Problem Solving to teach Processing Systems: Engineering learning objects based on anchored instruction

S. Impedovo, A. Ferrante, D. Impedovo, R. Modugno, G. Pirlo, E. Stasolla, C. A. Trullo

***Abstract*—** There is a great request of new and effective aids that facilitate information acquisition and knowledge construction. Researchers in the field of educational software development are continually involved in experimenting new approaches to improve students' learning capabilities. The new engineering methodologies allow developing learning objects that really support information transfer and knowledge building. This paper presents an investigation on the capacity of web-based, computer-assisted, anchored instruction to improve problem-solving skills while teaching scientific disciplines in the university environment. The experience provides empirical evidence of the usefulness of the adopted solution in developing learning objects to teach computer systems architecture to computer science students. The developed instructional software really provides a motivating, attractive and enjoyable environment. The experience has a positive impact on students, strengthening their problem-solving skills. The results suggest that web-based anchored instruction exhibits high potentials in offering useful instructional environments.

***Keywords*—** Anchored instruction, E-learning, Learning object, Problem solving, Software engineering.

I. Introduction

THE growing use of Information and Communication Technologies (ICTs) solicits a shift in ways of studying, knowing and doing. There is an always increasing interest in finding new strategies for organizing activities incorporating ICTs to enhance teaching and learning processes.

The excitement is mainly due to the recognition that digital information and communication technologies are the best tools for bridging the gap between the classroom and the real-world working conditions, between cognition and practical application, and between theory and practice.

This notwithstanding, nowadays, many students still rely upon memorization to master subjects instead of thinking and using problem-solving skills. Moreover, some teachers believe that a good approach for achieving good results rely on teacher-directed instruction and on students' practice in many related problems [1]. Therefore, instructional content is often presented to learners in simplified, de-contextualized, and isolated information chunks that encourage memorization rather than problem solving thinking. This kind of learning makes it difficult to help students to appreciate the value of the knowledge they learn. It is also hard for students to comprehend the content applicability to actual problems and meaningful situations, and to transfer learning experience to different situations [2], [3], [4].

A. *Software engineers proposals*

The educational software engineers have diagnosed these flaws and proposed several approaches as remedies.

They believe that:

- constructivist pedagogy, emphasizing student-centered rather than teacher-centered learning, can lead to a significant educational improvement;
- technology can make the difference;
- education should be reengineered by the integration of technology, pedagogy, and new curricula;
- learning to think critically, and to analyze and synthesize information for problem solving could become the new crucial educational goal.

Moreover, Dunlap and Grabinger [2] suggested that there are two main instructional issues that need to be addressed in order to make learning meaningful for students.

- The first one is to help students in applying the information they learn;
- the second one is to make the need and reason of learning clear and apparent by using examples and simulations.

Manuscript received November 29, 2008.
Revised manuscript received January 30, 2009.
This work was supported by the Italian Minister for University and Research (MIUR), and by the University of Bari.

Prof. Sebastiano Impedovo is with the "Centro Rete Puglia" - via G. Petroni, 15/F.1, Bari, Italy - and the Department of Computer Science - via Orabona, 4, Bari, Italy - of the University of Bari - Piazza Umberto I, n. 1 70100 Bari, Italy. (phone: +39 080 5575931; fax: +39 080 5575938; e-mail address: impedovo@retepuglia.uniba.it, impedovo@di.uniba.it;
web site: www.retepuglia.uniba.it/Impedovo/index.htm)

B. Anchored instruction fundamentals

By recognizing the need to find innovative, student-centered ways to design and deliver online courses, anchored instruction provides strategies for online learning software design and development. Anchored instruction allows developing context-based learning objects that encourages students to solve realistic problems in a technology-based learning environment and to acquire practical skills reusable in different environments and conditions. The software products are designed to aid:

- reflection,
- information transfer,
- critical thinking,
- and problem solving within realistic and authentic contexts [3], [4].

Indeed, anchored instruction requires putting students in the context of a problem-based story. In this way, the students 'play' an authentic role while investigating the problem [5]. In addition, anchored instruction suggests effective ways to organize learning objects to make students able to solve complex discipline related tasks. In effect, the demands of problem-solving skills, to correctly be satisfied, need to be linked to the cognitive processes of the specific learning task. Two main approaches in identifying problem-solving relevant constituent elements can be distinguished [6]:

- The factor-analytic approach attempts to identify distinct abilities as required in problem-solving tasks.
- The information processing approach tries of identifying the cognitive processes required in the problem-solving process.

Both approaches attempt to isolate various components involved in problem solving ways so that they can be examined to address. The acquired solving skills, according the Polya problem solving strategy [7], can fruitfully be spent in different activities to approach new tasks posed by everyday life, and solve new problems.

C. Paper organization

This paper, starting from the previous considerations, presents theoretical foundations and practical principles application of anchored instruction to develop a problem solving learning object for teaching processing systems to university students. It is organized as follows: Section 2 summarizes the Polya problem solving strategy; Section 3 presents the anchored instruction engineering principles; Section 4 deals with the organization of the learning experiment; Section 5 presents the problem solving learning object; Section 6 discusses the testing phase; and finally Section 7 presents discussions and conclusions.

