# Lecture Notes in
# The Philosophy of Computer Science

Matti Tedre

Department of Computer Science and Statistics, University of Joensuu, Finland

This .pdf file was created January 26, 2007.
Available at cs.joensuu.fi/~mmeri/teaching/2007/philcs/

# 1 Introduction to the Course

- What is *the philosophy of science*?
- What is the topic of *the philosophy of computer science*?
- Why is the philosophy of computer science important for a computer scientist?

Many people suppose that philosophy is about deep, difficult, fundamental, vague, and general questions, like "Why are we here?" and "What is the meaning of all this?". Well, some philosophers perhaps ask such questions. But there are other kinds of philosophy, too. In this course the questions are concrete, commonsensical, specific, and well-defined questions, such as "What kinds of methods do computer scientists use to investigate computing?" and "What is the subject matter of computer science?". These questions belong to a branch of philosophy called the philosophy of computer science, which parallels with other philosophies of specific disciplines, such as the philosophy of mathematics, the philosophy of chemistry, the philosophy of physics, and so forth.

These lecture notes are not focused on science in general, but on computer science in particular. In these lecture notes and in the Moodle fora we are not going to speak too much about how computer science should ideally be done, but we are going to focus on how computer science is *actually* done and analyze that from a philosophical point of view. We are also going to talk about the subject matter of computer science—what kinds of things do computer scientists study, how certain can we be about our results in computer science, and are our findings like "discoveries", "constructions", "laws", or "products".

In order to talk about the philosophy of computer science, one must first know what is philosophy, especially *the philosophy of science*, and what is *computer science*. Neither of the concepts is a straightforward one, and the meaning of both of them is debated. Although the topic of *the philosophy of computer science* is complex and perhaps ambiguous, I expect that by the end of this course the course participants will (1) understand some of the basic positions and concepts in the philosophy of science, (2) be familiar with the debates about the content and form of computer science, (3) be able to connect the debates about what computer science is with the basic positions in the philosophy of science, (4) be able to explain some aspects of computer science in terms of the philosophy of science, and (5) be able to explain their own research and its boundaries in terms of the philosophy of science. That is, by no means, an easy task, but I believe that reading the material a couple of times will be enough to gain an adequate familiarity with the topic.

This is not a course on the general philosophy of science, and in these lecture notes the philosophy of science is not covered very deeply. There is a number of good textbooks about the philosophy of science; I have liked, for instance, Chalmers, 1999. In these lecture notes I include a lot of references to classic works, oft-quoted "classic" journal articles, and to contemporary debates. Those who are interested in the topics can easily find those articles. Other good readings in the topics include:

- Bynum, Terrel Ward; Moor, James H. (2000) *The Digital Phoenix: How Computers are Changing Philosophy*. Blackwell Publishers: Oxford, UK.
- Campbell-Kelly, Martin; Aspray, William (2004) *Computer: A History of the Information Machine* (2nd ed.). Westview Press: Boulder, CO, USA.
- Chalmers, Alan F. (1999) *What is This Thing Called Science?* (3rd ed.). University of Queensland Press: Queensland, Australia.
- Colburn, Timothy R. (2000) *Philosophy and Computer Science*. M.E. Sharpe: Armonk, NY, USA.
- Couvalis, George (1997) *The Philosophy of Science: Science and Objectivity*. SAGE Publications: London / Thousand Oaks / New Delhi.
- Floridi, Luciano (1999) *Philosophy and Computing: An Introduction*. Routledge: London / New York.
- Mitcham, Carl (1994) *Thinking Through Technology: The Path Between Engineering and Philosophy*. The University of Chicago Press: Chicago, USA.
- Okasha, Samir (2002) *Philosophy of Science: A Very Short Introduction*. Oxford University Press: New York, NY, USA.
- Shapiro, Stewart (2000) *Thinking About Mathematics: The Philosophy of Mathematics*. Oxford University Press: New York, NY, USA.
- Yearley, Steven (2005) *Making Sense of Science: Understanding the Social Study of Science*. SAGE Publications: London / Thousand Oaks / New Delhi.


