Basic Concepts of Computer Architecture Through Games and Game Development

Edurne Larraza-Mendiluze, Olatz Arbelaitz Gallego, Olatz Arregi Uriarte, Jose Ignacio Martín Aramburu, and Jose Francisco Lukas Mugika

***Abstract*— The aim of this research is to study how games and game development can help students learn introductory concepts of a very abstract topic such as Computer Architecture. In a quasi-experimental scenario, quantitative and qualitative data has been collected from students before and after the intervention throughout three consecutive school years. Following the action-research methodology, year by year several changes have been implemented in the games, pursuing a better understanding of the concepts. Also during the last year a game development has been added in order to consolidate knowledge. No statistical differences were found between the knowledge acquired by students in the control and experimental groups during the first two years. Therefore, and given that motivation was higher when using games, the same game-based methodology was used in the third year in both groups. Afterwards, the students had to develop an interactive presentation in the form of an Escape Room to teach younger students the concepts they had learned. Only a few concepts gained knowledge after this intervention, but those concepts with less prior understanding did. Although advanced computer architecture concepts can be difficult to handle through games, there are basic concepts that can be worked on in this way. In addition, the excitement and motivation provided by games make a good introduction to the subject. Also the development of simple games helps to understand some of the concepts that were not well understood before.**

***Index Terms*— Computer architecture, basic concepts, game based learning, game development based learning.**

I. Introduction

IN JULY 2023 the article *Juegos para presentar conceptos básicos de Arquitectura de Computadores* [1] was presented at JENUI (Conference on University Level Informatics Education, from its Spanish acronym) and was chosen as one of the best papers of the conference and proposed to be sent to IEEE-RITA. This paper, therefore, is an extension of the previous one showing the results obtained during one more course in which the experience has been applied. In addition to several adjustments in the proposed games, another intervention is proposed through group development of games to reinforce the learned concepts.

This proposal addresses the lack of basic computer architecture knowledge among first-year Artificial Intelligence students. According to the SCIE/CODDI working group's report on pre-university education in Computer Science [2], students should have the following knowledge: in primary school, students should know the components of a computer and their roles in its operation; in secondary school, students should be able to design simple digital circuits with logic gates, understand the hardware and software components of a computer, and know how these components communicate with each other and other computers. Additionally, they should understand how instructions are stored and executed in a computer; in high school science programs, students should understand the Von Neumann architecture.

Computer architecture is a subject that students usually find difficult to assimilate [3]. As much as they are used to using computers, there are several levels of abstraction that hide the inner workings of the machine from the users. That is why getting to the bottom of the abstraction levels becomes complicated for students, even more so when there is no prior knowledge.

In the degree of Computer Engineering there are several subjects that teach the concepts little by little and deepening in each of them, from the point of view of the design of the architecture. On the other hand, in the Artificial Intelligence (AI) degree, the students' vision of the architecture is intended to be much more functional and with concepts more related to the performance of the machines they will use in the future. Therefore, students are expected to deal with concepts such as program execution in assembly language, memory hierarchy, cache memory, instruction level parallelism (ILP) or the problems posed by storage systems.

In order to reduce the initial knowledge gap, in this experience we propose to dedicate a limited amount of time of the teaching hours to the introduction of basic concepts through games. In the literature one can find an infinite number of references on game-based learning (GBL), not only in digital gaming [4], but also in analog [5]. All of them refer to

Received 21 December 2023; revised 19 April 2024 and 9 May 2024; accepted 21 May 2024. Date of publication 11 September 2024; date of current version 14 October 2024. This work was supported in part by the Department of Education of the Basque Government to research group ADIAN under Grant IT1437-22 and in part by the Educational Advisory Service of the University of the Basque Country/Euskal Herriko Unibertsitatea under Grant IKDi321-18. *(Corresponding author: Edurne Larraza-Mendiluze.)*

This work involved human subjects or animals in its research. Approval of all ethical and experimental procedures and protocols was granted by CEISH (UPV/EHU) (IF PROVIDED under Application No. M10/2022/126.

Edurne Larraza-Mendiluze, Olatz Arbelaitz Gallego, Olatz Arregi Uriarte, and Jose Ignacio Martín Aramburu are with the Department of Computer Architecture and Technology, University of the Basque Country (UPV/EHU), 20018 Donostia-San Sebastián, Spain (e-mail: edurne.larraza@ehu.eus).

Jose Francisco Lukas Mugika is with the Department of Research and Diagnostic Methods in Education, University of the Basque Country (UPV/EHU), 20018 Donostia-San Sebastián, Spain.

A Spanish version of this article is available as supplementary material at https://doi.org/10.1109/RITA.2024.3458863.

Digital Object Identifier 10.1109/RITA.2024.3458863

motivation, which is one of the elements to be analyzed in this work.

In addition, although with less depth since it is the experience of a single year, it is also proposed to work on game development based learning (GDBL). Already in 2006, El-Nasr and Smith [6] began to talk about the possibilities offered by video game development environments, since they make it possible not to design games from scratch but to modify them and place us in front of the constructionism of Papert [7], who argues that it is by building that we really learn.

From this point, section II intends to relate this work with others carried out with GBL and GDBL methodology, around computer architecture. Section III, presents the methodology used in the research while the games used in the different interventions are related in sections IV and V. Section VI analyzes the results obtained in the experiments carried out, and finally, section VII shows the conclusions obtained and the lines on which further work is to be done.

II. Related Work

An article published in 2016 [8] makes a systematic review of games used in higher education to teach computer science. Of the 107 articles it finally analyzes, only three talk about computer architecture (CA) and the article explicitly mentions that much of the material found in the area of CA are simulations rather than games, the reason why they have not included them in the study.

Of the three articles mentioned, the first two do not describe the games, but focus on the methodology, either teaching [9], or the design of serious games [10]. The last reference [11] is to a web page in which a live simulation is proposed, where the role of the components is played by the students. The proposal is for students as young as 4-6 years old to understand the operation of the computer and its peripherals.

Another live simulation, but with more advanced concepts is presented in [12]. Something similar to the simulators of the Little Man Computer machine [13], [14], [15].

On the other hand, the Moon game, which is presented in Section III and has been analyzed from the point of view of the development of computational thinking in [16], can be used to introduce Boolean logic.

Three additional proposals utilizing digital games have been identified [17], [18], [19]. The first two proposals do not specify which computer architecture concepts are covered in the game. The third proposal addresses the execution phases, detailing the steps involved in each phase, and also introduces the interruption phase.

