Questioning Two Myths in Computer Science Education

Paolo Rocchi

IBM and LUISS University, Roma, Italy
procchi@luiss.it

Abstract. This paper examines two statements regarding computer science as a discipline and its theoretical basis. We shall demonstrate how those statements are questionable and in addition they tend to hide the real root-causes of some significant educational issues. Those statements are very popular in the scientific community and have noteworthy negative effect on the researchers who frequently double their efforts and get around the same problems for years. This work concludes with the claim that experts on computer science education (CSE) should be more attentive to the theoretical aspects of this discipline and should pay more attention to speculative proposals.

Keywords: Theories on computing, computer technology, computer science education, strategies of research.

1 Introduction

In the preliminary stage, we specify that the terms 'computer science', 'information and communication technology (ICT)', 'computing' and 'informatics' will be used as synonyms hereunder.

Since the birth of the digital age, experts made significant efforts to divulge and explain the technology of systems and their use. It seems that pioneering articles appeared in the early 1950s [1].

By the mid-1960s computer science education (CSE) became a very active area due to the wide spread success of computers in businesses and organizations. There was great demand for skilled practitioners and the courses on ICT offered vital services in Western economies. Several research projects started in order to improve didactics of informatics. Also companies and businesses intervened to suggest amelioration of the contents to be taught.

The amount of contributions presented in the last half a century is simply immense. One could quote the hundreds of academic and professional societies on CSE; the thousands of books, publications, and journals specialized on computer education and not specialized; the thousands of occasional and regular events – see the analysis of SIGCSE symposia in [2]. We put forward two remarks upon this enormous global effort.

1) The vivid activity of CSE researchers and the large amount of contributions do not provide proportionally satisfactory outcomes so far. Sometimes experts give the impression of talking around the same topics [3]; the Royal Society talks

D. Passey and A. Tatnall (Eds.): KCICTP/ITEM 2014, IFIP AICT 444, pp. 34–41, 2014.

about the 'vicious circle' of modern computer science education [4]. Pears and others express an opinion amply shared: "decades of active research on the teaching of introductory programming has had limited effect on classroom practice" [5]. Education of software engineering has raised complaints for years [6]. Various studies attempting to delineate core competencies and skills in informatics provide uncertain guidelines rather than unequivocal answers [7] [8] [9].

2) The considerable energy spent to tackle educational issues in informatics cannot be compared with other disciplines. Nothing like this occurs elsewhere. Researchers are not so busy in discussing curricula, roles, skills and professional profiles in other technical areas.

One could ask: 'Why is there such a great difference between didactics of computing and didactics of other scientific fields?'

2 First Myth

Informatics appears as a rapidly changing discipline, which places considerable pressure on CSE and several commentators ascribe the responsibility of the difficulties, which we have pinpointed in 1) and 2), to the subject matter. The following passage summarizes an amply shared viewpoint:

"Computing has changed dramatically over that time in ways that have a profound effect on curriculum design and pedagogy." [10]

This common judgment unites the vast community of ICT educators whose inferential reasoning can be subdivided into two sentences:

"Informatics is a novel and fast evolving discipline. Hence, the pedagogy of informatics meets severe obstacles." (1)

The first line is the premise to the second line which exhibits the conclusion. Statement (1) could be defined as the statement of a theorem including the hypothesis and the thesis. Speaking in general, theorems are to be demonstrated; instead nobody substantiates (1) so far; nobody shows how the close necessarily follows from the premise. so far. Authors cite the various versions of (1) without any logical proof [11] [12]. This common belief seems a dogma rather than a statement sustained by logical reasoning.

The first part (i.e. "Informatics is a novel and fast evolving discipline") is established on the basis of facts and is true beyond any doubt. However the end point (i.e. "hence the pedagogy of informatics meets severe obstacles") is not so evident.

A proof is required to demonstrate a theorem; an example is sufficient to disprove it. Here we present two examples to show how assertion (1) is arbitrary. In particular, we shall examine two innovative and fast evolving scientific sectors which do not raise severe educational problems.

