Defining computer science

Article *in* ACM SIGCSE Bulletin · June 2000

DOI: 10.1145/355354.355379 · Source: DBLP

1 author:

James W. McGuffee
Christian Brothers University, Memphis, United States
25 PUBLICATIONS **129** CITATIONS

Defining Computer Science

James W. McGuffee
CIS/CSC - Riverside Campus
Austin Community College
1020 Grove Boulevard
Austin, Texas 78741 USA
<jmcguffee@acm.org>

Abstract

This paper explores the use and purpose of a definition of computer science from the perspective of an undergraduate student. In order to gain access to the topic, the nature and purpose of definitions are explored. Historical examples of computer science definitions are given. The paper concludes with an examination of how students define computer science and how we should use these definitions in computer science education.

1. Introduction

The debate over the definition of computer science is as old as our discipline itself. This paper tries to approach this subject from a point of view that has not been widely explored. Namely, what do our students think the definition of computer science is? The paper starts with an examination of the very nature of definitions.

2. Definitions

The first thing to remember when developing a definition is that definitions are a human invention. A definition is an arbitrary process that serves to include some cases and to exclude other cases [4]. In other words, a definition must not simply tell us what something is, it must also tell us what it is not. This dual function of a definition is difficult to achieve and there is often a struggle between a narrow definition and a broad definition. A narrow definition will be specific and make clear whether or not a case will be included in the concept. The problem with a narrow definition is that some legitimate cases of the concept may be excluded by the definition. On the other hand, a broad definition will include all relevant cases but might also include some concepts that are marginally legitimate at best.

An example of a narrow definition of computer science is one given by Edsger Dijkstra. In his definition, Dijkstra prefers to limit our discipline to an intellectual exercise. In 1987, he wrote "...the core challenge for computing science is hence a conceptual one: what (abstract) mechanisms we can conceive without getting lost in complexities of our own making." [3] The problem with Dijkstra's definition is that it ignores implementation issues. These issues are very important to many computer scientists.

A good example of a broad definition is the one given by Newell, Perlis, and Simon in 1967. Their definition simply states that "computer science is the study of computers." [8] While simple and elegant, this definition would include many cases that most people would not consider being computer science. The Computing Sciences Accreditation Board (CSAB) has tried to improve on this definition. CSAB defines computer science as "a discipline that involves the understanding and design of computers and computational processes." [1] However, this definition is also very broad and would allow many extraneous cases to fall within the umbrella of computer science.

Finally, an example that can be considered both narrow and broad is given by Long, et al. In their 1987 paper, they wrote that the central intellectual role of computer science is "the study and application of languages and methods for making precise and understandable descriptions of things." [7] It is narrow in the sense that it leaves out important computer science topics such as ethics in computing. It is broad in that it allows almost any topic from science to be included. What is science if not an attempt to use methods to make precise and understandable descriptions?

In addition to the function of a definition, there are many types of definitions. Some of the more common types are symbolization, similarity, spatial relations, temporal relations, and causation. *Symbolization* is when an object is defined by physically pointing to an example of the concept. If you are trying to define "apple" and you point to an apple, you are symbolically defining the term "apple." *Similarity* is when a concept is defined by examining similarities among objects. If you are trying to define "red" you might point to a red shirt, a red car, and a red house. When you use relations (spatial, temporal, or causation) you describe a concept by its relation to other concepts.

In an interim report from the ACM/IEEE-CS Curriculum 2001 task force, computer science is defined as "an integrated field of study that draws its foundations from mathematics, science, and engineering." [9] This is an example of a relational definition. It assumes that one can define mathematics, science and engineering and it implies that whenever these three disciplines intersect you have computer science. Another problem with this relational def-

inition is that it is not completely and totally accurate. Other disciplines can claim to have contributed to the foundations of computer science. For example, many concepts in artificial intelligence are drawn from psychology [11].

An interesting observation is that the ACM/IEEE-CS Curriculum 2001 task force definition of computer science was originally presented as merely the context for the definition given by the ACM task force on the core of computer science. Their definition, as printed in 1989, states that computer science "is the systematic study of algorithmic processes - their theory, analysis, design, efficiency, implementation, and application - that describe and transform information" [2].

One other troubling piece to the definition puzzle is that the name of our discipline has also undergone scrutiny. While I have used the term computer science, many others have used the term informatics, computing, algorithmics, and even computer studies to describe the same discipline [5]. While most people readily accept these various names as being different ways to name the same thing, I would propose that there are some who use certain names to try and emphasize their particular viewpoint of the definition of computer science. For example, Edsger Dijkstra clearly states that he uses the term computing science to remove from consideration the idea that the discipline should be concerned with a physical computer [3].

The problem that I see with all the definitions for computer science is that they have been directed at people who already think of themselves as computer scientists. The definitions have been used either to argue for inclusion or exclusion of certain items from the field of computer science or the definitions have been used to form a basis for creating a computer science curriculum. To my knowledge, there has not been a serious attempt to define computer science so that a person considering studying computer science would have an idea as to what they are pursuing.

3. Students Perspective

Over the past year in the computer science courses I have taught, I have given a brief assignment. The assignment is to answer the question "What is computer science?" The assignment is not graded and is anonymous. The assignment is given on the first day of class. The students that have taken this assignment include first year students in CS1 and CS2, students enrolled in a microcomputer assembly course, and juniors taking an operating systems course. While certainly not a scientific sampling, I believe the results of these assignments can shed some illumination on student's perception of the definition of computer science.