Regarding the use of game development as a learning methodology in the field of computer architecture, several proposals have been found. Most of these focus on assembly programming [20], [21], [22], [23], while one proposal addresses low-level Input/Output operations in law level C language [24].

This list of references shows that this is a field of interest but that there is still much to be done in it. The article presented here is intended to pave the way in this field of research.

III. Contextualization and Research Methodology

The study was conducted over three consecutive academic years, from 2021-2022 to 2023-2024, in the “Introduction to Computer Architecture” course for first-year AI degree students. An action-research perspective was employed, analyzing the results obtained each year and making adjustments accordingly for the following year.

In the first two courses, 4.5 hours of class time during the first week, including both theoretical and practical lessons, were dedicated to this purpose. Along with the game instructions, the learning objectives were explained. After each game, a discussion was held to review the game dynamics and the concepts covered, linking them to the course objectives.

The course is offered in two groups, allowing for a quasi-experimental design (the groups were not randomly selected), with a control group and an experimental group. Each group receives instruction in a different language, but the same instructor teaches the introductory subject in both groups.

To obtain quantitative data, a test was administered before the intervention (pretest) and another after (posttest). In both tests, students were asked to define the concepts covered in the introduction to the subject: programming, control structures, algorithms, compilation, binary system, bit, byte, main memory, CPU, control unit, and instruction execution phases.

The pretest showed that both groups (experimental and control) were equivalent, meaning neither group started with superior knowledge. Subsequently, the new methodology, involving board games that will be explained in section IV, was implemented in the experimental group. Meanwhile, the control group continued with the usual methodology, which is based on explanations and exercises.

In the post-test, it was assessed whether both the control and experimental groups had improved since the pre-test. Additionally, it was examined if there were significant differences between the performance of the control group and the experimental group.

In the 2021-2022 academic year, 23 students in the experimental group and 19 in the control group were fully tracked. In the 2022-2023 academic year, 19 students in the experimental group and 25 in the control group were tracked. In the last course, although the students were still divided into two groups, the data were analyzed jointly since the same methodology was used in both groups. The responses of a total of 50 students were analyzed.

Quantitative and qualitative data collection procedures were developed using a mixed design, specifically Creswell’s convergent design [25] (Figure 1).

In addition, the post-test included an evaluation of the games and methodology. In the first two courses, this evaluation was conducted with both the experimental and control group students, allowing for multiple comparisons. The experimental and control groups were compared with each other both before and after the intervention, and the improvements within each group were analyzed separately. In the final course, only the pretest and posttest results were compared, treating the students as a single group since the same methodology was used for both groups.

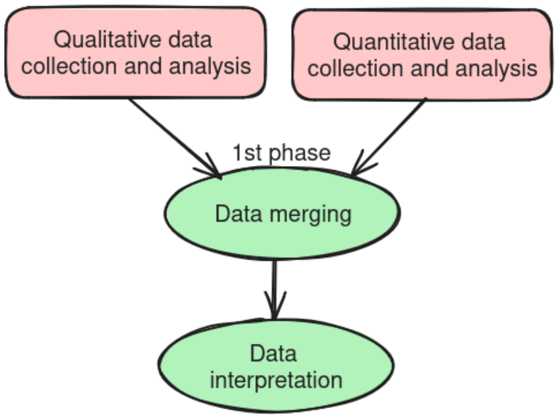

Qualitative data collection and analysis
Quantitative data collection and analysis
1st phase
Data merging
Data interpretation

Fig. 1. Creswell's convergent design.

The definitions provided by students in both the pretests and posttests were evaluated separately by two teachers who are proficient in the subject but did not teach the courses. First, the correct definitions of the concepts were agreed upon based on thematic bibliographies. The answers were then rated on a scale from 0 to 4, where 0 means no answer, 1 means incorrect answer, 2 means some idea but not correct, 3 means some errors but close to the correct answer, and 4 means the answer is correct. The mean score between the two evaluations was calculated. Finally, the data were analyzed using the SPSS statistical program (version 27). For group comparisons, Student's t-distribution was used: "Student's t for related groups" for comparisons within the same group and "Student's t for unrelated groups" for comparisons between different groups.

During the first course, focus groups were held with students to gather qualitative information. These discussions were recorded and transcribed with the students' permission, and questions about the new methodology were raised. The moderator of the focus group, although a member of the research group, was external to the subject and had no relationship with the students during the intervention. Several open-ended questions were also included in the post-test. The evaluation of the texts from the focus group discussions and the open-ended post-test questions showed little variation. As a result, in subsequent courses, only the open-ended questions in the post-test were used.

To analyze the qualitative information, a comprehensive analysis [26] of the texts collected from the focus groups was conducted.

In the last course, an additional activity was introduced where, after the introductory week, students worked in groups of four to develop a digital escape room. To enhance motivation, students were informed that their escape rooms would be used to teach the concepts they had learned to high school students. However, due to the limited time available for this activity, coordination with other centers was not feasible, and the escape rooms were not tested with high school students. The authors intend to review the escape rooms to correct any errors and eventually make them available to high school teachers.

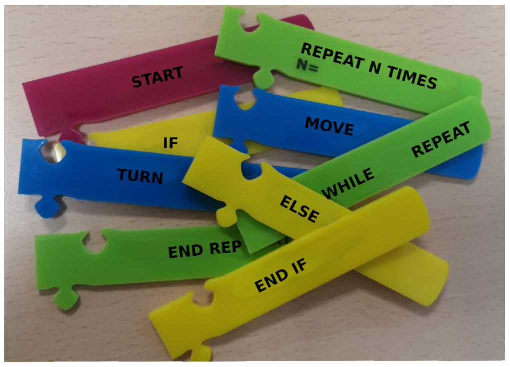


Fig. 2. Pieces to program the mazes.

The escape rooms were developed using Genially [27], a platform that does not require programming but incorporates programming concepts such as sequencing (of the images) and jumping (for interactivity). This activity was designed based on the game-based learning model. All students who participated in the study took part in the escape room design activity. After developing the escape rooms, the post-test was administered again to observe differences compared to their previous knowledge, specifically the knowledge they had at the time of the first post-test.

To draw conclusions, the results of the quantitative and qualitative analyses were combined and interpreted. The findings are presented in section V.