II. Polya problem-solving strategy

According Polya [7], problem-solving consists of different phases, not necessarily performed in a sequential order since loops and backtracking can occur in the various steps of the solving activity.

A. Problem solving phases

Essentially four main phases can be distinguished:

A) understanding the problem,
B) devising a plan,
C) carrying out the plan,
D) looking back and check.

A) *Understanding the problem* seems so obvious that it is often not even mentioned, yet students are often foiled in their efforts to solve a problem simply because they do not understand it fully, or understand it partially. This phase includes labeling and identifying unknowns, condition(s) and data, and determining the solubility of the problem.

B) *Devising a plan* means drawing on prior knowledge to frame an appropriate technique. Indeed there are many reasonable ways to solve a problem. The skill at choosing an appropriate strategy is best learned by solving many problems.

C) *Carrying out the plan* is usually easier than devising the plan. In general, all what is needed is care and patience. People need to be persistent with the plan they have chosen and need to control if it continues to work. If the chosen strategy does not work, learners need also to have the capability of discarding it and of choosing another strategy.

D) *Checking the correctness of the solution* represents the final phase and allows adding the problem to one's store of knowledge for use in solving future problems.

B. Problem solving strategy factors

When implementing the solving strategy at least the following factors need to be considered:

1. Domain Specific Knowledge;
2. Algorithms;
3. Heuristics;
4. Decision mechanisms;
5. Reflection.

1. *Domain Specific Knowledge*: to become a good problem solver, one must develop a base of knowledge; problem solving abilities, beliefs, attitudes, and performance develop in contexts that must be studied and learnt; how effective one is in organizing that prior knowledge also contributes to successful problem solving activities.
2. *Algorithms*: an algorithm is a procedure, applicable to a particular type of exercise, which, if followed correctly, is guaranteed to give the answer to the exercise; also the process of creating an algorithm and applying it to a specific application can be considered a problem solving activity.
3. *Heuristics*: heuristics are kinds of information, available to students to make decisions during problem solving; they are aids to the generation of a solution, plausible in nature rather than prescriptive; in spite of their empirical determination, they seldom provide infallible guidance also if can provide variable results.

4. *Decision mechanisms*: an extensive knowledge base of domain specific information, algorithms, and a repertoire of heuristics is not sufficient during problem solving activities; the student must also construct some decision mechanism to select from among the available heuristics, or to develop new ones, as problem new situations are discovered or encountered.
5. *Reflection*: looking back is the most important part of the problem solving activity; it is the set of activities that provides the primary opportunity for students to learn from the problem. The phase consists also of checking the result; validating the argument; deriving the result differently; using the result, or the method, for some other problems; reinterpreting the problem; interpreting the result under different point of view; or stating a new problem to solve. What people learn after they have solved the problem is what really counts. Polya [8] mentions that much can be gained by taking the necessary time to reflect and looking back at what has done, paying attention on what worked and what did not. Doing this will enable the solver to predict what strategy to use to solve future problems.

III. Anchored Instruction Engineering Principles

Anchored instruction, originally proposed by the Cognition and Technology Group at Vanderbilt University [3], [4], [9], [10], aims to help students develop the confidence, skills, and knowledge necessary to solve problems and become independent thinkers. With the widespread application of multimedia technology, the ideas of anchored instruction can be better achieved. Taking advantage of the emerging multimedia computing technology, computer based learning environments can be deployed to expand the power and flexibility of learning resources.

Anchored Instruction has its roots in constructivist thinking and can be linked to the ideas of Piaget [11], [12], who proposed that humans cannot be given information which they immediately understand and use. Instead, they must construct knowledge through experience.

The major features of anchored instruction are:

- the use of problem-scenarios to elicit students' problem-solving goals,
- strategies for solving these problems,
- and the connection of knowledge with every day life.

Based on the theories of situated learning, cognitive apprenticeship and cooperative learning, anchored instruction makes it possible to provide life-like inquiry situations, in which students can easily explore the content and which facilitates the teaching of scientific concepts and allow acquiring problem-solving strategies [13], [14].

A. Anchored instructions problems

The primary focus of anchored instruction is the development of interactive learning environments that encourage both teachers and students to pose and solve complex, realistic problems.

The environment has the purpose of creating interesting and realistic contexts that encourage the active construction of knowledge.

Well formulated problems exhibit their effectiveness in promoting problem-solving abilities as well as enhancing attitudes toward scientific disciplines [3], [9], [10], [15], [16].

Studies provide also wide evidence that situations involving the use of instructional technologies are authentic, relevant, and stimulating to learners' attitudes and performances [17].

The most important design principles that need to be considered in problems formulation are:

- narrative web-based format,
- explanation with realistic problems,
- generative format,
- embedded data design,
- problem complexity,
- and link across the pages and among the related arguments.

IV. Engineering the Learning Activity

A learning object developed according anchored instruction design principles can offer to learners one or more complex problems.

To solve the problem, the students must:

- generate appropriate sub-goals,
- identify relevant information,
- eventually cooperate with other colleagues in order to plan and reach the global solution.

Learners need also to evaluate and discuss the advantages and disadvantages of possible alternative paths and compare perspectives by pointing out and explaining key factors. Indeed, the proposed problems are not trivial and require various steps to obtain the solution. All the data needed to solve the problems are embedded in the learning object.

