# Example usage evaluation for the programming learning in the MELBA environment

Loé Sanou, Sybille Caffiau, Patrick Girard, Laurent Guittet

# EVALUATION OF THE EXAMPLE USAGE FOR THE PROGRAMMING LEARNING IN THE MELBA ENVIRONMENT

Loé SANOU, Sybille CAFIAU, Patrick GIRARD, Laurent GUITTET
*LISI-ENSMA (Laboratory of Applied Computer Science)*
*1 avenue Clement Ader - BP 40109 – Futuroscope, France*

**ABSTRACT**

This paper describes the validation or an example-based approach to learn programming. This validation was made by a controlled experiment in real conditions. With a computer assisted learning environment, MELBA (Metaphors-based Environment to Learn the Basic of Algorithmic), users were required to follow an experimental protocol that describes programming through examples. The evaluation focused not only on the feasibility and ease of programming tasks, but also on the acquisition procedural than declarative programming skills, which could be easily reusable with another tool.

**KEYWORDS**

Programming by example, Computer-dided learning and teaching, Experimentations, Evaluation, Usability, User testing.

## 1. INTRODUCTION

Many studies have shown that it is difficult for a novice programmer to realize programs (Boulay 1989). The programming includes not only learning a programming language, but also the ability to create an algorithm for solving the problem. It concerns three types of knowledge (Blackwell 2002): syntactic, semantic and schematic. The syntactic knowledge covers the lexical and syntactic elements of the programming language. The semantic knowledge refers to the concepts of variables, appeals, and object instantiation. Schematic knowledge uses previous knowledge to define generic structures of solution. To assist the novice programmer in learning programming, several techniques have emerged including learning programming “based on the example". The programming “based on the example” is characterized by the manipulation of concrete examples to build a program (Lieberman 2001). The use of an approach based examples for learning to program is the hallmark of programming called "practical". In contrast to the classic "abstract" approach, which uses a posteriori games of tests to evaluate a finished program, "an approach based on the example" allow you to specify the lead in the first. The context objects are either used to provide a progressive assessment (known as visualization program and programming with example), or manipulated by the programmer to create the program (programming on example). In the latter case, the program may be represented built literally, through a visualization tool or completely hidden (Booth 1992).

Following research carried out in order to identify problems related to programming (Boulay 1989; Canfield et al. 2001), solutions have been proposed to facilitate the learning of programming, both in terms of methodology and on the tools (Soloway 1985; Kahn 2001). The result is the provision of user environments integrating various techniques to make a simple abstraction of different concepts. Most of these environments provide examples on which the novice programmer must rely to produce her programs. Several studies have demonstrated the relevance of these systems, especially in terms of the final result. Learners, especially pure novices benefit greatly from the use of the tool. But learners do appreciate this method? The systems can be used in a mode where the example is the support of learning. Are they being used in this mode (Pane 2002)?

To answer these questions, we conducted a study in the context of an introductory course in programming. This study focused on the learning environment for programming “based on example" MELBA (Metaphors-based Environment to Learn the Basic of Algorithmic) (Guibert 2006), which has been used 3 years ago in the teaching at University of Poitiers.

In the first part of this article, we present the MELBA system, learning system for programming “based on the example". The second part focuses on the assessment itself. We present in first the used method, tasks to be solved, and the procedures for observation and collection, and then we analyze the collected data. Finally, a discussion allows us to discuss the usefulness of the example in the learning process. The conclusion we can give a few openings in this work.

## 2. DESIGN AND EVALUATION OF SYSTEMS OF PROGRAMMING LEARNING « BASED ON THE EXAMPLE »

The desirable characteristics in a learning system for programming differ greatly from those of integrated development environments (McDaniel et al. 1999). The learning systems programming should allow novices to gain experience. They will have to bear all the classical abstractions of programming, while allowing to introduce them in the most progressive possible manner (Canfield et al. 2001). The apprenticeship system of programming is based on the fact that the novice programmer does not want to program, but only to learn how to program.

