The Effect of Learning Objects on a C++ Programming Lesson

Carlos Argelio Arévalo Mercado
Autonomous University of Aguascalientes, México
carevalo@correo.uaa.mx

Estela Lizbeth Muñoz Andrade
Autonomous University of Aguascalientes, México
elmunoz@correo.uaa.mx

Juan Manuel Gómez Reynoso
Autonomous University of Aguascalientes, México
jmgr@correo.uaa.mx

ABSTRACT

The subject of teaching computer programming has been widely studied for the last 25 years. Different approaches and disciplines have given important knowledge about the problem, mainly cognitive science and the use of software tools and technology to aid teaching. In recent years, Learning Object (LO) technology has received world wide attention within the distance education community. However, literature research shows little experimental and empirical evidence about the effectiveness of LOs in academic performance. In this exploratory study, the effect of using LOs in a computer programming lesson is reported. Descriptive statistics and T test results show that there is no significant difference in student's performance, under the conditions of the study.

INTRODUCTION

Learning to program is difficult for many students. Dropout and failure rates in introductory programming at undergraduate courses are evidence to the fact that learning to program is a difficult task (Wiedenbeck, 2004). It can be said that algorithmic thinking is complex: more advanced concepts are layered on top of others, learned previously (Jenkins, 2002; Machanick, 2005)

Even though the subject of learning to program has been widely studied for the last 25 years, within several disciplines and from different approaches (psychological/cognitive, sociological, and software tools, among others) there is not yet a definite solution and understanding to the problem.

In the line of software tools to aid learning, learning objects (LOs) are receiving a lot of world wide attention. LOs are generally understood as digital learning resources that can be shared and accessed via Internet and used in multiple contexts (Wiley, 2000). The basic premise of LOs is to offer scalable and individually adaptive instruction, which can even be generated on the fly according to the learner needs by intelligent semantic technologies (Gibbons, 2000). And so, the adoption of LO technology has seen a significant increment within the distance learning community. Important efforts are made to establish standards (ISO, 2003), quality measures (Archambault, 2003), and efficient repositories. Substantial amounts of human and financial resources are being channeled to LOs related projects (CISCO, 2000).

However, the LO approach is not without criticism. For example, Jaakkola (2004, p.4), states: *When considering the effect of learning objects on students' learning performance, it is important to understand that it is impossible, and irrelevant, to separate the learner and the learning content to be learned from the context in which learning occurs.*

More so, research literature focused on the pedagogical effectiveness of LOs is limited (Moisey, 2003). There is little empirical evidence of the effect of LOs on academic achievement of students (Jaakkola, 2004; Kujansuu, 2006).

The purpose of this exploratory study is to statistically measure the effects of LOs in a first year C++ programming lesson, and find if students exposed to this technology (in a limited time frame) show better academic performance.

METHODOLOGY

In this study, we will describe an exploratory study that consisted of comparing the performance of two first year C++ undergraduate programming groups of students, under two different conditions:

1) Using traditional, teacher-led instruction, and
2) Using learning objects.

The study was conducted in Autonomous University of Aguascalientes (UAA), México.

The content was focused on the subject of file handling in C++. This subject was selected because it coincided with both the lecture plan and the thematic progress of the course. Thirty five first year computer science students participated in the study. Two groups were created: one using traditional methods (TRADG), and the other using LOs (LOG).

Before the formation of the groups, students with previous programming experience (Byrne, 2001; Holden, 2005; Fauxx, 2006) were identified and later randomly distributed within the two groups.

The study was conducted in the second half of the semester, so it was assumed that the participants already knew the basic programming concepts and structures (sequence, decisions, loops, variables) of the C++ language.

STRUCTURE OF THE TEACHING SESSIONS

Each group was given a two hour session with the following structure:

a. A lecture about the subject of file handling in C++ (reading and writing)
b. Free, individual study/use of corresponding learning materials
c. Solution of a small test

To control the teaching style variable, the same teacher conducted both sessions.

Both sessions were held in the same laboratory, using a projector.

DESCRIPTION OF THE TEST

To minimize a possible waste of time, the students were asked to answer a written exam. This was also designed to avoid the effect of stress and anxiety due to compilation errors.

