7 *Educational attribute of Serious Games* *33*
Table 8 *specific Arduino programming concepts and in-game tasks of each level* *54*
Table 9 *System Usability Scale (SUS) results with sus score* *76*

Introduction

In recent years, Serious Games have increasingly been considered as an innovative method of improving students' engagement and encouraging higher levels of interaction in education (Zyda, 2005; Michael & Chen, 2006). Serious Games are highly interesting educational tools, in which entertainment and pedagogical approaches are brought together to develop inclusive learning contexts that promote trial, error and creativity. The merging of play and learning is quite consistent with more recent theories of learning, such as constructivism (Tricot, 1999; Rahdoun & Djabbar, 2013), in which the play-facilitated opportunities for interaction between learner and environment are said to help the learner's construction of his or her knowledge.

Digital game-based learning holds significant potential in computer science education, particularly for learners who already possess foundational knowledge of programming and electronics. For such learners, Serious Games offer a dynamic platform to reinforce concepts, deepen understanding, and apply logic in practical contexts. Through immersive game scenarios, educators can engage students in authentic problem-solving experiences that simulate real-world programming and hardware integration. This approach enables learners to consolidate prior knowledge, refine their skills, and interact with technical content in meaningful, motivating, and cognitively challenging ways.

Yet, despite their promise, the use of serious games in educational environments are not very widespread, especially in developing countries. Limitations like lack of resources, educational compatibility and inadequate model of evaluation are obstacles to the successful implementation of Serious Games in courses. These voids highlight the importance of focused studies that examine the design and assessment of Serious Games in discipline-specific contexts to support active learning.

This dissertation explores the educational potential of serious game-based learning as an instructional strategy in computer science, with a particular focus on learners who already possess foundational knowledge of Arduino programming and electronics. It presents the design, development, and evaluation of a custom-built serious game, *Smart in the Dark*, developed to reinforce and extend understanding of embedded systems through gameplay. Aimed at intermediate learners, the game addresses key pedagogical objectives such as maintaining motivation, encouraging interactivity, and supporting the practical application of prior knowledge through simulation-driven scenarios and progressively challenging tasks.

In order to explore this, the study uses a multimethod design (Creswell, 2003) using both qualitative and quantitative methods. The game was designed through iterative process and based on pedagogical theories. The testing phase included pre-test and post-test, System Usability Scale (SUS), in-game metrics, and user feedback collected through online deployment. These tools provided comprehensive insights into both learning outcomes and user experience.

This dissertation is structured as follows. The Introduction presents the research context, objectives, and methodology. Chapter I, Introduces the concept of Serious Games , including their definitions, background, taxonomy, benefits and challenges. Chapter II, Investigates the connection between Serious Games and learning theories, the educational characteristics and the evaluation frameworks of serious games. Chapter III presents Arduino, its relevance in education, as well as its role in the proposed learning architecture.

Chapter IV details the design and development process of *Smart in the Dark*, the serious game created as part of this work. Chapter V focuses on the evaluation and validation of the game through empirical methods. Finally, the General Conclusion, summarizes the key findings, and discusses recommendations and future perspectives for Serious Games in computer science education.

Chapter I. Introduction to Serious Games

I.1) Introduction

In this chapter we will first define and trace the historical development of Serious Games, before examining the classification, benefits, and challenges associated with them. we will then address the development process, key components, and theoretical foundations underlying these games. Finally, we will present a selection of examples of Serious Games.

I.2) Serious Games

I.2.1) Definitions of Serious Games

The term "serious game", as we understand it today, appears to have been first coined in 1970 by Clark C.Abt in his book "Serious Games ". At that time, the term "serious game" was used for card or board games designed for educational purposes. Today, the term "serious game" is generally used in a computer context.

Michael and Chen (2006), researchers and specialists in the development of games and educational technologies, define the serious game as: “Games that do not have entertainment, enjoyment, or fun as their primary purpose.”

The researchers contend that Serious Games utilize gameplay mechanics designed to attract users; however, these mechanics are intentionally oriented towards significant objectives that benefit fields such as education, health, or vocational training. Consequently, the enjoyment derived from these games is not an ultimate goal but rather a tool to encourage users to interact with important content in a meaningful manner.

Mickael Zyda (2005), an American researcher who participated in the development of America's Army, provides a more formal definition in which entertainment is clearly identified as a fundamental component: "Serious game: a mental contest, played with a computer in accordance with specific rules, that uses entertainment to further government or corporate training, education, health, public policy, and strategic communication objectives.”

Zyda confirms in his definition that Serious Games extend beyond mere entertainment tools; they serve as instruments for attaining strategic objectives in specific domains. In this context, the human element is considered crucial, as the interplay of gameplay, enjoyment, and individual engagement facilitates effective interaction with both content and skills.

Zyda also says:

"Pedagogy must be subordinated to the game's scenario, and the playful component must take precedence" (Zyda, 2005).

Zyda means that the educational element in Serious Games should not be separate or imposed in the game, but rather should be integrated within the scenario of this game. In other words, education should be a clear and direct goal that surpasses the elements of entertainment.

In his research, Alvarez defines Serious Games as: “A computer application, the initial intention of which is to combine, with coherence, both serious aspects (Serious) such as, in a non-exhaustive and non-exclusive manner, teaching, learning, communication, or information, with playful springs from the video game (Game). Such an association, which is carried out by the implementation of an "educational scenario", which on the computer level would correspond to implementing a dressing (sound and graphic), a story, and suitable rules, therefore aims to move away from simple entertainment. This gap seems indexed on the prevalence of the "educational scenario" (Alvarez, 2007).

Alvarez presents the serious game as a computer application with the purpose of combining, at the same time, serious features in a non-exhaustive method such as teaching, learning, communication, or information with playful elements from video games. His definition emphasizes the intentional integration of educational or communicative objectives with gaming elements, ensuring that the game serves a purpose beyond pure entertainment.

Finally, we can say that a serious game refers to a type of video game that is primarily designed not just to entertain but also to inform, educate, train, or transmit knowledge and skills through a playful mediation process.

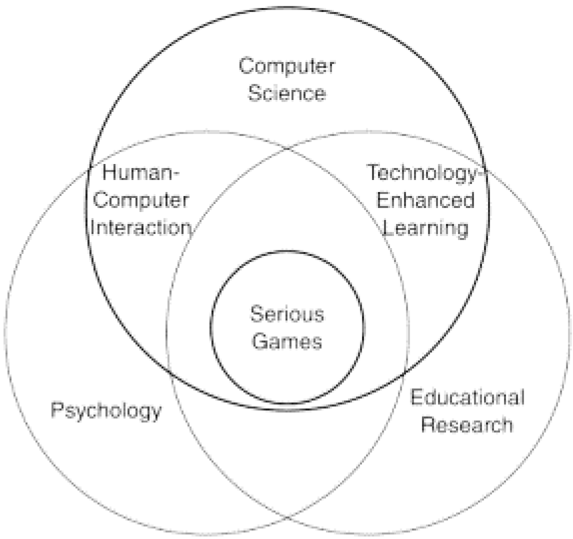


***Figure 1** Venn diagram of Serious Games and other related disciplines.*

I.2.2) The purpose of Serious Games

For the purpose of Serious Games, many researchers have mentioned these goals, as we find Peña-Miguel and Sedano Hoyuelos (2014) mentioning improving learning and interaction "Serious Games can provide not only entertainment but also training in areas such as health, marketing, education, and transmit contents and values attractively and efficiently".

Educational Games for Learning,2014.

Gatzidis and al., say about the skills development: "Serious Games use game technology and design principles for purposes beyond pure entertainment, such as training coaches".

Towards the Development of an Interactive 3D Coach Training Serious Game,2009.

They say also in their other research in 2015 that serious game help in Awareness raising and social change: "Serious Games are video games designed to improve knowledge, skills, or attitudes in the real world, with potential applications in healthcare."

Serious Games: A Concise Overview on What They Are and Their Potential Applications to Healthcare,2015.

Marsh confirms that one of the principles purposes of SG is Design and development: "Serious Games exist on a continuum between traditional games and experiential environments, with the purpose of conveying meaning."

I.2.3) The Key differences from entertainment games

Serious games differ from entertainment games in both intent and application. As Michael and Chen (2006) explain, serious games are “designed for purposes beyond entertainment, such as education, training, or raising awareness”. In contrast, entertainment games primarily aim to provide fun and excitement, without additional objectives.

Zyda (2005) supports this distinction, stating that “serious games use entertainment as a means to achieve strategic objectives, such as professional training or education”, framing enjoyment as a functional tool rather than an end in itself.

Similarly, Alvarez (2007) notes that serious games “combine seriousness and playfulness to deliver an educational or interactive experience,” while entertainment games focus exclusively on providing an enjoyable experience without explicit learning goals.

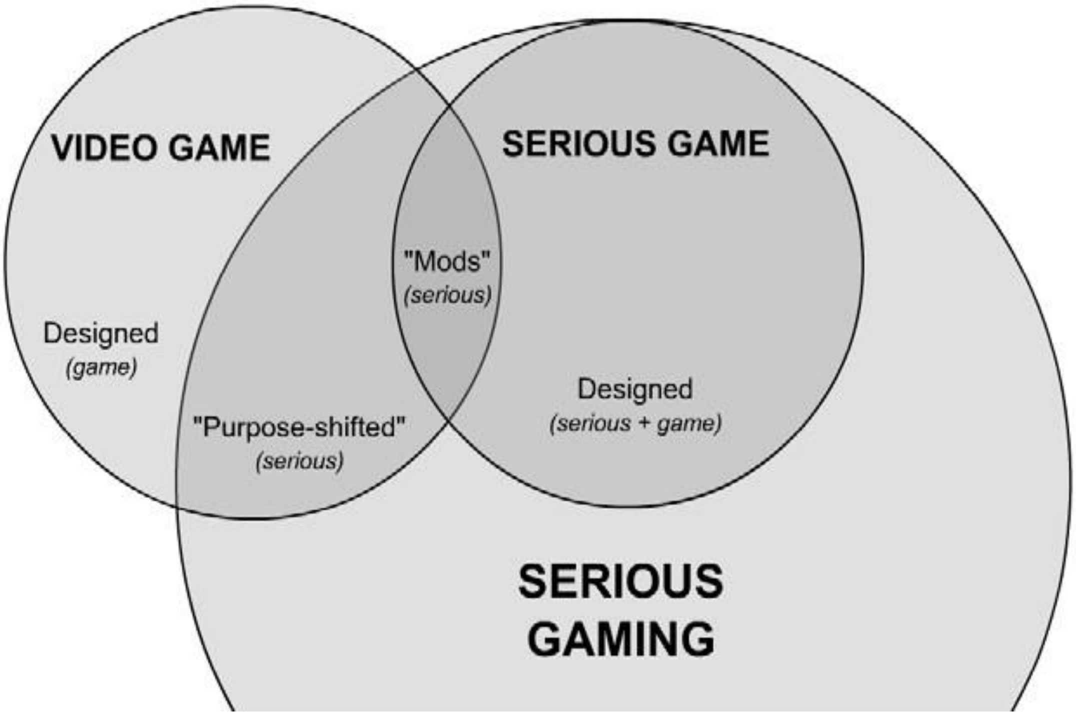


***Figure 2** Relationship between video game, serious game and serious gaming (Djaouti, Alvarez & Jessel, 2011, p. 4)*

This figure (figure 2) highlights the relationship between video games and Serious Games and shows that Serious Games are not completely separate from video games. We can use video games that are based on entertainment as Serious Games in two ways: "purpose-shifting" and "mods". For example, if we use the first technique, the teacher can add a serious scenario to a video game in order to reach a specific educational goal. This serious dimension is not directly integrated into the game, but the teacher uses it to influence the way his students play. Thus, we can consider that the "serious" and "playful" dimensions exist in this technique, and the difference between it and Serious Games lies in the design process. As for the second technique, it depends on introducing modifications to the video game in order to adapt it to the serious goals of the game. For example, Escape from Woomera is a software modification of the video game Half-Life. The main difference between mods and Serious Games is the relationship between the designers of each of the "serious" and "playful" dimensions. The designers of Escape from Woomera designed only the "serious" dimension that fits into a pre-existing "game" scenario.

Table 1 *Differences between entertainment games and Serious Games*

Aspect	**Serious Games**	**Entertainment Games**
Main Purpose	Achieving educational, training, awareness, or health-related goals.	Providing fun and entertainment to users.
Design and Content	Carefully designed to integrate educational or professional content with game elements.	Focused on enjoyable aspects like challenge, excitement, or competition.
Target Audience	Specific audiences (students, professionals, trainees) based on the game's purpose.	General audiences seeking entertainment.
Expected Outcomes	Enhancing skills, increasing knowledge, or raising awareness in a particular field.	Offering an enjoyable experience without direct educational goals.
Simulations	Assumptions necessary for workable simulations	Simplified simulation processes
Use of Technology	Leveraging technology to achieve practical and beneficial objectives.	Using technology to enhance the gameplay and entertainment.

Motivating Factors	Interaction with real-life scenarios or application of learned skills.	Challenges, competition, and exploration within the game environment.
Examples	Medical simulation games (e.g., virtual surgery), language learning games.	Adventure games (e.g., Minecraft), shooter games (e.g., Call of Duty).

I.3) History

The idea of Serious Games dates back to the 15th century with the Italian humanist movement, where the term "Serio Ludere" became famous, referring to the idea of dealing with a "serious" subject in a "fun" way. This idea was prevalent in literature and other forms of art, where a light and humorous tone was used to discuss important societal issues.

In the 19th century the Prussian army developed one of the earliest examples of Serious Games, "The American Kriegsspiel "a game invented by William Livermore and Hugh Brown in 1879, they are icons of variable units with indicators of strength, type, fatigue, ammo and mission time. It was used to train officers in strategic thinking and tactics.

The first computer war games were invented by the Army's Operations Research Office at Johns Hopkins University which was an "Air Defense Simulation" in 1948. There was also "CARMONETTE" which was a computerized Monte Carlo simulation in 1953.The war games have evolved as the following chart shows (figure 3):

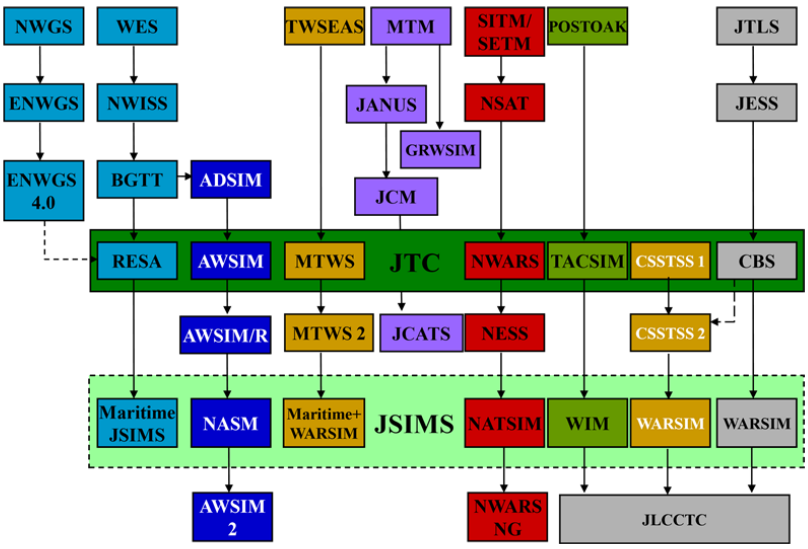


Figure 3 Military Computer Wargaming Evolution

The term Serious Games became popular thanks to the American researcher Clark C. Abt in his book in 1970, where he saw that card games, board games, and computer games (which were not well developed at the time) supported educational goals. He said:

“We are concerned with Serious Games in the sense that these games have an explicit and carefully thought-out educational purpose and are not intended to be played primarily for amusement” (Abt, 1970).

Abt participated in the development of TEMPER, one of the first computerized war games.

Other games appeared, such as the Greek "la petite" and the Roman "troncule"(Figure 4), which were war games whose primary purpose was not educational, but rather depended on thinking and planning, not chance.

Figure 4 *The game « la troncule »*

In 1980, Atari developed the first serious computer game, Army Battlezone (Figure 5), a game designed to train American soldiers. Over time and with the great development witnessed by the digital field, Serious Games have become used in fields such as education, health care and corporate training to achieve specific educational goals.

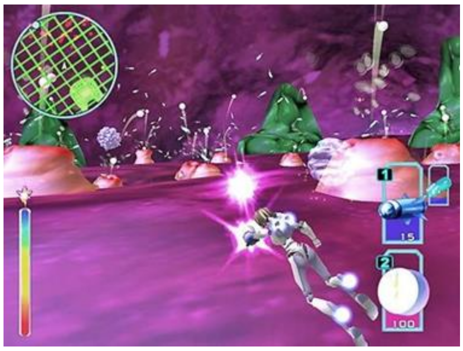
Figure 5 *First serious game «Army Battlezone»*

In 1985, one of the first Serious Games was invented, a modification of the flight program. Shooting interactions were added, a maximum of 10 players due to bandwidth limitations, and a time calculation was later added to reduce network flooding. (figure6)

Figure 6 *1985 Modification of Flight program*

Then games with political (like Peacemaker and Dean for America), health (like Re-Mission) and religious (like Left Behind and Interactive Parables) goals appeared.

Dean for America



Re-Mission

Interactive Parables

The following table summarizes the most important stations in the history of Serious Games.:

Table 2 Milestones in the history of Serious Games

Year	Serious game	Application
1970	Serious Games book by C. Abt	Academic book
1972	Magnavox Odyssey	Education
1973	The Oregon Trail	Education
1980	BattleZone	Training
1981	The Bradley Trainer	Training
1982/1983	Pole Position/Atari VCS 2600 console	Training
1996	Marine Doom	Military
2002	America's Army	Military
2003	DARWARS	Military
2005	VBS	Military
2006	BiLAT	Interpersonal communication
2009	VBS2/Game After Ambush	Military
2012	X-Plane 10	Training

I.4) Evolution of Serious Games

I.4.1) The history of Serious Games

In addition to the games Oregon Trail and SimCity, titles like Mavis Beacon Teaches Typing were now called "educational." These titles were supposed to teach children skills while they passed time, but not very often did that happen since the titles were almost commercial disasters.

I.4.2) The spike in CD-ROM sales (late 90s and early 2000s)

Educational software was more present on PCs in the '80s riding the crest of CDROM technologies, while some parents were actually trying to make the most out of the new technology for their children's learning on home computer.

I.4.3) Decline and disillusionment (2000-2010s)

The entertainment industry had little time for Serious Games , which were mostly left into neglect. There wasn't much if any new investment in the serious game business then.

I.4.3.1) Spike and growth (2010s)

- A lot of interest followed by funding from many foundations including MacArthur and Gates.
- Mobile applications were proving highly effective in the field-testing of

educational games.

- Government funding through SBIRs provided grants to nurture smaller indie

firms to develop learning games.

I.4.3.2) Now (2020s)

- Sustained growth with some downtimes, however.
- Costly development remains a challenge, and the established market has

nothing to lean back on.