**Knowledge of current and orientation studies.** The characteristics associated with the learning of programming, especially for beginners are evident (Guibert 2004). The systems typically used for learning about programming are built on a set of tools designed from the perspective of development and not in an explicit pedagogical way. Their sole purpose was to provide analytical tools supported by the program design. A recurring criticism (Mancy 2004) to the use of such systems in learning is that it induces a kind of implicit learning focusing on the resources and tools rather than on the activities of the learner. A pedagogical approach explicitly opposes this approach described as industrial. This approach aims to develop active learning situations, in which the user interactions with the environment first aimed are discovery and the construction of knowledge and not conducting a technical task. This approach is centered on the learner and its activities. This designation is not related to a particular area of learning. It is in this second perspective that MELBA has been designed (Guibert 2006).

**A learning environment for programming based on example: MELBA.** MELBA is an introductory computer programming and is based on the example (Guibert 2006). It is a learning system which consists of three main zones: a space program that lets you represent and edit it, an area representing the semantics running (and primitive structures of the language used), and an area pragmatic to illustrate the program (Figure 1). A similar approach can be found in the system ALVIS (Hundhausen 2007). Each zone is interactive, and the consistency of all is provided by the programming mechanisms on or with example (Guibert 2004). MELBA allows the user to model the progress of a job through a program; it is the knowledge target, the model that the user has to learn to generate and animate mentally (Mancy 2004). The user can from its know-how on the job, and lead a program with the help of programming mechanism with example, or try to understand a program through the animation of a sample.

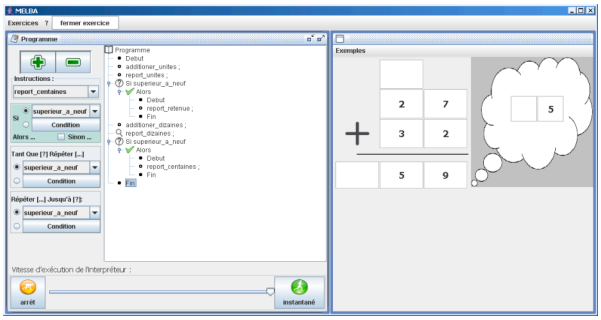


Figure 1: The MELBA environment

**Evaluations conducted.** On MELBA two scores in an ecological context were conducted (Guibert 2006). The first applied to a course of "introductory computer programming" and compared the performance of a group with MELBA to those in a control group who followed a curriculum of classical TD. The goal of the second experiment was to clarify the support provided by the tool, every skill and educational objective. The

interpretations of these results validate unambiguous use of the environment MELBA as part of the course "Introduction to Programming".

**Results.** The assessment results of MELBA are analyzed in thesis report of Guibert (Guibert 2006). The author observes that the use of MELBA causes an elongation of about 30% of the time spent on exercises TD by learners. This is common to most of EIAH (Kahn 2001) in qualitative studies. On the plan for the impact of the use of the tool in the learning process, studies have shown an improvement in the mastery of the concepts taught, with a pronounced effect for weakest students.

The uses and use difficulties experienced by learners were then trained at an experimentation conducted on a oculometer Tobii 1750 in collaboration with the team Multicom at Grenoble (Guibert 2006). This experiment has mainly studied the use of areas in the application, and tried to characterize learning styles. The study eye has shown a difference in rates of eye fixation in the program and operation areas to the detriment of the execution, depending on the task at hand (or correction program phased construction program). It also showed that users very often change fashion, going for the same task from editing mode static program to lively fashion, and vice versa.

It seemed interesting to try to characterize these behaviors, and especially to check whether users benefit derived from the use of the example in the learning process.