IV. Proposed Games

This section will explain the games used in the interventions. The following section will detail which games were used in each course and the reasons for changing some games from one intervention to another. Classroom sessions are 90 minutes long. The time spent on each game is indicated next to it, with an additional 15 to 20 minutes allocated for group discussion. This section will explain the games used in the interventions. The following section will detail which games were used in each course and the reasons for changing some games from one intervention to another. Classroom sessions are 90 minutes long. The time spent on each game is indicated next to it, with an additional 15 to 20 minutes allocated for group discussion.

A. Programming Mazes (75 min)

This is a challenge game where students must program paths to reach a desired object. Similar mazes can be found on the code.org website. However, the key difference here is that the programming blocks used in this activity are physical, not digital (Figure 2). Observing the use of these physical blocks has led us to conclude that, without the option to execute the program, students focus more on the logic of the code rather than simply whether it works or not.

B. The Lost Card (30 min)

According to the Merriam-Webster Dictionary [28] an algorithm is "*a step-by-step procedure for solving a problem or accomplishing some end*". It often involves a series of operations that must be performed in a specific order. It is an essential concept within AI and although it will certainly be introduced in programming subjects, it is also important in this subject.

This game is played using a deck of cards with multiple suits, each suit containing the same number of cards, and the cards within each suit arranged in a specific order.

The students are divided into groups of 3 or 4 people. Each group uses a deck of cards from which they draw one card without knowing its identity. Once the card is drawn, they must create a set of instructions to identify the missing card. The students know the deck's composition, with the drawn card being the only one they cannot see, and they have random access to the rest of the cards. Each group devises an algorithm to quickly find the missing card. Another group then uses this algorithm to search for the missing card and compares it with their own, analyzing the differences.

This game is very useful for students to understand the importance of writing clear and unambiguous instructions. Additionally, even without delving deeply into the algorithms, students can observe that some algorithms are faster than others.

C. Dice Race (30 min)

This game is proposed on code.org [29] and continues to explore the concept of algorithms, introducing control structures such as for and while loops and conditionals. It can be played in small groups or with the whole class and consists of three rounds.

In the first round, each player rolls the dice, and the winner is the one with the highest score. The concept of a while loop is illustrated as the action of rolling the die is repeated as long as there are players.

In the second round, each player rolls the die three times, and the winner is the one with the highest combined score. This round introduces the for loop concept, as the dice rolls are repeated three times.

The last round is similar to the second round, but if someone rolls a 1, they roll the die again, introducing the concept of conditionals.

In addition to introducing the three basic control structures, the game emphasizes the importance of unambiguous instructions. For example, issues such as ties or whether players should roll three times in a row or take turns in the second round highlight the need for clarity. These details, though easily resolved by people, require precise instructions for a computer to understand and execute correctly.

D. Magic: Guess the Day of My Birthday (15min)

It is important to explain that no matter how well an algorithm is followed or how well a program is written in a high-level language, this code will not be understandable by the computer. The computer only understands combinations of 0s and 1s, which is known as binary code.

To introduce the binary system of number representation, a magic trick is used [30]. After performing the trick, its workings are explained, highlighting that it functions because

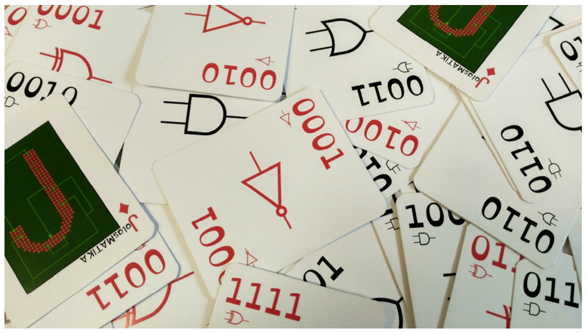


Fig. 3. Binary cards.

the binary system is a positional system where each position has only two possible values. The relationship between the binary and decimal systems, both positional systems, is then clarified, along with the patterns they follow.

E. Binary Cards (30 min)

The game involves using binary cards to play a Spanish card game called "sixes,". In this case the starting card is "0100" with an OR sign. The challenge lies in the binary nature of the cards, as depicted in Figure 3. Despite each card showing four bits, the values for each suit range from 0 to 7. This setup encourages students to grasp and internalize the binary representation of these values. Any game in which the card numbers are used consecutively would be valid.

F. The SUN Game (75 min)

This game is the most complex among all those previously used. It is built upon Little Man Computer [13], [14]. This game is a variation of these simulators.

On one hand, it aims to be more tangible than a digital game, drawing on research [32] that suggests tangible elements help connect physical forms and actions with symbolic representations. On the other hand, unlike the previously mentioned machines that use an accumulator as an input register for the Arithmetic Logic Unit (ALU), this one doesn't, as the machines used later in the course do not follow this design.

The objectives of the game are four:

1. To get to know the basic elements of a computer by means of a simplification (Figure 4).
2. To know the execution phases through which each of the instructions of a program must pass: search, decoding and execution. Here we can also talk about operand reading, execution and writing of results.
3. To understand how the instructions are executed step by step.
4. To become familiar with assembly language.

In the first challenge they have to execute (Figure 5), the variables x and y are set to 2 and 3, respectively. Additionally, a register (r3), which controls the loop, is set to 5 and decremented by 1 in each iteration until it reaches 0, at which point the loop exits. During these five iterations, the position (x, y) is painted on the screen and the value of x is incremented each time. Consequently, the positions from x=2 to x=6 are painted in the row where y=3 (Figure 6).

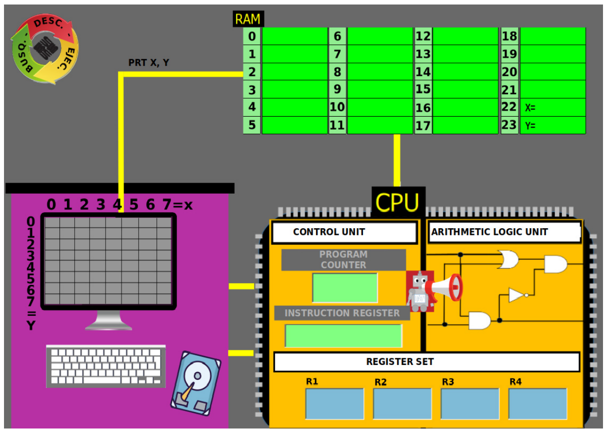
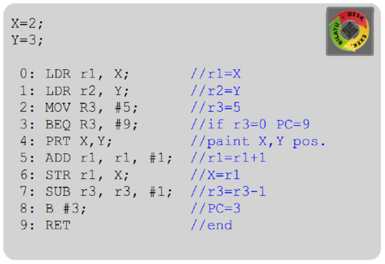


Fig. 4. SUN game board.



```
X=2;
Y=3;

 0: LDR r1, X;        //r1=X
 1: LDR r2, Y;        //r2=Y
 2: MOV R3, #5;       //r3=5
 3: BEQ R3, #9;       //if r3=0 PC=9
 4: PRT X,Y;          //paint X,Y pos.
 5: ADD r1, r1, #1;   //r1=r1+1
 6: STR r1, X;        //X=r1
 7: SUB r3, r3, #1;   //r3=r3-1
 8: B #3;             //PC=3
 9: RET               //end
```

Fig. 5. First program of the SUN game.

