Problem Resolution Processes in Computer Science Teaching

JOSÉ FRANCISCO ZELASCO¹, GUSTAVO LÓPEZ²
JOSE MANUEL CABO HERÁNDEZ³, JUDITH DONAYO⁴
Departamento de Computación, Facultad de Ingeniería
Universidad de Buenos Aires
Av. Paseo Colón 850, CABA, Argentina¹,²,⁴

Departamento de Didáctica de las Ciencias Experimentales,
Facultad de Educación y Humanidades, Campus de Melilla, Universidad de Granada
Ctra/ Alfonso XIII s/n Melilla 52005³

¹jfzelasco@gmail.com  ²glopez@fi.uba.ar  ³jmcabo@ugr.es  ⁴juddonay@gmail.com

Abstract: the resolution of problems in the field of computer science is based on cognitive processes whose goal is to find a way out to a difficulty, a path around an obstacle in order to obtain a particular result that is not easily attainable. Thus, some authors define thinking as problem solving. Others claim that the learner discovers a combination of previously learned rules, which may be applied to achieve the solution to new situations. This paper deals with this topic and presents a number of processes applicable to the resolution of problems in computer science.

Key-Words: Problem Solving in computer science, resolution of problems, Solution, Problematic Situations, Resolution Processes, Thought, Abstract Thinking in Computer science.

1 Introduction

For many authors solving problems as a cognitive activity is inherent of thought. In all his works, Mayer indiscriminately uses the words "thinking, cognition and problem solving", taking into account the following aspects: thought is cognitive, but is inferred from behaviour [1]. It happens internally and must be inferred indirectly, the thought is a process that involves some manipulation of, or a set of operations on, knowledge in the cognitive system. And finally, in the field of computer science thought is directed and results in the "resolution" of problems or leads towards the solution. Thus Mayer argues that: "Thinking is what happens when a person solves a problem."

Garret presents the dissimilarity between problems and puzzles, in the first case there may be no solution while in the second there is one. [2]. Then, he concludes:

"Problem solving can be seen as an element of thought but it is probably more appropriate to consider it as a complex learning activity that involves thinking."

Perales Palacios claims that "a problem generally is any planned or spontaneous situation that produces, on the one hand, some degree of uncertainty and, on the other, a behaviour aimed at finding a solution" [3].

Shaw then characterized the ability to solve problems as an integration of interpreting data, controlling variables, defining operationally, and formulating hypotheses [4]. This characterization is particularly useful and applicable to solve problems in the field of computer science.

It can be observed what the drawbacks are when trying to characterize the concept of problem solving, taking into account that only a small set of studies on the topic were mentioned. In the origin of the problem lies a conflict itself [5].

2 Formulation of the Problem

To solve a problem, students of computer science have to build mental models that are appropriate to predict or explain situations. The acquisition of such models is not a task to be carried out trivially, this is because it is based on a set of basic previously learned skills. Human beings tend to develop mental models of what surrounds them and to relate this world with their own language, see Safir-Whorf hypothesis, which states that what we think and what we are able to think is mainly determined by the language we speak. In addition, we relate our
language and the perception of reality through our mind. Thus, the models that computer science students have on a particular concept, algorithm or description are generally inconsistent, inaccurate and incomplete. Therefore, we consider that it is necessary to provide them with tools to make these temporary models evolve so that they lay the basis for generating their own mental models to solve problems in the field of computer science.

3 Solution of the Problem
After considering some aspects involved in the resolution of problems in computer science, a number of specific processes to be applied are set out below.

3.1 Abstraction: The information contained in a given concept or in an observable phenomenon should be reduced in order to retain only the important information in a given context. The concept or idea, for example of a chair, comes from the process of comparing different furniture and extract those similar characteristics. Thus, we can leave aside certain features of the set of objects and keep only those that they have in common. This example clearly illustrates the concept of abstraction which derives from the Latin root abstrahere which means to separate.

3.2 Divide and conquer: The process aims at breaking or dividing a problem into a set of smaller problems in order to have a better understanding of the problem studied. This proposal is based on the Latin proverb 'Divide et impera' used by the Romans as a rule of government. Three steps are suggested when applying this process:

- Divide: Divide the problem into two or more sub-problems.
- Conquer: If the subdivision into sub-problems results in a set of solvable problems, they will be solved.
- Combine: It will combine the solution of each sub-problem in order to obtain the solution of the original problem

3.3 Analogy: This proposal is based on a cognitive process to transfer information or meaning of a subject (the analogue) to another particular subject (the target). An example of the use of analogy in the resolution of problems could be the model of the atom presented as an analogy of the solar system model.