The first thing I noticed was the variety of answers that I received. Some of the answers were quite humorous, sometimes unintentionally so. One student wrote, "I think of someone working on a computer trying to solve any problems the computers might have." With this definition, computer scientists are reduced to the role of psychoanalyst for the computer. Another definition given by a CS2 student expresses some of the frustration the student must have already experienced. The student wrote, "It is the science of ideas, methods, and languages that make the computer the 'devil' that you must work with."

One trend that emerged was that the definitions given by CS1 students tended to be too broad and the definitions given by CS2 students tended to be too narrow. Some examples of the definitions given by the CS1 students include "computer science is the part of our life which deals with computers" and "computer science is the understanding of how computers works." Some of the definitions given by CS2 students include "computer science is programming," "computer science is the study of computer theory," and "computer science involves using a programming language to solve scientific problems."

I find this trend interesting because I always assumed that students naturally thought computer science was programming and it was my job to expose them to the larger world of computer science. It appears that my students are entering the program with a larger view of computer science and that their first course is causing them to narrow quickly their idea of what computer science is. I would agree with my colleagues who claim that students should see the breadth of computer science early [6]. However, I think we as educators need to see that we may be the reason students have such a narrow view of computer science.

The solution that I have to offer is to perform the "What is computer science?" assignment every semester. This is such a simple idea that it hardly seems worthy of writing this paper. However, I think it is important to remember that definitions are arbitrarily created by human beings. By empowering our students to create their own definitions we are encouraging them to examine what they are studying. I am not advocating that we simply ask for the student's definition and move on to another topic. The written assignment can lead to an important discussion about computer science and scope of our discipline. If the definitions given by your students are too broad, present certain cases that would be covered by their definition but are not considered computer science. Likewise, if the definitions given by your students are too narrow, give examples of computer science topics that are not covered by their definitions.

It is important to remember that people who saw the computer as a new and novel experience created the definitions of computer science that exist today. Most likely, they were educated in disciplines that were not computer science. Freshman who are entering college now were born after the emergence of the microcomputer, have most likely used computers all of their lives, and have used the Internet for nearly half their lives. Their perceptions of what is and what is not computer science will certainly be different from ours. It is our job to encourage their exploration of our discipline. We should encourage our students to help define our emerg-

ing discipline and should never try to burden them with old paradigms and turf wars.

4. Other Thoughts

In addition to being correct, a good definition of computer science should have aesthetic qualities. It should inspire as much as it should clarify. Computer science is a very exciting and dynamic discipline. A good definition should convey that excitement.

Developing a good comprehensive definition of computer science is quite difficult. A reasonable alternative might be to try and to describe what a computer scientist does as opposed to saying what computer science is. I really like the description given by Dirk Siefkes. He states, "As computer scientists we discuss problems, describe solutions, design and use computers and formalisms." [10]

It has often been argued that a survey of computer science topics during the first year does not have a great impact because students do not have the knowledge base to understand all of the concepts that are being presented. Describing what computer scientists do as opposed to what computer science is has the potential of giving the student a more understandable grasp of the scope of computer science.

5. Conclusion

An excellent way to begin a classroom discussion on the nature of our discipline is to ask students to define computer science. By empowering students to develop their own definition, we can encourage them to explore the breadth of computer science

References

1. Computing Sciences Accreditation Board, Inc. http://www.csab.org
2. Denning, Peter J, et al, "Computing as a Discipline", *Communications of the ACM*, Vol. 32, No. 1 (January 1989), 9-23.
3. Dijkstra, Edsger W., "Mathematicians and Computing Scientists: The Cultural Gap", *Abacus*, Vol. 4, No. 4 (Summer 1987), 26-31.
4. Foss, Sonja K., Karen A. Foss, and Robert Trapp, *Contemporary Perspectives on Rhetoric*. Waveland Press, Inc., Prospect Heights, Illinois, 1991.
5. Gal-Ezer, Judith and David Harel, "What (Else) Should CS Educators Know?", *Communications of the ACM*, Vol. 41, No. 9 (September 1998), 77-84.
6. Heliotis, James, et al, "What Concepts Of Computer Science Are Essential For Students Entering The Field?", *The Journal of Computing in Small Colleges*, Vol. 14, No. 4 (May 1999), 224-228.
7. Long, Timothy, et al, "Providing Intellectual Focus To CS1/CS2", *SIGCSE Bulletin*, Vol. 30, No. 1 (March 1998), 252-256.
8. Newell, Allen, Alan J. Perlis, and Herbert Simon, "What is Computer Science?", *Science*, No. 157 (1967), 1373-1374.
9. Roberts, Eric, et al, "Curriculum 2001: Interim Report from the ACM/IEEE-CS Task Force", *SIGCSE Bulletin*, Vol. 31, No. 1 (March 1999), 343-344.
10. Siefkes, Dirk, "Computer Science as Cultural Development: Toward a Broader Theory", in *Foundations of Computer Science: Potential-Theory-Cognition*. Springer, Berlin and New York, 1997.
11. Streib, James T., "Computer Science And The Liberal Arts", *The Journal of Computing in Small Colleges*, Vol. 13, No. 4 (March 1998), 282-287.