The learning activity is effectively accomplished through many problem-solving steps:

- problem disclosure,
- problem understanding,
- learning strategies for solving the problem,
- and effectively solving the problem.

In case of success the learner can try to solve another problem. In case of failure a sequence of slides provide an explanation that enables a new attempt to reach the correct solution.

The students, in this manner, are addressed to recognize the main topics in the problems and to provide themselves with necessary scaffolding to overcome obstacles.

A. Learning activity organization

To evaluate the usefulness of the anchored instruction to teach processing systems, a specific learning object has been developed and offered to students in Computer Science at the University of Bari - Italy.

The learning activity has been organized as an experiment. The experiment consisted of a pre-activity test, a learning phase and post-activity test.

The pre-activity test was used to investigate learners' prior knowledge. The scores of this test were used as a baseline to examine whether the learning phase changed the students' problem solving skills.

The learning phase then followed. It consisted of a teaching learning object and a problem solving learning object. The reader interested in the organization of the teaching learning object can refer to [18]. The problem solving learning object was developed according to the design principles of anchored instruction. It presented specific problems on main components of processing systems and their functional organization; it challenged university student to linearly provide a solution to each one of the proposed problems to complete the learning activity.

The learning object, published according to the AICC/SCORM standards [19], [20], imported in the Oracle iLearning platform [21], [22] and offered to learners in computer science [23], [24], was available via web and accessible through a portal [25]. The product could be used from classrooms, through a dedicated e-learning infrastructure [26], or from home through domestic personal Internet connections.

After the completion of the learning objects, all the students were given a post-activity test to evaluate their profits and a survey [27] to investigate how they judged the learning object. The survey allowed not only assessing student perceptions of the experience, but also investigating the attitudes enhancement in problem solving while approaching problems on processing systems.

V. The Problem Solving Phase

After a brief introduction (Figure 1) to the subject, the learning object drives the learner into the theme of the presentation:

problems on processing systems
architecture, organization and management.

A voice speaking in background also illustrates the strategy adopted in the learning environment to reach the final objective:

choice of the correct solution
among the various alternative proposed ones.

Figure 1: Introductory page

Three problems are sequentially presented:

1. A problem on the average information access time into main memory;
2. A problem on the cost of transferring information between main memory and I/O devices;
3. A problem on the cost of I/O devices management.

Each problem is organized as follows:

- the problem is formulated,
- necessary conditions and operation constraints are declared,
- and possible alternative solutions are proposed.

A. *The problem on information access time*

The first problem deals with calculating the average CPU access time to the main memory to retrieve a chunk of information in a system provided with cache.

The cache has a hit ratio, that is to say the probability of containing the requested information, of the 95%; the average access time to the cache is 100 nsec, while 800 nsec are required to access the random access memory (RAM).

The problem requires (Figure 2) the calculation of the average access time to retrieve information in the main memory. Four possible time values are proposed 125, 135, 145, or 155 nsec; among them only one corresponds to the correct value (135 nsec).

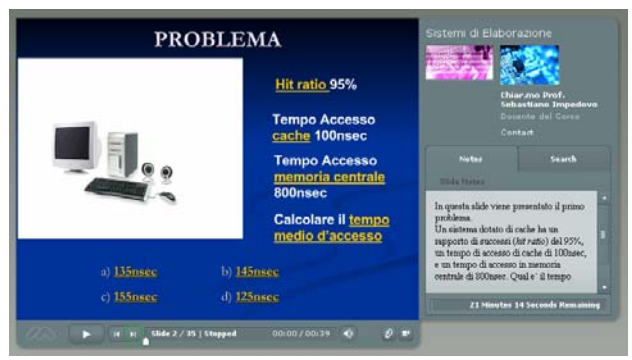

Figure 2: First problem presentation

A.1 Wrong choices management

If the learner selects one of the wrong solutions, the learning environment continues with the explanation of the fundamental concepts of main memory access techniques and associated costs.

It illustrates the concepts related with: the hit and the miss; the hit ratio and its complement, the miss ratio; the different types of memory systems, and their hierarchic tree organization; the properties, organization and addressing system of the cache (Figure 3); the motivation, peculiarities, functions and position of the various types of cache; the properties and contents of the RAM; the different storage devices access mechanisms; the different speed of registers, cache and RAM; and the information locality principle.

Figure 3: A page illustrating memory hierarchy

At the end, the learning environment explains how to calculate the hit time (the time necessary to access the nearest level of memory with success in information retrieval) and the miss time (the time necessary to access the following levels of memory in case of failure in retrieving information in the nearest level). The formula to calculate the average total access time is presented and commented. The specific calculation is not performed and the learner is invited to repeat the solving activity to reach the goal (Figure 4).

Figure 4: First problem solution explanation

A.2 Correct choice management

If the learner selects the correct solution (135 nsec) the system immediately confirms it.

