MULTIMEDIA LEARNING AND ITS ROLE IN EDUCATION

A. Kadriu

CST Department, SEE University

Ilindenska, b. b., 1200 Tetovo, Macedonia

a.kadriu@see-university.edu.mk

Abstract: This paper is about the application of multimedia technology in the teaching and learning process. We are all witnesses of the problems in the educational systems in many countries. Most of the theories on education goals agree that students should be taught to think independently and their knowledge should satisfy the needs of the society. In this context, the computer technology and, particularly, the multimedia technology can play an important role in improving the educational process. While developing educational software, except the technical aspects, one should also consider the pedagogical basis for the concrete problem on one side and the teaching theories on the other side.

Keywords: multimedia, interactive learning, educational software, constructivism

1 Introduction

Education represents the highest form of learning. It is a complex structure that contains many substructures. One of these possible substructures is multimedia technology. It brings a new learning philosophy: learning through pictures, voices and motions that appear on the computer screen. Besides passive teaching, it provides also an interactive environment, where the student can exercise and test his/her knowledge, influencing so his/her own teaching course.

2 Learning theories

In time, many different ways that explain how people learn developed. They relate to the learning process and the interaction between the learning subject and the teacher. All these theories have their own characteristics, their own names and their followers. Nevertheless, for all of them we can say that they can be depicted as a point in the spectrum between two main philosophies: behaviourism and constructivism.

Behaviorism' s idea is very similar to the traditional way of teaching and learning: there is an explicit teacher on one side, a human being that dictates what and how should be learned, and a student on the other side, as a passive learner that has only to absorb lectures given by the teacher, thus having no influence on the learning process. Furthermore, the teacher not only gives "authoritative lectures", but he commands and motivates the student's learning by distributing "rewards" and "penalties" corresponding to the degree of success in learning the lectures.

Constructivism' s philosophy on the other side struggles against the imposed learning and strives for "autochthonous" absorption of knowledge, aiming at accessing the knowledge base individually. In addition to the independent access, it emphasizes the importance of group work, that way preparing the students for the real life. This philosophy does not consider the knowledge as something separated from the human. It is considered as a process in the human being itself, a process that should be only "supported" by the teacher. In this way, instead of teacher-centered environment, we have a student-centered environment. In this atmosphere, the student can learn not only pre-packaged lectures, but also how to solve problems and think critically. Computers and, especially, multimedia technology with its "real world" nature can contribute a lot to the learning process within the framework of such a philosophy.

3 Developing educational multimedia software

To develop educational software, one should, from the very beginning simultaneously write both the document that will define the goal, and the procedure for creating the system. The process of writing these inputs is important as the documents themselves, because it helps in the process of thinking in a structured way, thus corresponding to the work that should be done. When writing these documents, all critical fields that can cause problems later can be identified. Furthermore, those documents will be an excellent basis for the further work and an opportunity to keep a clear vision on the final view of the overall system. The following figure shows the order how these documents should be written:

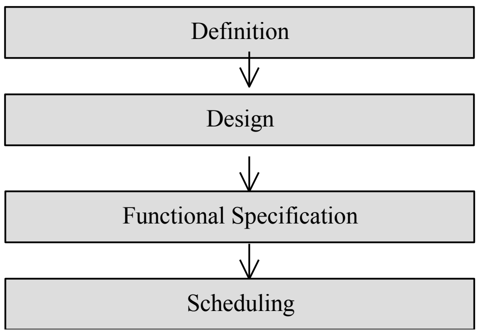

Fig. 2: The order of writing the design documents

The first document should define the overall goal of the project. It should be concise and clear: what has to be done and what tool will be used; how should the final product look like; will it be CD distributed or used for distance learning; etc. The developers should plan the timing and the expense, and also have in consideration the hardware power. They should be problem-solvers that cannot ignore students' requirements and those technology opportunities that can save money and time. There should be no surprises.

In this phase is very important to decide on the development tool that will best fulfil the requirements of the work. There are some aspects to bear in mind when choosing a tool, such as the hardware platform where the product will be delivered, the subject' s character, and the available budget. The easy use and the power of expression also take place in bringing decision about that what kind of tool will be used. Authorware Professional, ToolBook, Macromedia Director are some known multimedia development tools.

The next step is design, where a more detailed description is provided. Developing educational software, particularly for very young students, might look very simple, but it actually represents a very complex problem. In creating such an application, a number of independent components that interact with each other should be considered. The figure below contains a few of these components.