- Educational games continue to be funded by the Department of Education and the like.

I.4.3.3) Future Plans

- An increased focus will be laid on informal learning and exercises aimed at skill-building, which would not be traditional in the classroom.
- Potential opportunities for enriched engagement do exist through through-the-wall VR/AR.

There is a demand for experimentation and a low-risk build-out of Serious Games .

I.5) Classification of Serious Games

Serious Games can be classified in multiple ways based on their purpose, design features the audience (player) they are directed at, and application domains.

Drawing on the foundational G/P/S model proposed by Djaouti, Alvarez, and Jessel (2011), this section refines the classification of serious games along three interrelated dimensions:

I.5.1) By Purpose

Firstly, the **purpose** dimension refers to the reason a game is created. Some serious games are designed to **disseminate messages** educational, persuasive, or subjective in nature while others serve to **train** users by enhancing specific skills. A third category involves **data collection**, where games are used as tools for gathering user inputs and behaviors, often for research or evaluation purposes (Djaouti, Alvarez & Jessel, 2011).

I.5.2) By sector

Secondly, the **sector** dimension considers the context in which the game is situated. Serious games are applied across a range of fields, such as **healthcare, education, and military**, and are tailored to particular audiences **students, professionals, or the general public** (Djaouti, Alvarez & Jessel, 2011).

I.5.3) By gameplay

Thirdly, the **gameplay** dimension differentiates between two modes of interaction: **video toys** and **rule-based video games**. Video toys provide open-ended, goal-free exploration and play, enabling users to experiment creatively. In contrast, rule-based games impose structured objectives, rules, and clear success criteria, delivering a defined experience that aligns with gameplay progression (Djaouti, Alvarez & Jessel, 2011).

I.6) Benefits and Challenges of Serious Games

I.6.1) Challenges

- Game-based learning has promising capabilities, but results are mixed. Overall, research on games and interactive simulations find that they result in higher cognitive gains as well as better attitudes towards learning than traditional methods. No single answer can be given regarding if educational technology in the form of games results in higher cognitive gain for learners. Differences between populations, computer usage, skills taught, and instructional designs are a few of the factors that may account for inconsistencies between results. So far, there is no univocal framework for classifying educational games, and studies have used incoherent wording and formulations that make comparisons between studies on games difficult to rela te to learning effects.

- Simulation-based learning and game-based learning focus on improving learning with inquiry processes, visualization, experimentation, and critical thinking problems. However, it has been shown in the literature that inquiry learning can be effective if supported, for instance, by feedback, scaffolding, and explanations.

- In a game environment, support is necessary for players to learn and build their knowledge. Interpretative support in games includes background information, suggestions of relevant variables, and feedback.
  Reflective support elicits reflection by players on specific aspects of their knowledge. However, during a game, players reach a state of flow, where self-consciousness disappears and time becomes distorted.

  This state of full immersion in the game positively influences learning performances and enjoyment.

  Addition of support to a game may disrupt game flow and diminish the motivating properties of the game.

  The work of Johnson and Mayer (2010) indicated the interaction of support and flow, where students allowed to choose an explanation from a list outperformed those who had to generate their own self-explanations.
  While support is thus needed to provide an advantageous learning environment, it needs to be carefully fitted into the environment so as not to disrupt game-flow and lose the motivational qualities of the game.

- Poor working memory (WM) in children may result from attention problems at a basic level of task orientation, which involves implicit processing. Attention is a process that involves selectively describing relevant information to current thought or behavior. The brain establishes task relevance through a cumulative process, which involves pre-activating brain cells in response to the selection of attention targets.

- Using Serious Games for learning can turn out to be rather costly, especially when you consider shrinking L&D budgets. Although there is a wide variety of eLearning authoring tools available, a low price point usually indicates minimal room for customization. If you want your serious game to bring impactful results to your organization, personalizing it to fit your workforce's learning needs is of utmost importance. Some organizations opt to create their own Serious Games from scratch. This would incur additional expenses, as you would need to hire graphic designers, game designers, and programmers, among others. Paired with the delays or mistakes that might occur, leaving some wiggle room in your budget is the only way your serious game development project isn't cut short.

- Most businesses are dynamic and diverse, employing people of different ages and backgrounds. Although that's great for innovation and productivity, it can make using a serious game for learning even more challenging. The L&D team must design the game to be accessible and easy to use by everyone, including those who are less tech-savvy. Accessibility for employees with disabilities is further necessary in view of equity and fairness in the working

environment. This may involve reworking the game to include screen readers, adding settings aimed at reducing visual clutter
or color adjustment, among others. The positive effects that these accommodations will have on your workforce's development will surely make the extra work these accommodations demand worth it.

I.6.2) Benefits

Playing learning apps is supposed to boost cognitive competences, motivation, and engagement skills to articulate human activities, the academic work, or professional practices, alongside enhanced social interactions, self-discipline, empowerment, collaboration, and decision making. They get to be experienced in the form of simulation and involve customization feedback, and will offer training under the observation of very realistic conditions. They offer flexibility in learning pace and style, an interactive learning environment, and may be cost-effective. Serious Games could change education and training by offering innovative and engaging learning experiences.

Arias-Calderón, Castro, and Gayol (2022) conducted a quantitative investigation titled *"Serious Games as a Method for Enhancing Learning Engagement: Student Perception on Online Higher Education During COVID-19"*, published on April 27, 2022. In this study, serious games were used alongside synchronous online classes in a physiology course at Universidad Andrés Bello for 108 second-year nursing students.

The questionnaire results revealed high levels of perceived gain in areas directly corresponding to the mechanisms listed in **Table 3** specifically **intrinsic motivation**, **situated learning**, and **reflection through mistakes** as students reported positive enjoyment, contextual understanding, and adaptive learning experiences. Quantitative measures reported in the study align with the outcomes presented in **Table 3**, where over 63% of students agreed that the games increased their motivation and engagement, and more than 60% noted improved understanding and meaningful learning

Table 3 *Percentage of responses to questions on attitudinal competencies*

Questions	Strongly agree	Agree	Neither agree nor disagree	Disagree	Strongly disagree
Playing the Serious Games increased your	54.6%(59)	33.3%(36)	8.3%(9)	3.7%(4)	0.0%

motivation to study					
Serious Games improve your engagement in the learning process	63.0%(68)	27.8%(30)	8.3%(9)	0.9%(1)	0.0%
Performing Serious Games improve your confidence in concepts learned	60.2% (65)	25.9% (28)	7.4% (8)	5.6% (6)	0.9% (1)

Performing serious game activities also improved students' study habits, with a majority feeling that it helped optimize their personal study time (see Table 4). Furthermore, the study assessed the impact on academic performance, revealing that students perceived the games as beneficial for understanding associated concepts and preparing for assessments, as detailed in Table 5.

Table 4 *Percentage of responses to questions on self-efficacy*

Questions	Strongly agree	Agree	Neither agree or disagree	Disagree	Stongly disagree
Playing the Serious Games improves your concentration during self-study	44.4% (48)	33.3% (36)	18.5% (20)	2.8% (3)	0.9% (1)
Performing serious game activities optimize personal study time	55.6% (60)	26.9% (29)	13.9% (15)	1.9% (2)	1.9% (2)

Your concentration and class attention improves by performing serious game activities	42.6% (46)	31.5% (34)	19.4% (21)	2.8% (3)	3.7% (4)

***Table 5** Percentage of responses to questions on academic performance*

Questions	Strongly agree	Agree	Neither agree or disagree	Disagree
Playing the Serious Games improve the learning of the associated concepts	59.3% (64)	31.5% (34)	8.3% (9)	0.9% (1)
Performing Serious Games facilitate doing seminar activities	36.1% (39)	24.1% (26)	27.8% (30)	11.1% (12)
Performing Serious Games facilitate study for weekly assessments	32.4% (35)	25.0% (27)	29.6% (32)	11.1% (12)
Performing Serious Games facilitates study for final assessment	40.7% (4)	32.4% (35)	13.9% (15)	7.4% (8)
Performing serious game activities was a significant learning experience	60.2% (65)	25.0% (27)	13.9% (15)	0.0% (0)
Do you feel well prepared for unit assessment by performing serious games	44.4% (48)	29.6% (32)	14.8% (16)	8.3% (9)

I.7) Examples of Serious Games

I.7.1) KangaZoo

I.7.1.1) Overview

KangaZoo is a mobile simulation game developed by Chaos Theory Games to raise awareness of Australian wildlife and conservation efforts. As a park ranger, the player must rescue hurt animals and document the native plant life.

Figure7 kangaZoo game

I.7.1.2) Gameplay Mechanics

- **Exploration**: Navigate diverse Australian ecosystems (e.g., rainforests, deserts).
- **Rescue Missions**: Treat injured animals using realistic veterinary techniques.
- **Educational Modules**: Mini-games teach players about endangered species and conservation strategies.

I.7.1.3) Objectives

- To educate players about Australian wildlife and conservation strategies.
- To foster empathy for endangered species and encourage real-world conservation efforts.

I.7.1.4) Audience

- **Primary**: School-aged children and families.
- **Secondary**: Global audiences interested in wildlife conservation.

I.7.1.5) Domain

- **Education**: Aligns with Australia's National Curriculum for Science.
- **Conservation**: Promotes awareness of endangered species and ecological stewardship.

I.7.2) The Great Flu: Pandemic Awareness Game

I.7.2.1) Overview

"In an effort to further raise awareness, Dutch researchers have created a game that challenges players to control a new pandemic.

The game begins with images of bedridden patients and graveyards from the 1918 Spanish flu. As the head of the fictitious "World Pandemic Control,"

Figure 7 *The Great Flu: Pandemic Awareness Game*

players pick a flu strain, and then monitor that strain's spread around the world."
[The Great Flu | Center for Digital Games Research - UC Santa Barbara]

I.7.2.2) Objective

- To educate players about pandemic response strategies and public health principles.
- To raise awareness of historical pandemics like the 1918 Spanish Flu.

I.7.2.3) Gameplay

- **Virus Customization:** Select strain characteristics (e.g., transmission rate, lethality).
- **Resource *Management*:** Allocate medical supplies and enforce quarantines.
- **Real-Time Data:** Track global infection rates and mortality statistics.

I.7.2.4) Audience

- **Primary:** High school students and adults.
- **Secondary:** Public health professionals and educators.

I.7.2.5) Domain

- **Public Health:** Used in epidemiology courses to teach pandemic response.
- **History:** Highlights historical pandemics and their impact.

I.7.3) IBM CityOne: Urban Management Training

I.7.3.1) Overview

IBM CityOne (2010) is a strategy game where players solve real-world urban challenges related to energy, water, banking, and retail.

Figure 8 *IBM CityOne: Urban Management Training*

I.7.3.2) Objective

- To teach players about sustainable urban planning and resource management.
- To encourage creative problem-solving for complex urban issues.

I.7.3.3) Gameplay

- **Crisis Scenarios:** Address issues like water shortages or economic downturns.
- **Budget Management:** Balance financial constraints with sustainability goals.
- **Collaboration:** Multiplayer mode allows teams to design city-wide solutions.

I.7.3.4) Audience

- **Primary:** High school students and adults.
- **Secondary:** Public health professionals and educators.

I.7.3.5) Domain

- **Public Health:** Used in epidemiology courses to teach pandemic response.
- **History:** Highlights historical pandemics and their impact.

I.7.4) Duolingo: Gamified Language Learning

I.7.4.1) Overview

Since 2012, Duolingo has offered a gamified approach to language learning, allowing new languages to be learned in a fun and engaging manner. Spanning over 500 million users worldwide, it offers courses in such 40-plus languages as the endangered and artificial languages of Navajo and High Valyrian.

Figure 9 *dualingo game logo*

I.7.4.2) Objective

- To democratize language learning by providing free, accessible education.
- To motivate consistent practice through gamified elements like streaks and rewards.

I.7.4.3) Gameplay

- **Interactive Lessons:** Short exercises include translation, listening, speaking, and matching tasks.
- **Gamification Elements:**
    - **Streaks:** Track consecutive days of learning to encourage daily practice.
    - **Leaderboards:** Compete with other learners in weekly challenges.
    - **Lingots (In-App Currency):** Earned by completing lessons and used to unlock bonus content.
- **Adaptive Learning**: Adjusts difficulty based on user performance, though grammar explanations are minimal.

I.7.4.4) Audience

- **Primary:** Students, casual learners, and travelers.
- **Secondary:** Educators and institutions using Duolingo for Schools.

I.7.4.5) Domain

- **Education:** Aligns with language curricula and supports classroom learning.
- **EdTech:** Pioneers gamification in digital education, influencing broader trends in Serious Games .

I.8) The difference between gamification, Serious Games and game-based learning

Table 6 *The difference between gamification, Serious Games and game-based learning*

Aspect	**Gamification**	**Serious Games**	**Game-Based Learning (GBL)**
Definition	Use of game design elements in non-game contexts	Games designed for purposes beyond entertainment	Learning happens through the use of games
Primary Purpose	Motivation, engagement, behavior change	Education, training, awareness, change	Learning knowledge, skills, attitudes
Use of Games	No full games – only elements (points, badges, etc.)	Full games with a serious/educational intention	Can use any game (serious, commercial, etc.)
Design Intent	Not necessarily educational	Designed intentionally with educational or other serious goals	Focus on learning; not necessarily through games built for learning
Educational Setting	Can be used in education but not limited to it	Includes education but also extends to health, politics, advertising, etc.	Primarily educational (formal or informal)
Learner Role	Learner is influenced through motivational game-like structures	Learner engages with a game as a medium for achieving non-entertainment goals	Learner plays or creates games as a means of learning
Examples	Points, badges, leaderboards in a course or app	Games for health (e.g., surgery training), games for change (e.g., social justice)	Using “Civilization” to teach history or designing a game to learn programming
Theoretical Focus	Behavior design, motivation theory	Depends on domain (e.g., instructional design, simulation)	Learning theory (constructivism, experiential learning, etc.)
Requires a Game?	No	Yes	Yes, but can be any game – not necessarily designed for learning
Related Concepts	Learner-centered design, choice, narrative wrapping	Games for Learning, Educational Games, Simulations	Digital Game-Based Learning (DGBL), Game Creation, Digital Game Pedagogy (DGP)

I.9) Conclusion

Serious Games have evolved significantly from their early conceptualization in the 1970s to their widespread applications today in education, healthcare, business, and military training. Unlike traditional entertainment games, Serious Games integrate engaging gameplay mechanics with specific learning or training objectives, ensuring that users gain valuable knowledge and skills while remaining immersed in the gaming experience.

Various definitions provided by researchers highlight the fundamental balance between entertainment and pedagogical intent in Serious Games. While some definitions emphasize their serious purpose, others acknowledge that entertainment plays a crucial role in engagement and motivation. Furthermore, the classification of Serious Games demonstrates their versatility, ranging from educational and health-related applications to business simulations and public awareness campaigns.

The benefits of Serious Games include enhanced learning engagement, improved knowledge retention, and the ability to simulate real-world scenarios in a risk-free environment. However, challenges such as high development costs, accessibility concerns, and the need for careful integration of pedagogical elements into gameplay remain significant considerations for developers and educators.

Overall, Serious Games represent a powerful tool for education, training, and behavioral change. As technology continues to advance, their role in various fields will likely expand, leading to more innovative and effective applications. Future developments in artificial intelligence, virtual reality, and adaptive learning could further enhance the impact of Serious Games, making them an essential component of modern digital learning and training strategies.

Chapter II. Serious Games and Learning

II.1) Introduction

This chapter explores the connection between learning theories and Serious Games, highlighting the educational attributes that make them effective learning tools. It also examines the Structural Components of Serious Games. Additionally, we will discuss the fundamental qualities of a good serious game and explore methods for evaluating their effectiveness in achieving educational goals.

II.2) Learning theory and Serious Games

One of the most important elements in the quality of the game is the reliance on the learning process, the rules of the video game are generally very simple, but what makes them seem complex is that they are unknown to the player, as the player embarks on the adventure and discovers strategies in a gradual learning process that makes him feel superior to himself as he becomes stronger, understands more and more, and establishes himself as the hero of the game, but this is all a trick, the game designer has hidden the rules and changes them dynamically according to the player's progress and competence. However, the learning process enhances players' skills and keeps them motivated and interested (Natkin, n.d.).

This principle, which is familiar to all game designers, is summarized by the so-called learning and difficulty curve (Figure 11): at any given moment, the player's level of learning enables him to cope with the current difficulty by using a minimum of deduction or agility. When he overcomes it, he acquires new knowledge and must face a new level of difficulty which, once again, forces him to surpass himself.

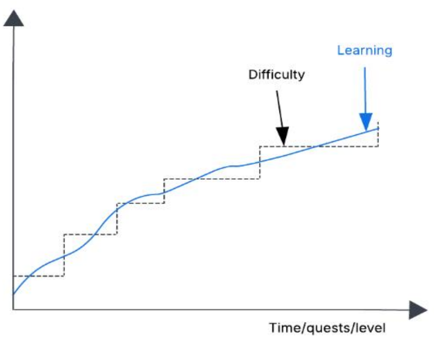


Figure 10 *The (ideal) difficulty and learning curves according to S. Natkin*

In his article, Jesper Juul (2005) sums up this principle using the strategy repertoire theory: ‘At any given moment in a game, a player has created a set of strategic methods and rules that he has devised and applies (the player's repertoire). A strength of a good game is its ability to constantly challenge the player, leading him to constantly find new strategies from those in his repertoire. A bad game is one in which the player is either unable to refine his repertoire, or has a dominant strategy (one that wins every time) and therefore has no need to improve his repertoire’.

This principle can also be explained according to the theory of intrinsic motivation, or more precisely the theory of flow: ‘The concept of flow describes the state of an individual who is completely immersed in the present, who directs all his sensory, mental and motor abilities towards the fulfilment of a very precise activity.’ (Csikszentmihalyi, 1990).

Here are some examples: Athletes while competing, artists while creating, the reader who is stimulated by description or revelation to imagine, the focused chess player, programmers while writing code...

The experience of flow occurs when there is a balance between the challenge proposed and the individual's skills in meeting it; we can speak of mastery in difficulty or, to use Csikszentmihalyi's terms, of ‘optimal challenge’. If there is an imbalance between the challenge and the skill, we move into states that are more or less far removed from flow: curiosity, anxiety, apathy, etc.’. The above principle also assumes that the player remains constantly ‘immersed’ in the game world. This is what Huitzinga (1950) calls the magic circle: the space and time in which the rules of the game surpass those of life. It's also linked to the notion of presence: the player loses awareness of the material device of the game (chess table or computer screen), and perceives only its unfolding.

Educational games or any interactive game try to exploit this mechanism, which is very immersive and addictive. We conclude from the above that a ‘good’ serious game should not abandon the principles that form the basis for writing the game, but this mechanism should be carefully guided, aimed at acquiring skills or knowledge (Rahdoun & Djabbar, 2013).