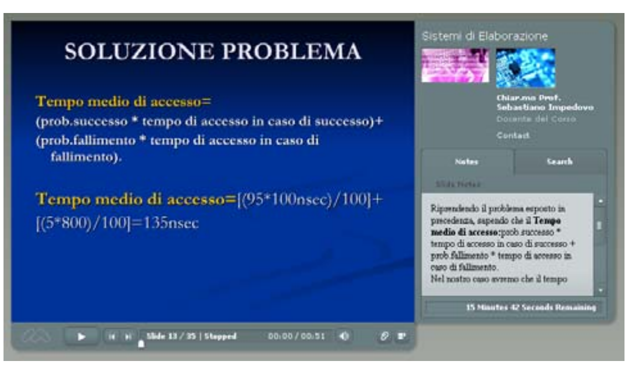

Figure 5: First problem numerical solution

The concepts related with the average information access time are presented and commented, and the formula (expressed as probability of success in finding the information in the nearest level of memory multiplied with the average time to access the nearest level of memory plus the probability of failure in finding the information in the nearest level of memory multiplied with the average time to access the following levels of memory) to calculate it is explained. A possible sequence of calculations to reach the particular solution is finally presented (Figure 5).

If the learner randomly selected the correct choice, the learning environment provides the essential elements to understand the value associated with the correct solution.

B. The problem on I/O techniques

The second problem investigates the efficiency of different techniques in transferring information between I/O devices and main memory (DMA, Interrupt-driven I/O, and Programmed I/O).

The learner is invited to compare the costs associated to the proposed I/O techniques and to select the one that allows the fastest termination of the process under consideration.

The CPU operates at 1 GHz and is provided with a 32 bits bus; the amount of information to transfer is equal to 64 MB; only blocks of information can be transferred, and each block contains exactly 64 Bytes; the CPU, when involved in information block transfer, needs 200 clock cycles to move a block.

The access time to the main memory is 300 clock cycles when programmed I/O is considered, and 400 clock cycles in the case of interrupt driven I/O. The DMA allow transferring 6400 words each time, but requires 500 clock cycles to be activated and 800 clock cycles to be terminated (Figure 6).

Figure 6: Second problem formulation

To solve this problem the learner needs to consider each one of the possible solutions, make all calculations and compare the obtained results to select the best I/O technique.

When, at the end of the calculations, the student tries to provide the solution, the environment firstly provides the necessary explanations related with the selected choice; then shows an example of the specific calculations by considering the given values; and finally comments the results by providing a measure of efficiency of the selected choice.

If the learner selects the correct solution, he is also informed about the success.

B.1 Programmed I/O selection

If the programmed I/O is selected, the environment explains that all I/O operations are handled by the CPU; that a program instruction is required to perform an information transfer; and that only atomic information can be moved. The CPU firstly checks and manages the I/O interface module by using control and test instructions and then performs transfer operations by means of read-write instructions.

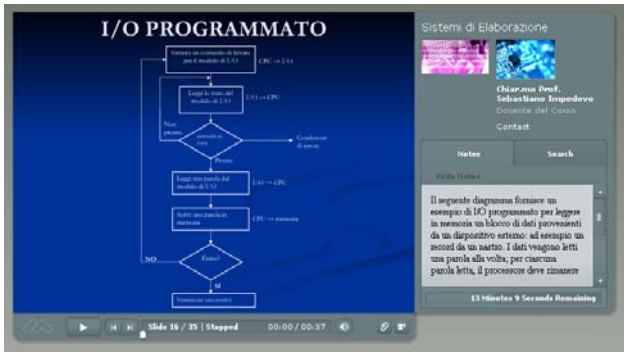

Figure 7: Programmed I/O flow chart

A flow chart (Figure 7) finally shows that the CPU is engaged in long busy waiting cycles during which no other operation can be performed by the entire processing system.

The calculation remarks that 1 Mega block transfer operations are requested to move 64 Mega Bytes of information organized in blocks of 64 Bytes. The CPU load is equal to 200+300 clock cycles in each block transfer (200 for CPU involvement and 300 for memory access). A total amount of 500 Mega clock cycles are requested to complete 1M block transfers (Figure 8).

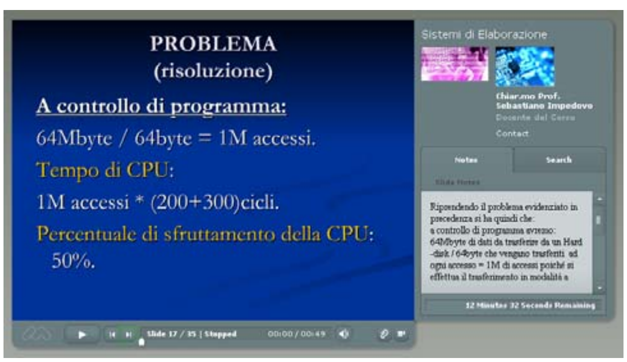

Figure 8: Programmed I/O calculations

B.2 Interrupt Driven I/O selection

If the Interrupt Driven I/O is selected, the environment shows how this technique operates. The slides show the properties associated with this technique, how information transfers are activated and terminated by I/O control devices and how the CPU is discharged from testing I/O devices (Figure 9). A flow chart shows the specific task sequence and points out that, also if this technique is expensive for each single process, it enables multiprogramming, allowing many processes to advance, at the same time, in the same system.