This course is based on my work on social studies of computer science, yet this course was influenced by William J. Rapaport's course *Philosophy of Computer Science: An Introductory Course* (Rapaport, 2005; see also Rapaport's course syllabus).

## 1.1 The philosophy of science

The philosophy of science deals with issues such as the foundations of science, its assumptions and limitations, its implications, and what constitutes scientific progress. The basic questions that appear often in the philosophy of science are the ontological, epistemological, and methodological questions: “What is real?”, “How do we get to know about the reality?”, and “By which principles do we form knowledge?” (we will later in this course discuss those questions in more detail). More specifically, the questions in the philosophy of science can be something like “What is scientific knowledge and how is it different from other kinds of knowledge?”, “With what kinds of methods is science done?”, “What are the limits of scientific knowledge?”, “How does science develop?”, “What is the role of argumentation, logic, confirmations, concepts, and consensus in science?”, “Are scientific results objective or subjective?”, “What can be proven?”, and “What is a law?”.

The philosophy of science is a relatively young branch of philosophy; it can be argued that the philosophy of science separated from general epistemology somewhere in the turn of the 1900s (Duran, 1998:4). During the 1900s the philosophy of science has grown significantly, and in the 1900s a number of very different theories about the soul of science have drastically altered the field

of the philosophy of science. In the beginning of the 1900s logical positivists, especially a group of philosophers called *the Vienna Circle*, were convinced that knowledge comes from experience, and that scientific claims are meaningful only if they can be *verified*. However, verification of many scientific claim turns out to be a daunting task. For instance, how can one verify, to absolute cer - tainty, that matter consists of atoms or that there are quarks?

In 1934 Karl Popper turned the positivists' thinking around and stated that there could not be a way to verify (prove) universal truths, but observations could be used in showing that some claims are wrong, that is, *falsifying* claims (see Popper, 1959). For instance, the theory that matter consists of atoms is consistent with a huge number of observations, experiments, and other theories. Still, a falsificationist would *not* say that the atomic theory has been verified, but that it has withstood an enormous variety of tests without having been falsified. For falsificationists, science consists of the best available theories, but falsificationists do not argue that any of their theories would be abso - lutely correct.

In 1962 Thomas Kuhn noted that historical study reveals that the progress of major sciences cannot be explained by either the positivist or the falsificationist accounts of science (see Kuhn, 1996). Kuhn argued that most of the time there exists a consensus among scientists about the explanations of the world. Most of the time scientists do not challenge the prevailing scientific theories, and they explain their findings within the commonly agreed framework of the prevailing scientific theories. Occasionally, however, the prevailing theories are found to be flawed and new theories bring forth a *scientific revolution*.

For instance, there were certain kinds of chemistry *before* the atomic theory, there were schools of chemistry *during the debates* about the atomic theory, and there is a certain kind of chemistry *based* on the atomic theory. Most chemists of each era have found their theories feasible, or even con - sidered their theories to be the best theories available. Similar, there were influential schools of automatic computation before the advent of the stored-program paradigm, there were competing schools during the formation of the stored-program paradigm, and there is a certain kind of con - sensus on automatic computation today. Scientists at each era have been quite convinced about the superiority of their own approach to automatic computation (for the early history of computing ma - chinery, see, e.g., Williams, 1985).

There are also *critics of science* who claim that science does not hold an intellectually superior posi - tion over other kinds of knowledge. For example, many modern sociologists of science have at - tempted to show the ways in which scientific results are interpretative and flexible, and that also scientific knowledge is essentially socially constructed (Yearley, 2005:25,29). In 1975 Paul Feye - rabend took skepticism about science to one sort of extreme when he argued that scientists are not really working according to any single rigorous methodology. Feyerabend argued that it is not even desirable that scientists would hold strictly to any preordained methods. Feyerabend did not argue that any method is as good as any other, but his argument was that the only principle that does not inhibit scientific progress is, “anything goes” (Feyerabend, 1993[orig. 1975]:14; Horgan, 1996:52).