3.4 Brainstorming: This is a group or individual technique whose goal is to find a conclusion for a given problem by generating a list of ideas spontaneously contributed by the members of the working group [6]. It is worth mentioning that though the technique is individual, it is recommended to use it in groups. This technique is based on two pillars: the first one, called 'deferred judgement', consists in retaining any judgment until the end of the session; the second one, called 'reaching the number', is focused on the number of potential solutions and not in the quality of them until the end of the session. The aim of this technique is to stimulate the generation of ideas, the group creativity and to reduce potential social inhibitions within the group. In the present work, it is proposed the use of this technique on the resolution of problems in computer science.

3.5 Analysis of means and ends: It is proposed to use this process to solve problems in the field of artificial intelligence as a heuristic method based on a search conducted by the discovery of differences between the current stage and the final stage that would lead to the resolution of the problem [7] [8]. The steps that are applied in the heuristic method “aims and ends” are:

1- set the difference between the current stage of the problem and the final stage that would lead to the solution. If there is no difference, it means that the problem was solved, otherwise, it should be continued.
2- select an operator that make it possible to reduce the differences.
3- apply the operator if this can be applied, otherwise, set a a sub-goal that admits the application of the operator. Apply again the analysis means-ends to the sub-goal until the operator can be applied or until it is decided not to apply it.
4- go back to the first stage.

3.6 Morphological Analysis: This process is particularly proposed to generate a big amount of ideas in a very short period of time. It is based on the decomposition of a concept or problem into its essential or structural elements, leading to the solution of it by analyzing the component parts. It is divided into three parts: analysis, combination and morphological search
For each of these steps can be performed, the following steps must be followed:

- Specify the problem to be solved.
- Select the attributes or parameters that comprise it.
- Analyze the possible alternative for each parameter.
- Perform combinations.

3.7 Lateral Thinking: "Lateral thinking" is proposed as a process for the resolution of problems of computer science because it involves an indirect and creative approach focused on non-obvious reasoning which includes those ideas that can not be used with traditional logic. Therefore, this is divergent because it is not restricted to a single dimension, but it moves in multiple and simultaneous dimensions. De Bono argues that lateral thinking aims at destroying patterns and at obtaining a set of processes that generate new ideas by structuring the concepts available in the mind [10] [11] [12]. This kind of thinking dismantles schemes, makes positions flexible, removes prejudices and creates new connections in the mind.

3.8 Root Cause Analysis (RCA): This methodology is proposed for solving problems in computer science focusing on the assumption that problems are best solved by attempting to correct or eliminate the root of the problem, as opposed to trying to solve its evident symptoms. When corrective measures to root causes are taken, it is expected that the chances of recurrence of the problem are minimized. However, it is known that total prevention of the recurrence of a single intervention is not always possible. Thus, the technique 'root cause analysis' can be considered as an iterative process and could be used as a tool for continuous improvement. This method is a reactive technique for the detection of problems and solutions. This means that the analysis is performed after an event has taken place. It may result in a pro-active process, which means that it could predict the probability of an event. Generally, this technique is connected with three essential questions:

- What is the problem?
- Why did it happen?
- What will be done to prevent it?

There are different schools of root cause analysis, but all of them use certain basic principles that include:

- taking measures for improving the causes or roots of a problem is more effective than treating its symptoms,
- RCA must be systematically performed to be effective, and its conclusions should be supported as documentary evidence,
- usually there is more than one potential cause of a problem,
- such analysis must establish all relationships between the causes and the problem.

In this technique, some tools can be used such as barrier analysis, tree factor cause analysis, Pareto analysis, Ishikawa diagram, etc..

3.9 Trial and Error: This heuristic approach to get either procedural or propositional knowledge is one of the fundamental pillars to solve problems in Computer Engineering. This method is based on testing an alternative and verifying if it works. If it works, a solution to the problem is achieved. If this test does not solve the problem, it will be another erroneous result and another alternative should be tried. Usually, there is a variant of this method when you have information beforehand that one alternative will solve the problem, so the alternative that has more chances to solve it will be the first to be tested. If there is no prior information, all the alternatives will be tested randomly. This method can be recognized in many examples in nature, for example, the natural selection has applied it for hundreds of thousands of years. In nature, a gene is modified randomly and a result represents a new species, if this mutation is successful the species will survive, and otherwise, the species will perish. Therefore, it can be said that nature takes knowledge through trial and error.