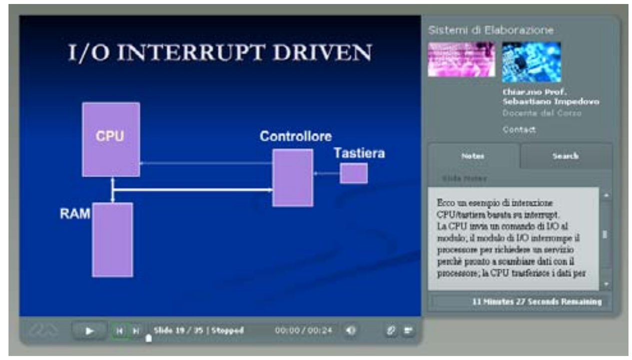

Figure 9: Interrupt driven I/O schema

The calculations shows that also in this case 1 Mega block transfer operations are requested to move 64 Mega Bytes of information in blocks of 64 Bytes. The CPU load is equal to 200+400 clock cycles in each block transfer (200 for CPU involvement and 400 for memory access). So a total amount of 600 Mega clock cycles are requested to complete 1M block transfers (Figure 10).

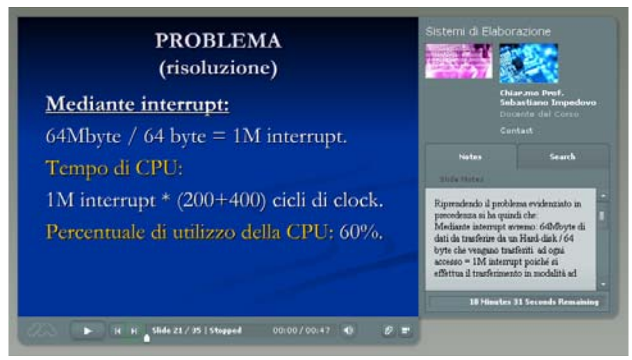

Figure 10: Interrupt driven calculations

B.3 DMA selection

If the DMA is selected, the learning environment shows that this technique allows directly transferring data between main memory and I/O devices. The principles regulating the specific information transfer method are presented.

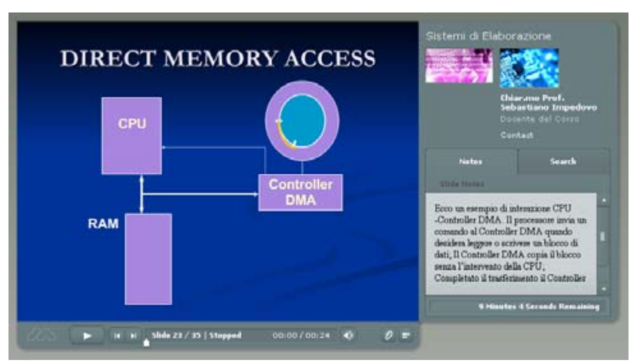

Figure 11: Direct Memory Access schema

The steps to activate and terminate I/O operations are illustrated; the way in which the information is transferred by using the shared bus is explained, the relatively low CPU charge is commented (Figure 11).

The principal parameters associated with a DMA command (operation request, I/O device addressing, starting location in main memory, number of words to transfer) are described. The cost of this solution is finally evaluated.

The calculation (Figure 12) shows that in this case to transfer 64 Mega Bytes of information, 2500 DMA transfer operations are necessary, each one transferring 6400 words (32 bit wide). Each DMA requires 500+800 CPU clock cycles to be activated and terminated. The total CPU involvement in opening and closing phases is 500+800=1300 clock cycles that, multiplied by the 2500 DMA transfer operations, results in a total number of 3250000 clock cycles, that is to say 3.25 M (1 M = 10^{-3} G) clock cycles.

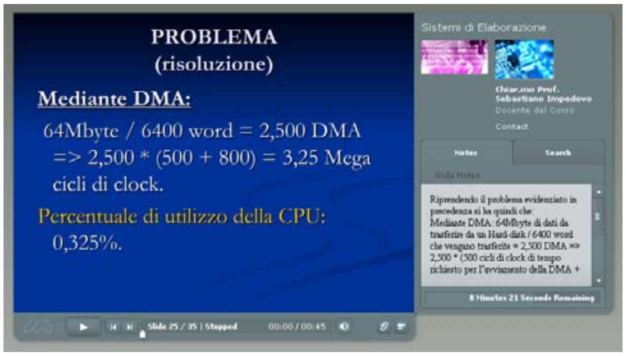

Figure 12: Direct Memory Access calculations

B.4 The problem solution

Comparing the three partial solutions, it results that the DMA is the less expensive technique since it requires only the 3.25% of the CPU time, while the other two techniques requires respectively the 50% and the 60% of the CPU time.

C. The problem on device management

The third problem considers the CPU charge to manage three I/O devices (a printer, a keyboard, and a display) while executing a heavy background user task (Figure 13).

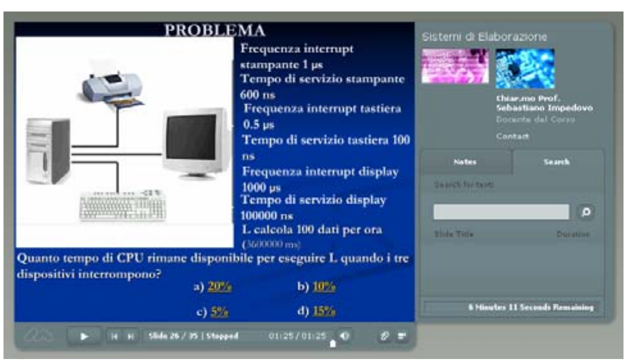

Figure 13: Third problem formulation

Each I/O device has specific interrupt frequency and service time. The printer has an interrupt frequency f_1 equal to 1 µsec and a service time t_1 equal to 600 nsec. The keyboard has f_2 equal to 0.5 µsec and t_2 equal to 100 nsec. The display has f_3 = 1000 µsec and t_3 = 100000 nsec. The problem asks for calculating the percentage of time that the CPU can spend to run the user program. The learner can select among four possible solutions (5%, 10%, 15% and 20%). Only a choice corresponds to the correct value (10%).