Philosophers of science often have a training in some branch of natural sciences, and they see the philosophy of science from the viewpoint of their own science. Physicists have been especially well-represented in the philosophy of science, and examples from the field of physics are often used

to back up claims about the philosophy of science. Many power figures of the twentieth century philosophy of science—such as Rudolph Carnap, Karl Popper, Thomas Kuhn, and Paul Feyerabend —were physicists and drew their examples from physics (see Popper, 1959; Kuhn, 1996; Feyerabend, 1993).

In addition to Popper, Kuhn, and Feyerabend, also many other twentieth century philosophers of science have used examples from physics to back up their arguments about how science works, and then assumed that other sciences work or should work in the same manner. However, the 1970s and the 1980s saw the beginning of a turn away from attempts to create explanations that would cover *all sciences*. Many recent philosophers of science have refrained from making arguments about science in general, but they have focused on specific sciences (Ylikoski, 1996). Another trend that began in the 1970s was acknowledging the roles of the scientific community, institutions, ideologies, power structures, cultures, personal preferences and beliefs, economic issues, irrationality, and chance in science. For several decades *the sociology of science* and *the philosophy of science* have been deeply intertwined. The Vienna Circle, Popper, Kuhn, Lakatos, and some recent discourses on the philosophy of science are discussed in more detail at the second half of this course.

## 1.2 What is Science?

The term *science* is already difficult to define. The term can be read in a number of ways, depending on the context. For instance, *science* has been used to refer to (1) a class of activities such as observation, description, and theoretical explanation; or (2) any activity that resembles or has characteristics like those: "He has taken boxing down to a science". Science can also refer to (3) a sociocultural and historical phenomenon: "Western science". Science can refer to (4) knowledge that is gained through experience: "According to our current scientific knowledge...", (5) knowledge that has been logically arranged in the form of general laws (Knuth, 1974c), or (6) structured knowledge derived from "the facts" (Chalmers, 1999:1).

Science can be understood as a (7) societal institution: "Humanity should be governed by science", or a (8) world view: "The scientific world view". Science can also be thought of as (9) a specific style of thinking and acting (Bunge, 1998b:3). Perhaps science can even be understood as (10) scientists' profession. Many scientists think that *pure* and *applied* sciences are separate things. Sometimes applied science is thought of as being intellectually inferior to pure science, yet sometimes pure science is accused of having alienated from practical, everyday matters.

Considering the different meanings of the term *science* described above, it is obvious that science is by no means a neutral term. As long as the term *scientific* is held as synonymous to *true*, *objective*, *justified*, or even *correct*, all the things that are labeled *science* are easily labeled as superior to other, *unscientific*, things. It is a different matter altogether if science really can meet the expectations of truth, objectivity, or correctness. Nevertheless, if *science* is not a neutral word but has a positive connotation, then many people certainly wish to label their work as science. And insofar as *science* carries a negative connotation—such as being cold, narrow, confined, or inflexible—there certainly are people who wish to *avoid* labeling their work as science. The term *science* is indeed a powerful tool for promoting certain agendas, it is a rhetorical key word, and it is an instrument of argumentation.

Furthermore, there are different *aims of science*, such as exploring, developing, refuting, verifying, explaining, understanding, and so forth. It is not clear at all which of these aims are acceptable or desirable of science. For instance, physicist and philosopher of science Pierre Duhem wrote that scientists should refrain from *explaining* anything—Duhem thought that the only thing scientists should do is to *describe* things. The purpose of science, according to Duhem, was to describe facts about the world—he thought that explanations of the world should be left to, say, philosophers (cf. Hitchcock, 2004:8).

> DESCRIPTIVE AND NORMATIVE CLAIMS
>
> There is an important pair of concepts that is going to be used throughout this course. That is the difference between *descriptive* and *normative* claims. Simply put, descriptive accounts of science describe what scientists *are* doing. That is, they describe how science is actually done, how scientists actually work, and how science works in general. Normative accounts of science prescribe what scientists *should be* doing. That is, they prescribe what kinds of actions are good or bad scientific practice, or what kind of science is desirable. Logically speaking, descriptive accounts of science cannot include stands towards how science should be done, and normative accounts of science cannot rely on how science is done (but in practice, they nevertheless do).