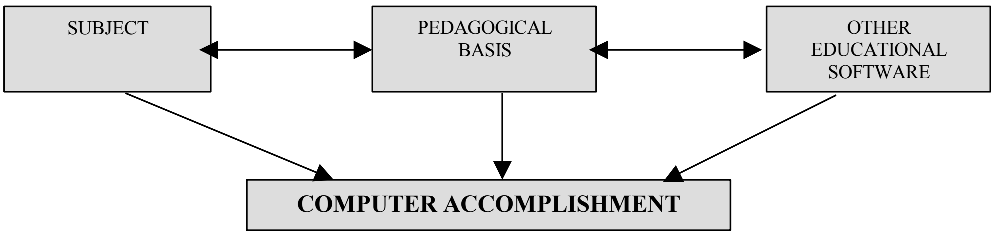

Fig. 3: Designing components

The very first step when designing an application for a concrete subject should be studying the books for that subject (for example, textbooks used in schools). The authors have to identify the main concepts of the topic and provide a structure that reflects the nature of the concrete topic. Inside this structure, all links between the main concepts will be kept. A poor initial structure will bring about a disorder when the application will be actually used.

One of the most frequently used structures is the hierarchical structure, where we have a number of substructures splitting up to other substructures and so on. This structure is shown in the figure below.

When designing an educational application, it is very important to take into account the pedagogical basis. The first step here is to consult experts in the specific field. For example, if one wants to make a math software for students aged 10-18, a math teacher should be consulted, but if software is prepared for children aged 6-9, one should consult a teacher that works with this class of students and knows better their psychology; he/she is not the best math expert, but is excellent in pedagogy and teaching "such simple things" to very young students. We will know that 1+1=2, and we don' t need a professional to tell us that. We need somebody that will help us give that information to a 6-year child through computers and multimedia.

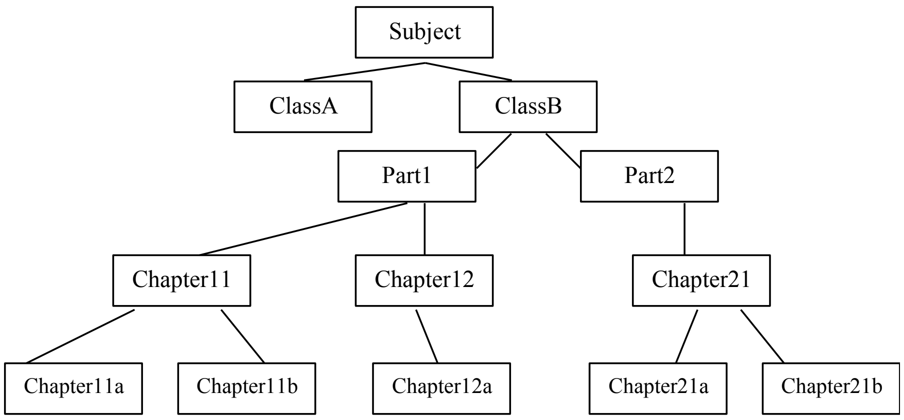

Fig. 4: Hierarchical structure

When deciding to make an educational product for very young children, it is taken for granted that some basic instructions have to be provided. As the learning material is simple and "self-evident", there is no need for prior learning or research of that learning material. The author can directly go into design and programming. When trying to make the first computer lecture, it is figured out how much creativity is needed, and how much should be invested to help the child to distinguish, say, a cube from the other three-dimensional forms.

Software for children is efficient and interesting if it is simple and clear, with as few as possible interface elements. The graphics components should be designed so that the colors are clear and lively - green, red, blue, yellow are the key colors in a child's head. The pictures used have to be also simple, childish and should preferably look as if drawn by a child's hand. The product will become much more attractive if characters from different stories and cartoons are used.

The voice as a special addition to multimedia applications can also contribute the software to become much more fascinating. A three years old child will show interest for a game only if a voice is heard all the time. These voices should have an effect of happiness when the child succeeds in a game, and show disappointment if not. If there are any non-interactive parts that only deliver lectures, there should be some melodies in the background that are pleasant for the child's ear, so it will not be bored while sitting in the front of the computer as a passive user.

The speech and the visibility are the core elements in the human-to-human communication. Speech is an essential part of every educational software, particularly for the younger children that do not have reading and writing skills, and so use this kind of communication intensively. Children will be especially impressed if they hear childish voices from the side of the computer-teacher.

Due to the need for combining voices from different sources (speech and imported melodies that are usually different file types), the nature of these voices (e.g. childish versus adult voices) should be synchronized, and their magnitudes normalized.

The text should also be not complicated, clean, with no scrolling attributes that can confuse children.

It is a good idea to consult also other educational software and see what is done on the same topic, although it can be quite difficult to find such software through Internet, and even more so through software groceries.