Figure 14: The instruction cycle with interrupts

C.1 Wrong choices selection

If the learner selects a wrong choice, the learning environment explains: the concept of interrupt (an external service request that can interrupt the regular CPU processing flow); the reason of its introduction (to increase the CPU processing efficiency); the different interrupt types (program, timer, I/O, and hardware error); the interrupt management method (a specific cycle introduced at the end of the instruction cycle) (Figure 14); and the pseudo-code trace sequence (a jump, at the end of an instruction, if an interrupt is waiting for being served, to the specific interrupt service routine) (Figure 15).

Figure 15: The management of interrupts

At the end of explanations, the device interrupt service time is presented and formalized (Figure 16). Finally, the learner is invited to repeat calculations to select another choice.

C.2 First path to the correct solution

When the learner selects the correct choice the environment shows that there are at least two possible paths to reach the goal and presents them.

The first one consists: in calculating the fraction of time, expressed in seconds, requested by each I/O device to be served, in summing partial time fractions to obtain the total time service fraction, and in subtracting the obtained value to the 100% of the unit time. In a second the printer interrupts 10^6 times and each time requires $600*10^{-9}$ sec. Its service time is $600*10^{-3}$ sec = $6*10^{-1}$ sec = 0.6 sec. In a second the keyboard interrupts $2*10^6$ times and each time requires $100*10^{-9}$ sec. Its service time is $200*10^{-3}$ sec = $2*10^{-1}$ sec = 0.2 sec. In a second the display interrupts $10^{-3}*10^6$ times and each time requires $100000*10^{-9}$ sec. Its service time is $10^{-3}*10^6*10^5*10^{-9}$ sec = 10^{-1} sec = 0.1 sec.

Figure 16: The interrupt time

The total service time fraction is 60%+20%+10%=90%. The time that can be dedicated to the background process is only 10%, since the remaining part is reserved to interrupts.

C.3 Second path to the correct solution

The second path shows that the solution can be reached: by calculating the amount of time spent by all the three I/O devices in an hour, by subtracting the obtained service times from the 1 hour total time, and by calculating the fraction of free of service time that can be assigned to the user task.

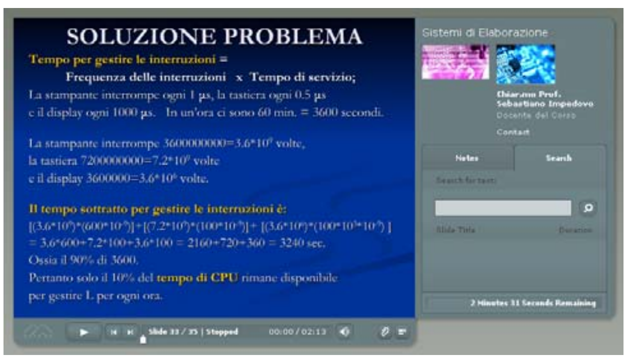

Figure 17: Example of problem solution

An hour consists of 3600 seconds. The printer in an hour spends $3600*10^6*600*10^{-9}$ =2160 sec. The keyboard in an hour spends $3600*2*10^6*100*10^{-9}$ =720 sec. The display in an hour spends $3600*10^{-3}*10^6*10^5*10^{-9}$ =360 sec (Figure 17). The total service time is 2160+720+360 = 3240 sec. The time that can be spent to let run the background process is equal to 3600 – 3240 = 360 sec. Compared with 3600, 360 equals the 10% of the available time.

D. *The end of the learning object*

At the end, when the learner correctly completes the three problems, the learning environment shows a congratulation page (Figure 18).

Figure 18: Congratulation page

VI. The Testing Phase

The testing phase has been organized as a pre-activity test and a post-activity test.

The pre-activity test contains only question on the fundamental concepts treated in the problems. The answers to these questions are used as base values of the following phase.

The post-activity test contains, beside the fundamental concept questions, also some stand-alone subtasks extracted from the problems proposed in the learning object. The analysis of the post-activity tests allows understanding not only the gain in the level of comprehension of concepts, but also to put in evidence the problems related to errors in calculations. It allows also discriminating the subtask(s) that posed particular difficulties.

A. *Fundamental concepts in the pre-activity test*

Among the fundamental concepts related with the first problem, the test investigates:

- the hit, the miss, the hit ratio, the miss ratio;
- the CPU speed, the cache speed, the main memory speed; the memory hierarchy;
- the cache access time, the main memory access time, the average information access time.

Among the fundamental concepts related with the second problem, the test investigates:

- the bit dimension, the byte dimension, the word length, the relation between words and bus size in system architecture;
- the relation between MB and GB, the relation between MHz and GHz;
- the clock cycle;
- the I/O test instructions, the I/O control instructions, the I/O operation instructions;
- the programmed I/O, the Interrupt driven I/O, the DMA.