Because the term *science* is difficult to define precisely, it might be good to start with referring to science via the *intuition of scientific knowledge* as stated by Alan Chalmers, and developing the concept along the course:

> *What is so special about scientific knowledge is that it is derived from the facts, rather than being based on personal opinion.*
>
> (Chalmers, 1999:*xx*)

However, this intuitive idea is problematic as it turns out. It is unclear which facts are being referred to and where do those facts come from. One might ask, "Are facts derived through reasoning or observation?". It is not certain whether the facts in *computer science* are derived through reasoning or observation, yet it seems that both methods of inquiry are common in computer science. One could continue asking, "Are facts in computer science universally true or not?". Many computer scientists support the universalistic view. A barrage of questions follows: "Are facts in computer science timeless or do they change? Are facts dependent on their intellectual or social context? How is the factuality of facts measured?"

Most scientists believe that science should be pursued through the scientific method. Unlike its name implies, *the scientific method* is not a single, well-defined "method". The scientific method is a collection of techniques, procedures, and methods that are used for investigating the world. Very shortly put, the scientific method includes examining and observing things, formulating hypotheses or explanations, and testing those hypotheses or explanations.

PROOFS

Note that scientists who work according to the scientific method do not *prove* anything correct. Proofs, at least in the mathematical sense as "proving correct", are demonstrations that some things are *necessarily* true, beyond any doubt. In physics, for instance, the general theory of relativity is not *proven*, although it can predict some physical phenomena well. No matter how much scientists test a hypothesis, it can never be said that the hypothesis is proven, but it can be said that it has stood up an extensive range of tests.

Now, what differentiates science from other intellectual activities such as mathematics, engineering, or philosophy? *Mathematics*, for one, is often considered to be different from science because mathematicians do not observe phenomena, formulate hypotheses, and test those hypotheses. Mathematical knowledge is generally considered to be *proven* instead of observed and tested (Shapiro, 2000:4). Although mathematics is often considered to be different from science, in most scientific activities mathematics is used as a tool.

*Engineering* is often considered to be different from science too—engineers do not focus on describing or explaining the world, but they focus on building things. Also *philosophy* does not generally work with observation and testing. It is often debated whether the term *science* is really applicable to every area of intellectual inquiry. For instance, whether or not there are *social sciences* is a debated issue. Some authors argue that there cannot be *laws* in the social sciences, and therefore social sciences should not be called *sciences* (Roberts, 2004; for an opposite view see Kincaid, 2004). However, if scientific laws cannot be verified, what actually makes laws in natural sciences (*laws of nature*—not "natural laws") so different from stable, consistent, and universally applicable theories in other intellectual fields? What actually are scientific laws?

No matter how interesting the question "*What is science?*" is, one cannot begin a course about the philosophy of science—or the philosophy of computer science—by defining science or computer science. That is for the reason that "*What is science?*" is a central question in the philosophy of science and "*What is computer science?*" is a central question in the philosophy of computer science. During this course a number of different views are presented and compared.

## 1.3 The Importance of Philosophy for a Computer Scientist

Disciplinary self-understanding is an important part of any mature discipline. Whether or not field-specific meta-knowledge can resolve any questions in the field is not the issue; the point of disciplinary self-understanding is to offer an account of a discipline and its relationship to other intellectual inquiries. The philosopher of computer science should have knowledge about the creation, maintenance, modification, and diffusion of knowledge in computer science; have knowledge about how computer science is done. Especially in a thoroughly interdisciplinary field such as computer science, where experts in different branches of computer science may not understand more than the general basics of each others' work, the philosophy of computer science should aim at providing orientation and direction for computer scientists (cf. Shapiro, 2000:15).

