Programming web-course analysis: how to introduce computer programming?

Romenig da Silva Ribeiro
Instituto de Matemática e Estatística – DCC
University of São Paulo
São Paulo, Brazil
romenig@ime.usp.br

Tulio Vitor Machado Faria
Escola de Artes Ciências e Humanidades – EACH
University of São Paulo
São Paulo, Brazil
tuliofaria@usp.br

Leônidas de Oliveira Brandão
Instituto de Matemática e Estatística – DCC
University of São Paulo
São Paulo, Brazil
leo@ime.usp.br

Anarosa Alves Franco Brandão
Escola Politécnica – PCS
University of São Paulo
São Paulo, Brazil
anarosa.brandao@poli.usp.br

Abstract— **Nowadays, computer programming and logical thinking skills have been proposed as a fundamental knowledge, even to young learners. On one hand, in undergraduate STEM (Science, Technology, Engineering and Math) courses, the first contact of students with the logic of programming usually results in high failure rates. The literature and experiments conducted by the authors point out that this occurs regardless the adopted programming language. On the other hand, the literature presents some positive results when the paradigm used to introduce the subject is Visual Programming (VP), where the learners use icons to build their programs. This approach is successful even with young learners. In this context, a relevant question is whether, and how, the Visual Programming can help learners to understand a traditional textual programming language. The proposal of this work is to study differences between visual and traditional programming by analyzing the mental workload of using both paradigms during the introduction of algorithms and basic concepts of programming in the context of an online course of introductory programming. In order to perform such analysis, we adopted the NASA TLX protocol.**

Keywords— visual programming; textual programming; MOOC; iVProg; iAssign; Moodle; VPL; NASA TLX; mental workload; web-learning; elearning; introduction to programming.

I. Introduction

Nowadays, logical reasoning and computer programming are proposed as fundamental abilities for students and their introduction in early stages of education has been adopted in an increasing rate. This adoption is part of the movement towards increasing the number of professionals that have degrees on **STEM** (*Science, Technology, Engineering, and Mathematics*) courses, since a bottled up demand of such professionals is a reality in several regions world wide, including Brazil.

Moreover, introducing logical reasoning and computer programming earlier may address some challenges that are currently faced by teachers and students of STEM courses that are related to teaching and learning introductory programming. Part of the problem is related to the introduction of a formal language (programming language), which allows students to solve problems computationally. This involves the use of some programming environment to describe the problem solution in the programming language (create the program), debugging, compiling and running the program. It seems simple but sometimes debugging activities may be so hard that compromise the students' performance [1].

In order to overcome this problem, there are some proposals of using visual systems to support the learning of introductory programming [1][2]. Such systems use visual resources like flowcharts or draggable codeblocks to guide students on building programs without any concern related to the programming language syntax, decreasing the effort of the debug activity. They also offer the possibility of compiling and running programs within them, avoiding the use of programming environments during introductory programming early stages. For instance, initiatives of using systems like Scratch [3] and Alice [4] have shown that the direct manipulation of the code blocks can reduce difficulties concerning the syntax of traditional programming languages and motivate students in the learning process [5]. This indicates that students may focus their attention to the problem solution and, consequently, increasing their logical reasoning skills. This approach of programming is called Visual Programming (VP). A definition for VP is given in [6] as "any system that allows the user to specify a program in a two (or more) dimensional fashion". In the same paragraph, Bentrad and Meslati also explain that "conventional textual languages are not considered two dimensional since the compilers or interpreters process them as long, one-dimensional streams".

In this scenario, we would like to investigate differences between the use of textual and VP environments for introducing computer programming, in order to analyze whether and how the adoption of a VP approach may help students to understand and use a programming language in its textual representation. Therefore, we initiate such analysis with the adoption of the NASA Task Load Index (**NASA TLX**) [7] for evaluating the mental workload of using both programming paradigms in the context of an online computer programming course.

An experiment was conducted in order to perform the aforementioned analysis. It consisted of the creation of two similar web-courses of Introductory Programming delivered through the **Moodle** (*Modular Object Oriented Dynamic Learning Environment*) [8]. Both courses used the same instructional content but different programming paradigms: one adopted VP and the other adopted textual programming. The course using VP used the new version of our *interactive Visual Programming* system (**iVProg**) integrated to Moodle by the **iAssign** package [9]. The textual programming course used the **C language** that was integrated to Moodle by the plugin Virtual Programming Lab (**VPL**) [10]. All these are free software, and iVProg [11] and iAssign [9] are developed by our research group LInE.

In both courses all programming activities used automatic evaluation, provided by iVProg+iAssign and by C+VPL. Nevertheless VPL allows the use of other compiled languages, we decided to adopt C by two reasons: it is one of the most used language in engineering schools and do not demand explanation of more sophisticated concepts, such as Oriented Objects.

Section II presents the two programming systems and the NASA TLX protocol, section III contains the experiment description, in section IV a discussion and the main findings are presented.