Among the fundamental concepts related with the third problem the test investigates:

- the interrupt, the interrupt types, the interrupt frequency;
- the instruction cycle, the interrupt cycle, the service time, the processing time;
- the µsec, the nsec, the msec definitions;
- the relation between instruction cycles and interrupt

cycle, the relation between interrupt cycle and machine clock cycle;
- the relation between frequency and number of interrupts.

B. Subtasks in the post-activity test

Subtasks related to the first problem require performing the following calculations:

- Average cache access time to retrieve certain information, given specific probability of success in finding information in the cache and a pre-determined cache access time;
- Average access time to successive memory levels to retrieve certain information, given a certain probability of failure in finding information in the cache and a specific access time to the main memory;
- Hit ratio estimation, given the number of times, over a total number of cases, in which the information is retrieved in the cache;
- Miss ratio estimation, given the number of times, over a total number of cases, in which the information is retrieved in the cache;
- Miss ratio estimation, given the hit ratio.

Subtasks related to the second problem require performing the following calculations:

- I/O programmed transfer time, given: the total amount of information to transfer, the block size, the CPU involvement and the device access time;
- I/O interrupt driven transfer time, given: the total amount of information to transfer, the block size, the CPU involvement and the device access time;
- DMA transfer time, given: the total amount of information to transfer, the DMA buffer size and management time;
- Number of data transfer operations, given the total amount of information to transfer and the block size;
- CPU load, given its operation frequency and number of clock cycles spent to manage the interrupts;
- Numbers of DMA transfers, given the total amount of information to transfer and the DMA buffer size;
- Number of words contained in a piece of information, in a processing system with a certain bus size;
- Total number of clock cycles, given the number of transfer operations and the CPU involvement.

Subtasks related to the third problem require performing the following calculations:

- Number of interrupt, given a frequency of interrupt;
- Number of μsec in a given number of nsec;
- Number of μsec in a given number of msec;
- Number of nsec in a given number of msec;
- Number of interrupt in an hour, given the per second interrupt frequency;
- The total amount of interrupt time in an hour, given the number of interrupts and the associated per interrupt service time;
- The total amount of interrupt time in an hour, given the interrupt frequency and the fraction of time unit (sec) required to serve an interrupt.

VII. Discussions and Conclusions

The purpose of this investigation was to conduct an empirical study on the effects of web-based anchored instruction on problem-solving skills while teaching/learning processing systems.

The learning object engaged students in planning for problem solving and in focusing attention on gathering the needed information. The students were led to individualize the different sub-problems hidden in the problems, to generate the solution for all of the identified sub-problems, to relate the partial solutions to reach the overall solution.

The obtained results show that problem-solving activities can effectively be used in university learning activities. The experience provided empirical evidence of the usefulness of anchored instruction in teaching computer systems architecture in the university environment. All the learners benefited from the effects of the learning object, developed following anchored instruction principles, and their problem-solving attitudes and capabilities enhanced significantly.

The anchored instruction web-based learning environment really provides a motivating environment that result highly attractive and enjoyable, and the experience has a positive impact on students, strengthening their problem-solving skills.

The results show that learning scientific disciplines such as "*processing system architecture and management*" can effectively be more pleasant and can be much more understood by students through the adoption of an instructional design that stimulates attention and participation.

The learning environment effectively affords students capabilities to understand problem structure and to provide suitable solutions. The experience shows how web-based technology coupled with anchored instruction, can be fruitfully integrated in university teaching curricula.

References

[1] S. Cheng, W. Chang, W. Ching, C. Guo, Development of a professional development program for science and scientific disciplines teacher - an action research, Annual meeting of the National Association for Research in Science Learning, San Diego, California, USA, 1998.
[2] J. Dunlap and R. Grabinger, Rich environments for active learning in higher education classroom, in B. Wilson (ed) Constructivist Learning Environment: Case Studies in Instruction Design, Educational Technology Publication, NJ, 1996, pp. 65–82.
[3] Cognition and Technology Group at Vanderbilt, Anchored instruction and its relationship to situated cognition, Educational Researcher 19, 1990, 2–10.
[4] Cognition and Technology Group at Vanderbilt, Anchored instruction and situated cognition revisited, Educational Technology, 33 (3), 1993, pp. 52–70.
[5] PennState College of Education, 2002, Anchored Instruction, http://www.ed.psu.edu/nasa/achrtxt.html
[6] M. Wu, R. Adams, Modeling Mathematics Problem Solving Item Responses Using a Multidimensional IRT Model, Mathematics Education Research Journal, 2006, Vol. 18, No. 2, pp. 93-113.
[7] G. Polya Scientific Disciplines Discovery: On Understanding, Learning, and Teaching Problem Solving, Wiley, New York, 1957.
[8] G. Polya, In the classroom, in How to solve it, Doubleday and C.ny Inc., Garden City, NY, 1957.