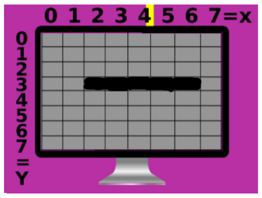


Fig. 6. Result.

Through these challenges, students can grasp the execution phases of an instruction and become familiar with the assembly language instructions that will be used later in the course. They will also learn how to convert the control structures from the game "Dice Race" into assembly language.

G. The MOON Game (45 min)

MOON is a board game designed by Pablo Garaizar [33]. It features 4 registers of 4 bits each, where players must conduct arithmetic and logical operations to obtain the values depicted on the target cards. Each operation incurs a specific energy cost. If players fail to reach the target value with their available energy, an additional target card must be added to the stack. If the stack reaches its maximum capacity, the lunar operation cannot be executed successfully. Conversely, successfully achieving all objectives indicates a successful mission (Figure 7).

H. Game 1-2-3 (15 min)

The objective of this game is to help students grasp the idea that all instructions executed in a machine go through

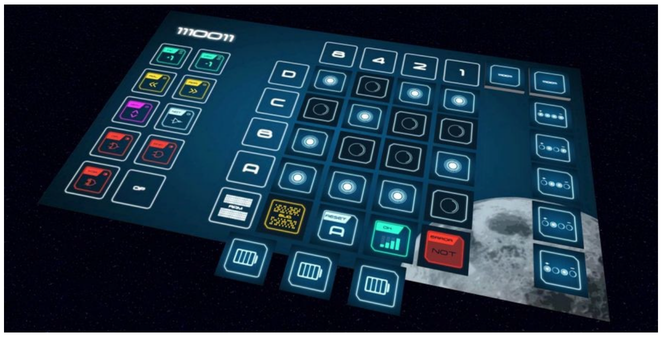


Fig. 7. MOON game board.

specific phases, each with its own timing. To facilitate this understanding, each student is provided with a deck of binary cards (Figure 4), with different suits representing logic gates such as AND, OR (black cards), XOR, and NOT (red cards).

Before the game begins, the teacher instructs the students to organize the cards into specific groups, for instance, separating the red cards from the AND cards, with the remaining cards forming a third group.

Then, in a synchronized manner, the teacher counts "1, 2, 3", prompting the students to draw a card from the deck without looking at it on "1" (fetch), revealing it on"2" (decode), and placing it in the designated pile on "3" (execute). This exercise helps students understand and differentiate the basic phases of instruction execution: Fetching, Decoding, and Execution.

While this game serves as an introductory activity to understanding the sequential phases of instruction execution, students will delve deeper into these phases in later lessons.

V. Games by Course

This section begins by listing the games utilized in each academic year. It then outlines the rationale behind the decisions to remove certain games and introduce new ones in subsequent academic years.

In the 2022-2023 course, the maze programming game utilized in the previous academic year (2021-2022) was substituted with the lost card and the dice race. This change wasn't primarily driven by the learning outcomes of the prior course, but rather by the specific attributes of these games. As first games to be presented, the lost card and the dice race offer quicker execution, providing more opportunities for play and facilitating group discussion about the objectives.

When working with the binary system, in the 2021-2022 course, binary cards were used to play "sixes" after an introduction to the binary system. Subsequently, the MOON game was introduced. However, upon analyzing the functioning of these games, it was evident that the MOON game required extensive explanation and execution time, despite the fact that logical operations, which are the main focus of the game, were not particularly relevant to the subject matter. Consequently, for the 2022-2023 course, the decision was made to introduce the magic game and remove the previous ones.

Additionally, there was a perceived lack of assimilation of the binary system in this experience. Therefore, in the

TABLE I
SELECTION OF GAMES FOR EACH COURSE

	A)	B)	C)	D)	E)	F)	G)	H)
2021-2022	X				X	X	X	
2022-2023		X	X	X		X		
2023-2024		X	X	X	X			X

TABLE II
RESULTS FOR COURSES 21-22 AND 22-23

item	concept	21-22 improvements in:			22-23 Improvements in:		
		Ctrl.	Exp.	Exp. vs Ctrl.	Ctrl.	Exp.	Exp. vs Ctrl.
(1)	Programming	✗	✓	=	✗	✗	=
(2)	Control structures	✗	✓	+	✓	✓	=
(3)	Algorithms	✓	✓	=	✓	✓	-
(4)	Compilation	✗	✗	=	✓	✓	-
(5)	Binary System	✓	✓	+	✗	✓	=
(6)	Bit	✓	✓	=	✓	✓	=
(7)	Byte	✓	✓	=	✓	✓	=
(8)	Main Memory	✓	✗	-	✓	✓	=
(9)	CPU	✓	✗	=	✓	✓	=
(10)	Control Unit	✓	✗	=	✓	✓	+
(11)	Instr.'s execution phases	✗	✓	=	✓	✓	=

(✓) Improvements in postest, (✗) No improvements in postest. Exp gr. iqual (=), better (+), worse (-) than Ctrl gr..

TABLE III
RESULTS OF FIRST AND SECOND POSTESTS IN COURSE 23-24

item	concept	First postest	Second postest
(1)	Programming	✓	✗
(2)	Control structures	✓	✗
(3)	Algorithms	✓	✗
(4)	Compilation	✓	✗
(5)	Binary System	✓	✗
(6)	Bit	✓	✓
(7)	Byte	✓	✓
(8)	Main Memory	✓	✗
(9)	CPU	✓	✗
(10)	Control Unit	✓	✓
(11)	Instr.'s execution phases	✓	✗

(✓) Improvements in postest, (✗) No improvements in postest.