A] In the late 1960s, driven by the growing public awareness of a need for action in addressing environmental problems and the arrival of more severe laws and regulations, environmental science (ES) came alive as an active field of scientific investigation. ES includes instructions in biology, chemistry, physics, geosciences, climatology, statistics, and mathematical modeling [13]. One could deem environmental science more complicated than informatics and in addition ES is fast changing because of the novel and multi-disciplinary technologies required to investigate and to solve large-scale environmental problems. However, educators in environmental science are not so pressed by educational problems as teachers in computing. Prerequisites to ES courses are the lessons of physics, chemistry, biology, geography and other matters. Specialized topics are placed after the introductory lessons. Environmental phenomena are often large scale yet the ES laboratory does not seem as awkward and tricky as computing drilling. Teachers update the courses without special struggle [14].

B] Environmental science probably seems far different in kind and somewhat incomparable to informatics. The second sector – electronics – is closer to computer systems and probably the reader feels it as a more fitting example.

The advances in the development of the electronic market appears remarkable even to common people. It may be said that electronics drags the advance of computer science [15]. Despite its rapid growth and evolution, the study of electrical devices does not result in pedagogical issues similar to those occurring in computing. Usually, the courses of electronics begin with fundamental principles – say Faraday's law of induction, Kirchhoff's equations, the Ohm's law, the Maxwell's formulas etc. – and proceed toward specialized topics.

Electronics does not give rise to severe educational problems regarding the subject contents. Teachers are not so much stressed to 'find a way to present' a certain topic. Students do not undergo disorientation and a sense of groundlessness so frequent in computing. They progressively enrich their culture and arrive at precise professional competencies.

The Association for Computing Machinery (ACM), the Institute of Electrical and Electronics Engineers (IEEE) and other organizations published several reports for experts who prepare courses on computing. This vivid activity is absent in the electronic domain. No organization felt the need to promote the publication of curricula or guidelines similar to those cited above. Experts arrange and optimize the lessons in electronics as technology advances without special efforts.

The reader perhaps objects that informatics has developed so much that now we face different sub-disciplines. The ACM/IEEE Curricula [10] holds:

"The scope of what we call computing has broadened to the point that it is difficult to define it as a single discipline." (2)

One can reply that also electronic experts have inaugurated several new areas such as nanoelectronics, photonics, robotics, power electronics, quantum electronics, and spintronics which do not subvert the didactical curricula. New specialist subject matters substitute the obsolete ones or are appended at the

bottom of the pedagogical pathways without great discussion [16]. The new topics give substance to new professional figures and do not raise lively debates comparable to the discussion occurring in CSE.

Concluding, the cases of electronics and environmental science make evident how an advanced and rapidly evolving technology does not oppose necessarily dramatic pedagogical difficulties. *Statement* (1) *sounds like a generic and untrustworthy myth*. Practical evidence demonstrates that *statement* (1) *is false* and results in repeated and prolonged misunderstandings.

3 Doubled Efforts

At this stage it is natural to ask the following question:

What is the substantial difference between computer science and electronics (and even environmental science) from the didactical viewpoint?

The answer appears to be self-evident as soon as one looks closely at the structure of the two disciplines.

Electronics has shared principles that guide and give order to the entire didactical process, no matter the process is running in a high school, a college or a university. Electronics is based on general laws, models, equations and universal properties that constitute its logical frame and inspire the didactical material. Instead, *informatics does not have a solid theoretical frame of reference*.

Factually "teachers need to know more than just their subject. They need to know the ways it can come to be understood, the ways it can be misunderstood, what counts as understanding: they need to know how individuals experience the subject" [17] and the theoretical basis of electronics provides the principal aid to the educators' complex activity. The fundamentals cast light into the entire area which one explains to students without extraordinary efforts. One can arrange the various topics and sub-topics in a rather straightforward manner. A teacher can proceed from general statements to particular cases. By contrast, *informatics is a science not equipped with an exhaustive and amply shared theoretical base*. The courses of electronics may be defined as 'theory-driven,' informatics cannot be defined as such; instead usual courses of informatics are 'practice driven,' 'competency-driven,' 'example driven,' or 'programming driven.'