II.3) Educational attribute of Serious Games

Table 7 *Educational attribute of Serious Games*

Attribute	Description
Active Learning	Serious Games encourage learners to interact with the content, make decisions, and solve problems, leading to a deeper understanding of the educational material. Al-Haila (2002) states that ***'play in general and educational games in particular are considered an essential entry point for the development of the child in psychological, physical, mental, social, and various linguistic and emotional skills, as it develops thinking and allows the adoption of social roles.'***
Motivation and engagement	Serious Games are characterized by their interactive and engaging nature that increases learners' motivation and makes the learning process more enjoyable and effective. Al-Huwaidi (2013) said***: 'Educational games contribute to increasing learners' motivation by providing fun and stimulating challenges.'***
Immediate feedback	Many Serious Games provide immediate feedback, allowing learners to understand the results of their actions and make necessary adjustments.
A safe environment for experimentation	Games provide a low-risk environment where learners can experiment and

	learn from mistakes without real consequences. Al-Huwaidi (2013) points out that ***'play is a tool of learning and exploration, a tool of compensation and expression'.***
Contextual learning	By placing learning in a meaningful context or story, Serious Games help learners apply knowledge in practical scenarios, enhancing the retention and transfer of skills.
Co-operation and social interaction	Serious Games play a big part in encouraging teamwork and communication, helping learners develop social and collaborative skills.
Assessment and progress monitoring	Many Serious Games have built-in assessment tools that allow learners' progress to be monitored, providing valuable data for both learners and teachers to help guide learning processes. Alkhodr (2017) said that***: 'Educational games provide continuous assessment tools that help measure learner progress and identify strengths and weaknesses, helping to optimize the learning process.'***

II.4) Core Elements of Serious Games

II.4.1) Serious Games Components

Serious Games are built on the premise that they help achieve educational objectives. The learning objective must be defined to make sure the game mechanics and content lead to the competencies we want to create. Often, pedagogical theories, for example, constructivism or Bloom's Taxonomy, are embedded to assist with skill development (Yusoff et al., 2009). Games, for example, might build challenges that allow players to move from simple recall of facts to analytical thought (Shute, 2008).

II.4.1.1) Game Mechanics and Dynamics

For engagement, game mechanics (rules, challenges, rewards and so on) and dynamics (player interactions) are crucial. It is also important to note the MDA framework (MechanicsDynamics-Aesthetics) from Hunicke et al. (2004) is oft-adapted because a balance of fun and functionality is needed. Mechanics such as points, levels, and quests are designed to promote and reinforce learning (Deterding et al., 2011).

II.4.1.2) Narrative and Storytelling

Narrative frameworks enhance immersion and contextualize learning. Dickey (2005) emphasizes that narratives in educational games provide meaning for tasks and stimulate emotional engagement. For example, a game for learning history can use a time-travel narrative to place historical problem-solving.

II.4.1.3) Feedback Mechanisms

Immediate and formative feedback is essential for learning. Shute (2008) identifies feedback loops that adjust difficulty or provide progress indicators, helping users correct mistakes and stay motivated. Positive reinforcement (e.g., badges) and corrective cues are common strategies.

II.4.1.4) Assessment and Evaluation

Embedded assessments quantify learning outcomes, which are usually performed using in-game analytics or quiz-like gameplay follow-ups. Bellotti et al. (2013) introduce the concept of stealth assessment in which player action is gathered non intrusively to assess proficiency. External measurements can also serve to verify efficacy (Calderón et al., 2020).

II.4.2) Structural Components of Serious Games

II.4.2.1) Game World

The game world refers to the virtual environment in which all gameplay interactions take place. It provides the spatial (e.g., landscapes, rooms, structures) and thematic coherence (e.g., futuristic city, historical setting) that give the player a sense of immersion and context. A well-designed game world is not just a visual backdrop; it supports the game narrative and mechanics by enabling meaningful exploration and interaction.

According to Salen and Zimmerman (2004), the game world acts as a representational space that players navigate and interpret through play. The structure of the world whether open-world, linear, or modular shapes how the player experiences the game and engages with its challenges. They emphasize that a coherent world reinforces player agency by offering environmental cues, spatial logic, and consistent feedback, all of which contribute to the game's systemic integrity.

II.4.2.2) Characters

Characters in Serious Games serve both narrative and interactive functions. These include player avatars, companions, enemies, and non-playable characters (NPCs). Characters are central to emotional engagement, as they humanize the experience and often carry the storyline, delivering quests, challenges, or critical information.

Gee (2007) argues that well-designed characters help establish a "projective identity" players project themselves into the character while simultaneously reflecting on that identity. This duality enhances learning, as players are more likely to internalize lessons or knowledge acquired through emotionally meaningful experiences. Moreover, characters can serve as learning agents, modeling behaviors or prompting reflection through dialogue and feedback.

II.4.2.3) Technological Infrastructure

Platform choices (e.g., VR, mobile) influence accessibility and interactivity. Westera et al. (2008) stress that technology must support scalability and real-time interactions without compromising pedagogical goals.

II.4.2.4) User Interface (UI) and Experience (UX) Design

Intuitive UI/UX ensures usability. Fabricatore and López (2012) note that clear navigation, responsive controls, and accessibility features (e.g., subtitles) reduce cognitive load, allowing users to focus on learning.

II.4.2.5) Levels and Missions

Levels and missions constitute the progression structure of a serious game. Each level presents a distinct set of objectives, challenges, or tasks that the player must complete to advance. This modular structure allows designers to gradually introduce new content, mechanics, or complexity, maintaining engagement and avoiding cognitive overload.

Zyda (2005) highlights that mission-based design is not only motivational but pedagogically effective. By offering incremental learning steps, missions can scaffold knowledge acquisition and reinforce concepts through repetition and variation. Moreover, levels can be aligned with Bloom’s taxonomy, progressing from simple recall to complex problem-solving tasks. This staged approach ensures learners remain in a state of flow, where the challenge matches their skill level.

II.4.2.6) Rules System

The rules system defines the logic and constraints of the game. It determines what players can and cannot do, how they win or lose, and how interactions unfold. Rules create a closed system in which players operate, offering structure, predictability, and meaningful consequences.

Salen and Zimmerman (2004) describe rules as one of the fundamental building blocks of games. They distinguish between operational rules (explicit instructions players follow), constitutive rules (underlying logic or mathematical models), and implicit rules (socially agreed-upon behaviors, like fair play). In Serious Games, rules serve an additional function: they can encode educational objectives. For example, requiring players to troubleshoot errors in a simulation reinforces problem-solving skills and mirrors real-world logic.

II.5) Serious Game Underlies "A Pedagogical Scenario"

To differentiate video games from Serious Games from a computational perspective, we refer to the writings of Zyda (2005), who posits that a video game is defined by its "story, art, and software."

He elaborates that Serious Games integrate, in addition to these three components, a pedagogical dimension: "Serious Games, however, are more than just story, art, and software. They involve pedagogy: activities that educate or instruct, thus disseminating knowledge or skills. This addition renders the games serious."

Figure 11, which reflects Zyda's (2005) schematic, illustrates the pedagogical dimension that complements the video game to create a serious game. This approach aligns with the views of André Tricot (1999), a professor of psychology at the University of Toulouse Le Mirail and the IUFM (Instituts Universitaires de Formation des Maîtres) of Toulouse, who studies and develops educational applications. In his interview, Tricot (1999) highlights the necessity of establishing a "pedagogical scenario" and a "usage scenario" for designing a computer application dedicated to "a pedagogical objective."

Tricot (1999) clarifies that this "pedagogical scenario" should not be set parallel to the game. Both components must be coherently integrated. According to Zyda (2005), "A production team must work closely with the design team to ensure proper integration of the pedagogical aspect." Similarly, Tricot (1999) emphasizes, "Both levels must be perfectly coherent with each other."

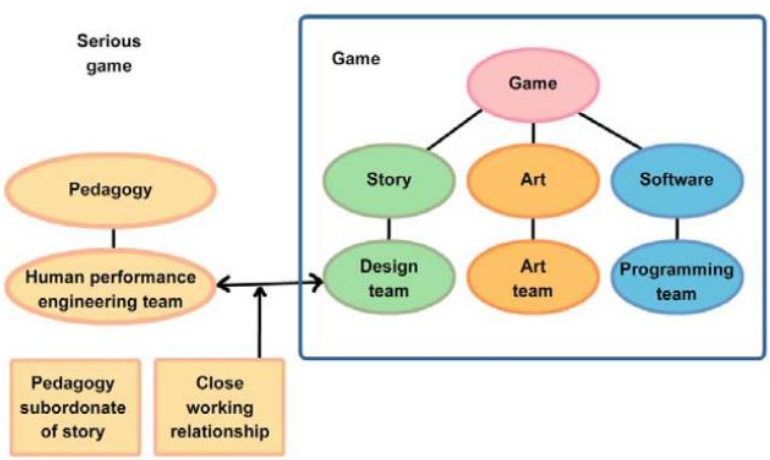


Figure 11 *Diagram showing the link between serious gaming and the pedagogical component of video game development (diagram reproduced by kind permission of M.Zyda)*

II.5.1) Defining the Pedagogical Scenario

Continuing to delineate the serious game, Tricot (1999) offers his definition: "There is a level I would call 'the usage scenario.' It refers to what makes a child (a user) engage in this game space? In this reading space? In this question/answer space? What makes them interact with a machine? Ensures that this interaction interests them? Is easy for them? Doesn't discourage them? This is the level that cannot be missed (It's the video game aspect in the case of a serious game). But to succeed in an educational product, meaning one whose goal is not to play but to reveal possibilities, a second level is needed. For simplicity, I call this 'the pedagogical scenario.' It encompasses how you will achieve an effect on the user. This effect could be awakening their curiosity, leading them to self-question, stimulating their desire to explore, etc. It must succeed in reaching this pedagogical objective. What will you implement to achieve this?"

II.5.2) Defining the "Pedagogical Objective"

Having identified different types of Serious Games in the first chapter, which seem to have different purposes, the focus here is not on the learning context (military, medical, political applications, etc.), but on the type of learning: Is it about teaching knowledge, a practice, or both? Teaching knowledge involves conveying information or book-type knowledge. This term doesn't imply that this type of knowledge is exclusive to reading. For instance, a video report can convey such knowledge. Teaching practice involves training or exercising the learner in an activity that could be physical or intellectual.

These two types of teaching are not confined solely to educational or training settings. With this approach to the concept of learning, different applications are grouped according to the criteria "teaching knowledge" and "teaching practice":

Military games and health games generally prioritize teaching practice. Titles with military connotations, such as America's Army, Battlezone (The Bradley Trainer), Game DIS, and Steel Beasts Professional, as well as applications potentially for health improvement, such as Dr. Kawashima's Brain Training or Wii Fitness, are primarily aimed at enhancing psychomotor performance: improving movement finesse and speed, and rapid response to psychotechnical questions.

Advergames, informative games, and activist games primarily focus on teaching knowledge. This knowledge is often informative, promoting a brand (e.g., Pepsi Invaders, Kool Aid Man) or advocating for a cause (e.g., McDonald's Videogame) or addressing a political issue (e.g., Kabul Kaboom!). This message can combine information and book-type knowledge by addressing social issues (e.g., AIDS and Us, Earthquake in Zipland).

Edugames often combine both types of teaching: knowledge and practice (e.g., The Oregon Trail, Lemonade Stand, Auto junior).

II.6) Fundamental Qualities of a Good Serious Game

II.6.1) introduction

Serious Games are educational tools that integrate video game technology with pedagogical objectives. Serious Games are used in many fields like education, professional training, medicine, and defense to provide skills and knowledge transfer in an engaging and interactive manner. A good serious game comprises a balance between instructional content and entertaining gameplay and meets a range of essential criteria that enable the overall serious game to be successful.

II.6.2) Balance Between Learning and Entertainment

One of the fundamental challenges in creating a serious game is finding a proper balance between its instructional objectives and level of entertainment. Too much focus on instructional content can render the gaming process boring, while too much prioritization of entertainment can compromise its instructional objectives. The ideal serious game succeeds in integrating learning objectives with enjoyable activities and challenging environments and thereby sustaining the interest of players and enabling them to achieve their instructional objectives. (Caserman et al., 2020; Charsky, 2010).

II.6.3) Realism and Immersion

Immersive experience lies at the heart of the efficacy of learning acquired through gaming. The integration of real-world situations and immersive environments frequently augmented by virtual or augmented reality enables the game to present an experience that is realistic and applicable. Such a depth of immersion not only commands the attention of the player but also fosters a more profound comprehension of the subject matter. (Ravyse et al., 2016).

II.6.4) Interaction and Feedback

Effective interaction and instant feedback are key elements of an effective serious game. Players should have to make decisions and, at the same time, experience the effects of their choices. Real-time feedback, provided in the form of scores, step-by-step explanations, or performance metrics, enables learners to identify their progress, modify their strategy, and comprehend the concepts taught better. (Ravyse et al., 2016; Abdellatif et al., 2018).

II.6.5) Adaptability and Artificial Intelligence

An effective serious game should be adaptable to the individual needs of each learner. By leveraging artificial intelligence, these games can dynamically adjust their difficulty based on player performance, offering personalized learning pathways. For instance, a math-focused educational game might present more challenging exercises to advanced players while providing additional support to those who struggle, thus optimizing the learning experience (Ravyse et al., 2016).

II.6.6) Choices and Challenges

There is a need to sustain the diversity in choices and difficult tasks gradually to retain players. If players have freedom of choice and to try alternative paths in the game, their intrinsic motivation is stimulated. This process of interaction not only promotes active learning but also allows for sustaining interest in the long run, as the game keeps changing to offer the right challenge. (Charsky, 2010; Martin & Shen, 2011).

II.6.7) Usability and Accessibility

A well-designed serious game must be user-friendly and usable for a wide variety of individuals, including those lacking extensive game-playing or technical experience. The use of an accessible and straightforward interface, with uncomplicated instructions, allows for the user to attend to learning objectives without being thwarted by technical problems. In addition, accessibility considerations in the form of including subtitles, control modifications, and varied audio-visual alternatives are instrumental in involving students with different needs. (Abdellatif et al., 2018).

II.7) Evaluation of a Serious Games

To design an educational game is just one part of many; it is just as important to choose tools that allow one to effectively determine and evaluate its educational value. This applies especially for such games since they can be created with detailed educational goals in mind or focus on developing certain skills in the players. Without appropriate evaluation instruments it tends to be impossible to assess if those goals are being achieved efficiently.

There are gaps in existing evaluation methodologies as well as frameworks because strategies do not fully capture the essence of the need in this case. Michela Mortara alongside Chiaro Eva Catalano and Angel Marco Luccini have identified the most common frameworks for extending the use of modern technologies in Serious Games as well as in education. Some of these are KLEAD The TILT, CIAO! and Flaslight frameworks that try to assess the impact of technology on learning. Of note too has been the development of the Game Object Model aimed at assessing the applicability of video games for learning purposes such as their "educational utility".

However many of these tools have proven to be less effective than expected, leading researchers to identify significant gaps in their utility and call for the development of more robust evaluation methods.

The significance of evaluation tools cannot be stressed enough as they are key components in revealing design imperfections and validating whether the game achieves its educational objectives as such increasing its usefulness for learning purposes.

Some of the significant methods and tools are outlined below, which are divided into two broad categories:

II.7.1) Quantitative tools

Quantitative evaluation is critical in determining how sophisticated games work. This involves the systematic collection and analysis of numerical data to define specific consequences, such as improvement in learning, engagement, and changes in behavior. Such an approach gives objective, quantifiable evidence that can be statistically analyzed for the effect Serious Games have.

II.7.1.1) Pre/Post Game Questionnaires

Post-game and pre-game questionnaires are crucial in determining the knowledge a player of a game had before engaging with the game as well as the knowledge they've acquired through playing the game. These questionnaires are commonly used in a variety of fields which include education and game design alongside interviews as they are some of the most effective ways of assessing knowledge a participant had prior to the game and the learning obtained after playing the game. (Catalano et al., 2014)

According to a study by (Belotti et al. ,2013), these questionnaires are a standard tool in educational research to measure learning outcomes.

Despite their effectiveness in assessing the acquisition of basic knowledge, before/after questionnaires often struggle to measure complex cognitive processes and higher thinking skills.

II.7.1.2) Activity logs and tracking

Serious Games , designed for educational purposes, often include activity logs and tracks to analyze player interactions and performance. These logs can provide valuable data for learning analytics, helping to understand how players interact with the game and how their performance relates to their actions in the game. This method is also non-intrusive, like its predecessors, and frequently cited in studies supporting methods intended to minimize disruption of gameplay (data collection should interfere as little as possible with the gameplay experience).

Taking the example of the Tracking Students' Activities in Serious Games study, it found that game tracking logs typically include parameters such as the number of clicks, frequency of use of the tool and duration of the interaction, which can be interpreted as behavioral indicators. For example, the rate of mouse clicks can indicate different types of player activity, such as confusion or enthusiasm.

II.7.2) Qualitative tools

Qualitative evaluation permits an in-depth understanding of the user experience, the user's perceptions, and the contextual factors influencing gameplay. Unlike their quantitative counterpart, whose focus is on numerical data, qualitative methods emphasize the understanding of "why" and "how" the user behaves and acts, thus specifically suited for capturing those subjective, subtle characteristics of all games and play.

II.7.2.1) Interviews

Interviews provide a rich and powerful means to gain insights into the diverse experiences students encounter in learning contexts, thus giving rise to the possibility of understanding their motivational influences, feelings, and understandings that could not be mirrored through statistical numbers. Interviews permit one to examine the "how" and "why" dimensions of learning, which, in turn, enable teachers and researchers to look into the personal and contextual factors that shape knowledge construction, self-reflection, and application of skills in real-world contexts.

An illustrative example of this is Calderón and Ruiz's work to conduct structured interviews with students participating in a serious game based on teaching programming concepts. Results indicated that not only could learners use their knowledge to solve real-world problems, but they also vocalized different insights regarding motivation, challenges, and level of engagement. In addition to these accounts, which could hardly have surfaced had quantitative data alone been employed, they show the depth of understanding that qualitative methodologies such as interviews can bring about. Combining the qualitative and quantitative approaches enables researchers to interpret the processes of simulation into wider understanding contexts with respect to their influences on student learning and performance.

II.7.2.2) Live observation

Live monitoring can help measure the effectiveness of Serious Games by tracking player interactions and learning outcomes. This methodology involves monitoring time-closed events that describe simple interactions between the player and the game, which are then sent to a central server for further analysis. The game itself does not make any assessment calculation: only raw events are sent to the server. Events are raw interactions rather than results, providing flexibility and the ability to recalculate points if necessary.

II.7.3) Frameworks for Evaluation

Evaluating the effectiveness of Serious Games requires structured approaches that consider both the educational and entertainment aspects of the game. Many frameworks exist to support this process, each detailing a different philosophy toward the assessment of Serious Games . Two of the most prominent frameworks are as follows:

II.7.3.1) The Four-Dimensional Framework

Developed by Kiili (2005), the Game-Based Learning Evaluation Model emphasizes the alignment between game design and learning objectives. This framework is based on the idea that Serious Games should balance engagement and educational value. Key components of the model include:

- **Learning**

Assessing the achievement of learning objectives is actually a dimension that evaluates the educational potential of a game. The performance of the game is measured in the way that it encourages knowledge acquisition, skills development, and a general transfer to real-life contexts.