# 3. A STUDY TO EVALUATE THE USE OF THE EXAMPLE IN THE PROGRAMMING LEARNING

The study presented here is based on a technique collectively used in evaluation, technical observation. The principle is to observe the user action and to collect systematically data during its activity (John, 1996). A set of pre-written questionnaires enabled us to collect data on the subjects and to appreciate their degree of satisfaction. The results obtained qualitatively were analyzed and interpreted whereas the quantitative data are exploited by crossed analysis.

**The study objective and the research ambition.** The study aims at estimating the use and the usability of the example in MELBA and the ease of understanding of the declarative and procedural knowledge in programming. This study will complete the evaluation of the efficiency and MELBA's use in universities. The results of the study will allow to bring the other visions on the improvement of the various concepts proposed in MELBA and will facilitate the improvements of the usability of the tool. The study is conducted, from a point of view of research, to think about the contributions of the example in the facilitation of understanding of the concepts of the programming.

**The used material.** For the sessions, MELBA was installed in a room of Windows PCs, in which the subjects had regularly accesses for their practical works. The working environment was thus familiar to them and the computer hardware as well as its use was not new for them.

**Test environment.** The main objective of the study being the proof of the efficiency of the example in the learning of the programming, it is essential to the learner in a situation of active learning, in which its interactions have for first purpose the discovery and the construction of programming knowledge. MELBA was conceived in this optics (Guibert 2006) and is intended to support the fundamental concepts of the programming. Its main characteristics are to be an autonomous tool that offers the learner programming exercises to resolve interactively. The edition, the interpretation, and the execution of the program written by learner are fully managed. Finally, MELBA is a flexible environment according to the wishes of the teacher.

To facilitate learning, the learning environment can be customized by the teacher according to the exercise and the target concepts of learning. MELBA is to "learn to program" and not "to program without learning" (Guibert 2006). The main benefits of MELBA in its use are the support of an experimental learning and its environment adapted to the educational activity (and not the opposite). The exercises of the tool are designed according to three levels (pragmatic approaches, hybrid, symbolic representation of the context).

## 3.1 The method

The study concerned 35 students in biology bac+1 / bac+2 (L1/L2) level and in bio-computing bac+4 (M1) level. The age of the subjects varies between 18 years and 25 years. According to their knowledge in

computing, we identified three groups of subjects: the "novices", the "intermediaries" and the "advanced". The "novices" have no knowledge of the programming but only received 12 hours of theoretical lesson in initiation to programming. They represent 51,4 % of the tested population. The "intermediaries" have a small experience of the programming, that is possess some basic notions in programming. In quantitative term, they have already benefited about 30 hours of formation in programming. They constitute 28,6 % of the tested population. The remaining 20 % are the "advanced". They benefit from a wide knowledge in programming because they possess more than a half of the year of programming in their asset. The subjects "novices" and "the intermediaries" attended a demonstration of the software MELBA during a quarter of an hour.

The evaluation was conducted in the form of supervised work on machine. They had the possibility of using some paper/pencil. The subjects had any knowledge of the tasks to be realized, recorded on paper at the beginning of the session. The location of the software on workstations and the objective to achieve was given. Three responsible of TD were available to students for any matters relating to the understanding of questions and often to guide them in the event of deadlock. At each post office, was willing an observer. At the end of sessions, a pooling of notes was conducted, to be able to proceed to a global and generalized analysis.

### 3.1.1 The tasks to be realized

The work asked to the subjects concerned an analysis of program, the correction of an algorithm and the complete realization of programs, the whole in two exercises. The first exercise contains a series of four questions and the second contains only two questions. The themes of the exercises are inspired by Charles Duchateau's book, Images to program (DuChâteau 2002).

The goal of the first exercise (DuChâteau 2002) (reporter of notes) is to evaluate, and possibly to modify a program that automates a task of adjournment of notes by automate. It has of before him a stack of copies, a sheet of notes containing a list of names associated with the note of the student, and a stack of straightens copies (Figure 2). At first, we suppose that every student of the list returned his copy. The automate is capable of executing certain number of elementary operations such as move atop the list of the sheet of notes, move to the next line or save a copy of it on the discards.