The test consisted of two exercises: one to demonstrate knowledge of “write” C++ related instructions, and the other related to the “read” counterpart instructions. The established scoring criteria were the following:

- Include the correct libraries (iostream.h ; fstream.h)
- Include the correct instructions, to open files (ofstream write/read). This had to be done in the correct context of the program.
- A logically correct use of a “for” loop.
- Include the correct writing and reading instructions (write; read.getline)
- Include the correct instructions to close files (write.close; read.close) in the correct context of the program

Each one of these requirements was assigned one point. The complete test summed up a total maximum score of 10 points.

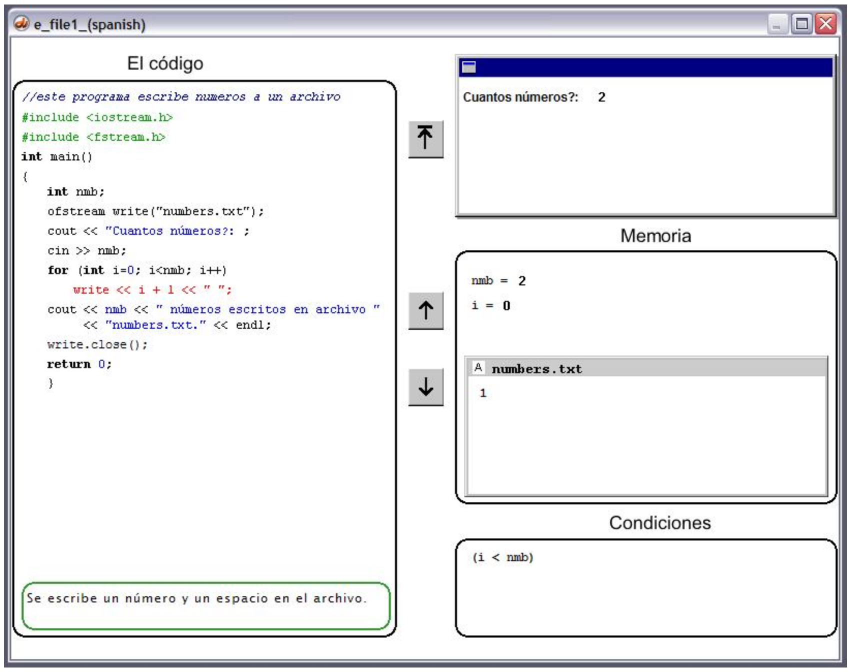
DESCRIPTION OF THE LEARNING OBJECTS

A more detailed description of the LOs used in the study is required. These LOs are part of the CodeWitz[1] international repository project, which is formed by several higher educations institutions. Its main objective is to help in the instruction of the basics of computer programming, specifically C++ and Java programming languages, through the use of high quality reusable learning objects.

For our study, the LOs were translated for a Spanish speaking audience.

The LOs interface consisted primarily of clicking three navigation buttons (see Figure 1), to go forward and backwards through the execution of an example program (in this case, writing and reading files in C++).

Figure 1: Adapted LO from the CodeWitz Project, for learning C++ file writing sentences.



```
//este programa escribe numeros a un archivo
#include <iostream.h>
#include <fstream.h>
int main()
{
   int nmb;
   ofstream write("numbers.txt");
   cout << "Cuantos números?: ;
   cin >> nmb;
   for (int i=0; i<nmb; i++)
      write << i + 1 << " ";
   cout << nmb << " números escritos en archivo "
        << "numbers.txt." << endl;
   write.close();
   return 0;
   }
```

```
Se escribe un número y un espacio en el archivo.
```

```
Cuantos números?:   2
```

```
nmb = 2
i = 0
```

```
numbers.txt
1
```