- **Immersion**

This dimension evaluates the extent to which players are engaged and absorbed in the game. It considers factors such as narrative depth, visual and auditory design, and the overall player experience.

- **Social Interaction**

This dimension focuses on the collaborative and competitive aspects of the game. It examines how well the game supports communication, teamwork, and social learning among players.

- **Usability**

This dimension assesses the ease of use and accessibility of the game. It considers factors such as interface design, navigation, and the overall user experience.

II.7.3.2) The Game-Based Learning Evaluation Model

The Game-Based Learning Evaluation Model propounded by Kiili (2005) serves as a discussion on game design and learning objectives. The framework emphasizes that A serious game should maintain a balance between engagement and educational value. It consists of the following main components.

- **Learning objective**

This model begins with a careful definition of the learning objectives to be manifested within the game as emphasized during the design and evaluation of games.

- **Game design**

This model includes an evaluation of how the mechanical, narrative, and feedback systems of game design support learning goals (ensuring the game is engaging and educational).

- **User experience**

In the model proposed, players' interactions with the game, as well as the player motivation and enjoyment factor, are evaluated. This will include immersion and feedback.

- **Learning results and assessment**

This comprises assessments of learning-also referred to as measuring-the realization of key learning goals through pre-and post-testing, observation, or self-report.

II.8) Conclusion

In short, an effective serious game must integrate educational objectives and entertainment invisibly. It must be immersive and realistic, allow for meaningful interaction with real-time feedback, and be adaptive to the specific requirements of its users. Usability and accessibility are also essential to achieving broad impact. As technology further advances especially in artificial intelligence and virtual reality Serious Games will have an increasingly important function to fulfill in educational environments and professional training programs.

Chapter III. Arduino

III.1) Introduction to the Arduino Platform

The Arduino platform, an open-source electronics system, was developed in 2005 at the Interaction Design Institute Ivrea, Italy, by Massimo Banzi, David Cuartielles, Tom Igoe, Gianluca Martino, and David Mellis to provide an affordable and user-friendly tool for students and hobbyists to create interactive devices (Arduino, n.d.; Banzi & Shiloh, 2014). Designed as an alternative to costly platforms like Basic Stamp, Arduino simplifies prototyping with its open-source hardware and software, based on programmable microcontrollers and the Arduino Integrated Development Environment (IDE). Priced at approximately $30, Arduino boards are cost-effective for educational and individual use (IEEE Spectrum, 2011). The IDE, using a simplified C++-based language, enables beginners to quickly learn programming, supported by a vibrant community offering tutorials and forums (Arduino Project Hub, n.d.).

Arduino's educational efficacy stems from its accessibility, allowing novices to undertake projects with minimal prior knowledge (Milani et al., 2020). It provides immediate feedback through tangible outputs like LEDs or motors, enhancing learner engagement (Nugroho et al., 2019). Its alignment with real-world applications, such as industrial prototyping and Internet of Things (IoT) systems, motivates students by demonstrating practical relevance (Arduino Education, n.d.). Additionally, Arduino supports project-based and constructivist learning, fostering active knowledge construction through hands-on activities (Milani et al., 2020; Nugroho et al., 2019).

III.2) History and Evolution of Arduino

The Arduino project, initially named Arduino Ivrea, was inspired by the need for a low-cost, open-source development environment for microcontrollers. The first board, Arduino Serial V1, released in late 2005, used the ATmega168 chip from Atmel's AVR family. Subsequent developments led to boards like Arduino Uno, Mega, Due, and Leonardo, all based on AVR microcontrollers, with designs published under open-source licenses (Banzi & Shiloh, 2014). In 2009, Leaflabs introduced the Maple board, powered by ARM Cortex chipsets, offering higher processing speeds (72 MHz to 1.5 GHz) while maintaining Arduino's programming simplicity (IEEE Spectrum, 2011). In 2010, the Papilio Arduino project introduced FPGA-based boards for high-speed parallel processing, though their complex programming and higher costs limit their use to industrial applications (Kushner, 2011). By 2016, 17 commercial Arduino board versions were produced by companies like Smart Projects, SparkFun Electronics, and Adafruit.

III.3) Arduino Board Types

Arduino boards vary in size, capabilities, and applications, catering to diverse project needs. The Arduino Mega series, including Mega, Mega 2560, and Mega ADK, offers up to 54 digital ports and 16 analog inputs, ideal for large-scale projects. The Due board, resembling the Mega ADK, uses a 32-bit ARM-based AT91SAM3X8E microcontroller with an 84 MHz processor, suitable for high-performance applications (Banzi & Shiloh, 2014). The Leonardo board, similar to the Uno, employs the ATmega32u4 microcontroller with built-in USB communication, functioning as a Human Interface Device (HID) for projects like keyboard emulators. Miniaturized boards like Micro and Nano, based on Leonardo and Uno respectively, are designed for breadboard integration, while the Mini lacks a USB-serial adapter.

The Pro and Pro Mini boards are intended for semi-permanent installation, allowing custom soldering, and the Lilypad series, designed for e-textiles, offers lightweight, round designs with large ports for wearable applications.

Figure 12 *Arduino Boards: Uno, Mega, Nano*

III.4) Hardware Components

Arduino boards integrate several hardware components to facilitate electronics projects. The microcontroller, a compact integrated circuit, serves as the core, comprising program memory (ROM, EEPROM, or Flash), a processor for executing code, RAM for temporary data storage, and input/output pins for interfacing with external circuits (Banzi & Shiloh, 2014). Common microcontrollers include the ATmega328P, featuring an 8-bit RISC architecture, 32KB flash memory, 2KB SRAM, 1KB EEPROM, and a 20 MHz clock speed. Additional features like Analog-to-Digital Converters (ADCs), timers, and communication protocols (UART, I2C, SPI) enhance functionality (Banzi & Shiloh, 2014).

Voltage regulators, such as the 1117ST50T3G (+5V, 1A) and LP2985-33BVR (+3.3V, 150 mA), ensure stable power supply for the Arduino Uno and connected circuits. LEDs indicate board status: the ON LED (green) signals power, the L LED (red) connects to digital pin 13, and TX/RX LEDs (yellow) show USB data transfer. The USB port supports programming, power delivery, and data exchange, protected by a fuse against short circuits (Banzi & Shiloh, 2014). A power jack (7-12V) and Vin port provide alternative power sources, while the reset button restarts the board. The LM358 integrated circuit, with dual operational amplifiers, manages power source selection and LED buffering. In-Circuit Serial Programming (ICSP) ports allow programming via external programmers, and power ports (Vin, GND, +5V, +3.3V, RESET, IOREF) support external circuit integration (Banzi & Shiloh, 2014). Analog input ports (A0-A5) convert analog signals to digital with 10-bit resolution, and digital ports (0-13) handle input/output, supporting PWM, UART, SPI, and I2C protocols.

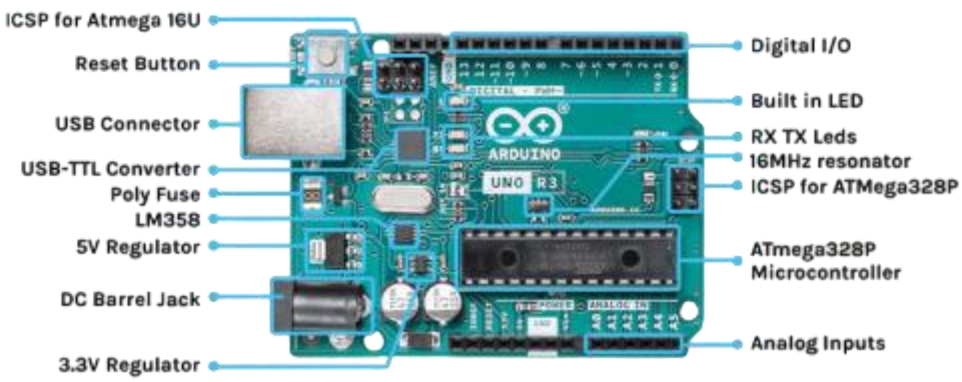


Figure 13 *Arduino Uno Components*

III.5) Software Environment

The Arduino software environment simplifies programming and project development. The Arduino IDE, available for Windows, macOS, and Linux, is downloadable from https://www.arduino.cc/en/Main/Software and supports code writing, compilation, and uploading . For Windows, the installation package (e.g., arduino-2.3.6-windows.exe) integrates seamlessly, storing sketches in a user directory. Linux distributions use packages like deb or rpm, while macOS requires unzipping Arduino.app . Alternatives include the browser-based Arduino Cloud Editor, ideal for collaboration and Chromebooks, and the Command Line Interface (CLI) for automation and advanced users.

The Arduino programming language, a simplified C/C++ variant, includes functions like setup (runs once) and loop (runs continuously), with standard data types and control structures. Core libraries, such as Wire (I2C), SPI, EEPROM, WiFi, and HTTP Client, facilitate communication with sensors, displays, and networks (Banzi & Shiloh, 2014). Third-party libraries, sourced from the Arduino Library Register or GitHub, extend functionality for specialized hardware. Debugging tools include the Serial Monitor for real-time output, logic analyzers for digital signals, and oscilloscopes for analog signals. Bootloaders manage code uploads, running for 8 seconds post-reset, while simulation tools like Wokwi and VSCode support code testing without hardware.

III.6) Teaching Electronics with Arduino

Teaching electronics with Arduino involves foundational concepts delivered through hands-on projects. Circuit theory principles, including voltage, current, and resistance, are applied practically, enabling students to design functional circuits (Milani et al., 2020). Digital and analog signals are managed via Arduino's input/output pins, with sensors (e.g., temperature, light) and actuators (e.g., motors, LEDs) enhancing environmental interaction (Arduino Project Hub, n.d.). Programming fundamentals, including setup, loop, pinMode, digitalWrite, analogRead, and delay, are introduced through accessible examples.

Educational strategies emphasize project-based learning, where students build projects like traffic light systems or temperature monitors to reinforce theoretical knowledge (Instructables, n.d.). A sample code for blinking an LED illustrates digital output and timing:

```
void setup()
 {
  pinMode(13, OUTPUT); // Designate pin 13 as an output
}

void loop()
 {digitalWrite(13, HIGH); // Activate the LED

  delay(1000); // Pause for one second

  digitalWrite(13, LOW); // Deactivate the LED

  delay(1000); // Pause for one second}
```

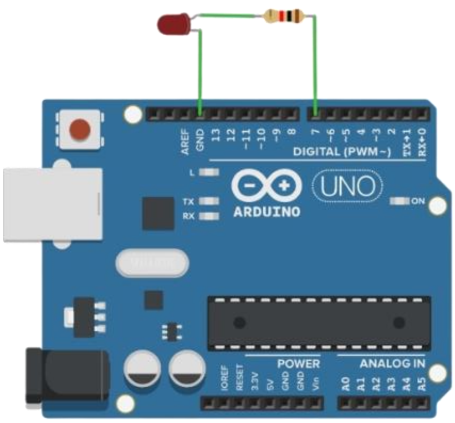


Figure 14 *LED Circuit Setup*

This code, accompanied by explanatory comments, is designed to support beginner learning by illustrating the basic structure of Arduino programming. Troubleshooting guidance typically addresses common issues such as incorrect wiring or basic coding mistakes. However, one significant challenge for novices remains syntax-related errors such as misplaced semicolons, unmatched brackets, or incorrect function names which can hinder early progress. This concern is explicitly addressed in the following chapter, where the visual programming environment BlocklyDuino is introduced to help learners bypass such syntactical obstacles through a block-based interface that enforce correct structure by design.

III.7) Applications and Integration

Arduino is extensively used in educational and practical environments, supporting project-based learning in schools to teach electronics, programming, and embedded systems, and powering real-world applications like smart home devices, automation systems, environmental monitoring, and health technologies (Milani et al., 2020; Arduino Education, n.d.). Its versatility makes it ideal for both teaching and innovation.

Typical beginner projects include blinking an LED to demonstrate digital output or using a temperature sensor to monitor environmental changes, as shown in Figure 14 LED Circuit Setup. Advanced applications involve controlling a relay to operate household appliances or building a line-following robot, bridging theoretical knowledge and engineering practice

Typical beginner projects include blinking an LED to demonstrate digital output (as shown in Figure 14) or using a temperature sensor to monitor environmental changes. Advanced applications, such as the smart lighting system shown in Figure 15, involve combining sensors and actuators to bridge theoretical knowledge and engineering practice (Instructables, n.d.; Springer, 2016).

Arduino's strength lies in interfacing with external components. It reads data from sensors (e.g., temperature, motion, light) using analogRead or digitalRead, and controls actuators like motors, LEDs, or relays via digitalWrite, enabling interactive systems and automation (Banzi & Shiloh, 2014).

While this section has covered how Arduino is used for educational purposes and real-world problem-solving, the next chapter focuses on how these principles are applied within a serious game. In *Smart in the Dark*, Arduino concepts are integrated into game mechanics using Blockly (a visual programming tool), allowing players to simulate hardware behavior and solve challenges. This bridges the gap between theoretical instruction and immersive, applied learning.

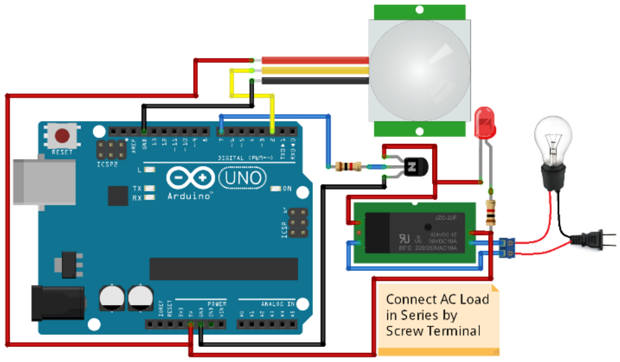


Figure 15 *Smart Lighting System with PIR Sensor*

III.8) Advantages and Limitations

Arduino offers several advantages. Its user-friendly IDE and library support simplify coding for beginners (Banzi & Shiloh, 2014). Affordability makes it accessible for rapid prototyping (Kushner, 2011). A rich community provides extensive resources, enhancing learning (Mellis et al., 2007). Its flexibility supports diverse applications, from simple LED circuits to complex automation (Giacomin et al., 2016). However, limitations include limited processing power (e.g., Arduino Uno's 8-bit architecture), constrained memory (2KB SRAM), and unsuitability for complex commercial applications requiring real-time processing or high-speed communication (Margolis, 2011; Nussey, 2013; Monk, 2017).

III.9) Serious Games in Arduino Education

Serious Games offer a dynamic and motivating framework for teaching Arduino concepts by embedding technical tasks within interactive challenges. They promote engagement, improve retention, and align well with constructivist learning, which emphasizes learning through doing (Hanselman, 2016; All3DP, 2019).

In Arduino education, gamified tasks such as controlling LEDs or responding to sensor input can help learners apply code in context, develop problem-solving skills, and explore electronics in an accessible, error-tolerant environment. These games often simulate real-world applications and support collaboration, self-paced learning, and immediate feedback (Springer, 2016).

This pedagogical model sets the foundation for the next chapter, which presents *Smart in the Dark*, a serious game designed to teach Arduino programming through visual coding, scenario-driven interaction, and embedded evaluation.

III.10) Conclusion

This chapter has explored the Arduino platform as a versatile tool for electronics education and prototyping. Its open-source design, affordability, and community support make it accessible to beginners, while its hardware and software capabilities support diverse applications. Educational strategies, including project-based learning and Serious Games, enhance engagement and learning outcomes. Despite limitations in processing power and memory, Arduino's flexibility bridges theoretical and practical knowledge, paving the way for innovative applications like *Smart in the Dark* in subsequent chapters

Chapter IV. Design and Implementation

IV.1) Design

IV.1.1) Introduction

The design of *Smart in the Dark* aims to blend narrative-driven gameplay with structured, level-based Arduino programming challenges. Drawing upon constructivist learning theory, the game immerses players in a scenario where they must repair malfunctioning electronic subsystems in a dark environment using block-based coding (**forked version of** BlocklyDuino editor) and point-and-click interactions (GDevelop) (Rubin et al., 2012). This chapter details the design rationale, educational objectives, system architecture, user interface, and pedagogical scenario. It also explains how technical choices such as embedding BlocklyDuino via an iframe , which involves loading the **forked and adapted version of the open-source** BlocklyDuino editor as an embedded HTML page and communicating with the game engine through the postMessage() API support the overall goals of accessibility (Rubin et al., 2012),, extensibility, and data-driven evaluation. And leveraging Firebase Authentication and Firestore support the overall goals of accessibility, extensibility, and data-driven evaluation.

IV.1.2) Project Overview and Objectives

Smart in the Dark is a serious game intended to teach Arduino fundamentals and embedded system debugging to beginners. The player's avatar is trapped in a home without power or functioning smart systems after a blackout. To progress and eventually escape, the player must repair each subsystem lighting, doors, air conditioning, security locks, and neighborhood streetlights by reconstructing and fixing the underlying Arduino code.

IV.1.2.1) Primary Educational Objectives

The main educational goals encompass the introduction of fundamental Arduino concepts to learners, including digital input and output (e.g., *pinMode*, *digitalWrite*), analog input (*analogRead*), conditional logic (if/else), loops, and basic timing functions like *millis()*. Furthermore, the objectives seek to foster problem-solving abilities and debugging skills through iterative code refinement. Learners are also encouraged to apply electronics principles to practical scenarios inspired by real-world situations, such as diagnosing and fixing a malfunctioning air conditioning system with the use of a temperature sensor.

IV.1.2.2) Technical Objectives

The technical objectives concentrate on utilizing an open-source, event-driven game engine known as GDevelop to develop lightweight gameplay suitable for developers using low-performance laptops. Integration of a visual programming environment called BlocklyDuino (**forked version**) is achieved through an iframe, facilitating bidirectional communication between the code editor and GDevelop game scenes to ensure smooth interaction. Moreover, user authentication and performance data are securely stored using Firebase Authentication and Firestore, enabling continuous progress monitoring, analytics, and laying the foundation for future research on learning outcomes.

IV.1.2.3) Success Criteria

The success criteria encompass the requirement that a minimum of 80% of players can successfully complete each level with minimal external assistance, reflecting the efficacy of instructional design and intuitive gameplay. Additionally, the game should operate seamlessly on laptops with restricted graphics capabilities, such as those equipped with 2 GB of RAM and integrated GPUs, in order to guarantee accessibility. Moreover, all pertinent user data including duration of play, click sequences, and level achievements ought to be accurately recorded and retrievable from Firestore for the purpose of behavioral analysis and facilitating potential enhancements or research endeavors.