TABLE IV
MINIMUM, MAXIMUM, AND AVERAGE NULL AND CORRECT ANSWERS FOR YEARS 21-22 AND 22-23

		Experimental		Control	
		Pretest	Postest	Pretest	postest
Nulls 21-22	Max.	69,57%	47,83%	84,21%	47,37%
	Min.	13,04%	0,00%	26,32%	0,00%
	Avg.	43,48%	17,39%	59,33%	16,75%
Corrects 21-22	Max.	34,78%	86,96%	21,05%	73,68%
	Min.	0,00%	0,00%	0,00%	5,26%
	Avg.	13,65%	20,55%	8,61%	25,36%
Nulls 22-23	Max.	96,74%	63,16%	100,00%	58,33%
	Min.	15,79%	0,00%	16,67%	0,00%
	Avg.	57,42%	19,14%	56,82%	21,21%
Corrects 22-23	Max.	21,05%	63,16%	16,67%	33,33%
	Min.	0,00%	0,00%	0,00%	0,00%
	Avg.	2,87%	16,27%	3,03%	12,88%

2023-2024 course, the binary cards game was reintroduced after being preceded by an explanation provided through the magic game.

The SUN game, which integrates various concepts, has been utilized in the first two courses. While it has effectively achieved its objectives, there has been a perception that it demands too much time, considering that assembly language, a subject requiring extensive time, is still being covered in the course. Therefore, for the third course, the 1-2-3 game has been introduced as a replacement. This game is much quicker, allowing more time to discuss the significance of program execution phases. Análisis de los resultad

VI. ANALYSIS OF RESULTS

A. Quantitative Analysis

In summary, TABLES II and III illustrate the variances in results between the pretest and posttest. The outcomes for each course are outlined below. It's worth noting that there were no significant differences observed between the pretests of each group across all courses. Hence, it's reasonable to assume that both groups commenced the experiment under similar circumstances.

B. First Year (2021-2022)

Both groups have shown significant improvements in various aspects. Regardless of the teaching methodology employed, both groups have enhanced their initial understanding. Upon comparing the posttests between the two groups, it was observed that the experimental group demonstrated a better comprehension of control structures and the binary system, whereas the control group exhibited a stronger grasp of the concept of main memory.

C. Second Year (2022-2023)

Both the experimental group and the control group experienced notable enhancements across nearly all items. Four items exhibited significant disparities between the experimental and control groups, with two favoring the experimental group and two favoring the control group. Notably, neither group demonstrated significant improvement in defining programming, possibly because it is a concept extensively covered in pre-university studies.

D. Third Year (2023-2024)

Both groups were analyzed collectively as they utilized the same methodology. Significant differences between the pretest and posttest were observed across all items. This outcome validates the efficacy of the methodology employed and strengthens the findings from previous courses.

In the second post-test, notable improvements were observed only in three items: bit, byte, and control unit. These specific items previously exhibited a high occurrence

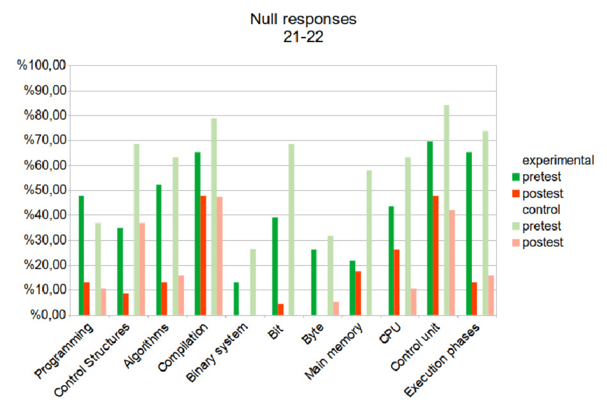


Fig. 8. Null responses in course 21-22.

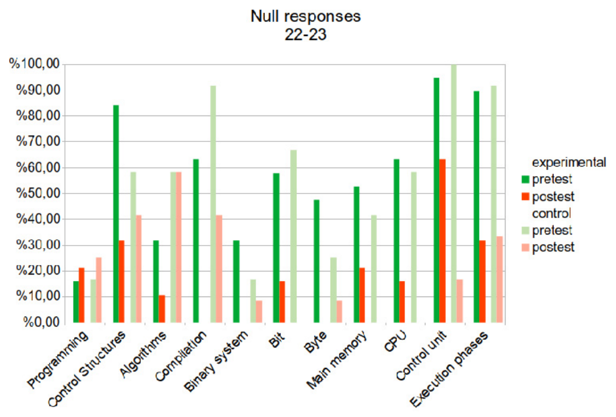


Fig. 9. Null responses in course 22-23.

of null answers and a low rate of correct responses in the first post-test.

E. A Different Approach on the Data Analysis

Additional analyses have been conducted considering the scoring of responses. Given the one-week intervention timeframe, a perfect understanding of concepts isn't expected, but partial improvement is anticipated. Hence, we focused on null answers from the pretest, indicating a lack of recognition of the concept, and completely incorrect responses, suggesting a misconception despite perceived knowledge. We then assessed how many of these responses transitioned to non-null in the post-test.

Figures 8 to 10 illustrate the null responses from years 21-22 to 23-24. Unlike TABLES II and III, which primarily indicate statistically significant differences between pretest and posttest, these graphs provide insight into the percentage of students who either didn't define or poorly defined each concept. It's evident that although the difference may not always be statistically significant, there's improvement across all cases, with some concepts even completely resolving.

Figure 10 depicts the progression from pretest to the first posttest and further to the second posttest. While most concepts show substantial improvement in the initial phase with limited room for enhancement in the second phase, it's noteworthy that concepts like compilation, bit, byte, and

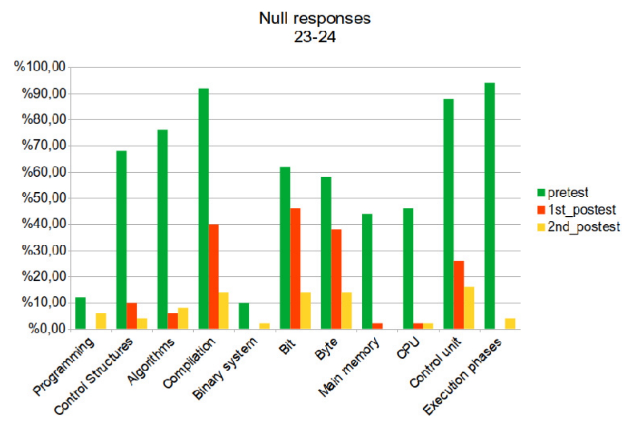


Fig. 10. Null responses in course 23-24.