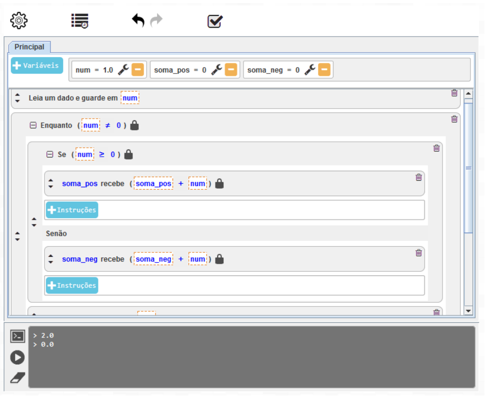
II. Background: iVProg, VPL and NASA TLX

Before describing the experiment, we briefly present the systems that were used in it and the protocol adopted for analyzing the mental workload.

A. iVProg

iVProg, was firstly deployed in 2009. It was developed as an Interactive Learning Module (iLM) based on Alice. In this version, the mouse was the interaction instrument for dragging-and-dropping some code components to build algorithms. In 2012, our research group started a Software Product Line (SPL) for iLM [12][13] and released the first version of a framework to iLM. The new version of iVProg was developed using this framework. The system still uses drag-and-drop for ordering the code components, but interaction was improved based on usability issues and currently, drag-and-drop is combined with the pointing and click approach. Even being designed from the Alice software, both the first and the second version of iVProg were created and adapted to teach procedural programming, while Alice is used for teaching object-oriented programming.

Automated assessment of iVProg is based on test cases. Basically, the system compares the expected outputs provided by the teacher with those generated from the student's algorithm. Moreover, iVProg can execute code with a single click on the play button. The default output is done at a built-in console simulator.



```
num = 1.0    soma_pos = 0    soma_neg = 0

Leia um dado e guarde em num
Enquanto ( num ≠ 0 )
    Se ( num ≥ 0 )
        soma_pos recebe ( soma_pos + num )
    Senão
        soma_neg recebe ( soma_neg + num )
```

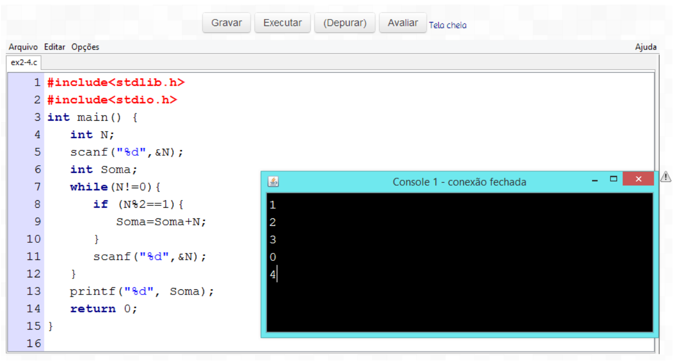
```
> 2.0
> 0.0
```

Fig. 1. iVProg screenshot on Moodle.

As an iLM, iVProg can be used as an applet integrated with Moodle through iAssign. Figure 1 shows a screenshot of the system running on Moodle.

B. VPL

The VPL system is a Moodle module developed at Universidad de las Palmas Gran Canaria, Spain. It allows algorithms constructing under a range of textual programming languages inside an applet. The applet contains a code editor with syntax highlighting. Furthermore, the student can test the algorithm, since the code is compiled and executed (at the server side) with a single button click. The VPL also has an automated assessment based on test cases constructed by the teacher. Due to security issues, VPL needs a jail system installed on a virtual machine different from the one of the Moodle server, meaning that two servers are needed to execute it properly: one hosting Moodle and another running the VPL jail system. Figure 2 shows a VPL screenshot running on Moodle.



```c
#include<stdlib.h>
#include<stdio.h>
int main() {
   int N;
   scanf("%d",&N);
   int Soma;
   while(N!=0){
      if (N%2==1){
         Soma=Soma+N;
      }
      scanf("%d",&N);
   }
   printf("%d", Soma);
   return 0;
}
```

```
1
2
3
0
4
```

Fig. 2. VPL screenshot on Moodle

C. NASA TLX

The NASA TLX protocol was initiated in 1980 years and has been used by the community Human-Computer Interaction to evaluate software. It is an assessment tool that rates perceived workload, been used to measure how the user faces a task or a system. The literature points that it is used in the development of complex computer interfaces to aviation industries and power plants. However, we can also find its usage in educational software, link in Ph.D. thesis of Santos [14], where the author studied the influence of the insertion of new computational tools in distance learning courses using two groups, one of them as control group. The author measured the cognitive workload considering during some activities in both groups. This measures can be important in educational process [15].

In our case, we used the protocol to measure the workload incurred to the students during the execution of course activities, intending to compare the influence of the programming paradigm. The workload is defined in [7] as a hypothetical construct that represents the cost of someone finishing a task and reaching a certain level of performance. Thus, the workload is not defined only by the tasks' demand themself, but they also reflect multiple attributes that may have different relevance for different individuals. Therefore, the workload is an implicit combination of several factors. The protocol divides the workload into six components (named as scales) that are listed and described in the sequence:

1. Mental Demand (MD): How much mental and perceptual activity was required (e.g., thinking, deciding, calculating, remembering, looking, searching, etc.)?