IV.1.3) Scenario and Pedagogy

IV.1.3.1) Narrative and Flow

The narrative immerses the player in Rami's intelligent home immediately following a devastating thunderstorm that results in a complete power outage. Upon waking up in darkness, Rami realizes that all automated systems have malfunctioned:

- **Level 1 – Equipment Gathering:** Rami gathers his Arduino kit, flashlight, and sensor modules scattered throughout the living room. This initial level introduces players to basic point-and-click navigation and inventory collection.
- **Level 2 – Illuminating the Hallway:** The main light fixture is inoperative. Rami must construct a straightforward LED circuit on a breadboard and create an Arduino sketch using BlocklyDuino to control the LED (*digital output*, *pinMode*, *digitalWrite*).
- **Level 3 – Fixing the Air Conditioning:** The A/C fan fails to operate due to a defective temperature sensor circuit. Rami identifies the analog input issue, corrects the code to interpret sensor values (*analogRead*), and incorporates conditional logic (if statements) to activate the fan when the temperature exceeds a predetermined threshold.
- **Level 4 – Restoring the Security System:** An erroneous PIR motion detection triggers the smart door's servo, locking it. Rami must implement signal stabilization or filtering (debounce delays) and, if necessary, program a manual override button, teaching about motion detection and input noise management.

- **Level 5 – Reinstating Neighborhood Lighting:** Venture outside, Rami discovers that the streetlights are offline. By combining data from an LDR (light sensor) and a PIR motion sensor, he uses logical operators (e.g., &&, ||) to only illuminate the lights when it is dark and motion is detected, imparting knowledge of sensor fusion and energy-efficient control mechanisms.

Each level in the program is structured around a Narrative–Challenge–Solution–Feedback loop with the aim of creating an immersive and educational environment. The Narrative phase initiates with a brief dialogue or cutscene that presents a practical problem, serving as a contextual backdrop. Following this, the Challenge Setup stage requires the player to investigate a malfunctioning subsystem, such as a dimly-lit hallway, in order to grasp the nature of the issue. The subsequent Coding Phase involves BlocklyDuino displaying a flawed Arduino sketch that necessitates the player to reorganize code blocks to rectify the problem. In the final Execution & Feedback phase, the corrected code is transmitted to GDevelop; if accurate, the desired in-game action is executed (e.g., illuminating an LED), enabling the character Rami to advance. In cases of incorrect code, the player is provided with specific error feedback and encouraged to make another attempt. This iterative process is consistent with constructivist pedagogy, empowering learners to actively construct knowledge through the diagnosis and resolution of code challenges embedded in meaningful, context-rich scenarios.

IV.1.3.2) Arduino Concepts Integration

Table 8 structures learning from basic to advanced Arduino concepts, like digital output in Level 2 to sensor fusion in Level 5. This aligns with constructivist learning theory, where players build knowledge through interaction, and flow theory, balancing challenge and skill for engagement. However, ensuring balanced difficulty to avoid cognitive overload is a noted challenge (Hektner & Csikszentmihalyi, 1996, p. 4).

Table 8 *specific Arduino programming concepts and in-game tasks of each level*

Level	**Environment/Task**	**Key Concepts**	**Learning Objectives**	**Validation Criteria**
1	Living Room Equipment collection	None (game orientation)	Familiarize with 3D navigation, point-and-click interactions	Collect all items; inventory full
2	Hallway – Build LED circuit	*Digital Output (pinMode, digitalWrite)*	Configure digital pins; write sketch to power an LED	Sketch uses *pinMode(pin, OUTPUT), digitalWrite(pin, HIGH)*

3	Living Room Repair A/C with sensor	Analog Input (*analogRead*), Conditionals (if/else)	Read sensor data; use threshold logic to control fan	Code uses *analogRead*, if, *digitalWrite* for fan control
4	Main Door – Fix motion-based door lock	Digital Input *(digitalRead), Servo Control (servo.write(angle)*), Signal Filtering (Debounce)	Process digital inputs from motion sensor; implement logic to ensure reliability; control servo to unlock door	Code uses *digitalRead* for motion sensor, applies logic to avoid false triggers, and controls servo to unlock door only on valid motion
5	Neighborhood Build a smart streetlight system	Sensor Fusion (*analogRead*, *digitalRead*), Logical Operators (&&, ||)	Integrate multiple sensors for energy-efficient automation using light and motion sensors	Code combines *analogRead* (light sensor) and *digitalRead* (motion sensor) with logical conditions to control the streetlight

The progression table outlines the levels of advancement achieved through the gradual introduction of digital output, analog input, conditional logic, servo control, and sensor fusion, allowing players to establish a strong base in Arduino programming.

Verification of each level's completion is determined by comparing the player's code to the correct code provided in GDevelop, as outlined in the Validation Criteria (refer to **Table 8**). This instantaneous, automated validation process guarantees that learners promptly receive feedback on the accuracy of their code.

IV.1.4) System Architecture

The system architecture of *Smart in the Dark* comprises three primary components:

IV.1.4.1) GDevelop Game Engine version 5.5.231 (Front-End)

The front-end of the GDevelop Game Engine Version 5.5.231 is responsible for implementing interactive gameplay mechanics and scene transitions through GDevelop's event-based system. It displays the player's view of the environment, including 2D sprites, dynamic lighting effects, and user interface panels. The game communicates with the embedded BlocklyDuino (**forked version**) iframe using the `*postMessage*` API to receive generated Arduino code and interaction metrics. The received code is then compared to a predefined correct solution stored in a scene variable. Upon a successful match, the engine triggers level completion, transitions to the next scene, and stores the player's progress to Firebase.

IV.1.4.2) BlocklyDuino Editor version 9.4.0 (Embedded via iframe)

The BlocklyDuino Editor Version 9.4.0 was **forked** and integrated within the application through an **iframe** (Rubin et al., 2012),. This editor serves as the interface for the player to piece together Arduino code using a block-based system derived from the Google Blockly framework. By utilizing the function

`Blockly.Arduino.workspaceToCode(MainWorkspace)`,

the editor dynamically creates Arduino C++ code based on the player's block arrangement. The resulting code string is then sent to the parent GDevelop frame using the method:

`window.parent.postMessage({type: "blocklyCode", code: generatedCode}, "*») `

Additionally, the **forked and adapted version of the open-source BlocklyDuino** editor manages level-specific XML configurations *(`levelXML`)* and defines the toolbox contents for each challenge to ensure that players have just necessary blocks for problem-solving (Rubin et al., 2012),.

IV.1.4.3) Firebase Backend 10.7.0 (Authentication + Firestore)

- **Authentication:** Guarantees that every player possesses a distinct user identity.
- **Firestore (NoSQL Database):** is used to store performance metrics for individual users. These metrics include the time spent on the programming editor (a forked version of BlocklyDuino) and the time spent on the Arduino simulator then the total time spent on the level (Rubin et al., 2012), the number of code submissions (clicks), and the status of level completions.
- **Data Flow**
  - ➢ At the beginning of each level, GDevelop verifies the authentication status of the user. If the user is not authenticated, a login prompt is displayed.
  - ➢ Upon clicking on "Upload to Arduino Board," the code (or click count) is sent from BlocklyDuino (**forked version**) through *postMessage*.

- GDevelop's event logic then compares the code. If it is deemed correct, GDevelop proceeds to create a record in Firestore (e.g., *{userID, level, timeSpent, clickCount}*) using the Firebase SDK (Firestore).
- After a successful write operation, Firestore updates in real-time (optional) and the game transitions to the next scene.

The complete sequence of these interactions, from user authentication to progress tracking, is illustrated in the system architecture diagram in ***Figure 16***.

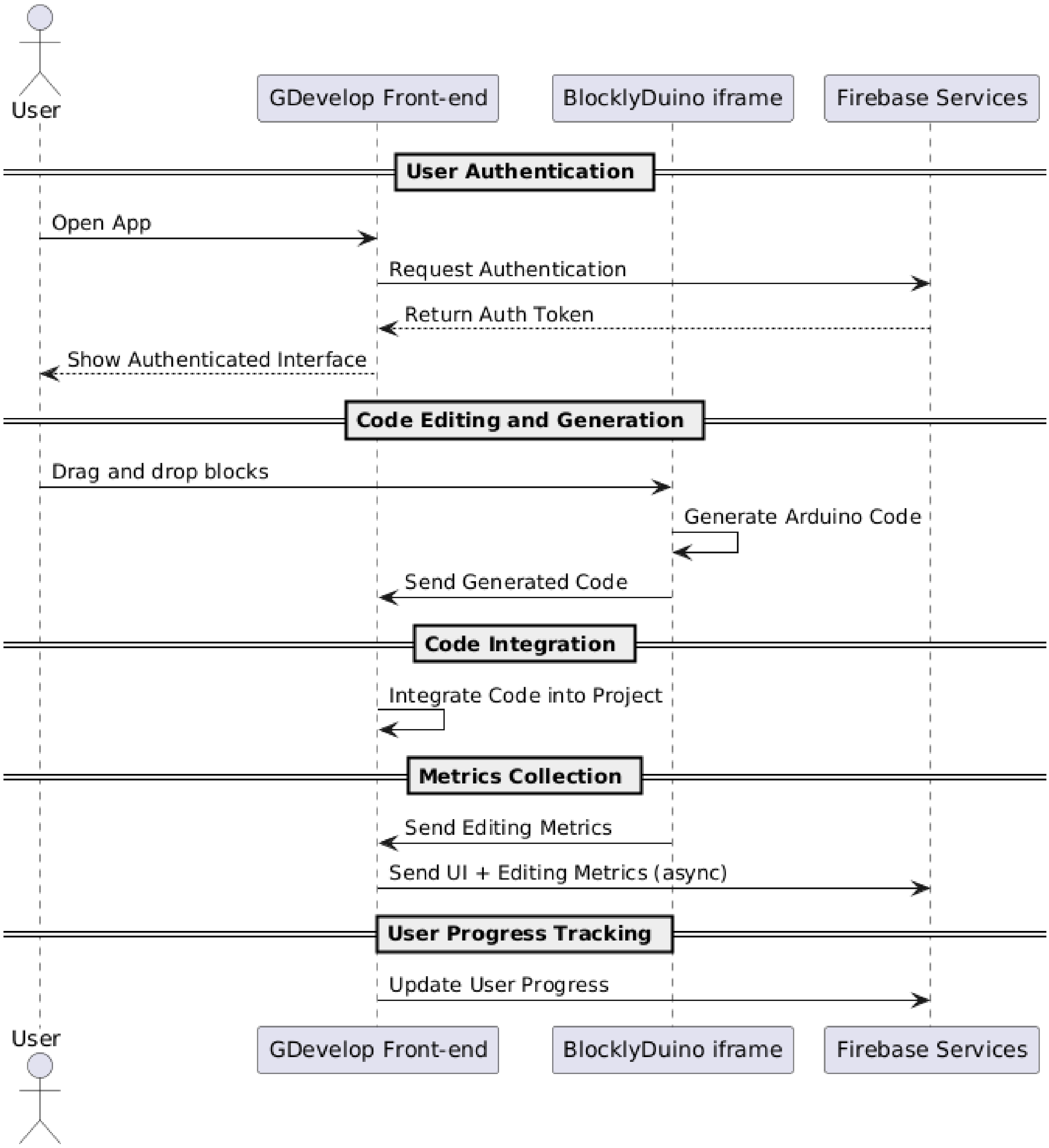


Figure 16 *UML Sequence Diagram of the Smart in the Dark System Architecture*

IV.1.4.4) Integration Strategy

The integration between BlocklyDuino (**forked version**) and GDevelop is based on a well-defined communication and control flow to ensure smooth interaction and efficient data handling:

- **Communication Protocol**

The system uses the `postMessage` API to facilitate communication between frames. The `sendCodeToGDevelop()` function in BlocklyDuino (**forked version**) sends the generated code string and the number of clicks to GDevelop. On the GDevelop side, a listener is set up via `window.addEventListener('message', ...)` to receive these messages and store the incoming data in scene variables such as `BlocklyCode` and `*ClickCount*`.

- **URL-Based Level Detection**

The BlocklyDuino custom editor reads the current URL path (formatted as *`/level/{n}`*) to identify which level configuration and toolkit to load, enabling dynamic content delivery based on the active game stage.

- **Event-Driven Logic**:

All code evaluation and Firebase data writing are fully managed within GDevelop's native event system, eliminating the need for external scripts. For example:

Conditions: *Scene Variable BlocklyCode == Scene Variable Correct*

Actions: *Firestore::Write { userID, level, timeSpent, clickCount } → Change to scene*

This integration approach leverages GDevelop's built-in event capabilities to minimize the complexity of custom code while providing a robust and maintainable game logic framework.

IV.1.5) User Interface

The user interface (UI) effectively combines narrative immersion with clear coding tasks, comprising two main panels.

- **Game View Panel**

The Game View Panel showcases a 2D environment where various sprites are displayed, including a dark hallway, a malfunctioning A/C unit, and a locked door. This combination of elements creates a captivating and immersive scene for the player. The panel is enhanced with dynamic lighting effects that replicate a dimly lit setting with restricted visible areas. These areas are illuminated only when the player's code is successfully executed, such as activating an LED light. Additionally, the panel features heads-up display (HUD) elements like a timer that measures elapsed time using GDevelop's *TimerElapsedTime* function to monitor the player's progress. Moreover, a hint button is included to provide optional assistance to the player through a textual hint dialog whenever requested.

- **forked version of the open-source BlocklyDuino editor Panel**

The forked BlocklyDuino Code Editor Panel is composed of two primary tabs: the Blocks Tab, where drag-and-drop programming blocks from the level-specific toolbox defined by the *toolboxXML* are displayed, and the Arduino Tab, which presents the generated C++ code in a read-only format. In addition, the panel features user interface buttons such as "Upload to Arduino Board," which activates the `*sendCodeToGDevelop()*` function within the Blockly iframe to send the code to GDevelop, and "Reset Blocks," which clears the workspace and reverts it to the predefined level configuration specified by the *levelXML*.

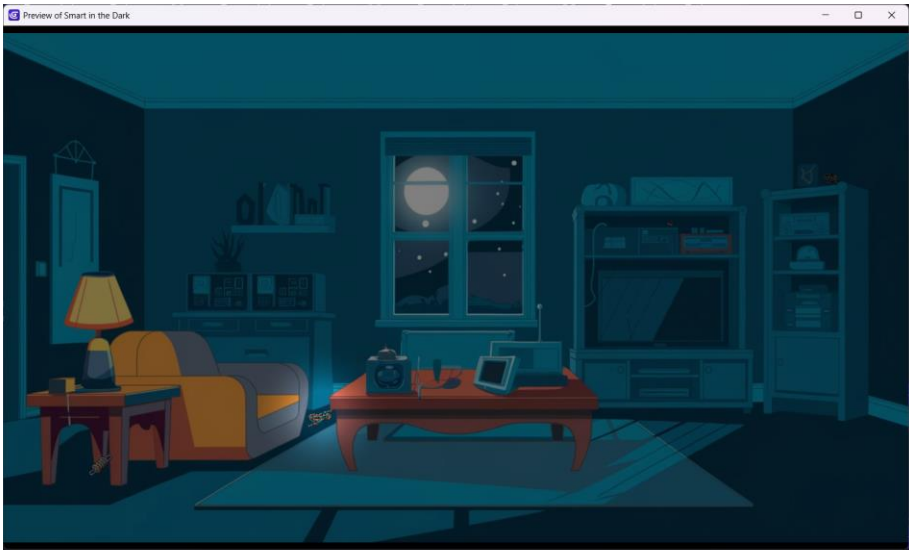


Figure 17 *Game View Panel Mockup*

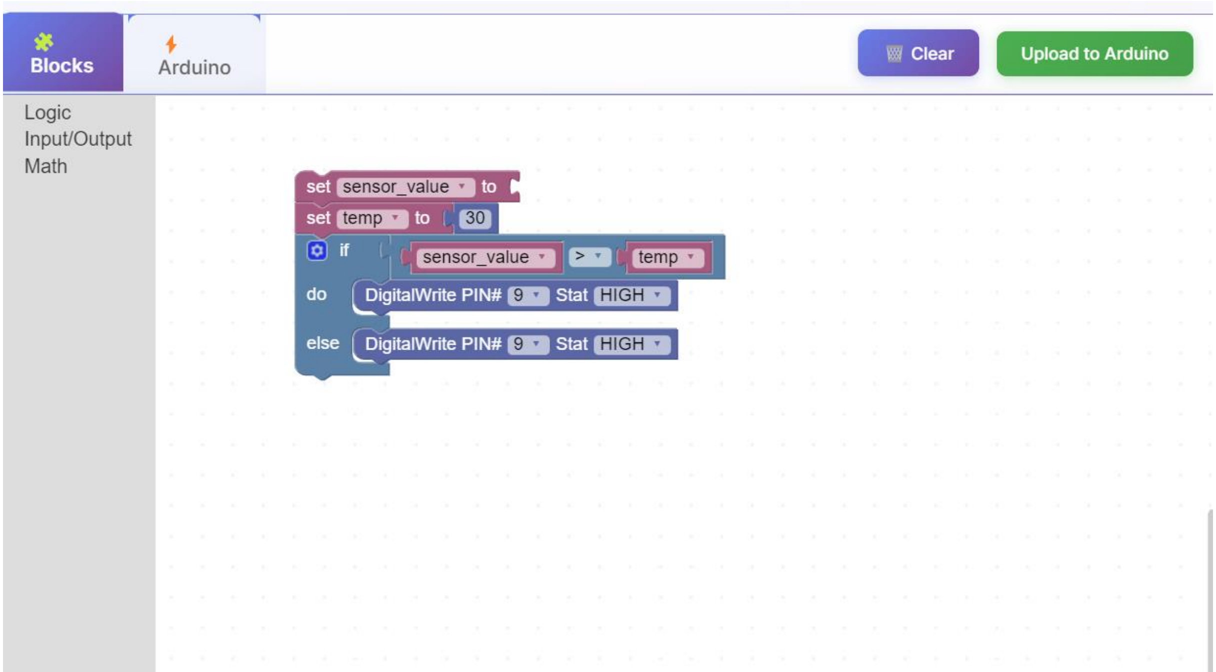


Figure 18 *A forked version of the open-source BlocklyDuino Editor Panel Mockup*

IV.1.5.1) Visual and Accessibility Considerations

The design incorporates a dark theme with high contrast in order to ensure the easy readability of UI text and to make lighted elements clearly distinguishable against dark backgrounds. Responsive typography utilizes clean, sans-serif fonts like Roboto set at 16–18 pt for body text to maintain consistency and readability across various devices. Keyboard navigation and focus indicators are supported through the utilization of Blockly's integrated keyboard accessibility features for block manipulation in conjunction with GDevelop's standard UI navigation. This ensures that users with limited dexterity are accommodated. Furthermore, feedback pop-ups are designed with color coding red for errors and green for success alongside icons to clearly communicate the outcomes of user actions.