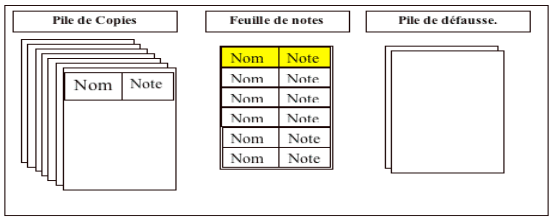


Figure 2: Plan of the exercise on " Reporter of note ".

It was first asked the student to explain the behavior of the program provided, the use of animation of the program is strongly suggested. In a second phase, a proposition of solution is asked to correct the error in the program. Then, another version of the program is proposed, including new control structure. The tasks are the same as above. Finally it is asked to propose a program from the two examples corrected conceding that all students have not return their copy (more complex case).

Exercise 2 (DuChâteau 2002) purpose is to write a program that automates the task of filling of glasses with a account-drop. The performer has in front of him an alignment of glasses, all initially empty and capacity equal; an account-drop, initially empty is positioned above the first glass (Figure 3); number of glasses, the contents of the pipette and glasses can vary. The performer is capable of operations as follows spend the glass, pressing a drop or complete the account-drop.

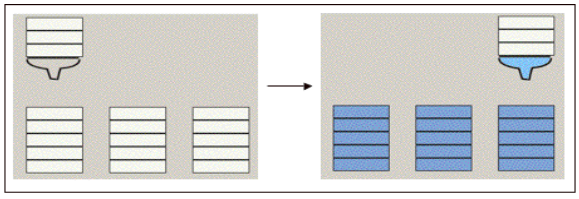

Figure 3: Plan of the exercise on " the account-drop ".

It is asked the subjects to write a program allowing filling 3 glasses of size 5 by means of an account-drop of size 4. Then, the subjects have to validate their program with 2 glasses of size 4 and an account-drop of size 5. An explanation is asked, then if need be a modification of the program must be realized.

### 3.1.2 The testing procedure

Every subject works on a post with beside him an observer from taking notes during the progress of the session, following a main carrying order on " what does the user? " And " how does he make it? ". The observer takes into account the popular purpose and the techniques used by learner who wills always supplies explanations concerning his interactions. The observer can intervene by small questions to encircle better the objectives of learner, and that too without excess to perturb him. The subject may request the intervention of a TD responsible to clarify points of understanding (evaluation takes place within the framework of a classic TD). The charge of TD, in his interventions, is careful not to provide real solutions. He generally responds by simple explanations for using MELBA or on the merits of the exercise. For every subject, there is a time limit of two hours to answer six questions of both exercises. Before the beginning of the session, the observer submits a part of a questionnaire to the subject and the rest at the end of the session. The first part of the questionnaire focuses on the knowledge of the learner (computer expertise, especially in programming, the tool MELBA, etc.) whereas the second part asks for its degree of satisfaction after usage of examples to realize its programs.

## 3.2 The data and information gathered

The collected data arise from notes of the observers and from answers to questionnaires. To confirm the data of satisfaction, in particular concerning the ease to use examples to understand the programming, the session of following TD began with an evaluation of the subjects and allowed the obtaining of supplementary data. The notes of the observers clarify the levels of resolution of the questions as well as the quality of the answers. Times of execution of the tasks are mentioned. A comparative opinion of the observer on the resolution of both exercises also allows measuring the state of evolution in qualitative term of the subject.