2. Physical Demand (PD): How much physical activity was required (e.g., clicking, typing, pushing, pulling, turning, controlling, activating, etc.)?

3. Temporal Demand (TD): How much time pressure did you feel due to the rate or pace at which the tasks or task elements occurred?

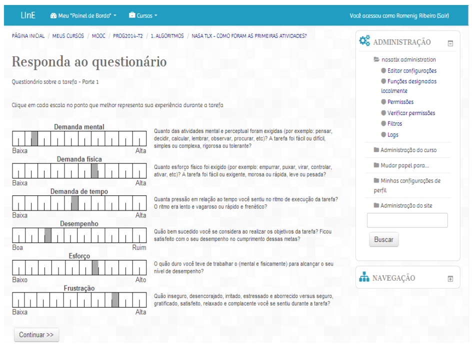
4. Own Performance (OP): How successful do you think you were in accomplishing the goals of the task set by the experimenter (or yourself)?

5. Effort (EF): How hard did you have to work (mentally and physically) to accomplish your level of performance?

6. Frustration (FR): How insecure, discouraged, irritated, stressed and annoyed versus secure, gratified, content, relaxed and complacent did you feel during the task?

The protocol consists of filling out a questionnaire with the above six scales varying from 0 to 100. After filling scales, the students need to choose, among fifteen screens, one of two components that strongly appeared while performing the task (exercises). To apply the protocol during the course we adapted the HTML available in http://keithv.com/software/nasatlx/ and created a Moodle module that displays both: the six scales of the questionnaire and the fifteen screens for a pairwise choice between the components of the workload. Filling scales occurs by simply selecting a cell within a range. Choices between pairs of components are made from button clicks.

Figure 3 shows a screenshot of the scale filling in Moodle. Figure 4 shows one of the fifteen screens that may occur during the choice between pairs of components of the workload.



Responda ao questionário

Questionário sobre a tarefa - Parte 1

Clique em cada escala no ponto que melhor representa sua experiência durante a tarefa

Demanda mental (Baixa – Alta): Quanto das atividades mental e perceptual foram exigidas (por exemplo: pensar, decidir, calcular, lembrar, observar, procurar, etc)? A tarefa foi fácil ou difícil, simples ou complexa, rigorosa ou tolerante?

Demanda física (Baixa – Alta): Quanto esforço físico foi exigido (por exemplo: empurrar, puxar, virar, controlar, ativar, etc)? A tarefa foi fácil ou exigente, morosa ou rápida, leve ou pesada?

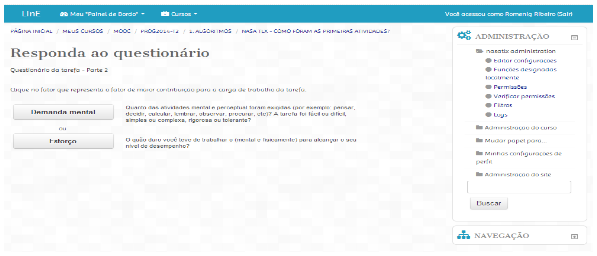
Demanda de tempo (Baixa – Alta): Quanta pressão em relação ao tempo você sentiu no ritmo de execução da tarefa? O ritmo era lento e vagaroso ou rápido e frenético?

Desempenho (Boa – Ruim): Quão bem sucedido você se considera ao realizar os objetivos da tarefa? Ficou satisfeito com o seu desempenho no cumprimento dessas metas?

Esforço (Baixo – Alto): O quão duro você teve de trabalhar o (mental e fisicamente) para alcançar o seu nível de desempenho?

Frustração (Baixa – Alta): Quão inseguro, desencorajado, irritado, estressado e aborrecido versus seguro, gratificado, satisfeito, relaxado e complacente você se sentiu durante a tarefa?

Fig. 3. Screenshot of NASA TLX scales on Moodle.


LInE
Meu "Painel de Bordo"
Cursos
Você acessou como Romenig Ribeiro (Sair)
PÁGINA INICIAL / MEUS CURSOS / MOOC / PROG2014-T2 / 1. ALGORITMOS / NASA TLX - COMO FORAM AS PRIMEIRAS ATIVIDADES?
ADMINISTRAÇÃO
nasatlx administration
Editar configurações
Funções designadas localmente
Permissões
Verificar permissões
Filtros
Logs
Administração do curso
Mudar papel para...
Minhas configurações de perfil
Administração do site
Buscar
NAVEGAÇÃO

Responda ao questionário

Questionário da tarefa - Parte 2

Clique no fator que representa o fator de maior contribuição para a carga de trabalho da tarefa.

Demanda mental: Quanto das atividades mental e perceptual foram exigidas (por exemplo: pensar, decidir, calcular, lembrar, observar, procurar, etc)? A tarefa foi fácil ou difícil, simples ou complexa, rigorosa ou tolerante?

ou

Esforço: O quão duro você teve de trabalhar o (mental e fisicamente) para alcançar o seu nível de desempenho?

Fig. 4. Screenshot of NASA TLX pairwise choice on Moodle.

After obtaining data from the protocol, it is possible to analyze what were the components that had the greatest influence during the activities. Following the protocol, we also can get an overall value of the workload during the activities and identify which were the most weighted factors.