The philosopher of computer science should be able to *interpret* the workings and results of computer science to computer scientists and scholars from other disciplines alike. In a field that is suc-

cessfully being utilized in a large number of other disciplines, there is a risk of intellectual chauvin – ism, of exporting the explanatory models of computer science into the more "backwards" discip – lines. For instance, the algorithmization of other disciplines (Easton, 2006) seems like such project. The philosopher of computer science should be able to locate computer science in the framework of research traditions and to illuminate the prospects and constraints of computational models, al – gorithms, and other models of explanations that computer science can offer. The philosopher of computer science should, therefore, know computer science inside out.

### 1.3.1 Promises and Challenges of Interdisciplinarity

Computer science is a relatively young discipline. Although people have used mechanical aids to calculation much longer than sixty years, the birth of the stored-program paradigm and the con – struction of the first fully electronic, digital, Turing-complete computer ENIAC in 1945 changed the face of automatic computation. It took some 20 years for computer science to achieve a discip – linary identity distinct from fields such as mathematics, electrical engineering, and logic. Throughout the short disciplinary history of electronic digital computing, there has been a great variety of different approaches, definitions, and outlooks on *computer science* or *computing as a discipline*. Arguments about the content of the field, its methods, and its aims have sometimes been fierce, and the rapid pace of extension of the field has made it even harder to define computer sci – ence faithfully.

During the last 60 years, researchers in the academic field of computing have brought together a variety scientific disciplines and methodologies. The resulting science, computer science, offers a variety of unique ways of explaining phenomena, such as computational models and algorithms. The increased investments in research efforts in computer science have been paralleled by the growth of the number of branches of computing, such as computer engineering, scientific computa – tion, electrical engineering, decision support systems, architectural design, and software engineer – ing. In the first parts of these lecture notes, the term *computer science* is used as a loose collection of computing-centered fields such as the aforementioned branches. That is, the term is used as an umbrella term, and more specific terms are used whenever necessary.

Although interdisciplinarity made the development of computer science possible in the first place, it also poses a very real challenge to computer scientists. Firstly, it is not certain what kinds of topics should be considered to be computer science. Secondly, it is very difficult to come up with a com – mon understanding of how computer science research should be done. The same rules of computer science should cover research in fields such as software engineering, complexity theory, usability, the psychology of programming, management information systems, virtual reality, and architectural design. The subjects that computer scientists study can be, for instance, programs, logic, formulæ, people, machines, usability, or systems. It can be debated if an overarching, all-inclusive definition of computer science is even necessary.

The attempts to describe computer science are invariably either very narrow and applicable to only some subfields of computer science, or so broad that they do not exclude much. A good example of a broad definition of computer science is the one given by three famous computer scientists Allen Newell, Alan Perlis, and Herbert Simon, in 1967. They argued that for any phenomenon in the world, there can be a science that describes and explains that phenomenon (Newell et al., 1967).

They continued by noting that there is indeed a phenomenon called computers, and by arguing that "computer science is the study of the phenomena surrounding computers" (Newell et al., 1967; see longer quotation on page 14 of these lecture notes).

Newell et al.'s account of computer science is a descriptive one (i.e., it describes what computer sci – entists actually study). It also emphasizes that knowledge in computer science is derived from de – scription and explanation. However, if computers are also a product of computer science, then Newell et al.'s definition seems circular. That is, computer scientists study computers, which are made by computer scientists. In other words, computer scientists create the topic of their research. Yet, many other disciplines have been defined much in the same way. For instance, C. Wright Mills wrote that social science is defined by what duly recognized social scientists do and have done (Mills, 1959:19). Richard Hamming wrote that "mathematics is what mathematicians do" and that "mathematicians are people who do mathematics" (Hamming, 1969). One could also say that the crux of the debate lies in the subject matter. The things that chemists and physicists study mostly exist and happen without chemists and physicists; Do the subject matters of computer sci – ence exist and happen without computer scientists? (Some say they do, some say they do not.)

Under a somewhat loose interpretation of Newell et al.'s description "computer science is the study of the phenomena surrounding computers", studies of individual-level phenomena, such as net ad – diction; interpersonal phenomena, such as net date and virtual rape; social phenomena, such as vir – tual communities; or perhaps even abstract subjects, such as information society and collective in – telligence would all belong to the field of computer science. However, because Newell et al. could not have predicted the phenomenal changes in computing and its uses, it is perhaps best not to cel – ebrate their statement as a wide-open and all-embracing account of computer science. Nonetheless, despite being 40 years old, Newell et al.'s definition is still today often referred to.