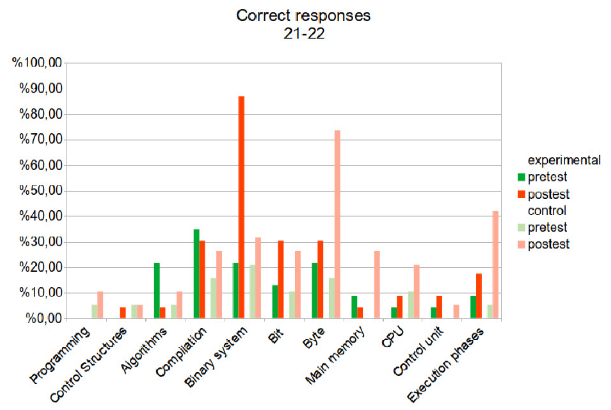


Fig. 11. Correct responses in course 21-22.

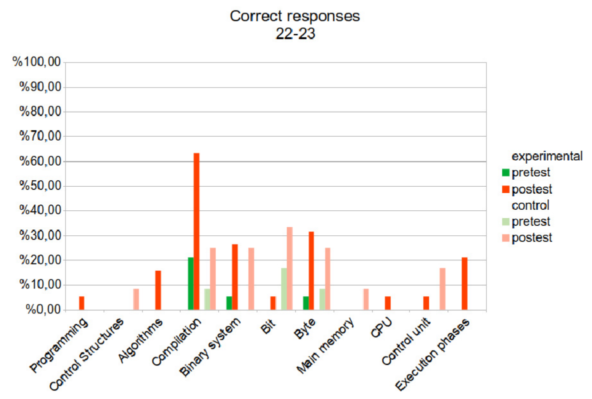


Fig. 12. Correct responses in course 22-23.

control unit, the ones with room for improvement, significantly improve in the second phase.

Furthermore, while it's acknowledged that students aren't expected to provide completely accurate definitions within just one week, the analysis of correct answers has also been conducted (Figures 11 to 13). These figures reveal that certain concepts are understood perfectly by only a few students, although there's an overall improvement in the number of correct answers across all concepts.

In the most recent course, following the second post-test, there's a significant increase in the percentage of correct

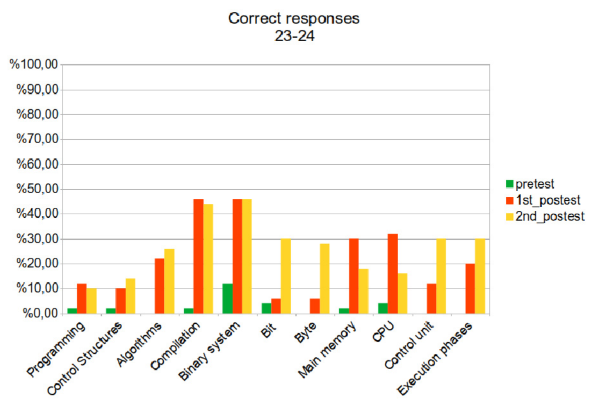


Fig. 13. Correct responses in course 23-24.

TABLE V
MINIMUM, MAXIMUM, AND AVERAGE NULL AND CORRECT ANSWERS FOR YEARS 23-24

		Pretest	Postest1	Postest2
Nulls	Max.	94,00%	46,00%	46,00%
23-24	Min.	0,00%	6,00%	10,00%
	Avg.	2,55%	22,00%	26,00%
Corrects	Max.	12,00%	46,00%	46,00%
23-24	Min.	0,00%	6,00%	10,00%
	Avg.	2,55%	22,00%	26,55%

answers for some concepts, such as “bit” and “byte,” while for others, like “main memory” and “CPU,” there’s a decrease. Although pinpointing the exact reason for these variations is challenging, analysis of the escape rooms suggests that certain terms have been utilized more frequently than others due to their utility in posing challenges. This discrepancy in exposure during escape room activities may explain the differences in students’ ability to define these concepts.

TABLE IV provides details of the minimum, maximum, and average null and correct answers for years 21-22 and 22-23, while TABLE V does the same for years 23-24.

F. Student Satisfaction

In addition to defining concepts, students were asked to rate their satisfaction with the games and methodology on a Likert scale ranging from 1 to 5 in the posttests. A score of 1 indicated total disagreement, while a score of 5 indicated total agreement. TABLES VI to VIII present the responses of students in the experimental group, the control group, and the second posttest of the last course, respectively.

G. Qualitative Analysis

Qualitative analysis was conducted on discussion group transcripts and responses to open-ended questions in the posttests. Below, interspersed with a brief explanation, are some low-inference testimonies extracted from this analysis.

Students in the experimental group demonstrated higher motivation. They found the classes to be more dynamic, enjoyable, and less burdensome.

- “The classes are not as burdensome, and certain games have been effective in introducing certain topics.”

TABLE VI
EXPERIMENTAL GROUP SATISFACTION

Questions	21-22	22-23	23-24
I liked working on the basic contents through games	4,38	4,50	4,80
By working on the basic contents with games I have learned more than with direct explanations.	3,17	3,58	3,92
I found the maze programming game to be a good way to get a first idea about programming.	4,38	-	-
The game of searching for the missing card is suitable for introducing the algorithms.	-	4,17	4,38
The dice race is suitable for introducing control structures.	-	3,67	4,29
The binary card game is suitable for exercising binary numbers.	4,13	-	4,63
The magic game is suitable for introducing binary code.	-	4,08	4,08
The MOON game is suitable for getting an idea of how binary operations work.	4	-	-
The SUN game is suitable for introducing the Von Neumann architecture, execution phases and assembly language.	3,88	4	-
The 1-2-3 game is suitable for introducing the execution phases.	-	-	3,98
Average	3,88	4	4,26

TABLE VII
CONTROL GROUP SATISFACTION

Questions	21-22	22-23
I liked the way the basic concepts have been worked on.	4,38	4,50
I would have liked to work on these concepts through games	3,88	4

TABLE VIII
ESCAPE ROOM SATISFACTION

Questions	23-24
I liked the fact of developing a game to teach the introductory concepts.	3,47
Knowing that the work will be used in high school has helped me to work harder.	3,51

- “The practical classes have engaged our attention, making it easier for us to grasp how things work.”
- “By incorporating games, we’ve been able to participate more actively and assist one another. If someone doesn’t understand something, the others pitch in to help.”
- “Through this methodology, we students develop a better understanding of how different systems operate. Even complex concepts become more comprehensible as we grasp the basic processes.”