In section III – B, we explain how this analysis was made. In the next section, we will make a description of the experiment.

III. Experiment Description

A. Setting the course

The course of Introductory Programming was created as short course, to introduce the first concepts of programming, from variable concepts to looping constructs.

The course registration was completely opened to the public, not requiring any document or verification, nor restricted to a specific educational institution. However, its propaganda was performed for a short period of time (4 weeks), and mainly restricted to the *University of São Paulo* (**USP**). Some member of our research group made propaganda of it in two more institutions, resulting in 54% of the students enrolled were from USP.

The course divulgation was conducted through 3 different channels, an oral disclosure (a presentation of no more than 5 min) to 15 classes of freshman at the USP, with the distribution of printed flyers in USP and the use of social networking of two research group team.

The registration to the course was open, through the web, under the confirmation of a valid email.

The course had 144 volunteers enrolled, from several educational institutions. However, most of the volunteers were from the engineering school of the University of São Paulo.

The reason must be that this school concentrated the majority of the freshmen's classes with oral disclosure, 11 classes.

The enrolled students were divided into two groups, G1 and G2. The group G1 worked with Visual Programming, and G2 with textual programming.

Since our purpose is to evaluate the mental workload of using the visual programming model versus the textual programming model, we created equivalent environments within Moodle for both groups. They were not randomly assigned. Instead, we tried to balance them by adopting the following criteria: a) we balanced the number of students per institution in each group, b) for each institution, we balanced the number of students who had experience with programming in each group c) for each institution, we also balanced the number of students who had no experience with programming in each group. Following this protocol, we could check the rate of permanence and participation for groups of students with different profiles.

Instructional content was designed independently of the programming paradigm and it consisted of four modules, each one containing a block of activities to be done followed by discursive activities and an evaluation of the mental workload to perform them using the NASA TLX protocol.

For some explanation related to the course, a flowchart model was introduced for both groups. Furthermore, to avoid Java installation problems and some settings that would prevent the applet execution, tutorial videos were prepared to explain how to prepare supporting tools for conducting the course. Students received the credentials to access the system and had access to tutorials for a period of one week before the beginning of the courses' activities.

Theoretical and practical content were presented in four modules and involved concepts of algorithms, variables an associated types, data input and output, arithmetic and boolean expressions, selection and looping constructs. More specifically, module 1 was called "Algorithms" and it is composed of the definition of algorithms and basic concepts of programming (variables and their types, data input and output, boolean expressions and arithmetic expressions). Module 2 was called "Selection" and it is composed of comments about the previous module, definition of selection with examples. Module 3 was called "Looping Constructs" and it is composed of comments about the previous module, definition of the looping constructs *while*, *for* and *repeat*. Module 4 was called "Closing" and it is composed of complex activities involving the content of the previous modules and discursive activities related to the course as a whole and a final NASA-TLX activity.

The course was delivered through Moodle in two versions: one using iVProg as the programming environment and the other using VPL with C. The first was prepared for G1 and the second for G2. Having established the course content, a methodology for conducting the experiment was defined and it is described in the next section.

B. Methodology

After setting the course, we generated data on the workload required by obtaining the answers from NASA TLX forms of each list of exercises. These data were separated and prepared for analysis. The protocol scale provides the students' perception about the incurred workload during the activities performance. The pairwise choice between the components (mental demand, physical demand, temporal demand, etc) provides a weight for each one of them. It is possible to obtain an overall workload, which is calculated as the sum of the products of each scale by its respective weight. The higher is the overall value the higher is the workload during a task performance. However, we decided to analyze the scales separately.

However, it is important to note which are the factors that most influenced the overall value. After all, the variable with the greatest impact in the study was the programming environments used by G1 and G2.

The reason we used a nonparametric test to analise the data is the low number of respondents, besides the apparent non-symmetrical distribution of data. Thus, the analysis as a whole will suffer less influence of outliers and the number of respondents. Considering the distribution of values to the same scale in both groups, we used the nonparametric method Wilcoxon-Mann-Whitney (WMW). Briefly, the test consists of defining ranks based on the samples values. The higher the value of the collected data, the higher its rank. Thus, is possible to find which group has the highest ranks for a given scale with a certain level of significance (we used $\alpha=0.05$), inferring about the population from which that sample was obtained. The analysis of ranks allows the identification of the sample distribution, if it is balanced or uneven. Thus, consider that H_0 : $distribution_{G1} = distribution_{G2}$ e H_1: $distribution_{G1} < distribution_{G2}$ to any of the scales (eg MD, PD, etc.).

In addition to the data from NASA protocol, we evaluated the number of submission attempts to each activity. We also conducted a qualitative analysis with students through an online form. Since the course was delivered through the internet, the option of in-loco interview was not possible, since many participants were physically distant. This research tried to lift especially if there were problems in the student access to course, and their opinions about the teaching methodology adopted and the tools used.

On the next section, we will describe the analysis of the data collected.