**The data and the information analysis.** The analysis of the notes of the observers concerns their capacities of learning not only to understand the structure of the programs examples and to modify them, but also to use these examples to write a new program. She also allows determining the frequency of use of the various modes of MELBA: asynchronous mode (classic edition, without example), synchronous mode (with example), or one of these modes with the animation. Our interest is particularly situated with regard to the usage of the synchronous mode with or without animation and of the asynchronous mode with animation. Always according to the taken temporal notes, it will be easy to compare the time put for the understanding of an example in its execution, and also to writing a new program. The exercises to be treated are defined so as to be able to profile the understanding of learning it to validate its control of the structures of control in algorithms. The evaluation of the personal satisfaction of learning him is numerically expressed on a scale from 1 to 5. The process used by learning it to resolve the various exercises is interpreted from the notes of the observer, the reports of the responsible for the TD and the reports of interview of the end of session with the subjects. Through the data collected, difficulties of MELBA's use were noticed. Their analyses are made according to feature and services proposed by MELBA.

## 3.3 Results

Information analysis allowed identifying tasks resolved by the subjects as well as the resolution time. The strategies used determine the capabilities to use examples for understanding of programming and especially, changes in execution method in the MELBA environment are essential to validate the use of examples in the programming initiation. The interviews with the subjects and notes parts on observer’s behaviour bring others aspects of the test environment as a possible improvement.

### 3.3.1 Tasks realization

Of the 35 subjects, 16 subjects were able to answer all questions (E1 & E2), 11 subjects were arrested at question 1 of exercise 2 (E1 & E2a), and 6 subjects in the exercise 1 (E1) . Only 2 subjects did not complete

the exercise 1 (E1). We considered the latter group at the same level as those who have completed the exercise 1. We summarize accomplishments rate of the tasks according to the percentages of subjects as well as the rate of exact solutions (correct) on matters conducted solely in table 1.

| Group | % Subjects | % Realised | % Exact |
|---|---|---|---|
| E1 | 22,9% | 66,7% | 89% |
| E1&E2a | 31,4% | 83,3% | 95% |
| E1&E2 | 45,7% | 100% | 98% |

Table 1: Correspondence realization rate (depending on the tasks required) and success rate (based on the achievement).

By increasing temporal data with the composition of subjects by skill level programming, we get the following rate in table 2. We will notice that 63.6% of people who came to realize 83.3% of asked work are « novices » (or approximately 39% of novice). Moreover nearly 28% of the « novices » have managed to achieve all programs.

| Group | % in E1 | % in E1&E2a | % in E1&E2 |
|---|---|---|---|
| « Novices » | 75% | 63,6% | 31,25% |
| « Intermediaries » | 12,5% | 27,3% | 37,5% |
| « Advanced » | 12,5% | 9,1% | 31,25% |

Table 2: Rate of the subjects by group with regard to the levels of the realized tasks.

We analyzed in depth these results in terms of realisation time, and strategic performance on the effectiveness of the approach and degree of understanding of the interactions.

- **Times performance:** concern the time taken by the various subjects per group depending on questions (the tasks realised). Total temporal dimension is 2 hours for six questions. We give the results as a time percentage on 120 minutes. The questions of exercise 1 are graded Q1 to Q4 and exercise 2 Q5 and Q6. The time is considered a statistical average time set by learners ordered by group by rate achievement. We get the data in table 3.

| | E1' | E1 | E1&E2a | E1&E2 |
|---|---|---|---|---|
| Q1 | 50% | 40% | 35% | 35% |
| Q2 | 30% | 30% | 25% | 24% |
| Q3 | 20% | 15% | 10% | 10% |
| Q4 | -- | 15% | 15% | 11% |
| Q5 | -- | -- | 15% | 12% |
| Q6 | -- | -- | -- | 8% |

Table 3: Temporal rate of realization of the tasks.