However, there are also individuals who feel uncertain about the methodological shift. They express a sense of inadequacy in their learning. This suggests a need for more thorough explanations when introducing games, aiming to reinforce the connection between the game and the content.

- “The games need more depth to provide a clearer understanding of the subject. Each game should include an introduction to the topic.”

- "You don't learn as much as you would in a traditional classroom. It's difficult to grasp concepts through games, and they don't clarify things."
- "Some games are too complex for someone with no prior knowledge of the subject and limited time to comprehend them thoroughly."

Regarding the control group, in both courses, they have demonstrated interest in the new methodology.

- "I would have welcomed the opportunity to explore more innovative methodologies."
- "The use of games to enliven the subject matter would have further facilitated content assimilation, although the current presentation of the subject is equally excellent."
- "I believe that the introductory content has been adequately explained, although I would have preferred to learn them in a more interactive manner, perhaps through gaming."

The perception in the last year remains largely consistent with previous interventions. The student body generally feels more motivated, as evidenced by the following testimonials:

- "It's more interactive than other methods, making it easier to stay focused."
- "It's less burdensome than traditional theoretical classes and encourages more active student participation in learning and understanding concepts."
- "Games facilitate a natural understanding of the content and help maintain attention more effectively."

They also express concerns about the limited time devoted to the material. They feel they don't have enough opportunity to fully grasp the concepts and are easily distracted by the game elements. Some worry they haven't correctly related the concepts to real-world scenarios. For instance:

- "I wish I had more time to jot down notes in my notebook."
- "Having cards or dice on the table can be distracting, leading to off-task behavior."
- "At times, the analogies used in the games are overly abstract. While they make sense logically, it's challenging to envision how these concepts apply practically to computer operations."

VII. Conclusion

The main goal of this study was to explore the effectiveness of using games to teach computer architecture basics quickly. The article outlines the games utilized in various interventions. Analysis of data from two courses indicated that, despite initial student insecurity in front of the new approach, learning outcomes were comparable to traditional methods, with the added benefit of increased motivation. Consequently, in the latest course, the approach was universally implemented, incorporating an exercise where students had to develop an Escape Room-style game to teach the learnt concepts to younger students, with the aim of reinforcing learning. While further data collection is needed for conclusive results, preliminary findings suggest the effectiveness of this addition, particularly in reinforcing weaker concepts. Additionally, the literature review highlighted the ongoing potential for integrating games and game development into computer architecture education, prompting the team to continue exploring new gaming approaches for more advanced topics.

Acknowledgment

The authors extend their gratitude to the students of the Artificial Intelligence Degree Program who participated in this initiative.

References

[1] E. L. Mendiluze, O. A. Gallego, O. A. Uriarte, J. I. M. Aramburu, and J. F. L. Mujika, "Juegos para presentar conceptos básicos de arquitectura de computadores," in *Proc. Actas de las Jornadas sobre la Enseñanza Universitaria de la Informática (JENUI)*. Spain: Asociación de Enseñantes Universitarios de la Informática, 2023, no. 8, pp. 49–56.

[2] G. de Trabajo. (Jun. 2018). *Informe Del Grupo De Trabajo Scie/coddi Sobre La Enseñanza Preuniversitaria De La Informática*. [Online]. Available: https://coddii.org/wp-content/uploads/2021/07/InformeSCIE-CODDI-2018-06.pdf

[3] N. Thomas, F. Carroll, R. Kop, and S. Stocking, "iBook learning experience: The challenge of teaching computer architecture to first year university students," in *Proc. P Worldcomp*, 2012, pp. 16–19. [Online]. Available: http://worldcomp-proceedings.com/proc/p2012/EEE2439.pdf

[4] S. Tobias, J. D. Fletcher, and A. P. Wind, "Game-based learning," in *Handbook of Research on Educational Communications and Technology*. New York, NY, USA: Springer, 2014, pp. 485–503.

[5] A. I. A. Jabbar and P. Felicia, "Gameplay engagement and learning in game-based learning: A systematic review," *Rev. Educ. Res.*, vol. 85, no. 4, pp. 740–779, Dec. 2015.

[6] M. S. El-Nasr and B. K. Smith, "Learning through game modding," *Comput. Entertainment*, vol. 4, no. 1, p. 7, Jan. 2006.

[7] S. Papert, "Children, computers and powerful ideas," *New York, Basic Books*, vol. 10, no. 1990, 1990, Art. no. 1095592.

[8] P. Battistella and C. G. von Wangenheim, "Games for teaching computing in higher education—A systematic review," *IEEE Technol. Eng. Educ.*, vol. 9, no. 1, pp. 8–30, Mar. 2016.

[9] K. Kuk, D. Rancic, P. Spalevic, Z. Trajcevski, and M. Micalovic, "Use game based interactive multimedia modules to learning basic concepts on courses for computing science," *Przeglad Elektrotechniczny*, pp. 150–153, Jan. 2012.

[10] J. Melero, D. Hern'ndez-Leo, and J. Blat, "Considerations for the design of mini-games integrating hints for puzzle solving ICT-related concepts," in *Proc. IEEE 12th Int. Conf. Adv. Learn. Technol.*, vol. 45, Jul. 2012, pp. 154–158.

[11] *Workshop: Simulate Computer*. [Online]. Available: https://sites.google.com/site/childrenandtechnology/workshop-simulatecomputer

[12] G. Kacmarcik. (2007). *How Computers Work*. [Online]. Available: https://www.cse4k12.org/how_computers_work/

[13] P. Higginson. *Little Man Computer*. [Online]. Available: https://www.peterhigginson.co.uk/lmc/?F5=16-Apr-24_08:58:02

[14] 101Computing. *Little Man Computer (LMC) CPU Simulator*. [Online]. Available: https://www.101computing.net/LMC/

[15] J. Gordon. *Binary LMC—Little Man Computer Simulator*. [Online]. Available: http://www.mathsuniverse.com/lmc

[16] B. Maraza-Quispe, O. M. Alejandro-Oviedo, W. Choquehuanca-Quispe, F. Huallpa-Gonzalez, L. M. Quispe-Flores, and S. A. C. Palomino, "Hacia el desarrollo del pensamiento computacional en los estudiantes de educación superior," *Int. J. Emerg. Technol. E-Learn.*, vol. 2, no. 1, pp. 1–10, 2023.