C. Enrollment analysis

Despite more than 300 student requested inscrition in couse, only 144 students have confirmed registration and 46 of them never accessed the system. Another negative data is the number of students that did not perform a single activity: 88 students. This is showed in Table I, in which the column labeled "With exp." means the students that declared having previous experience with programming, while the column "Without exp." is the opposite situation. Tables I to II also presented the data separating the students in accordance with their group in the web-course, the group G1 with iVProg and the group VPL.

TABLE I – No show in G1 and G2

Group	System	With exp.	Without exp.	Total
G1	iVProg	9	16	25
G2	VPL	7	14	21

Considering all the students enrolled in the web-course, about half of them have declared previous experience with programming. This is presented at Table II.

TABLE II – Student distribution over the groups G1 and G2

Group	System	With exp.	Without exp.	Total
G1	iVProg	31	41	72
G2	VPL	31	41	72
		62	82	144

The low participation could be explained by three facts: the course period; the origin of the students enrolled; and, the students are volunteers. About 82 of them were freshmen at the University of São Paulo (**USP**), and the course period colided with the final exams of the students in the USP.

Besides, the last week in our web-course occurred at the end of the semester at the USP, probably explaining the very low number of students doing the activities. This is showed at Table III, with only 16 students doing the final activities, that included the last NASA TLX questionnaire.

TABLE III – Students that accessed the last week of the course

Group	System	With exp.	Without exp.	Total
G1	iVProg	3	3	6
G2	VPL	2	8	10

Since we did not interviewed the students it is not possible to explain the low rate of participation. However the hypothesis of volunteers' basis and the absence of certification is in accordance with other author [16].

In the next section are analyzed the data obtained.

D. Analysis of activities (programming and questionnaires)

The NASA TLX protocol allowed interesting observations about the use of VPL and iVProg tools during the execution of course activities. The protocol analyses: the mental demand (MD), physical demand (PD), temporal demand (TD), own performance (OP), effort (EF) and frustration (FR), in accordance with the student point of view.

The objective of each NASA TLX questionnaire is to identify the perception of the student considering the six scales MD, PD, TD, OP, EF, and FR. It was applied after each block of activities with iVProg (in G1) or with C (in G2). Each application were formed by 15 pairwise choice (each combination of two scales) to order them considering the difficult identified by the student. Our interest was to identify the item (scale) which demands more effort in each activity, comparing iVProg with C.

In figures 5 and 6 are presented the NASA TLX to groups G1 and G2 to the first block of activities. In it is observed that, in G1, the most significative demand was EF (effort), with median 8. In group G2 the EF was smaller, however MD (mental demand) and TD (temporal demand) was significantly bigger. The MD was 4 in G1, and 8 in G2. However, the biggest value in G2 (C) are between 25 and 30. To confirm this observation it was used the WMW test, that is presented in table IV. In this table the values p is calculated to all scale to G1 and G2, with hypothesis: H_0 : *distribution*$_{G1}$ = *distribution*$_{G2}$; and H_1: *distribution*$_{G1}$ < *distribution*$_{G2}$, with one exception to the OP* scale, to which was considered as H_1: *distribution*$_{G1}$ > *distribution*$_{G2}$.

To each value of *p* there is no evidence to reject the null hypothesis. However, it is noteworthy that the values relatad to mental effort and time, respectively, MD and TD is quite smaller, as observed in table IV.

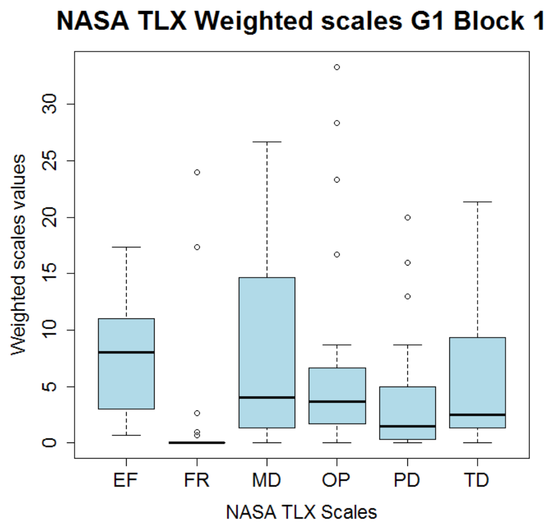


Fig. 5. NASA TLX: G1 block 1 weighted scales (22 students' responses)

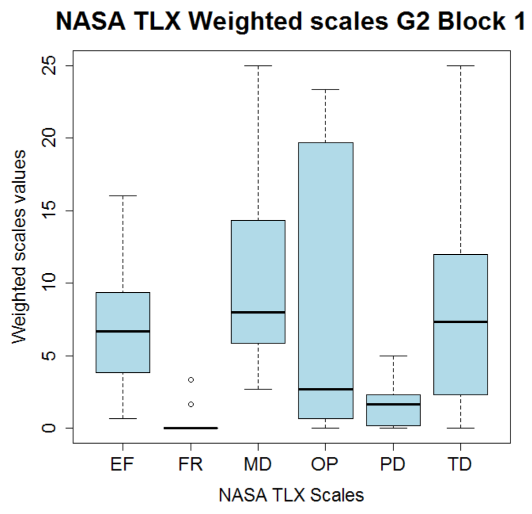


Fig. 6. NASA TLX: G2 block 1 weighted scales (12 students' responses)

TABLE IV – WMW *p-value* for G1 and G2, block 1

	EF	FR	MD	OP*	PD	TD
p-value	0.6409	0.6676	0.1167	0.3002	0.8272	0.1132

The data collected from the students activities is in accordance to the perceptions above. The number of attempts of solving the problems in G2 group, using C+VPL (textual programming), were up to 15 times and it was not uncommon to find a number greater than five attempts. The G1 group (iVProg) used 4 attempts at most (only 1 student), more than that, the most common situation was the student submit the correct answer in first trial.