In consequence of fundamental theory shortage, researchers in computing education have a double job to do: they tackle the usual pedagogical issues and in addition they are required to compensate for the lack of a general and logical reference. Experts in CSE counterbalance the missing cultural basis, and arrange the subject contents and the targets. They also establish the importance of the various topics, the sequel of the lessons and their logical relationships. They take on several other 'structural' issues.

This double job becomes stressing in a particular manner when there is something to update. The change of a detail can result in dramatic consequences. One can quote the dispute about the substitution of the language Pascal with Java or C++ [18]. By contrast, novelties do not overthrow the educational systems of electronics. Any

didactical update complies with the theoretical order already established. If a technical solution, an instrument or a work method evolves, the didactical process is not completely redesigned but rather, just the intended part varies.

Educators of informatics – lacking an exhaustive theoretical frame – prevalently illustrate technical solutions. A teaching process – even if optimized and updated – turns out to be provisional when it is based on specialist topics because specialist solutions change. Only general principles can yield a reliable pedagogical pathway. Researchers on CSE spend most of their time with specialist topics that are transitory and cannot ensure a logical order to the overall matter. Hence, not only education specialists do a double job, in addition they are doomed to failure. They spend much energy but create doubtful cultural 'buildings'.

The lack of shared fundamentals yield lessons which often seem to exist only in response to a request for help; e.g. request for coding a program, for the use of spreadsheets, and emailing a message [19]. In other words, theoretical omission forces a teacher toward a 'service-orientated activity'. This somewhat obligatory didactical style emerges even in the introductory lessons [20]. The contrary occurs in electronics which is 'theory-driven'. Teachers spend several lessons explaining abstract statements and general equations. Practical skills are gained later.

Concluding, undisputable evidence shows how most significant difficulties in CSE do not derive from the fact that we teach a novel and fast evolving technology, but rather from the fact that *informatics does not have a consistent theoretical basis* and CSE cannot advance as a 'theory driven' activity.

4 Second Myth

Perhaps the reader challenges the ensuing sentences just mentioned:

- Informatics does not have a solid, theoretical frame of reference.
- Informatics is a science not equipped with an exhaustive and amply shared theoretical base.
- Informatics does not have a consistent theoretical basis.

Several experts believe the contrary is true. They claim that computer science is assisted by a large set of abstract references. They cite the Turing theory, the Shannon theory, the theory of graphs etc. and conclude that computer science has a significant mathematical base. The common opinion upon the theoretical foundations of informatics can be summed up in the following terms:

Computer science has a large mathematical base. (3)

This statement is right in the sense that computing has 'several' theoretical bases. However, when one examines each theory, one by one, he can remark that each formal construction gives support to a small technical area, and no theory covers the entire computer domain. For example, Shannon deliberately ignores semantics, and one cannot use the Shannon entropy to qualify the expressive power of a Web page. The various constructions are not connected either logically or causally or by shared characteristics, and thus do not constitute a consistent framework. The theories do not

illustrate the body of computing as an organized description of the knowledge of the field.

It may happen that two or more theories revolve around a single subject matter and pursue different targets. For example, a variety of theoretical constructions treat the programming languages: the Turing theory explains imperative programming [21]; functional programming has a theoretical basis in lambda calculus and combinatory logic [22]; data base languages refer to relational algebra [23], and a special theory supports object-oriented programming [24].

Some theories of computer science overlap and one begins by believing it is hard to say which model is more 'basic'. They are often designed with different goals in mind and are not in agreement with themselves. That is why, the numerous and unrelated conceptualizations do not provide a unified guideline to CSE experts.