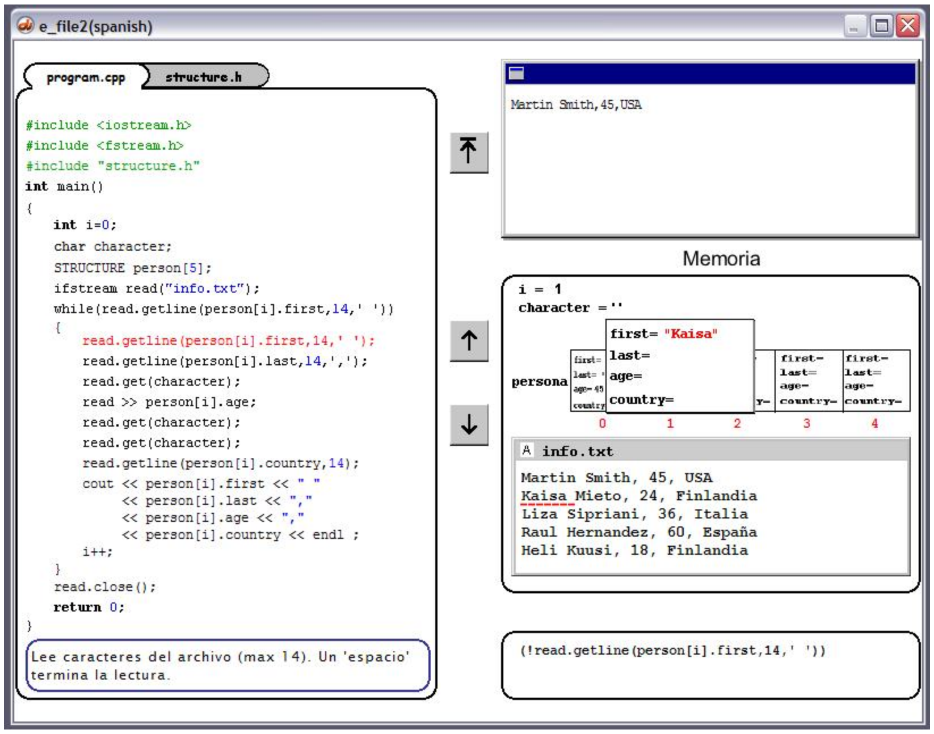
```
(i < nmb)
```

[1] http://www.codewitz.net/

The LOs have five different sections (see Figures 1 and 2):

1) "The code", which shows the example program.
2) An explanatory section of each one of the lines of the program (the content of these sections changes when the user navigates the program) located at the lower left of the LO.
3) An "Output" section, to simulate how the computer screen could give input and output in the monitor.
4) The "memory" section, to show the state and contents of variables and data structures. This section is particularly important, given that previous cognitive studies (Norman, 1983; Fixx, 1993; Ramalingam, 2004; Ma, 2007) have identified "proper mental models" as a relevant factor. The visualization style tries to follow a "debug like" behavior (Kujansuu, 2006).
5) Finally, a "Conditions" section to show the state of the logical conditions guiding the behavior of the loops and decision structures of the example program.

Figure 2: Adapted LO from the CodeWitz Project, for learning C++ file reading sentences.



```
#include <iostream.h>
#include <fstream.h>
#include "structure.h"
int main()
{
   int i=0;
   char character;
   STRUCTURE person[5];
   ifstream read("info.txt");
   while(read.getline(person[i].first,14,' '))
   {
      read.getline(person[i].first,14,' ');
      read.getline(person[i].last,14,',');
      read.get(character);
      read >> person[i].age;
      read.get(character);
      read.get(character);
      read.getline(person[i].country,14);
      cout << person[i].first << " "
           << person[i].last << ","
           << person[i].age << ","
           << person[i].country << endl ;
      i++;
   }
   read.close();
   return 0;
}
```

Lee caracteres del archivo (max 14). Un 'espacio' termina la lectura.

```
Martin Smith,45,USA
```

```
i = 1
character =''
first= "Kaisa"
last=
age=
country=
```

```
info.txt
Martin Smith, 45, USA
Kaisa Mieto, 24, Finlandia
Liza Sipriani, 36, Italia
Raul Hernandez, 60, España
Heli Kuusi, 18, Finlandia
```

```
(!read.getline(person[i].first,14,' '))
```

As stated before, this particular design tries to reinforce the mental model of the student using visualization and showing what happens "inside" and "outside" of the program. This is also in accordance to the "notional machine" concept stated by DuBoulay (1989).

CONDITIONS FOR THE TRADITIONAL METHOD GROUP (TRADG)

For the TRADG group (n = 18), the teacher gave one lecture of the subject of writing and reading files in C++ using a PDF file, containing the theory and two examples. The approximate time was 15 minutes for theory and 10 minutes for explanation of the examples. After this, a 20 minute period was given for the students to freely study and experiment with the PDF examples. Past this time, the teacher asked the students to complete the written test (described previously). Individual completion times were recorded. A one hour limit was given to the participants.