[9] Cognition and Technology Group at Vanderbilt, Technology and the design of generative learning environment, Educational Technology, 31, 1991, pp. 34–40.
[10] Cognition and Technology Group at Vanderbilt, The Jasper Series as an example of anchored instruction: theory, program description, and assessment data, Educational Psychologist 27 (3), 1992, pp. 291–315.
[11] J. Piaget, R. Garcia, L. Banks, P. M. Davidson, J. A. Easley, Toward a Logic of Meanings, Lawrence Erlbaum Associates, 1991, 178 pages.
[12] J. Piaget, B. Evert, Mathematical Epistemology and Psychology, D. Reidel Publishing Company, Dordrecht-Holland, 1966.
[13] J. Bransford, N. Vye, A perspective on cognitive research and its implications for instruction, in Resnick L and Klopfer L (eds) Toward the Thinking Curriculum: Current Cognitive Research, ASCD, Alexandria, VA, 1989, pp. 173–205.
[14] P. Cobb, E. Yackel, T. Wood, A constructivist alternative to the representational view of mind in scientific disciplines, education Journal for Research in scientific disciplines Education, 23, 1992, 2–22.
[15] D. Hickey, J. Pellegrino, S. Goldman, N. Vye, A. Moore and CTGV, Interest, attitudes, and anchored instruction: the impact of one interactive learning environment, American Educational Research Association, Annual Meeting, Atlanta, GA, 1993.
[16] J. Van Haneghan, L. Barron, M. Young, S. Williams, N. Vye, J. Bransford, The Jasper series: An experiment with new ways to enhance mathematical thinking, in D. Halpern (ed) Enhancing Thinking Skills in the Science and scientific disciplines, Lawrence Erlbaum, Hillsdale, NJ, 1992, pp.15–38.
[17] M. Simonson, N. Maushak, Situated learning, instructional technology, and attitude change, in H. McLellan (ed) Situated Learning Perspectives, Educational Technology Publications, Englewood Cliffs, NJ, 1996, pp. 225–242.
[18] S. Impedovo, S. Campanella, G. Dimauro, A. Ferrante, D. Impedovo, R. Modugno, G. Pirlo, L. Sarcinella, E. Stasolla, C. A. Trullo, "Developing Situated Learning Teaching Courses: A Practical Experience at the University of Bari", Proceedings of the 5th WSEAS / IASME International Conference on ENGINEERING EDUCATION (EE '08), Heraklion, Crete Island, Greece, July 22-24, 2008, WSEAS publishing, pp. 226-231.
[19] http://www.imsglobal.org/
[20] http://www.aicc.org/
[21] http://www.oracle.com/applications/human_res ources/ilearning.html
[22] http://ilearning.oracle.com/
[23] S. Impedovo, G. Dimauro, A. Ferrante, N. Greco, M. G. Lucchese, R. Modugno, G. Pirlo, L. Sarcinella, The PROTEO Project: New Advances in e-Learning Activities at the University of Bari, WSEAS Transactions on Communications, Issue 1, Volume 5, Jan. 2006, WSEAS press, pp. 23-30.
[24] G. Dimauro, D. Impedovo, R. Modugno, A LMS to Support e-Learning Activities in the University Environment, WSEAS Transactions on Advances in Engineering Education, Issue 5, Volume 3, May 2006, WSEAS press, pp. 367-374.
[25] http://www.web-learning.uniba.it
[26] D. Impedovo, M. G. Lucchese, R. Modugno, Dedicated e-Learning Infrastructure in a Metropolitan Academic Network, WSEAS Transactions on Advances in Engineering Education, WSEAS press, Issue 2, Volume 3, February 2006, pp. 80-85.
[27] S. Impedovo, S. Campanella, G. Dimauro, A. Ferrante, D. Impedovo, M. G. Lucchese, R. Modugno, G. Pirlo, L. Sarcinella, E. Stasolla, C. A. Trullo, Quality Enhancement in E-Learning Activities: Improvements by mean of a Newly Engineered E-Learning Survey, WSEAS Transactions on Advances in Engineering Education, Issue 4, Volume 5, April 2008, WSEAS press, pp. 242-251.

Prof. Sebastiano Impedovo was born in Putignano (Bari-Italy) on May 17, 1947. He obtained his degree in Physics with honours at the University of Bari in 1972. He became Assistant Professor in Electronics, then Associate Professor in Cybernetics in 1981, and in 1987 he became Full Professor in Operating Systems at the University of Bari.
He is an IAPR Fellow and a member of IAPR, ACM, IEEE, IGS, ANIPLA and AICA. He has published more than three hundred papers and seven books in the field of handwriting recognition and intelligent systems for document analysis. He is a member of the editorial board of the International Journal of Pattern Recognition and Artificial Intelligence, and of the International Journal of Document Analysis and Recognition.
Sebastiano Impedovo organized several international conferences, workshops, schools and international panels on Image Analysis and Document Processing, Tele-teaching, Tele-working and E-learning.
Sebastiano Impedovo was also the Director of the Department of Computer Science of the Bari University, the President of the Degree Course in Computer Science and a Member of the "Senate" of the Bari University; he was the Coordinator the PhD Course in Computer Science, the Director of the Bari Unit of the National Inter-University Consortium in Computer Science, a Member of the Scientific Board of the “Tecnopolis” Consortium and the Chairman of the Council of Department Directors of the University of Bari. Now he is the President of the Inter-Faculty Centre “Rete Puglia” of the University of Bari.