IV.2) Implementation

IV.2.1) Tools and Technologies

This section enumerates and justifies the main tools used in implementing *Smart in the Dark*:

IV.2.1.1) GDevelop Engine

The GDevelop Engine is an open-source game engine that operates on an event-driven architecture and has capabilities for both 2D and 3D game development. Its selection was based on its efficient performance, user-friendly visual event system, and compatibility with HTML5 export, facilitating deployment across various platforms including Windows, macOS, Linux, and web browsers. Additionally, GDevelop enables the customization of behaviors and event bundles, promoting a design approach that is both modular and adaptable.

IV.2.1.2) BlocklyDuino Editor

The **forked and adapted version of the open-source** BlocklyDuino Editor is a specialized version of Google Blockly designed specifically for creating Arduino code efficiently (Rubin et al., 2012). Integrated into the GDevelop environment through an iframe, this editor includes dynamic block categories that adapt to the user's learning level. By assembling blocks, the editor automatically produces valid Arduino C++ code, simplifying the learning process for novices and intermediates by eliminating the need to deal with intricate syntax intricacies and enabling them to concentrate on fundamental programming principles.

IV.2.1.3) Firebase (Authentication & Firestore):

Firebase is utilized for the management of both user access and data storage in the application. The Authentication feature ensures secure login through various methods like email/password and OAuth providers, assigning distinct user IDs for monitoring personal advancement. As for Firestore, it serves as a real-time NoSQL database for storing user performance indicators like task duration, click frequencies, and level completion details. This configuration presents notable benefits such as inherent scalability, reduced setup complexity, and instantaneous data synchronization, rendering it particularly suitable for educational gaming applications aimed at expanding user populations.

IV.2.1.4) Web Technologies:

Web technologies are essential for the integration and management of the front-end components of the application. These technologies include HTML/CSS for structuring and styling the BlocklyDuino iframe and various UI panels to create a clean and responsive interface. Furthermore, JavaScript plays a critical role in enabling core interactivity by managing message passing between the BlocklyDuino (**forked version**) iframe and the parent GDevelop frame through `*window.postMessage*`. Additionally, JavaScript processes important data such as click counts, generated code strings, and current URL paths for level detection, facilitating smooth communication and dynamic behavior across different components.

IV.2.1.5) Additional Libraries:

The integration of additional libraries significantly improves the visual coherence and user experience of the application. Material Icons are utilized to visually depict UI actions, such as "Run" and "Error," offering intuitive and familiar signals for users. Google Fonts, in particular the Roboto typeface, guarantee uniform and legible typography across various devices and platforms, thereby enhancing the overall polish and accessibility of the interface design.

Together, these technologies support a seamless integration of narrative gameplay, block-based coding, and cloud-based data tracking, all without requiring end users or local developers to obtain specialized hardware or proprietary software.

IV.2.2) Implementation Highlights

IV.2.2.1) Core Mechanics

The core mechanic of *Smart in the Dark* is centered around a continuous code repair and validation loop that promotes iterative learning and real-time feedback. This mechanic is carried out through a structured, event-driven process:

A. **Level Initialization**: upon loading a level scene in GDevelop, the embedded the custom BlocklyDuino workspace is initialized using the level-specific XML configuration stored in `*levelXML[level]*`.
B. **Dynamic Toolbox Generation:** BlocklyDuino presents a curated set of block categories and types based on `*toolboxXML[level]*`, ensuring learners interact only with relevant programming constructs for that specific level (e.g., *LED control, sensor input, servo actions*).

```
const toolboxXML = {
  "1": `<xml id="toolbox" style="display: none">
          <category name="Input/Output">
            <block type="inout_digital_write"></block>
          </category>
        </xml>`,
  // Additional levels ...
};
```

C. **User Edits Blocks:** When the user clicks the “Upload to Arduino Board” button, BlocklyDuino converts the block structure into Arduino C++ code. This code string is then transmitted from the iframe to the GDevelop parent frame using the `*postMessage*` API.
D. **Code Generation & Transmission:** When the player clicks *“Upload to Arduino Board,”*the embedded script in the Blockly iframe executes:

```
function sendCodeToGDevelop() {
  if (typeof Blockly !== "undefined" && Blockly.mainWorkspace) {
    let generatedCode =
       Blockly.Arduino.workspaceToCode(Blockly.mainWorkspace);
    window.parent.postMessage({ type: "blocklyCode", code:
       generatedCode }, "*");
    showNotification("Code uploaded successfully!", "success");
  }
}
```

The generated Arduino code (C/C++) is sent to the parent frame (GDevelop) via *postMessage*..

E. **Code Comparison in GDevelop:** GDevelop, listening for blocklyCode messages, processes the incoming string:

```
function removeWhitespace(str) {
  return str.replace(/\s+/g, '');
}

if (!window.blocklyCodeListenerAdded) {
  window.addEventListener("message", function(event) {
    if (event.data && event.data.type === "blocklyCode") {
      let blocklyCode = removeWhitespace(event.data.code);
      const sceneVariables = runtimeScene.getVariables();
      sceneVariables.get("BlocklyCode").setString(blocklyCode);

      // Compare with correct code stored in "correct" variable
      if (sceneVariables.get("correct").getString() ===
         blocklyCode) {
        console.log("correct answer");
        // Trigger success actions
      } else {
        // Indicate incorrect code and remain in the editor
      }
    }
  }, false);
  window.blocklyCodeListenerAdded = true;
}
```

Prior to comparing the code submitted by the user, GDevelop eliminates all whitespace characters from the `BlocklyCode` string in order to avoid potential discrepancies in formatting that could mistakenly identify accurate logic as incorrect. If the purified code string matches precisely with the correct solution stored in the scene (loaded at the commencement of the level), GDevelop proceeds to trigger predefined success events such as illuminating LEDs or unlocking doors indicating successful completion of the task. In the event of a code mismatch, the player will remain within the Blockly editor without any progression, urging them to revise and rectify their arrangement of blocks until a valid solution is reached.

F. **Performance Metrics Logging:** Simultaneously, whenever a code upload is requested,the script in the Blockly iframe also sends the click count:

```
function computeClickCount() {
  clickCounter++;
  window.parent.postMessage({ type: "clickCount", clicks:
     clickCounter }, "*");
}
```

GDevelop listens for *clickCount* messages to update its *ClickCount* scene variable:

```
if (!window.clickCountListenerAdded) {
  window.addEventListener("message", function(event) {
    if (event.data && event.data.type === "clickCount") {
      const clickCount = event.data.clicks;
      const sceneVariables = runtimeScene.getVariables();
      sceneVariables.get("ClickCount").setNumber(clickCount);
    }
  }, false);
  window.clickCountListenerAdded = true;
}
```

G. Firebase Integration

On success, GDevelop triggers Firebase writes

- **Actions**

Following successful code validation, GDevelop executes a series of actions to progress the game and record user performance. Initially, it writes a data entry to Firestore at the designated path `users/{userID}`, saving an object containing the current level, time elapsed (retreived using `*TimerElapsedTime("levelTimer") + "s"*`), and the number of code submission attempts (`*ClickCount*`). Subsequently, it eliminates the Blockly iframe or UI overlays to reset the coding interface. Lastly, it transitions the game to the subsequent level scene (for example, *“Level{n+1}”*).

These Firestore entries maintain user data in real-time within the individual user document, enabling educators or researchers to later retrieve and analyze metrics like task duration and user involvement throughout the levels. This event-driven, message-passing architecture guarantees minimal delay between code correction and in-game feedback, while upholding a clear division of roles: BlocklyDuino manages code assembly, GDevelop oversees gameplay and validation, and Firebase securely stores and synchronizes user performance data.

IV.2.2.2) Scenario Execution

Each level’s narrative scenario is implemented as a distinct GDevelop scene configured as follows:

A. Scene Structure:

he scene structure within *Smart in the Dark* has been meticulously designed to integrate narrative, interactivity, and logic seamlessly within each level. The visual context is established by the background layer featuring darkened rooms or outdoor settings depicted through sprites such as hallways, living rooms, or streets. Positioned on top of this are interactive objects, including malfunctioning devices like an LED lamp, A/C unit, or electronic door lock, which react to player input with actions like *"Click to Inspect."* A dialogue layer provides narrative guidance through on-screen text, such as “The hallway light remains dark. Repair the LED circuit to proceed.” This prompts player to engage in the coding challenge in a meaningful manner. Driving the interaction is a finite state machine that governs scene logic through well-defined states: “Inspecting” (identifying the issue), “Coding” (editing code in BlocklyDuino), “Validating” (submitting and verifying the code), “Success” (providing feedback and progressing), and “Retry” (allowing additional attempts if needed). This multilayered, event-driven framework fosters immersive gameplay while reinforcing learning through a structured problem-solving process.

B. Initialization: Load level variables (correct code string, timer, clickCount = 0).

Each level within the game *Smart in the Dark* functions as a finite state machine with clearly defined states aimed at managing game logic as well as player progression. The initial state, known as “Initialization”, is responsible for loading all necessary level variables, such as the appropriate Arduino code string, a timer, and ensuring the click count is set to zero. Moving on to the “*AwaitingCode*” state, players are prompted to interact with the BlocklyDuino interface in order to analyze and correct any faulty code present. Upon clicking “Upload to Arduino Board,” the game transitions into the **Validating** state, where the submitted `BlocklyCode` is evaluated against the correct solution. If the code matches, the game progresses to the **Success** state; otherwise, it reverts back to “*waitingCode*” for further revisions. Once in the **Success** state, the designated in-game action is executed (for example, turning on an LED or unlocking a door), while user performance data is stored in Firestore. Subsequently, the game advances to the subsequent level or displays a “Victory” scene, contingent upon the player's progress.

C. Asset Pipeline

The asset pipeline for the game *Smart in the Dark* consists of creating and preparing visual and audio resources that are optimized for use within GDevelop. Initially, sprites and backgrounds are designed or edited using Inkscape, a vector graphics editor, and then exported as PNG files with resolutions of either 512 × 512 pixels or 1024 × 768 pixels to comply with GDevelop's asset requirements. As for audio, sound effects like electric sizzling or servo motor noises are sourced from royalty-free libraries and imported into the project in the OGG file format to ensure efficient playback and compatibility. This workflow guarantees the production of high-quality, lightweight assets that enhance the immersive gameplay experience.

D. Scripts & Events

GDevelop primarily handles game logic through its event system, which includes conditional checks and corresponding actions. For example, verifying if the level timer has expired may trigger actions such as changing an LED's animation to "Lit". Custom events are utilized to streamline repetitive processes by grouping frequently used logic, such as Firebase data writes, into reusable behaviors. Level detection is seamlessly integrated into the modified BlocklyDuino editor, which dynamically loads the correct level configuration by reading the current URL path. This event-driven structure efficiently connects narrative elements, such as dialogue text and environmental cues, to coding challenges, enhancing player immersion and supporting a coherent pedagogical progression.

IV.2.2.3) User Interaction

Throughout each level, player interactions are tracked and used both for real-time feedback andpost-hoc analysis:

A. Point-and-Click Navigation

The point-and-click navigation feature enables players to interact seamlessly with the game environment by selecting sprites, such as a breadboard positioned on a table, in order to acquire components or examine faulty devices. For instance, in Level 1, players procure essential equipment by clicking on pertinent objects. Through GDevelop's "On object clicked" event conditions, responses are generated to these interactions by activating highlight animations that visually accentuate the chosen item and presenting contextual text to offer further information or directions, ultimately improving engagement and comprehension.

B. Blockly Block Manipulation

Users engage with the Blockly editor by dragging blocks from the toolbox and placing them within the workspace, enabling them to construct valid structures of Arduino code. The integrated syntax checker in Blockly provides ongoing guidance to users by identifying incompatible or erroneous blocks through greying them out, thus reducing the likelihood of errors during the compilation of code and supporting individuals in creating syntactically accurate programs.

C. Data Logging

Whenever the player selects the option "Upload to Arduino Board" on the Blockly iframe, a message containing the variable `clickCount` is sent to GDevelop. GDevelop actively monitors these messages and adjusts the scene variable `ClickCount` accordingly. Following the successful validation of the code, GDevelop records performance data, specifically

`{ timeSpend: TimerElapsedTime("levelTimer") + "s", clickCount: ClickCount }`

in Firestore at the location *`users/{userID}/levels/{levelNumber}`*. The elapsed time is accurately calculated through the utilization of GDevelop's *`TimerElapsedTime`* function, which measures the number of seconds since the *`levelTimer`* was reset at the commencement of the scene. This seamless integration of point-and-click interactions, block-based coding, and immediate feedback forms a robust cycle of action, reflection, and revision which is essential to experiential learning.

```
function computeClickCount() {
  clickCounter++;
  window.parent.postMessage({ type: "clickCount", clicks:
     clickCounter }, "*");
}
```

IV.2.3) Screenshots

Below are representative placeholders illustrating key gameplay and coding interfaces. Figure 19 shows the main menu of Smart in the Dark. To illustrate a mid-game scenario, Figure 20 depicts a narrative moment from Level 4 where the player must diagnose a faulty security system. The corresponding coding challenge is presented in Figure 21, which shows the Arduino circuit simulator interface for that level.

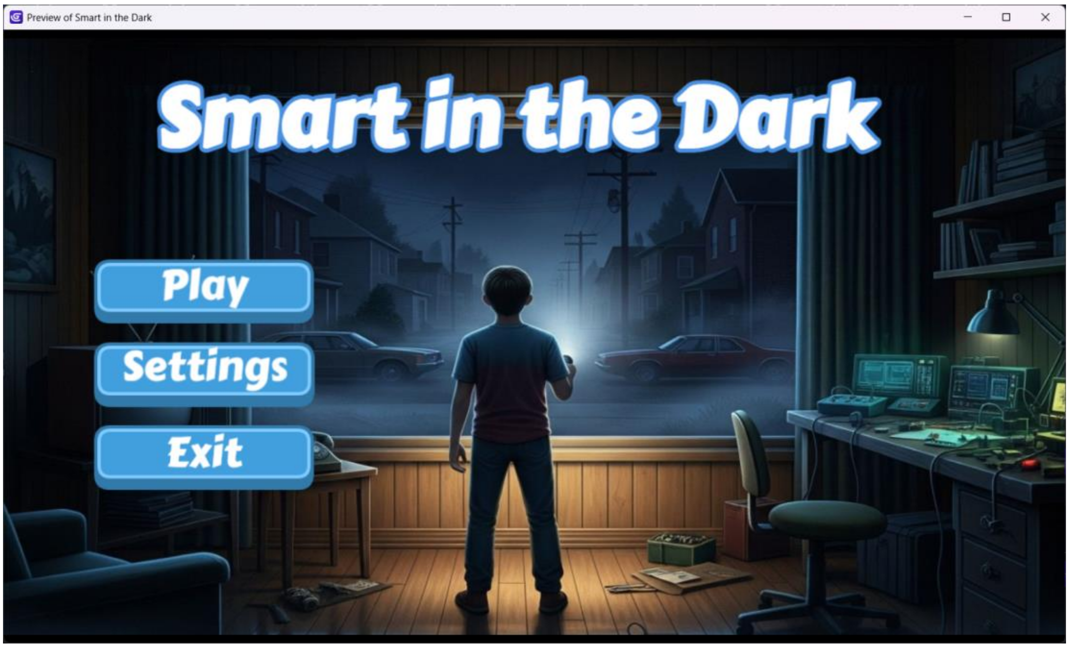


Figure 19 *The Menu of Smart in the Dark*

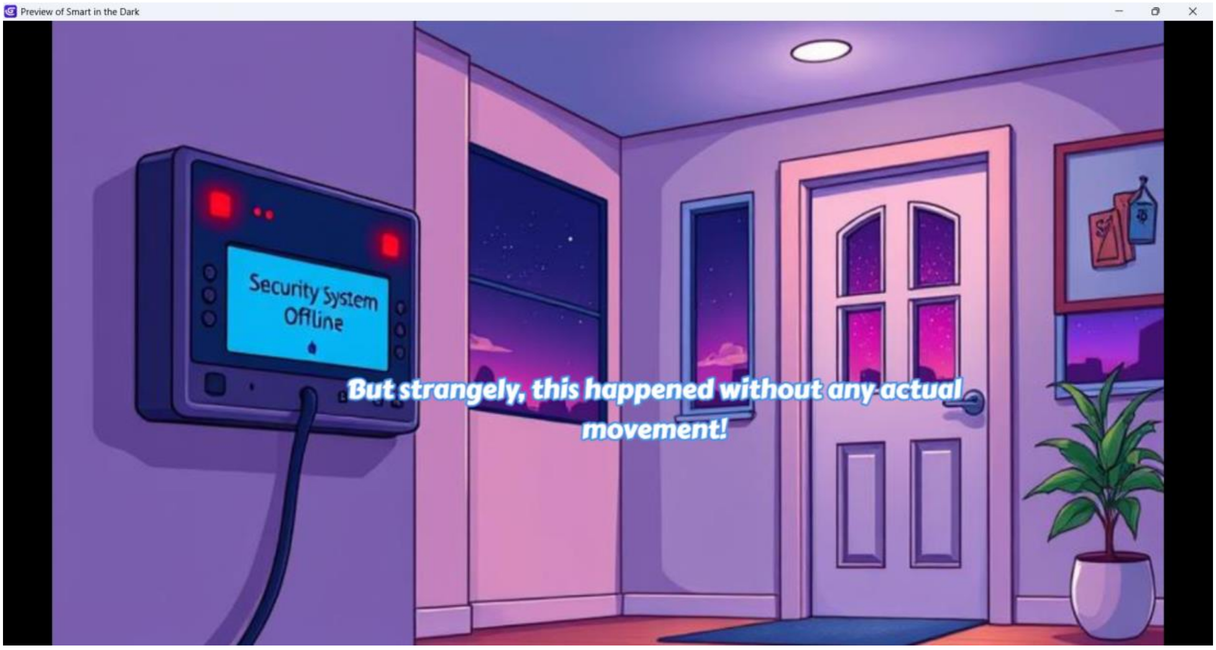


Figure 20 *Level 4 Restoring the Security System*

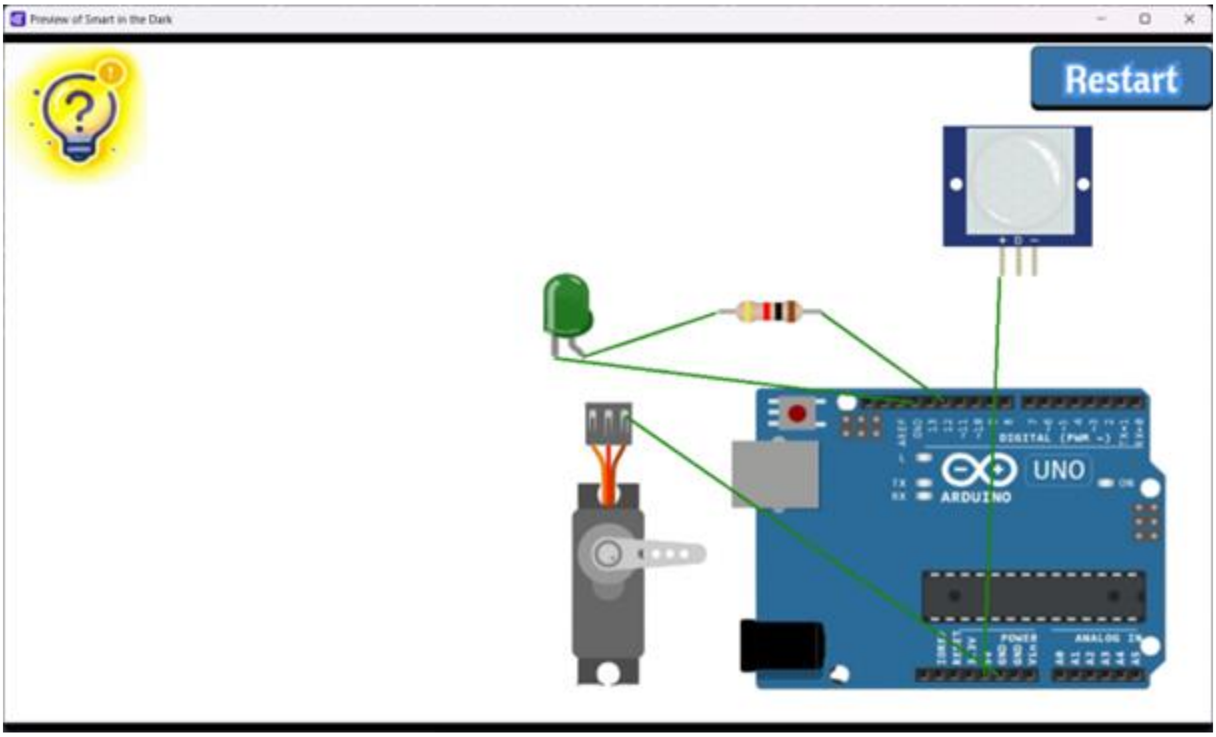


Figure 21 *Level 4 arduino circuit simulator*

IV.3) Conclusion

This chapter has elucidated the integrated design and thorough implementation of *Smart in the Dark*. Through the integration of BlocklyDuino into GDevelop using an iframe and utilizing event-driven message passing, the game provides immediate and contextually relevant feedback to learners as they troubleshoot Arduino code. The utilization of Firebase Authentication and Firestore facilitates the maintenance of persistent user profiles and detailed analytics, allowing educators to monitor progress and identify common obstacles. The structured gameplay, infused with a narrative and level-based format, connects gaming experience to practical tasks, thereby reinforcing each new Arduino concept in a scaffolded approach. In the subsequent chapter, we will undertake an evaluation of the game through user testing, evaluate learning outcomes, and analyze performance data collected from Firestore.