  More subjects advancing in question, more time spent diminish. This means a breakthrough in work understanding and learner knowledge. More he can be adapted to use system, more he advance and his quickly. The Q3 can be seen relatively because it is a question relatively deductible of Q2. This explains a little time to put its resolution compared to the following.
- **Strategic performances:** affect the techniques used to realise those tasks. The subjects of "advanced" level have quick grip that the subjects of "intermediate" level and even more than the "inexperienced". All subjects in general went on aboard premium to the understanding of test environment functioning. Quantitatively, it resort that more than 10% of time set for Q1 and Q2 completion relate to familiarization with MELBA. It is also mentioned that about 5% of time to realise Q5 served with the same intentions. The analysis of temporal notes also leads to deduce that majority of subjects spent about 11% of time allocated to each question to get an idea by an animated simulation and also to check in execution the program written or modified.
- **Efficiency and degree of tasks realisation:** the subjects knowing that they are being tested gave the best they can to achieve tasks as much as possible without TD responsible intervention. Given the time taken and percentages achievements, subjects were an average efficiency of 4 points on a scale of 5 in the aggregate. The rate of conforms implementation expected, i.e. programs and correct answers, which is about 94% confirms more this efficiency, and above all the degree of programming understanding.

### 3.3.2 Strategies and behaviors

To reach their end, the students have adopted three techniques. If at the beginning of the session there were enough questions raised at TD responsible, the subjects were more communicative, or at least more explicit after the first three questions with observer.

- **Information retrieval:** the strategy of seeking information about algorithmic checks structures was exploring environment toolbar. Nevertheless, it should be noted that this was kind on indications of TD responsible, which can be regarded as a learning of test environment. After attempts to retrieve more detailed guidance from the TD responsible in the first half hour, more than 90% of subjects were placed investigated by tinkering technique. They try action then often cancel. There was no guides or manual to their provision.
- **Questions asked:** they were during first hour quarter, on the environment understanding. Then, specific questions on the scheduling of predefined operators followed. Though curiously, 5.7% have received replies from the TD responsible. This rate is the same that group E1' (those who have not completed exercise 1) in relation to total number of subject. The critical questions can be summed up: « How do I know loop to use? » « Is it possible to embed this loop with that here? », « Why my program does not stop? ».
- **Switching Mode:** on environment using with or without animation in programme performance and especially programme writing in synchronous mode (with example) or asynchronous. All subjects (100%) have used animated mode for all tasks. A mix using is made depending on the questions. For example, approximately 20% did not use animation to Q3. However 86.85% of those who have reached Q5 used the synchronous mode with animation. Some average statistics, 12% of subjects have achieved almost 20% of tasks asynchronously. These achievements are generally the tasks Q3, Q4 and rarely Q6. Statistically, the using rate of subjects in different ways depending on the questions is summarized as presented in table 4 below. These rates are calculated on basis of staff that have made task and not on basis of the total number of subjects. For example, there are 33 subjects who were realising Q4 while only 16 subjects have achieved Q6. The utilization rate of environment in « with such » mode are much higher and can be the cause of high levels of achievement. That route case is also deducted from anecdotal information gathered during interviews with subjects at the end of meeting.

| | « Animated » | With example | Without example |
|---|---|---|---|
| Q1 (35) | 100% | 94,29% | 5,71% |
| Q2 (35) | 100% | 97,14% | 2,86% |
| Q3 (35) | 100% | 71,43% | 28,57% |
| Q4 (33) | 100% | 78,79% | 21,21% |
| Q5 (27) | 100% | 92,6% | 7,4% |
| Q6 (16) | 100% | 93,75% | 6,25% |

Table 4: Rates of subjects by use of modes depending on the questions.

### 3.3.3 User interviews data

Notes taken by the observers are rich enough, but could not qualify for best satisfaction of the learner or to better understand their views on the work and its way of seeing the example in programming. An analysis of questionnaires after meeting and notes of the interview shows that over 70% of users have learned more in programming through the example using. The quantitative data in table 5 illustrates knowledge advances in programming topics. We made an analysis in relation to the three levels of subjects. The assessment of knowledge degree is on a scale of 5. The rates are based on the respective effective level.

| | % Before | | % After | |
|---|---|---|---|---|
| | < 3 | >= 3 | < 3 | <= 3 |
| « Novices » | 83,33% | 16,67% | 27,78% | 72,22% |
| « Intermediaries » | 60% | 40% | 20% | 80% |
| « Advanced » | 0% | 100% | 0% | 100% |

Table 5: Percentage of self-assessment notes subjects.