In figures 7 and 8 are presented the median to NASA-TLX to the second block of activities. Again the MD is smaller in G1 (6.33), than in G2 (11.33).

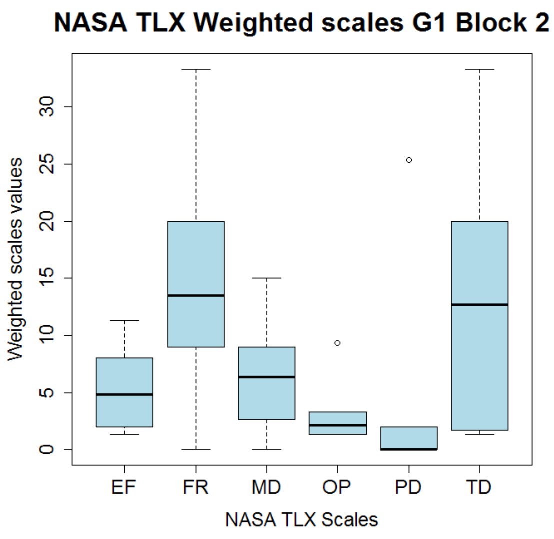


Fig. 7. NASA TLX: G2 block 2 weighted scales (6 students' responses)

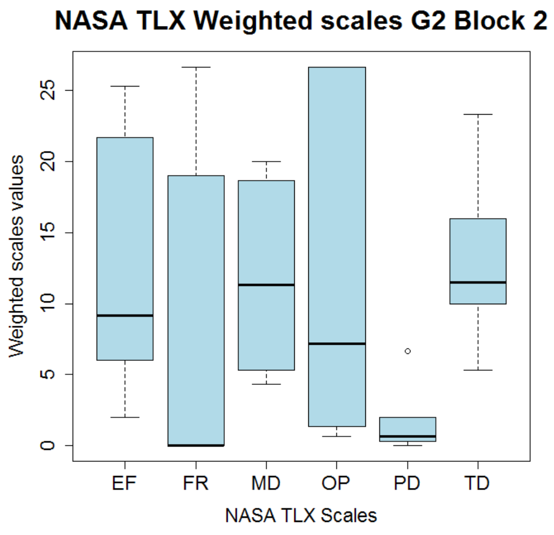


Fig. 8. NASA TLX: G2 block 2 weighted scales (6 students' responses)

In table V is presented the WMW test. The value of *p* were computed to each scale to the second block of activities. Once more, there is no evidence to reject the null hypothesis. However, must be noticed that the *p-values* to the scales EF and MD are significantly smaller than the others. This is confimed under the figures 7 and 8, in which EF and MD is smaller in G1 than in G2, respectively, 4.835 against 9.165 and 6.33 against 11.33.

TABLE V – WMW *p-value* for G1 and G2, block 2

	EF	FR	MD	OP*	PD	TD
p-value	0.07441	0.8935	0.1473	0.2071	0.2023	0.532

The number of submissions in G2 (C+VPL) is significantly higher than in G1 (iVProg). Again, in G2 the maximum number of attempts to solve a problem was 12 and the most common situation is the use of 5 attempts. Nevertheless, in G1, the maximum number of attempts was 4 and the most common situation was students sending the correct answer in their first trial.

In figures 9 and 10 are presented the median to NASA TLX to the third block of activities. In this blok the highlight is scale EF, about 6 in G1, against 13.33 in G2.

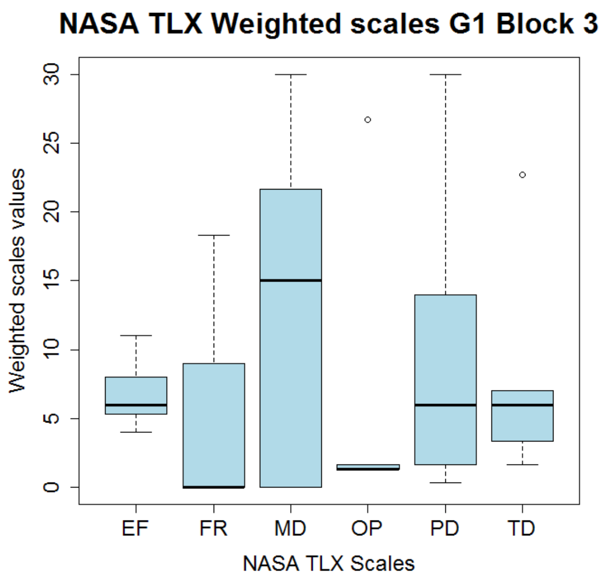


Fig. 9. NASA TLX: G1 block 3 weighted scales (5 students' responses)

Again, we constructed a table showing the calculated *p-values*, that is presented in table VI.