Most of the applications possess professional approval from a pedagogue that has estimated the age of children who can use the program. The children software is usually split in three groups with the age as a criteria:

- educational software for children from 2 to 5;
- educational software for children from 6 to 10;
- educational software for children from 10 to 14;

Normally, the age band is indicative and should not be construed to represent a strict barrier to browse programs below or over the age band of an actual child. A "paint" program can be both fun and useful for children aged from 4 to 14 years.

There are products that, at first impression, do not have some educational nature, but they nevertheless can be useful for improving logic reasoning (e.g. puzzle, word and card games).

In general, the design document provides more details compared to the definition document, and the functional specification includes greater depth compare to the design. In the third step of functional specification, all resources that will be used should be involved. The resource development represents a significant element of every product. A well-organized planning of assets will result in cost-effective progress.

The first step is defining the type of resources. Usually they can be external (imported) and internal (made within the framework of the development tool), as shown in the figure below.

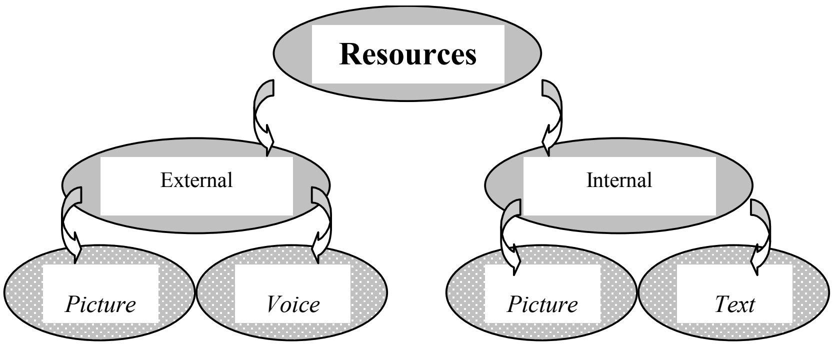

Fig. 5: Using resources

The external resources usually contain pictures and voices, whereas internal include (simpler) pictures and text.

In this stage of the expansion, the interface that will be used and its resources should be outlined. The interface should not require strong effort and it has to have navigation elements. There are a few general design rules in this context. These interface tools should lead the students through the knowledge area. They have to know where they are at every single moment. The developers should be aware of these student requirements and expectations.

As the development expands, the schedule should be outlined and filled in. This is very useful when following the growth of the project and taking care the product to be finished on time and within the budget. Filling in the schedule goes parallel with the computer realization of the software using the selected tool.

If the project has to be done by a team, there must be a supervisor with the aim of managing and controlling the expansion of the application. Another issue is heterogeneity of the team, as the team can include programmers, educational experts and sometimes designers. Often pedagogues are not aware of the technology opportunities, and programmers cannot fully understand the pedagogical aspects. In such situation, it is a quite significant issue to harmonize the team and give team members an overall explanation of what and how should be made.

The computer realization of the educational software goes step by step. After all components are completed, they have to be integrated in a single structure. If this integrated unit consists of independent sub-units that correspond to a main file, it is very easy to add later components, for example, games that will improve the interactivity of the software. This main file will contain the most important menu of the product.

As time goes by, the structure, interface and routing tools should be tested continuously. This is important because it will be difficult to make changes once it is distributed.

After the software is submitted, the last phase is its distribution and evaluation. The evaluation should be done by those to whom this software is intended, and this process is not that simple. The final results should justify the investment.

4 Conclusion

Education is more successful when a methodology corresponding to the latest technology is used. Multimedia is an opportunity to accomplish that. It is a tool that helps students to extend their knowledge through interactive learning and the student needs should be actively taken into consideration. Developing educational software is a complex task that involves more disciplines and requires a number of professional of diverse backgrounds (designer, programmer, pedagogue, manager) combined into one effective team.

5 References

1. Boyle, T., *Design for Multimedia Learning*, London: Prentice Hall, 1997

2. Bearman, M., Kidd, M., Cesnik, B., *Methodology for Developing Educational Hypermedia Systems*, Centre of Medical Informatics, Morash University, Proceedings of MEDINFO, 1998
3. Barker, P., and King, T., *Computers in Education*, 1993
4. Duchastel, P.C., Structures and the Methodologies for the Evaluation of Educational Software, *Studies in Educational Evaluation*, 13, 111-117, 1987
5. Ennals, R., Overview of Computer Assisted Learning, *Journal of Artificial Intelligence*, 2(2) 5-16, 1991
6. Wilson, B., and Cole, P., *A Review of Cognitive Teaching Models*, Educational Technology Research and Development, 39(4), 47-4, 1992