These self-assessment data subjects are not as significant in quantitative « advanced » subjects, but very interesting qualitatively relative to their evaluation feedback. Nearly 85% of « advanced » felt the use of such much more comprehensive in structuring programme and only about 35% find the using of environment rather complex for a « novice ». On this question, 82% of « novices » find environment suitable and nearly 20% says he lost at experimentation start in understanding programs construction.

- **Difficulties with protocol:** as a general rule, either observers or subjects, the protocol was simple to follow set off the failure to attend the learner during certain difficulties in test environment manipulation. By contrast, about 27% of participants found that the questionnaires before and after meeting were

somewhat limited and could not fully express their qualitative feedback. Some of the topics (approximately 17%) would like an initiation meeting prior to MELBA environment.

- **Suggestions for improvement:** they wear for most of test environment. Only 17.14% of subjects had a full appreciation on test environment. Difficulties such as « removing loop in a program without affecting the result of the code », « impossible moving instruction in the program structure » or « not availability of contextual help » are the accusations principals collected. Qualitative analysis of this information can cast a prospect of improving the system MELBA.

# 4. DISCUSSION

The results presented are derived from data and information recoilless in experimentation. They are not following hypothesis topic of discussion or personal interpretation. The aspects that deserve further discussion can be self evaluation subjects, types of exercises selected, and especially non comparison with a control group who worked to achieve the same tasks without use of the environment or examples answer questions. It is easy to justify the choice of exercises by the fact that these exercises were used, along with other subjects, validation of the environment and in the classroom, they have proven themselves in the course of introductory algorithms. Self-evaluation topics are a guarantee of safe return of the subject to his vision of the value of work in relation to our goals. It is easy to be the same whether or not it has assimilated a technique or not. It is true that they need some more time to better understand some factors unlike others who, after a few moments are completely at ease in the application of technology. But is this as long as these factors affect the evaluation results from an acquaintance?

## 4.1 Usefulness and Usability of the example in MELBA

The experience we had not completed its objective related to the quality of the system software tested, but much more in reference with the user in order to monitor their understanding of programming through examples of programmes. Obviously through the figures given that the ability of the example for the performance of a programme by a learner is high. The effect that we expect can be regarded as expected. Learning programming with "examples" is very efficient and rewarding subjects refers to the level of comfort felt by the use of the concept. Although he has some difficulties afferent thus reducing the efficiency of production with the result, it does not demure unless performance in implementing programmes is acceptable, and even very satisfactory. One could discuss the fact that this performance depends topics and conditions, but there exists a protocol clear, precise and rigorous allowed to make these kinds of tests that without these factors are influential?

The utilizability the example in learning programming is the ability to create programs by students from examples of programs. We can measure the result. The success of tasks is the fundamental factor in the quality of the performance of the strategy for reachability of the target. The discussion possible on the subjectivity of the acceptability of the performance is no longer subject to that level of achievement. Also, satisfaction, a factor for usability, does not have the same importance in the context of professional evaluation of the technique to use the example that staff in the context of the learner. In our case, the student had the opportunity to restrict the mode example and without animation. To end on the efficiency and effectiveness of the example in learning programming, we will say that learners have provided minimal effort of understanding to produce a program called from the example. The yield of comportment usual learners is significant.

## 4.2 Some methodological

Qualitative analysis of the study showed the effectiveness of the approach introductory computer programming through examples. The exercises analysis and correction programme have enabled subjects to understand in their own way, programming structures. The measurement of the efficiency required prior to the definition of objectives to be achieved, in a precise manner. An analysis of the tasks to be performed was welcome. The setting in terms of the percentage allows an orientation based on the topics to confirm to learn

and remember knowledge of programming. The assessment staff made before and after a chance also to define the nature of the process intellectual understanding of programming through examples.