[17] A. Tlili, F. Essalmi, and M. Jemni, "A mobile educational game for teaching computer architecture," in *Proc. IEEE 15th Int. Conf. Adv. Learn. Technol.*, Jul. 2015, pp. 161–163.

[18] A. Tlili, F. Essalmi, M. Jemni, and D. Kinshuk, "An educational game for teaching computer architecture: Evaluation using learning analytics," in *Proc. 5th Int. Conf. Inf. Commun. Technol. Accessibility (ICTA)*, Dec. 2015, pp. 1–6.

[19] K. H. A. Tan, Y. Cai, B. T. Lokesh, Y. Zhu, C. Su, and Q. Cao, "VR serious game for learning the computer organisation and architecture course," in *Proc. 3rd Int. Conf. Educ. Technol. (ICET)*, vol. 10, Sep. 2023, pp. 1–6.

[20] N. C. Cruz, J. L. Redondo, J. D. Álvarez, and P. M. Ortigosa, "Programación de un juego en ensamblador CHIP-8 como actividad complementaria en la asignatura Arquitectura de Computadores," in *Proc. XXIX Jornadas de Paralelismo*, 2018, pp. 183–192.

[21] J. Kawash and R. Collier, "Using video game development to engage undergraduate students of assembly language programming," in *Proc. 14th Annu. ACM SIGITE Conf. Inf. Technol. Educ.*, Oct. 2013, pp. 71–76.

[22] F. J. Gallego-Durán, P. Compañ, R. S. Cuerda, and C. J. V. Arnedo, "El código máquina mola," in *Proc. Actas de las JENUI*, vol. 3, 2018, pp. 149–156. [Online]. Available: https://aenui.org/actas/pdf/JENUI_2018_028.pdf

[23] R. Bourbia, N. Gouasmi, M. Hadjeris, and H. Seridi, "Development of serious game to improve computer assembly skills," *Proc. Social Behav. Sci.*, vol. 141, pp. 96–100, Aug. 2014.

[24] E. Larraza-Mendiluze et al., "Game-console-based projects for learning the computer input/output subsystem," *IEEE Trans. Educ.*, vol. 56, no. 4, pp. 453–458, Nov. 2013.

[25] J. W. Creswell, "Revisiting mixed methods and advancing scientific practices," in *The Oxford Handbook of Multimethod and Mixed Methods Research Inquiry*. Oxford, U.K.: Oxford Univ. Press, 2015, pp. 61–71.

[26] S. J. Taylor and R. Bogdan, *Introducción a Los Métodos Cualitativos De Investigación*, vol. 1. Barcelona, Spain: Paidós, 1987, p. 348.

[27] *Paidós*. [Online]. Available: https://genial.ly

[28] *Merrian-Webster Dictionary*. [Online]. Available: https://www.merriam-webster.com/dictionary/algorithm

[29] *Code.org—Dice Race*. [Online]. Available: https://code.org/curriculum/course3/10/Teacher

[30] *Cse4k12—Binary Magic Trick*. [Online]. Available: https://www.cse4k12.org/binary/magic_trick.html

[31] *Little Robot Computer*. [Online]. Available: http://casne.ncl.ac.uk/events/conference/session/71/

[32] A. Manches and C. O'Malley, "Tangibles for learning: A representational analysis of physical manipulation," *Pers. Ubiquitous Comput.*, vol. 16, no. 4, pp. 405–419, Apr. 2012.

[33] *Moon*. [Online]. Available: https://compus.deusto.es/es/moon-1110011

Edurne Larraza-Mendiluze received the M.S. and Ph.D. degrees in computer science from the University of the Basque Country (UPV/EHU), Donostia-San Sebastián, Spain, in 1999 and 2014, respectively. She is currently an Assistant Professor with UPV/EHU. She is a member of the ADIAN Research Group. Her research interests include computer science education at all educational levels.

Olatz Arbelaitz Gallego received the M.S. and Ph.D. degrees in computer science from the University of the Basque Country (UPV/EHU), Donostia-San Sebastián, Spain, in 1993 and 2002, respectively. She is currently an Associate Professor with UPV/EHU. She is also a member of the Aldapa Research Group (http://www.aldapa.eus/en/). Her research interests include data mining, web mining, and unsupervised learning techniques. She is also interested in teaching innovation and implementation of new methodologies in the context of EHEA.

Olatz Arregi Uriarte received the M.S. and Ph.D. degrees in computer science from the University of the Basque Country (UPV/EHU), Donostia-San Sebastián, Spain, in 1986 and 1991, respectively. She was the Director of the Department of Computer Architecture and Technology from 2019 to 2022. She is currently an Associate Professor with UPV/EHU. She is also a member of the Ixa Research Group (http://www.ixa.eus) and the HiTZ Center (http://www.hitz.eus/). Her research interests include natural language processing. She is also interested in teaching innovation and implementation of new methodologies in the context of EHEA.

Jose Ignacio Martín Aramburu received the M.S. and Ph.D. degrees in computer science from the University of the Basque Country (UPV/EHU), Donostia-San Sebastián, Spain, in 1990 and 1994, respectively. He is currently an Associate Professor with the Computer Architecture and Technology Department, UPV/EHU. His research interests include data mining and pattern recognition. He is also interested in teaching innovation and implementation of new methodologies in the context of EHEA.

Jose Francisco Lukas Mugika received the Diploma degree in teaching, the B.A. degree in philosophy and educational sciences (pedagogy section), and the Ph.D. degree in pedagogy. He was the Deputy Director of the Institute of Educational Sciences, UPV/EHU, from 1997 to 200, and the Director of the Department of Research and Diagnostic Methods in Education, UPV/EHU, from 2003 to 2009. He is currently a Senior Lecturer with the Department of Research and Diagnostic Methods in Education, UPV/EHU. He is an expert in educational measurement and evaluation. He was a member of the Scientific Committee of the Basque Institute for Non-University Educational Evaluation and Research (ISEI-IVEI) from 2002 to 2014.