Chapter V. Evaluation and Validation

V.1) Introduction

The aim of this evaluation is to measure the success of *Smart in the Dark* in meeting its educational goals and engaging users. In serious game development, systematic evaluation and validation are essential to verify that the game effectively promotes learning and user engagement(Berkmen & Lande, 1975).

As one review notes, evaluation "helps determine if a serious game meets its educational objectives and truly promotes learning" and provides feedback on design and gameplay for improvement(Berkmen & Lande, 1975).

Conversely, a lack of rigorous evaluation has been cited as a major challenge in the field (Damaševičius et al., 2023)

Thus, assessing both outcomes and user experience will establish the game's credibility as an educational tool.

V.2) Evaluation Objectives and Tools

The purpose of this evaluation is to systematically assess both the educational effectiveness and the usability of *Smart in the Dark*, a serious game designed to enhance learning through interactive problem-solving and engagement. The game targets intermediate learners by simulating real-world challenges in a puzzle-based environment that promotes logical reasoning and strategic thinking. Grounded in constructivist learning theory, the game's design encourages active knowledge construction through experiential play.

In order to arrive at a sound and sufficiently thorough evaluation, a mixed methods approach was chosen. In other words, qualitative as well as quantitative research instruments were applied to collect complementary data. This approach is widely supported in serious game and educational technology research (Pacheco-Velazquez et al., 2023).

In this evaluation, four data collection tools were applied:

Pre-Game Questionnaire: To collect user characteristics (age, education, technical experience) and knowledge/skills related to the game's topics, in order to establish a knowledge baseline before the game was played (de Freitas & Oliver, 2006).

Post-Game Survey: To collect information about knowledge/skills gained, player experience/satisfaction and involvement after playing the game. Pre-game and post-game comparisons can be used for assessing the cognitive and affective effects of the game (Connolly et al., 2012).

System Usability Scale (SUS): A 10-item Likert scale survey conducted after gameplay for measuring the usability of the game, offering a quick, dependable instrument for measuring the usability of many different kinds of products and services alike (Brooke, 1996).

In-Game Analytics: The player's total clicks and time spent playing were recorded digitally inside the game. Such gameplay metrics are useful for gauging player engagement, which is key for the success of gamified applications (Hamari et al., 2014). In order to examine the research questions, we utilized an array of tools and data sources, employing both subjective (self-report questionnaires) and objective (digital logging) data collection methods. This triangulation enhances the credibility of the results and assists in understanding the role of the digital game as an educational tool and player experience hub.

V.3) Description of the Tests (Pre/Post, SUS, In-Game Metrics)

We tested *Smart in the Dark* remotely, sending the game to participants and inviting them to complete evaluation forms. We used the online user guide (https://smart-in-the-dark-guide.netlify.app/) to guide them to ensure as consistent an experience as possible.

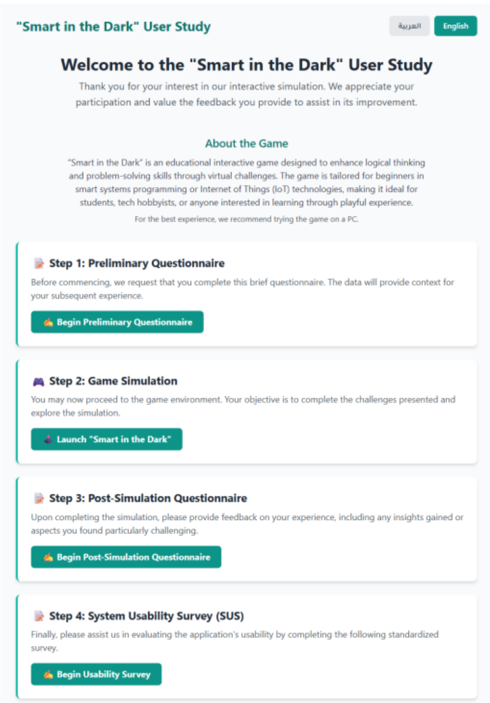
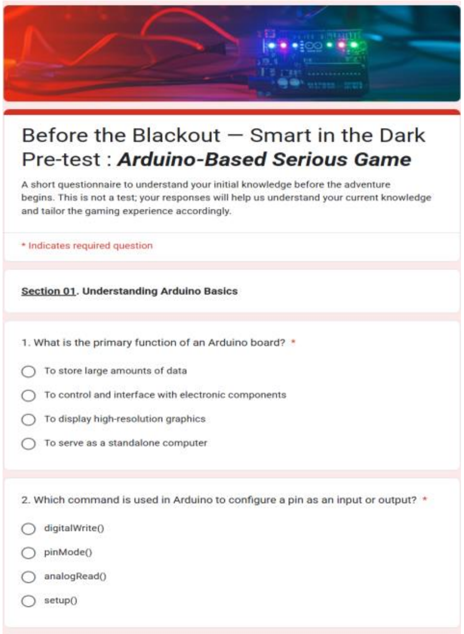


Figure 22 *Smart in the Dark" User Study*

To establish a baseline, we asked participants to complete a pretest, available via a Google Form (https://forms.gle/Zg8pkXsXxuyCcwnm8).

This survey collected demographic information and explored players' knowledge of the educational concepts presented in the game. This allowed us to measure players' learning levels by comparing their responses before and after the game.



Before the Blackout — Smart in the Dark
Pre-test : ***Arduino-Based Serious Game***

A short questionnaire to understand your initial knowledge before the adventure begins. This is not a test; your responses will help us understand your current knowledge and tailor the gaming experience accordingly.

* Indicates required question

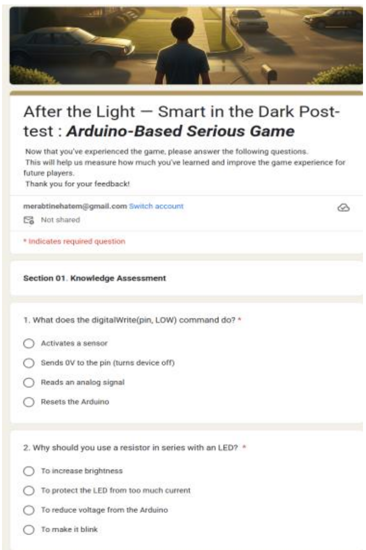
Section 01. Understanding Arduino Basics

1. What is the primary function of an Arduino board? *
- To store large amounts of data
- To control and interface with electronic components
- To display high-resolution graphics
- To serve as a standalone computer

2. Which command is used in Arduino to configure a pin as an input or output? *
- digitalWrite()
- pinMode()
- analogRead()
- setup()

Figure 23 *Screenshot from Google Form pre-test survey*

After playing the game, participants were invited to complete a pretest (https://forms.gle/G2ULdWjMZky4AVso6).



After the Light — Smart in the Dark Post-test : ***Arduino-Based Serious Game***

Now that you've experienced the game, please answer the following questions.
This will help us measure how much you've learned and improve the game experience for future players.
Thank you for your feedback!

merabtinehatem@gmail.com Switch account
Not shared

* Indicates required question

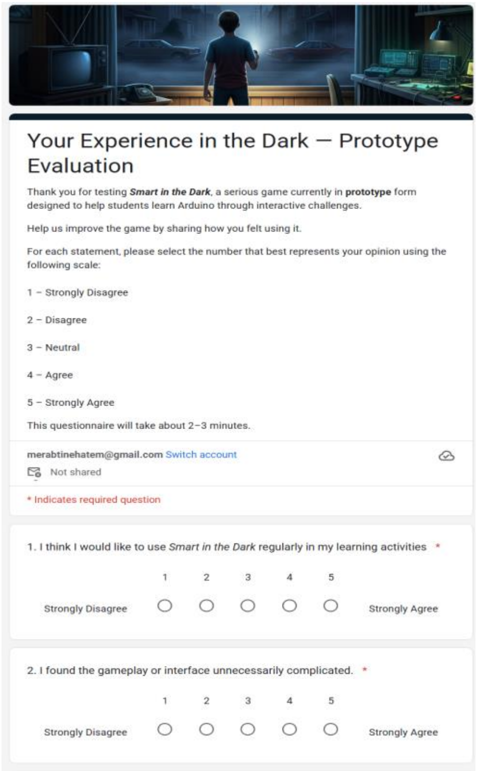
Section 01. Knowledge Assessment

1. What does the digitalWrite(pin, LOW) command do? *
- Activates a sensor
- Sends 0V to the pin (turns device off)
- Reads an analog signal
- Resets the Arduino

2. Why should you use a resistor in series with an LED? *
- To increase brightness
- To protect the LED from too much current
- To reduce voltage from the Arduino
- To make it blink

Figure 24 *Screenshot from Google Form post-test survey*

We primarily use the results of this test to determine what and how much our players learned from the game. Additionally, this survey helped us understand the audience's level of satisfaction with the game and their assessment of the educational value of their gaming experience. Finally, participants were directed to the “System Usability Scale” survey link via Google Forms (https://forms.gle/mqfkdjn4FyRzBWow5).


Your Experience in the Dark — Prototype Evaluation

Thank you for testing ***Smart in the Dark***, a serious game currently in **prototype** form designed to help students learn Arduino through interactive challenges.

Help us improve the game by sharing how you felt using it.

For each statement, please select the number that best represents your opinion using the following scale:

1 – Strongly Disagree

2 – Disagree

3 – Neutral

4 – Agree

5 – Strongly Agree

This questionnaire will take about 2–3 minutes.

merabtinehatem@gmail.com Switch account

Not shared

* Indicates required question

1. I think I would like to use *Smart in the Dark* regularly in my learning activities *

	1	2	3	4	5	
Strongly Disagree	○	○	○	○	○	Strongly Agree

2. I found the gameplay or interface unnecessarily complicated. *

	1	2	3	4	5	
Strongly Disagree	○	○	○	○	○	Strongly Agree

Figure 25 *Screenshot from Google Form SUS survey*

In addition to subjective evaluations, **in-game behavioral metrics** were automatically recorded by the system. Each player's total number of mouse clicks was logged, offering insight into the complexity of user interactions and the strategic demands of the game environment. Simultaneously, the total duration of gameplay was tracked for each participant to assess immersion and engagement levels.

The integration of these digital tools and automated data tracking mechanisms allowed for an efficient and scalable evaluation process. By capturing both qualitative feedback and quantitative performance data, the study achieved a well-rounded assessment of the game's usability and pedagogical effectiveness.

V.4) Analysis of results

V.4.1) Learning Outcomes: Pre- and Post-Game Questionnaire Analysis

This section presents a comparative analysis of participant performance on conceptually matched pre- and post-game questionnaire items, aimed at evaluating the educational impact of *Smart in the Dark*, a serious game designed to introduce foundational Arduino principles, components, and programming logic.

Methodology

Participants completed two distinct instruments: a pre-game questionnaire (n = 18) and a post-game questionnaire (n = 6). Both instruments included multiple-choice items designed to assess understanding of Arduino concepts such as digital signal functions, component roles (transistors, relays), and sensor selection. While the pre- and post-game questionnaires were not respondent-matched, questions were analyzed based on conceptual alignment. For each aligned question pair, correct responses were assigned a score of 1, and incorrect responses a score of 0. Mean accuracy (i.e., the proportion of correct responses) was computed for each question across all respondents.

The following conceptually aligned questions were analyzed:

- digitalWrite function: Pre-game Q3 vs. Post-game Q1
- Relay function and comparison: Pre-game Q2 vs. Post-game Q4
- Transistor role: Pre-game Q3 vs. Post-game Q3
- Sensor selection: Pre-game Q1 vs. Post-game Q5

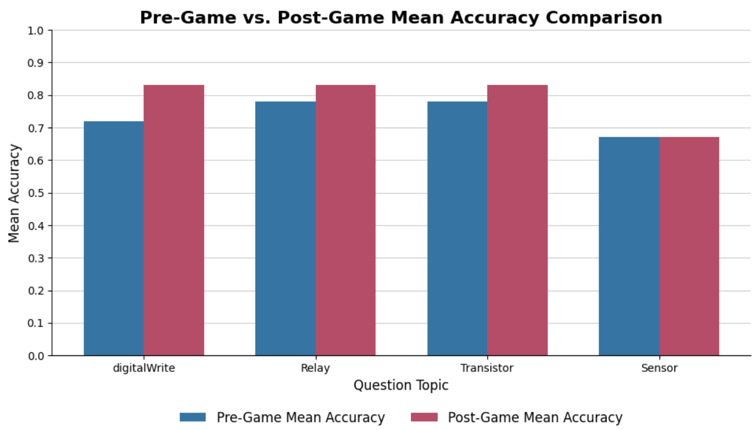


Figure 26 *Pre-Game vs Post-Game Mean accuracy comparison*

As shown in Figure 27, three out of the four conceptual areas demonstrated improved mean accuracy in the post-game responses. The largest gain was observed in understanding the function of digitalWrite, with a +11.11% improvement. Slight improvements were also noted in comprehension of relays and transistors (+5.56% each).

As results, as shown in Figure 28, the mean score increased from 79.01% (pre-game) to 83.33% (post-game), indicating a gain of: 4.32%

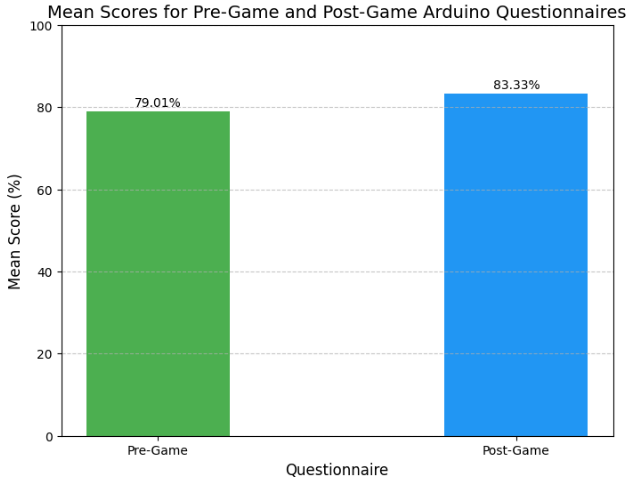


Figure 27 *Mean Scores for Pre-Game and Post-Game Arduino Questionnaires*

The results suggest a modest increase in Arduino-related knowledge following interaction with the game. The observed improvement of 4.32% in mean percentage scores may reflect the game's effectiveness in reinforcing key concepts. However, this result is constrained by a key limitation. The post-game sample (number (n) = 6) is significantly smaller than the pre-game sample (n = 18), reducing statistical comparability.

V.4.2) System Usability Scale (SUS) Evaluation of "Smart in the Dark"

The SUS is a standardized tool comprising ten items designed to assess a system's overall usability by evaluating perceived ease of use, learnability, and satisfaction. Responses are recorded on a 5-point Likert scale ranging from 1 ("Strongly Disagree") to 5 ("Strongly Agree").

- **SUS Items**: The ten items alternate between positive (odd-numbered) and negative (even-numbered) statements to mitigate response bias. The statements were adapted to the context of the serious game as follows:

1. Regular use of the game in learning activities
2. Perceived complexity of gameplay or interface
3. Ease of understanding and navigation
4. Need for technical support
5. Integration of Arduino puzzles, challenges, and feedback
6. Consistency of mechanics and instructions
7. Learnability for typical students
8. Frustration or difficulty in use
9. Confidence during gameplay and problem-solving
10. Prerequisite knowledge required before use

The SUS score was calculated for each participant according to the standard methodology:

- For positively worded items (odd-numbered): **score contribution = response − 1**
- For negatively worded items (even-numbered): **score contribution = 5 − response**
- The total SUS score per respondent is derived by summing the individual contributions and multiplying the result by 2.5 to yield a score on a 0–100 scale.

the table 9 summarize the results:

Table 9 *System Usability Scale (SUS) results with sus score*

id	Q1	Q2	Q3	Q4	Q5	Q6	Q7	Q8	Q9	Q10	SUS mean
1	1	4	3	5	2	5	2	4	1	4	17.5
2	3	1	2	5	5	3	2	4	4	4	47.5
3	3	2	4	2	4	4	3	2	4	4	60.0
4	4	3	4	4	4	4	4	2	4	2	62.5
5	4	3	4	4	4	5	4	2	4	2	60.0
6	5	2	4	1	5	2	4	3	5	2	82.5

Mean SUS Score: 55.0

Interpretation of Results

The mean SUS score of **55.0** is below the commonly accepted usability benchmark of **68**, indicating that while the game offers some positive user experiences, its overall usability may be rated as suboptimal. The variance in scores from **very poor usability** (17.5) to **excellent usability** (82.5) reflects heterogeneous experiences among users.