3.10 Reduction: This methodology is particularly proposed for the Theory of Computation and the Theory of Complexity. Basically, it deals with the transformation of a problem into another one. It may happen that, depending on the transformation used, one problem may lead to a set of problems. This
technique works as follows: a particular problem 'X' is reduced to a problem 'Y' if there is any way to transform a solution for 'Y' in a solution for 'X' so that, if a solution for 'Y' is found, it will also be found for 'X'. This method is widely used in problems closely related to mathematics.

3.11 **Hypothesis Test:** This process is proposed for the resolution of problems in Computer Engineering by applying four key steps:

1. Consider options or possibilities for solving a problem.
2. Select the best explanation (develop a hypothesis).
3. Test it by conducting a series of tests or experiments.
4. Develop a conclusion or theory.

3.12 **Seven disciplines to solve problems:** This methodology is proposed to solve problems in engineering and particularly in quality assurance processes. A series of considerations must be taken into account to apply it [13].

1. Forming a team of experts on the topic covering all functions.
2. General definition of the problem.
3. Implementation and verification of an action of provisional containment.
4. Identification and verification of a root cause (RCA).
5. Determination and verification of permanent corrective actions.
6. Prevention of the re-occurrence of the problem and/or its root cause.
7. Recognition of the team's efforts.

3.13 **GROW Model:** a Ingeniería de Software. A este modelo se lo adjudican varios autores: Graham Alexander, Alan Fine, Sir John Whitmore [14] [15]. This methodology is proposed specially in the area of Software Engineering. Many authors deal with this method: Graham Alexander, Alan Fine, Sir John Whitmore [14] [15]. This proposal considers:

**Goal:** This is the point of arrival.

**Reality:** This point shows us how far we are from reaching the point of arrival.

**Obstacles:** There may be several obstacles to reach the point of arrival which must be clearly identified.

**Options:** Once obstacles have been defined, the course of action to reach our point of arrival must be decided, and these will be our options.

**Way Forward:** To achieve our point of arrival, the options must be put into action.

3.14 **DCA Technique:** This methodology is especially proposed as a management method of control and continuous improvement of processes and products. Four steps will be considered for its application:

1. **Plan:** Objectives and processes necessary to obtain the expected results are defined at this stage.
2. **Do:** A plan is implemented, processes are executed, and products are made. Data are collected for further analysis.
3. **Check:** Results obtained are studied and compared with those expected results so as to adjust the differences. Moreover, deviations are also considered in the implementation of the plan.
4. **Act:** If necessary, corrective actions are taken and the differences to determine the cause of the problem are analyzed. With these data, it is determined where changes to improve processes and products are to be applied.

4 **Conclusion**

1. The solution of problems from the perspective of the students of Computer Science is a complex competence closely related to situations that require skills involving human intelligence. Since the resolution of problems associated with the design of software and data structure conceptualization is intimately linked to how reality is represented by a model and also how the solution adapts to a computable model easy to implement. It should be noted that the greater the degree of abstraction, the shorter the semantic gap with the real problem is, so the increase of that semantic gap must be controlled, since the final physical or operational solution will be arrived when the level of abstraction is reduced.

2. Therefore, the resolution of problems in the field of Computer Science requires a series of processes that allow students to build the solution through a model of reality which must consider the other restrictions.
associated with the problem, such as the efficiency of the resolution, the performance in terms of time, etc. [16]. Another point to be considered is that, unlike other disciplines (such as Physics, Mathematics and Chemistry), problems in the field of Computer Science can be solved in different ways even starting at the same point; this implies that the same problem could be solved in different forms. Notwithstanding this and the particularity of these methods for solving problems in the field of Computer Science, it is also important to highlight that Science Education Institutions have been demanding for a long time a program with an orientation in Science-Technology-Society to give higher importance to the formation of future citizens and to their access to scientific and technological information in order to achieve greater social participation in techno-scientific decision making [17].

3. In future works, issues in the field of computer engineering and even beyond, in the field of computing in general should be addressed, where new and unexplored paths that are on the edge of current scientific knowledge are constantly appearing. New fields of knowledge are being defined and new application areas of computing are being emerged. Also, models of processes for the resolution of problems in new areas should be proposed in future works.

**References:**