A mathematical theory should have adept epistemic capabilities, that is to say it should elucidate the why, the when and the how of the phenomena under observation. Unfortunately a small theory – even if correct – has little explicative qualities because of its proper dimension, and in turn the sum of small theories says nothing or very little as the summation of zeros returns zero. Many CSE researchers have gone through this cultural void. The various attempts to ground a didactical pathway on abstract concepts has not yet gained much success [25]. Educators sometimes prefer to develop the lessons on the basis of practice and experience gained. The students are trained to build knowledge by themselves and actively search for solutions to the problems they experience.

One can reasonably conclude that informatics has several mathematical theories but not a single theory, nor the entire set of theories is able to provide the exhaustive logical framework needed by educators. Practical evidence shows how *statement* (3) *is false*, and the lack of thorough support can but provoke the didactical difficulties discussed in this paper.

5 Practical Development and Conclusion

This paper is an attempt to illustrate how the idea that 'Computer science has a large mathematical base' has not ground and it paves the way to the following deceptive notion 'Informatics is a novel and fast evolving discipline. Hence, the pedagogy of informatics meets severe obstacles'. The two sentences are nothing more than false legends whereas the small and pretentious theories put forward in the computer domain are the authentic root-causes of most problems which current CSE literature is discussing. Hence – in our opinion - researchers should enhance their strategies. They should pay greater attention to the investigations conducted to clarify the fundamentals of CS. They should stimulate and even offer assistance to theorists as frequently occurred in the past with success in various domains.

Computer science as a discipline has always struggled with its identity [26][27] and CSE experts are able to provide a significant contribution. For example, the UK Open University has recently inaugurated a debate on the multifaceted notion of information [28].

Lethbridge [29] claims: “Many important ideas in science and technology have been developed and refined by educators, attempting to determine how to explain complex concepts.”

Theories are not carved in stone. Old theories may be replaced by new theories that have broader scopes than the old ones. Sometimes an old construction is not to be thrown away; it can just be viewed as an approximation – applicable in certain restricted circumstances – of the new more comprehensive theory. It is therefore, quite reasonable to think that CSE should bring greater attention to innovative visions and theoretical proposals that outline a comprehensive frame [30][31][32]. It is worth mentioning that some ideas have been already used in education with success [33].

In conclusion, the argument discussed in these pages is far from being abstract and avulse from practical consequence; it entails behaviors that are really innovative.

References

1. Hopper, G.M.: The education of a computer. In: Proc. of the ACM National Meeting, pp. 243–249 (1952)
2. Valentine, D.W.: CS educational research: A meta-analysis of SIGCSE technical symposium proceedings. In: Proc. of the 35th SIGCSE Symposium, pp. 255–259 (2004)
3. Huggins, P.: Universities failing to provide adequate background DP. Computerworld 25, 3 (1970)
4. AA. VV. - Shut down or restart? The way forward for computing in UK schools. The Royal Society Education Section (2012)
5. http://royalsociety.org/uploadedFiles/Royal_Society_Content/education/policy/computing-in-schools/2012-01-12-Computing-in-Schools.pdf (accessed December 2013)
6. Pears, A., Seidman, S., Malmi, L., Mannila, L., Adams, E., Bennedsen, J., Devlin, M., Paterson, J.: A Survey of Literature on the Teaching of Introductory Programming. In: Working Group Reports on ITCSE on Innovation and Technology in Computer Science Education, pp. 204–223 (2007)
7. Ghezzi, C., Mandrioli, D.: The challenges of software engineering education. In: Inverardi, P., Jazayeri, M. (eds.) ICSE 2005. LNCS, vol. 4309, pp. 115–127. Springer, Heidelberg (2006)
8. Connolly, R.: Criticizing and modernizing computing curriculum: The case of the web and the social issues courses. In: Proc. of the 17th Western Canadian Conference on Computing Education, pp. 52–56 (2012)
9. Dagienė, V.: Informatics education for new millennium learners. In: Kalaš, I., Mittermeir, R.T. (eds.) ISSEP 2011. LNCS, vol. 7013, pp. 9–20. Springer, Heidelberg (2011)
10. De Kereki, I.F.: Work in progress: Transversal competencies contributions to computer science 1 course. In: Proc. Frontiers in Education Conference, pp. S3G1–S3G3 (2011)
11. ACM-IEEE Computing Curricula 2001 Joint Task Force on Computing Curricula (2001), http://www.computer.org/education/cc2001/
12. Goldweber, M., Impagliazzo, J., Clear, A.G., Davies, G., Flack, H., Myers, J.P., Rasala, R.: Historical perspectives on the computing curriculum. ACM SIGCUE Outlook, Special Issue 25(4), 94–111 (1997)
13. Tucker, A.B., Wegner, P.: Computer science and engineering: the discipline and its impact. In: Handbook of Computer Science and Engineering. CRC Press, Boca Raton (1996)
14. Enger, E., Smith, B.: Environmental Science, 12th edn. McGraw-Hill, New York (2009)
15. Elstgeest, J., Harlen, W.: Environmental Science in the Primary Curriculum. SAGE Publications Ltd., Thousands Oaks (1990)