TABLE VI – WMW *p-value* for G1 and G2, block 3

	EF	FR	MD	OP*	PD	TD
p-value	0.1452	0.6028	0.4353	0.7435	0.9642	0.6028

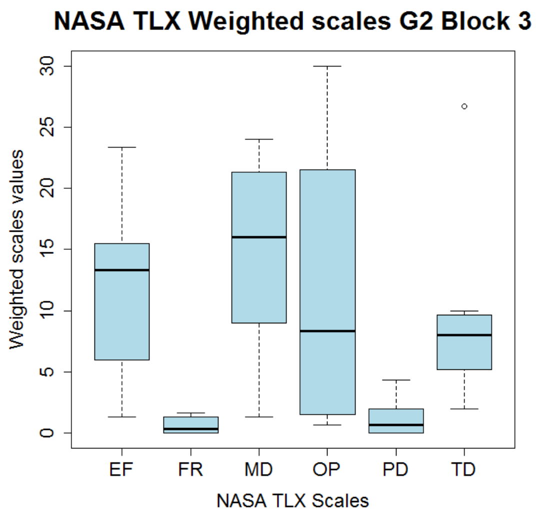


Fig. 10. NASA TLX: G2 block 3 weighted scales (7 students' responses)

The analysis of the third block of activities allowed us to observe that the median of MD were 15 in G1 and 16 in G2, a small difference compared with the other blocks. EF in G1 were 6 and in G2 were 13.33. Regarding the number of submissions for G1 few students did the activities, however, their submission was correct on only 1 attempt. In G2, The number of submissions was slightly higher, however, the number of attempts reached the maximum of 17.

As aforementioned, the number of NASA TLX submissions for the forth and fifth blocks were not sufficient to make any comparison. The NASA TLX also showed that in some cases, users have shown a little bit frustrated during the execution of the proposed tasks.

In order to understand this phenomenon and, moreover, to collect qualitative data about the web-course we designed a simple online survey. We also would like to find out why such a high rate of users never accessed the system. The survey questionnaire was answered by 26 students. Among them, 23 were able to access the course satisfactorily and carry out the necessary tasks. In addition, only 3 responses were from users who had never accessed the system. Two of them claimed they did not receive the e-mail with information on the course and one person said he forgot the password and could not retrieve it.

The survey was basically composed by three questions:

1. If you have not accessed the course system or did not accomplished the module I could share the reason with us? If yes, fill out the form below telling us why.

2. If you did the activities and read the instructional material, do you have any suggetion of improvement to the environment or to the material?

3. What is your opinion about the tool used to create algorithms?

Generally, participants reported being very satisfied with the course, with the methodology and the tools used. Some participants had had problems with the Java Applet, that was a central technology to VPL and iVProg. We received 3 emails from students asking for help with the installation of Java and the security level setting to allow execution of Java in the browser. Additionally, another participant wrote at the questionnaire that "... Despite a slight problem with the java configuration at the beginning, it worked very well, I could use it without any trouble".

Praise for the course and methodology were numerous, as in: "I was quite intrigued with the tool, how it provides inputs to the application without arguments in main function and after reading it selectively outputs is very practical for the correction of exercises, plus this the fact that there is instant feedback was very useful. I have nothing to complain about the tool, in fact, if possible, would like to know more about it because I found the concept interesting: the interaction between the tool and the algorithms at run time, I was pleased to see that application in teaching programming". Another one: "I believe that this tool will serve for the initial teaching of algorithms". However there were two students who criticized the agility of the iVProg. One of them wrote: "The idea of the tool as a method for teaching programming logic is good, but who spends much time to make the code (declare variables all the time, etc.)".

Considering the content there were some suggestions, like this one: "Working examples of various algorithms (parity check, count digits, etc..) could be presented next to its theoretical content. This would facilitate the learning and appreciation of these topics to users without solid mathematical basis".

Considering the questionnaire answers, some relations could be established between it and the NASA-TLX protocol. An example is identified in the third block of activities, when the algorithms became more complexes. The current interface of iVProg demands much more time when the algorithm is bigger, i.e., the time consumption seems to increase more then a linear function with complexity.

In the next section we present a discussion and intentions for future work.

IV. Discussion

Since the course adopted an online approach, complete optional to the students, with no certification at all, we observed a consistent reduction in the students participation.

Initially the course had the enrollment of 144 students, however, 32% of them have never accessed the system. Some of them because their email served blocked the email sent to them with user name and password. Furthermore, the students participation in the last activities were drastically reduced. The last block of activities had only 8 students performed the last activity, considering both groups (G1 and G2). We believe that this can be explained by the liberty of the course model, students were volunteers, almost all of them freshmen in university. With the activities in the university increasing, the participation drop down.