CONDITIONS FOR THE LEARNING OBJECTS GROUP (LOG)

For the LOG group (n = 17), the lecture was made using the same PDF file given to the TRADG group, *minus the examples*, which were substituted with the LOs described above.

The teacher and the LOG group students used the LOs for approximately 15 minutes for the explanation of the examples. After that, the students used the objects freely for another 20 minutes. For the last part of the session the LOG group participants were asked to complete the same written test. A one hour limit was also given to the participants of this group.

OBSERVATION OF THE PARTICIPANTS

The teacher conducting the sessions was asked to take notes about the behavior of the participants, to provide complimentary qualitative data to the analysis. She recorded the following:

- The participants of the LOG group were found to be nervous and stressed about using the LOs. Comments such as "how do we use this?" were recorded.
- In general, the LOG group took more time to complete the test and 53% did not complete the second exercise of the test. In comparison, 33% of the TRADG group did not complete the second exercise of the test.
- The participants of the TRADG group were less anxious and nervous about the test.

RESULTS

Descriptive statistics obtained for both groups are shown in Table 1.

Table 1: Descriptive statistics of the study.

Group Statistics

	grupo	N	Mean	Std. Deviation	Std. Error Mean
Exercise A	1	18	3.1111	2.11824	.49927
	2	17	3.6176	2.10304	.51006
Exercise B	1	18	3.0000	2.26222	.53321
	2	17	2.2941	2.51283	.60945
Grade obtained in the exam	1	18	6.1111	4.23068	.99718
	2	17	5.9118	4.11262	.99746

An initial analysis of the data shows that the mean of the TRADG group was slightly superior (6.11) to that of the LOG group (5.91). This results are somehow discouraging and surprising. It is evident by the data obtained that the overall performance of the LOG group was lower than the TRADG.

Only in the first exercise (Exercise A) the LOG group (mean = 3.61) had slightly better performance than the TRADG group (mean = 3.11). However, for Exercise B, the performance of the LOG group was clearly inferior (mean = 2.29) than that of the TRADG group (3.00).

To corroborate what descriptive statistics seem to indicate, a standard t test was applied to both groups, with the following results:

Table 2: T test results for TRADG and LOG groups.

Independent Samples Test

		Levene's Test for Equality of Variances		t-test for Equality of Means						
									95% Confidence Interval of the Difference	
		F	Sig.	t	df	Sig. (2-tailed)	Mean Difference	Std. Error Difference	Lower	Upper
Exercise A	Equal variances assumed	.183	.671	-.710	33	.483	-.50654	.71390	-1.95898	.94591
	Equal variances not assumed			-.710	32.912	.483	-.50654	.71375	-1.95882	.94575
Exercise B	Equal variances assumed	3.450	.072	.874	33	.388	.70588	.80729	-.93656	2.34832
	Equal variances not assumed			.872	32.144	.390	.70588	.80978	-.94330	2.35506
Grade obtained in the exam	Equal variances assumed	.057	.812	.141	33	.889	.19935	1.41160	-2.67257	3.07126
	Equal variances not assumed			.141	32.969	.888	.19935	1.41042	-2.67028	3.06897

As expected (see Table 2), the t test results are not significant in any case (Exercise A = .483, Exercise B = .388, Final Grade = .889). The null hypothesis for equality of means has to be accepted, meaning that there is no significant difference in the performance of both groups.

Given the results, it was considered that no further statistical analysis was needed.

DISCUSSION AND FUTURE STUDIES

Clearly, the results obtained were not the ones expected; however, they provide hints about what factors may have influenced this phenomenon. The qualitative report provided by the teacher that conducted the study, lead us to think that the participants of the LOG group likely felt pressured and anxious (Brosnan, 1998). Another factor could have been the limited time of exposure to the learning objects, even though the user interface seemed to be very easy to use. It is possible that the participants of the LOG group could have perceived that they had two different assignments: learn to use the LOs, and complete the test.

As mentioned previously, there are few studies that show empirical evidence about the effectiveness of LOs regarding student achievement (Moisey, 2003). The results of our study seem to concur with those found by Jaakkola (Jaakkola, 2004). In his study, three experimental studies using LOs were reported, and two of them did not show any significant differences in the performance of the participants (high school students). However, a third experiment did show significant differences due to the combination of simulation and hands-on experimentation (the topic covered by this experiment was learning of DC circuit design concepts).