Key strengths include perceived ease of navigation, coherent integration of game mechanics with educational objectives, and player confidence. However, several participants expressed challenges with the game's onboarding experience, highlighting the need for additional technical guidance and clearer instructions.

V.4.3) In-Game Metrics

The dataset under analysis was exported from **Google Firestore**, a cloud-based NoSQL database, and stored locally in a JSON file format (InGameMetrics.json). The file encapsulates user interaction data collected during gameplay within the serious educational game *Smart in the Dark*. Structurally, the JSON object includes a top-level __collections__ field, with the principal collection titled users.

Each user is identified by a unique Firestore-generated document ID (e.g., *AyzK2qdCzLWGm1iGpFjETaESa1w2*) and contains nested fields representing gameplay metrics. These metrics include time spent in specific game areas, number of user interactions (clicks), and boolean flags denoting level completion.

Each user record captures detailed gameplay metrics, including time spent (in seconds), number of clicks, and level completion status. All 12 users interacted with the initial game area, room_0, spending between 11 and 215 seconds and registering between 5 and 9 clicks. The game also includes other distinct areas, such as Blockli_X and similator_X, which represent subcomponents of different levels. Beyond room_0, eight users advanced to these subsequent areas, and only two participants completed all four levels.

User progression varied: six users completed level 1, two reached level 2, and two finished levels 3 and 4. Engagement tended to increase in higher levels, both in time and click volume. For instance, one user spent 257s in Blockli_1, while another logged 324s and 47 clicks in level 1. Levels 3 and 4 required 143–191s and up to 33 clicks, indicating greater complexity or engagement.

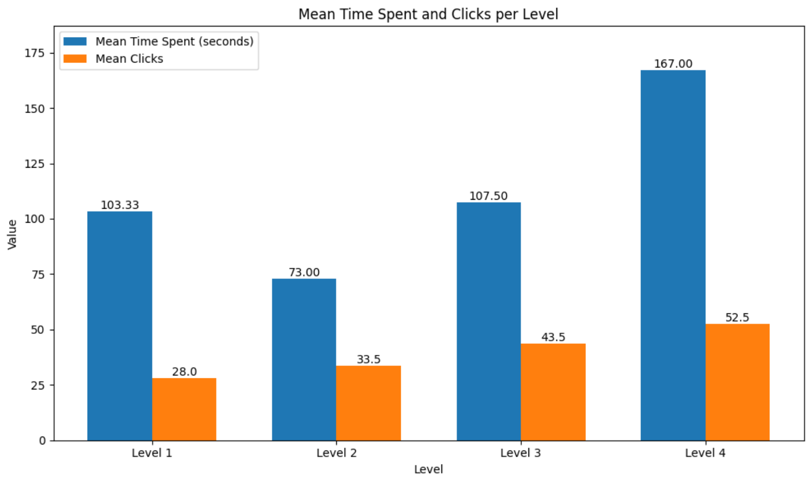


Figure 28 *Mean time spent and clicks per level*

In summary, the Firestore-exported JSON dataset provides detailed and structured insights into user engagement with *Smart in the Dark*. It captures user behavior through quantitative metrics such as time spent, interaction frequency, and progression tracking. The data suggests a varied pattern of user engagement, with some users exiting early and others completing all game stages. Figure 28 provides an overview of player's interactions (clicks) and engagement (time spent).

The observed increase in time and interaction in higher levels may point to escalating complexity, thereby offering valuable implications for future iterations of the game's design, particularly in terms of onboarding, pacing, and instructional support.

V.5) Learner feedback

Feedback from the post-game questionnaire reveals that *Smart in the Dark* effectively engages users with some prior Arduino knowledge, who praised the game's educational value and interactivity. However, novice users struggled with unclear instructions, limited guidance, and an absence of foundational explanations. Key user suggestions included clearer hints, video tutorials, hands-on projects, enhanced UI/UX, and a clearer distinction between beginner and intermediate learning paths. While most feedback was positive, indicating the game's potential as a learning tool, results highlight the importance of incorporating adaptive support and differentiated design to accommodate diverse user needs in educational serious games.

V.6) Limitations and Future Perspectives

While remote evaluation offered logistical advantages—such as broader reach and reduced cost—it also introduced several methodological limitations that impacted the validity and scope of the findings. First, the lack of control over participants' environments (e.g., device performance, screen size, internet stability) may have negatively influenced their engagement and comprehension. Additionally, the absence of real-time support restricted opportunities for clarification, particularly regarding gameplay hints and English-language instructions, which may have affected performance for non-native speakers.

Another significant limitation is the discrepancy in sample sizes: 18 participants completed the pre-game questionnaire, while only 6 completed the post-game version. This disparity hinders statistical comparability and limits the generalizability of observed learning gains. Moreover, the inability to pair pre- and post-game responses on an individual level prevents within-subject analysis, reducing the strength of causal inferences about learning outcomes.

To address these challenges, future evaluations should adopt a mixed-methods approach (Creswell & Plano Clark, 2017), combining remote and face-to-face testing to enhance ecological validity and capture contextual nuances. In addition, expanding the assessment toolbox to include psychological and motivational measures would enrich the understanding of learner experience. For instance:

The State-Trait Anxiety Inventory (STAI) (Spielberger et al., 1970) could assess affective states related to cognitive performance and engagement.

The Intrinsic Motivation Inventory (IMI) (Ryan, 1982) and the Flow State Scale (FSS) (Jackson & Marsh, 1996) are suitable for evaluating motivation, cognitive absorption, and emotional engagement during gameplay.

Integrating these tools alongside observational and gameplay metrics would provide a more comprehensive, contextually grounded evaluation of Smart in the Dark and its impact on both learning and learner psychology. This holistic framework is essential for the future refinement and empirical validation of serious games in educational settings.

V.7) Conclusion

This chapter presented an empirical evaluation of Smart in the Dark using a triangulated approach that combined pre-/post-game questionnaires, the System Usability Scale (SUS), and behavioral gameplay data. Together, these instruments offer a multidimensional view of the game's effectiveness in conveying basic Arduino concepts and fostering hands-on engagement through a block-based programming interface.

The pre/post comparison indicated modest improvements in conceptual understanding—most notably in functions like digitalWrite and the roles of transistors and relays—highlighting the potential of interactive learning environments to support foundational electronics education. Despite the limited sample size in the post-game phase, results suggest that Smart in the Dark enhances learner comprehension through simulation-based experimentation.

Qualitative feedback reinforced these findings: players appreciated the intuitive interface and practical application of concepts, though challenges with in-game guidance and theoretical depth were noted. To address this, future iterations of the game should prioritize clearer instructional scaffolding, progressive difficulty balancing, and improved conceptual framing.

Moreover, the study underscores the importance of complementing usability and performance metrics with psychological constructs. Future evaluations should incorporate instruments like the STAI, IMI, and FSS to capture motivational and emotional dimensions of the learning experience—critical yet often under-assessed aspects of educational game design.

In summary, Smart in the Dark demonstrates strong potential as a serious game for computer science education, particularly in Arduino instruction. Its continued development should emphasize both instructional refinement and deeper empirical evaluation, ensuring the game not only teaches effectively but also engages learners meaningfully.

Conclusion

This dissertation explored the design, development, and evaluation of *Smart in the Dark*; a serious game aimed at facilitating the learning of Arduino programming and embedded systems concepts. Through a pedagogically driven design approach, the game was developed to foster engagement, interactivity, and cognitive reinforcement for intermediate learners who have prior knowledge of Arduino programming and electronics. The research employed a mixed-methods evaluation framework consisting of pre-and post-tests, the System Usability Scale (SUS), in-game metrics, and qualitative feedback to assess both educational effectiveness and user experience.

The evaluation results indicate that *Smart in the Dark* holds considerable potential as a learning tool that effectively bridges theoretical concepts with practical application. While the small post-evaluation sample size limits the generalizability of the findings, several participants demonstrated notable improvement in their understanding of foundational Arduino concepts—particularly regarding digital outputs and basic circuit components. Behavioral interaction data, such as event tracking and activity duration, revealed moderate levels of engagement with the game mechanics, suggesting that the block-based interface and simulation elements were sufficiently intuitive to promote active exploration, though opportunities remain for enhancing sustained cognitive involvementHowever, the study also highlighted several limitations, particularly the challenges posed by remote evaluation settings, including reduced control over testing environments, limited access to the target learner demographic, and constraints in collecting observational feedback. Additionally, while the implemented tools captured cognitive outcomes effectively, emotional and motivational dimensions were underexplored due to the absence of psychological assessment instruments.

To build upon the foundations established in this thesis, several future directions are proposed for the continued development and educational deployment of *Smart in the Dark*.

Future Considerations for *Smart in the Dark*

1. *Enhancements in Gameplay and Learning Mechanics*

Future improvements should deepen the game's pedagogical impact through adaptive AI-based difficulty, collaborative multiplayer models, and the inclusion of advanced Arduino topics like PWM, interrupts, and IoT communication.

2. *Technical Improvements*

Integrating the AVR8js JavaScript-based emulator into BlocklyDuino would enable real-time code simulation and reduce hardware dependency (Wokwi, n.d.). Additional improvements include optimizing performance for low-end and mobile devices, expanding Firebase analytics, and ensuring multilingual, cross-platform compatibility.

3. *Narrative and Immersion Enhancements*

To sustain engagement, the game should offer more complex storylines, interactive characters, and dynamic environmental effects (e.g., weather conditions) that influence gameplay.

4. *Future Research and Educational Applications*

The game can be integrated into formal curricula in electronics and embedded systems. Institutional partnerships, ongoing usability testing, and future adoption of augmented reality (AR) will further expand its educational value.

In conclusion, *Smart in the Dark* has demonstrated its potential as a pedagogical tool that aligns with the goals of active learning and constructivist education. While the current implementation has laid a solid foundation, the proposed future directions offer a roadmap for transforming the game into a robust, scalable, and adaptable educational platform. Continued development, informed by research and user feedback, will ensure that the game remains both pedagogically effective and technologically innovative serving learners across multiple languages, educational systems, and cultural contexts.

References

Abdellatif, A., McCollum, B., & McMullan, P. (2018, March 14–16). *Serious games: Quality characteristics evaluation framework and case study* [Paper presentation]. 2018 IEEE Integrated STEM Education Conference (ISEC), Princeton, NJ, USA. https://doi.org/10.1109/ISECON.2018.8340460

Abdullah, A. A. (2012). *Simply Arduino* (4th ed.). [Self-published].

Al-Haila, A. (2002). *Al-Tufulah wa-wasa'il al-i'lam: Madkhal tanmawi* [Childhood and the media: A developmental approach]. Dar Al-Fikr Al-Arabi.

Al-Huwaidi, A. (2013). *Al-Al'ab al-tarbawiyya al-iliktruniyya* [Electronic educational games]. Maktabat Al-Anjlo Al-Misriyyah.

Alvarez, J. (2007). *Du jeu vidéo au serious game: approches culturelle, pragmatique et formelle* [From video game to serious game: Cultural, pragmatic, and formal approaches] [Doctoral dissertation, Université de Toulouse le Mirail].

Arias-Calderón, M., Castro, J., & Gayol, S. (2022). Serious games as a method for enhancing learning engagement: Student perception on online higher education during COVID-19. *Education Sciences, 12*(5),315. https://shorturl.at/3McnZ

Banzi, M., & Shiloh, M. (2014). *Getting started with Arduino: The open source electronics prototyping platform* (3rd ed.). Maker Media.

Becker, K. (2021). *Gamification, serious games, and game-based learning in higher education: An international review*. Springer.

Bellotti, F., Kapralos, B., Lee, K., Moreno-Ger, P., & Berta, R. (2013). Assessment in and of serious games: An overview. *Advances in Human-Computer Interaction*, *2013*, 1–11. https://doi.org/10.1155/2013/136864

Bergeron, B. (2006). *Developing serious games*. Charles River Media.

Caserman, P., Garcia-Agundez, A., Göbel, S., Tregel, T., & Steinmetz, R. (2020). Quality criteria for serious games: Serious part, game part, and

balance. *Entertainment Computing, 33*, 100341. https://shorturl.at/Y22Lt

Catalano, C. E., Luccini, A. M., & Mortara, M. (2014). The serious games evaluation framework (SEGA). *International Journal of Serious Games, 1*(1), 1– 11.

https://doi.org/10.17083/ijsg.v1i1.8

Charsky, D. (2010). From edutainment to serious games: A change in the use of game characteristics. *Games and Culture, 5*(2), 177–198. https://shorturl.at/96HVe

Csikszentmihalyi, M. (1990). *Flow: The psychology of optimal experience*. Harper & Row.

Damaševičius, R., Blažauskas, T., Štuikys, V., & Toldinas, J. (2023). On educational technologies: Serious games and gamification in engineering education. *Applied Sciences, 13*(3), 1452. https://doi.org/10.3390/app13031452

Deterding, S., Dixon, D., Khaled, R., & Nacke, L. (2011). From game design elements to gamefulness: Defining "gamification." In *Proceedings of the 15th International Academic MindTrek Conference* (pp. 9–15). ACM. https://rb.gy/vqpsuu

Dickey, M. D. (2005). Engaging by design: How engagement strategies in popular computer and video games can inform instructional design. *Educational Technology Research and Development, 53*(2), 67–83.

Dubois, F. (2021, April 11). *Use Firebase in your GDevelop games*. GDevelop Blog. https://gdevelop.io/blog/use-firebase-in-gdevelop

Fabricatore, C., & López, X. (2012). Sustainability learning through gaming: An exploratory study. *Electronic Journal of E-Learning, 10*(2), 209–222.

Firebase. (n.d.). *Cloud Firestore*. Firebase Documentation. Retrieved November 21, 2023, from https://firebase.google.com/docs/firestore

García-Tudela, P. A., & Marín-Marín, J. A. (2023). Use of Arduino in primary education: A systematic review. *Education Sciences, 13*(2), 134.

https://doi.org/10.3390/educsci13020134

Gee, J. P. (2007). *What video games have to teach us about learning and literacy* (2nd ed.). Palgrave Macmillan.

Giacomin, A., et al. (2016). Arduino: A low-cost solution for physical computing education. *Journal of Computing Sciences in Colleges, 31*(6), 116-118.

Google. (n.d.). *Blockly*. Retrieved November 21, 2023,

from https://developers.google.com/blockly

Hektner, J. M., & Csikszentmihalyi, M. (1996, April 8–12). *A longitudinal exploration of flow and intrinsic motivation in adolescents* [Paper presentation]. Annual Meeting of the American Educational Research Association, New York, NY, United States.

Huizinga, J. (1950). *Homo ludens: A study of the play-element in culture*. Beacon Press.

Hunicke, R., LeBlanc, M., & Zubek, R. (2004). MDA: A formal approach to game design and game research. *Proceedings of the AAAI Workshop on Challenges in Game AI*, 1–5.

Hurrell, S. A. (2017). The eLearning benefits and challenges of simulation-based learning. *The Journal of Teaching and Learning for Graduate Employability, 8*(1), 163–171. https://doi.org/10.21153/jotlge2017vol8no1art686

Ilyukha, V. (n.d.). *What is Firebase?* Jelvix. Retrieved November 21, 2023, from https://jelvix.com/blog/what-is-firebase

Juul, J. (2005). *Half-real: Video games between real rules and fictional worlds*. MIT Press.

Kartaca. (2024). *Implementing analytics in HTML5 games with Firebase*. https://kartaca.com/en/articles/firebase-analytics-html5-games

Kiili, K. (2005). Digital game-based learning: Towards an experiential gaming model. *The Internet and Higher Education, 8*(1), 13–24. https://tinyurl.run/RwVUYj

Kolb, D. A. (1984). *Experiential learning: Experience as the source of learning and development*. Prentice-Hall.

Kushner, D. (2011, October 26). The making of Arduino. *IEEE*

Spectrum. https://spectrum.ieee.org/the-making-of-arduino

Margolis, M. (2011). *Arduino cookbook*. O'Reilly Media.

Martin, T. L., & Shen, J. (2011). Gaming to learn: Serious games to train information technology professionals. *International Journal of Cyber Behavior, Psychology and Learning, 1*(4), 45–61.

Mellis, D. A., Banzi, M., Cuartielles, D., & Igoe, T. (2007, April 28 – May 3). *Arduino: An open electronic prototyping platform* [Conference session]. CHI '07 Extended Abstracts on Human Factors in Computing Systems, San Jose, CA, USA.

Michael, D., & Chen, S. (2006). *Serious games: Games that educate, train, and inform*. Thomson Course Technology.

Monk, S. (2017). *Programming Arduino: Getting started with sketches*. McGraw-Hill Education.

Natkin, S. (n.d.). *Video game design and learning*. [Unpublished lecture material].

Nussey, J. (2013). *Arduino for beginners: Essential skills every maker needs*. John Wiley & Sons.

Ravyse, W., Blignaut, A. S., Leendertz, V., & Woolner, P. (2017). Success factors for serious games to enhance learning: A systematic review. *Virtual Reality, 21*(1), 31–58. https://doi.org/10.1007/s10055-016-0298-4

Rival, F. (n.d.). *GDevelop*. Wikipedia. Retrieved November 21, 2023, from https://en.wikipedia.org/wiki/GDevelop

Rubin, M. B., Burgstahler, S., & Means, D. R. (2012). Blocklyduino: A web-based visual programming editor for Arduino. *Journal of Computing Sciences in Colleges, 28*(2), 35–42.

Salen, K., & Zimmerman, E. (2004). *Rules of play: Game design fundamentals*. MIT Press.

Samčović, A. B. (2018). Serious games in military applications. *Vojnotehnički glasnik/Military Technical Courier, 66*(4), 869-893. https://tinyurl.run/ggKKw3

Shute, V. J. (2008). Focus on formative feedback. *Review of Educational Research, 78*(1), 153–189. https://doi.org/10.3102/0034654307313795

Tricot, A. (1999). *Apprentissage, nouvelles technologies et scénario pédagogique* [Learning, new technologies, and pedagogical scenarios]. Interview by O. Glassey. Tecfa, University of Geneva. https://tinyurl.run/wKzaX6

Westera, W., Nadolski, R., Hummel, H., & Wopereis, I. (2008). Serious games for higher education: A framework for reducing design complexity. *Journal of Computer Assisted Learning, 24*(5), 420–432.

Wokwi. (n.d.). *Wokwi Arduino simulator*. Retrieved November 21, 2023, from https://wokwi.com/

Yusoff, A., Crowder, R., Gilbert, L., & Wills, G. (2009). A conceptual framework for serious games. In *9th IEEE International Conference on Advanced Learning Technologies* (pp. 21–23). IEEE. https://doi.org/10.1109/ICALT.2009.23

Zyda, M. (2005). From visual simulation to virtual reality to games. *Computer, 38*(9), 25–32. https://doi.org/10.1109/MC.2005.297

s