Another common problem in online courses is the relation between the student participation and the level of difficulty in the activitiy, that is inversely proportional. A clear example of this relation is observed in the article [16] that analysed a **MOOC** (*Massive Open Online Course*) about circuits and electronic components promoted by the **MIT** (*Massachusetts Institute of Technolog*). The authors stated that the rate of students who never accessed the course was 29% (46000 from 154000 never appeared). This makes us think that this number may be an intrinsic part of the MOOC mode. Moreover, in [16] is reported that students competing for certificate are more dedicated and the drop-out rate decrease in MOOC. Our web-course did not offer certificates due to its experimental characteristic.

Despite the reduced quorum in our web-course, the NASA TLX protocol indicated that visual programming seems to be a nice option to introduce programming concepts. Indeed, the number of submissions to both models indicated that visual programming led less mental demand and less effort for users to accomplish the tasks. In terms of frustration, to carry out the activities, students in G1 felt more frustrated than students in G2 while accomplishing more complex exercises.

At the end of the web-based course a survey questionnaire was used to identify the reason to low quorum and possible frustrations. From the answer it became clear that the main reason to frustrations were problems with the Java applet technology and the difficulties with its configuration.

Considering all collected data, from NASA TLX, activities log, and the survey, we can observe that visual programming is a good model to teach algorithms and programming. However the low number of respondents do not allow stronger assertions.

V. Future works

Since the reduced number of enrolled students prevented us of any statistical conclusions, we intend to perform a new course edition, this time as MOOC.

Another future work is to analyse if a new version of iVProg, now implemented using HTML5 technology can reduce the students frustrations with Java security issues.

Besides, this first course edition comparing visual with textual programming arose several questions that must be investigated in future.

One of them is how to compare the effective learning. Is it possible to compare both models?

Another question is related to the necessity of the traditional textual programming. Is it enough the visual model to Science, Technology, Engineering, and Mathematics (STEM) students?

Another future work must be a quantitative experiment. With the first tested version of content materials and activities, our intentions is start a new course, this time with a more general invitation, not restricted to some freshmen in one university.

Acknowledgment

This work is partially supported by grant #011/10926-2 São Paulo Research Foundation (FAPESP).

References

[1] Carlisle, M. C. "Raptor: a visual programming environment for teaching object-oriented programming".Journal of Computing Sciences in Colleges, vol. 24, issue 4, April 2009, pp. 275-281.

[2] Kölling, M. "The Greenfoot Programming Environment".ACM Transactions on Computing Education (TOCE), vol. 10, issue 4, November 2010.

[3] Maloney, J., Resnick, M., Rusk, N., Silverman, B., Eastmond, E. "The Scratch programming language and environment".ACM Transactions on Computing Education (TOCE), vol. 10, issue 4, November 2010.

[4] Cooper, S., Dann, W., Pausch, R. "Alice: a 3-D tool for introductory programming concepts". Journal of Computing Sciences in Colleges, vol. 15, issue 5, May 2000, pp.107-116.

[5] Hundhausen, C. D., Farley, S., Brown, J. L. "Can direct manipulation lower the barriers to programming and promote positive transfer to textual programming? An experimental study".Visual Languages and IEEE Symposium on Human-Centric Computing, September 2006, pp. 157-164.

[6] Myers, B. A. "Taxonomies of visual programming and program visualization".Journal of Visual Languages & Computing, vol. 1, issue 1, March 1990, pp. 97-123.

[7] Hart, S. G., Staveland, L. E. "Development of NASA-TLX (Task Load Index): Results of Empirical and Theoretical Research". 1988, pp. 139-183.

[8] Moodle: Modular Object-Oriented Dynamic Learning Environment, http://moodle.org. Accessed: 20/04/2014.

[9] VPL: Virtual Programming Lab, http://vpl.dis.ulpgc.es/. Accessed: 20/04/2014.

[10] Brandão, L. O., Ribeiro, R. S., Brandão, A. A. F. "A system to help teaching and learning algorithms". Frontiers in Education Conference (FIE), October 2012, pp. 1-6.

[11]Rodrigues, P. A., Brandão, L. O., Brandão., A. A. F. "Interactive Assignment: a Moodle component to enrich the learning process".Frontiers in Education Conference (FIE), October 2010, pp. T4F-1-T4F-6.

[12] D. L. Dalmon, L. O. Brandão. "Uma linha de produtos de software para módulos de aprendizagem interativa".Anais do Simpósio Brasileiro de Informática na Educação, 2012, pp..

[13] Dalmon, D. L., Brandão, L. O., Brandão, A. A. F., Isotani, S. "A Domain Engineering for Interactive Learning Modules".Journal of Research and Practice in Information Technology, vol. 44, 2012, pp.309-330.

[14] Santos, L. M. A. "A inserção de um agente conversaciaonal animado em um ambiente virtual de aprendizagem a partir da teoria da carga cognitiva".PhD Tesis, 2009, pp.114.

[15] Windell, D., Wiebe, E. N. "Measuring Cognitive Load in Multimedia Instruction: A Comparison of Two Instruments".Annual meeting of the American Educational Research Association, 2007.

[16] Seaton, D. T., Bergner, Y., Chuang, I., Mitros, P., Pritchard, D. E. "Who Does What in a Massive Open Online Course? (MOOCs)".Communications of the ACM, vol. 57, issue 4, April 2014.