In summary, the results of our exploratory study seem to indicate two main ideas:

a) A computer programming instructor cannot assume that the sole use of LOs, in a single lecture, will make a significant difference in the performance of students. There are other important contextual factors that may yet have to be identified to improve academic achievement. On the other hand, a more prolonged exposure to LOs has to be explored and measured under carefully controlled conditions.

b) In accordance to other experimental studies, the combined use of simulation and hands-on experimentation seems to be a promising line of research and development in the field of learning computer programming.

It can be suggested that the field of LOs can benefit with more experimental studies to give instructors a clearer vision of circumstances under which LOs can be more effective to academic performance.

Finally, in a purely speculative spirit, a kind of learning artifact that interactively gives both “guided experimentation” and visual feedback to the student, seems to be a desirable goal. However, the time and cost associated with this kind of development could be a significant barrier.

REFERENCES

Archambault, J. V. J. C. N. K. B. A. (2003). Learning Object Evaluation: computer-mediated collaboration and inter-rater reliability. *International Journal of Computers and Applications* 25(3).

Brosnan, M. J. (1998). The impact of computer anxiety and self-efficacy upon performance. *Journal of Computer Assisted Learning*. 14: 223-234.

Byrne, P., Lyons G. (2001). The effect of student attributes on success in programming. ACM SIGCSE Bulletin. 33(3): 49 - 52 .

CISCO (2000). Reusable learning object strategy. Definition, creation process, and guidelines for building. Version 3.1.

Du Boulay, J. B. H. (1989). Some difficulties of learning to program. Lawrence Erlbaum Associates, Hillsdale.

Fauxx, R. (2006). Impact of pre-programing course curriculum on learning the first programming course. *IEEE Transactions on education*. 49(1): 11-15.

Fixx, V., Wiedenbeck, S., Scholtz, J. (1993). Mental representations of programs by novices and experts. Conference on Human Factors in Computing Systems Amsterdam, The Netherlands ACM Press New York, NY, USA .

Gibbons, A. S., Nelson, J. & Richards, R. (2000). The Nature and Origin of Instructional Objects. In The Instructional Use of Learning Objects. Bloomington: Association for Educational Communications and Technology.

Holden, E., Weeden, E. (2005). Prior Experience and New IT Students. *Issues in Informing Science and Information Technology*. 2: 189.

ISO (2003). Business Plan for JTC1/SC36 (Standards for Information Technology for Learning, Education, and Training).

Jaakkola, T. N., S. (2004). Final Report on CELEBRATE Experimental studies. Learning Objects - A lot of smoke, but there is fire? Turku, University of Turku, Finland. Educational Technology Unit.

Jenkins, T. (2002). On the difficulty of learning to program. 3rd annual LTSN-ICS Conference, Loughborough University, LTSN Centre of information and computer sciences.

Kujansuu, E. (2006). Using program visualization learning objects with non-major students with different study background. Proceedings of Methods, Materials and Tools for Programming Education conference, Tampere, Finland.

Ma, L. F. J. R., M; Wood, M. (2007). Investigating the viability of mental models held by novice programmers. ACM SIGCSE Bulletin. 39(1): 499-503.

Machanick, P. (2005). Peer Assessment for action learning of data structures and algorithms. Australasian Computing Education Conference, New Castle, Australia.

Moisey, S. M., A. (2003). Fulfilling the promise of learning objects. In Handbook of Distance Education, Lawrence Erlbaum. 1st ed.: 323.

Norman, D. A. (1983). Some observations on mental models. Erlbaum, Hillsdale, NJ.

Ramalingam, V. L., D.; Wiedenbeck, S. (2004). Self-Efficacy and mental models in learning to program. Proceedings of the 9th annual SIGCSE conference on Innovation and technology in computer science education. Leeds, United Kingdom ACM Press New York, NY, USA.

Wiedenbeck, S. L., D.; Kain, V.N.R (2004). Factors affecting course outcomes in introductory programming. 16th Workshop of the psychology of programming interest group, Institute of Technology, Carlow, Ireland.

Wiley, D. (2000). The Instructional Use of Learning Objects. Bloomington, IN, Association for Educational Communications and Technology.