An improved protocol is feasible for another experiment with the same subject but through other exercises. This improvement may be on the side of the definition observation techniques learner. In the environment, incorporating a timer taking into account accurately for the work time on each mode (which he referred to as executions in a file) will be more effective and more precise than taking notes manual. An initiation of manipulation or to the environment familiarization test is a possible increase in the performance of learners. This could decrease factor frustration of the learner at the beginning of meetings.

## 5. CONCLUSION

The programming “based on example" is a paradigm particularly attractive in view of the initiation into the programming. Using the example allows the learner to present the information of its programme in its business logic from the example. It allows the learner to think about specific examples, and it restores direct manipulation, thereby allowing an assessment of progressive behaviour of the program. This dimension is critical to allow the construction and consolidation of mental models by the learner. The immediate return provided by the example allows the learner to build your own mental models of experimentally, testing them and supplementing them as and when, thus validating the effectiveness of the lead in learning the programming.

MELBA is unusual for introductory computer programming because it combines the two approaches "with" and "by" example, to adapt to a broad audience possible, with different learning styles.

## ACKNOWLEDGEMENT

We particularly thank Nicolas GUIBERT, MELBA environment author as part of his doctoral work and also to the experiences of the environment validation and its use. Our thanks also go to students in IUP M1 Physiology and Computer Engineering (some observers, other "advanced" learners) and L1/L2 Biology (learners), at University of Poitiers, and Gaëlle Largeteau, responsible for TD of computer programming initiation, which allowed us to integrate this assessment in the normal education.

## REFERENCES

Blackwell, A. (2002). What is Programming? London, UK, PPIG, Brunel University.

Booth, S. (1992). Learning to program : a phenomenographic perspective.

Canfield Smith, A. Cypher, et al. (2001). Novice Programming Comes of Age. Your Wish is My Command. H. Lieberman**:** 7-20.

Charles DuChâteau, J. A. (2002). Images pour programmer : apprendre les concepts de base. .

Du Boulay, B. (1989). Some Difficulties of Learning to Program. Studying the Novice Programmer.

Guibert, N. (2006). Validation d'une approche basée sur l'exemple pour l'initiation à la programmation. LISI / ENSMA. Poitiers, Université de Poitiers. **Docteur de l'Université de Poitiers:** 246.

Guibert N., G. P., Guittet L. (2004). Example-based Programming: a pertinent visual approach for learning to program. Advanced Visual Interfaces, Gallipoli, Italy, acm press.

Hundhausen, J. L. B. C. D. (2007). "What You See Is What You Code : A Live Algorithm Development and Visualization Environment for Novice Learners. ." Journal of Visual Languages and Computing **18(1)**: 22-47.

John, B. E. and D. E. Kieras (1996). Using GOMS for User Interface Design and Evaluation: Which Technique? ACM Transactions on Computer-Human Interaction. **3:** 287-319.

Kahn, K. (2001). How Any Program Can Be Created by Working with Examples. Your Wish is My Command. H. Lieberman**:** 21-44.

Lieberman, H. (2001). Your Wish is my command, Morgan Kaufmann.

Mancy, R. N. (2004). Aspect of Cognitive Style and Programming. PPIG Workshop, Carlow, Irland.

McDaniel, R. G. and B. A. Myers (1999). Gamut : Creating Complete Applications Using Only Programming by Demonstration. Submitted for Publication in CHI'99 - publié !!!

Nicolas Guibert, L. G., Patrick Girard (2004). **Apprendre la programmation par l'exemple : méthode et système.** Technologies de l'Information et de la Connaissance dans l'Enseignement Supérieur et l'Industrie (TICE), UTC Compiègnes- France.

Pane (2002). A programming System for Children that is Designed for Usability. Computer Science, Carnegie Mellon University.

Soloway, E. a. S., J. C. (1985). Studying the Novice Pogrammer, Lawrence Erlbaum Associates.