16. Morton, D.L., Gabriel, J.: Electronics: The Life Story of a Technology. Johns Hopkins University Press, Baltimore (2007)
17. Mc Shane, E.A., Trivedi, M., Shenai, K.: - An improved approach to application-specific power electronics education: Curriculum development. IEEE Transactions on Education 44(3), 282–288 (2001)
18. Laurillard, D.: Rethinking University Teaching: A Framework for the Effective Use of Educational Technology. Routledge, London (1993)
19. Abelson, H., Bruce, K., van Dam, A., Tucker, A., Wegner, P.: The first-course conundrum. Comm. of the ACM 38(6), 116–117 (1995)
20. Hatziapostolou, T., Kefalas, P., Sotiriadou, A.: Promoting computer science programmers to potential students: 10 myths for computer science. In: Proc. of the Informatics Education Europe III Conference, pp. 125–133 (2008)
21. Bruce, K.B.: Controversy on how to teach CS1: A discussion on the SIGCSE-members mailing list. SIGCSE Bulletin 36(4), 29–34 (2005)
22. Hoare, C.A.R.: An axiomatic basis for computer programming. Comm. of the ACM 12(10), 576–580 (1969)
23. Hindley, J.R., Seldin, J.P.: Lambda-Calculus and Combinators: An Introduction. Cambridge University Press, Cambridge (2008)
24. Molková, L.: Theory and Practice of Relational Algebra: Transforming Relational Algebra to SQL. Lambert Academic Publishing, Saarbrücken (2012)
25. Gunter, C.A., Mitchell, J.C.: Theoretical Aspects of Object-Oriented Programming, Types, Semantics, and Language Design. MIT Press, Cambridge (1994)
26. Golshani, F., Panchanathan, S., Friesen, O.: A logical foundation for an information engineering curriculum. In: Proc. of 30th Annual Frontiers in Education Conference, vol. 12, pp. T3E/8–T3E12 (2000)
27. Demeyer, S.: - Research methods in computer science. In: Proc. of the 27th IEEE International Conference on Software Maintenance, p. 600 (2011)
28. Denning, P.J.: The science in computer science. Comm. of the ACM 56(5), 35–38 (2013)
29. In: Proc. of Workshop 'The Difference that Makes a Difference'. Open University (2011), http://www.dtmd2011.info/ (accessed December 2013)
30. Lethbridge, T.C., Diaz-Herrera, J.D., LeBlanc, R.J., Thompson, J.B.: Improving software practice through education: challenges and future trends. In: Proc. of Future of Software Engineering Congress, pp. 12–28 (2007)
31. Denning, P.J., Wegner, P.: Introduction to what is computation. Computer J. 55(7), 803–804 (2012)
32. Floridi, L.: Philosophy and Computing: An Introduction. Routledge, London (1999)
33. Rocchi, P.: Logic of Analog and Digital Machines, 2nd edn. Nova Science Publishers, New York (2012)
34. Rocchi, P.: Lectures on CS taught to introduce students with different background. In: Proc. of the Informatics Education Europe III Conference, pp. 115–124 (2008)