An example of a narrow definition of computer science is, for instance, Khalil and Levy's descrip – tion of computer science as a study of programming computers:

> *Our (first order) definition is that computer science is the study of the theory and prac – tice of programming computers. This differs from the most widely used definition by emphasizing programming as the central notion and algorithms as a main theoretical notion supporting programming.*
>
> (Khalil & Levy, 1978)

Khalil and Levy foregrounded programming perhaps because their intention was to market their view of a graduate program in computer science at the time of the software crisis. There are many implicit positions that can be clearly seen in Khalil & Levy's argument, such as their view that knowledge is derived from experience instead of being derived solely by reflection and thinking. Note that both Newell et al.'s and Khalil & Levy's definitions represent single views among hun – dreds of definitions of computer science, dozens of which will be presented in this course.

Now, it is important for a computer scientist to understand the challenges (and possibilities) that the vast diversity of computer science causes. Many arguments about how computer scientists *should* work have their roots in different conceptions about what computer science is. Many misunder –

standings and controversies between scientists from different branches of computer science can be avoided by just knowing the philosophical traditions in which people in those branches work.

Even more importantly, a computer scientist must know that the same methods cannot be used with the whole variety of subjects that computer science studies. Mathematical and computational mod – els are precise and unambiguous, yet they are confined to the abstract world of mathematics and they fail to capture the richness of reality. Narratives and ethnographies are rich in dimensions and sensitive to detail, yet equivocal and context-dependent. Narratives have little use in deriving for – mulæ, and formal proofs have little explanatory power regarding usability issues or diffusion of in – novations.

Computer science is increasingly drawn into other fields such as physics, chemistry, biology, psy – chology, and even sociology and anthropology. The better computer scientists understand the as – sumptions, constraints, limitations, and premises of their own science, the easier it is to accommod – ate computer science for the uses of other disciplines. The better computer scientists understand computer science, the easier it is to utilize other disciplines in computer science. Valuation of other intellectual traditions comes with understanding the strengths and weaknesses of one's own intellec – tual tradition. Similarly, contributions to other fields often require an understanding of the intellec – tual foundations of computer science.

### 1.3.2 Philosophical Questions Specific to Computer Science

One can easily take the basic questions from the philosophy of science into computer science and ask, for instance, “What is *knowledge* in computer science and (how) is it different from *beliefs* and *assumptions*?”, “How can one differentiate between knowledge and beliefs in computer science?”, “How *do* computer scientists work and how *should* computer scientists work?”, “Do computers sci – entists prove formulas like mathematicians do, build things like engineers do, test hypotheses like natural scientists do, or something else?”, “Can computer scientists know some things for sure?”, “Are there things computer scientists cannot know for certainty?”, “What is *progress* in computer science?”, “Are new algorithms or programs *progress*?”, “How does new knowledge about comput – ing become a part of commonly held knowledge about computing?”, “What kinds of things in com – puter science are universal, or objective, or timeless?”, “What kinds of things in computer science can be proven correct?”, “Are there *laws* in computer science?”, and so forth.

But computer science raises also philosophical questions that are specific to computer science, and that all computer scientists should be aware of. For instance, the question “What is the relationship between a program, a model, and the world?” accompanies all programming activities (yet not all programmers are aware of the difficulties and consequences of that question). Researchers of com – puter science should be aware about the consequences of different positions towards the question “Are programs *theories*?”. Concepts such as *implementation level*, *hardware*, and *software* belong to the commonplace terminology of computer science, but it is uncertain what exactly *is* an imple – mentation level, and where does the line between hardware and software go. Note that in many cases hardware could be implemented as software, and in many cases software could be implemen – ted as hardware.

On a more general level, one can ask that if botany is the study of plants, zoology the study of animals, and astronomy the study of stars, what is computer science a study of? Is it a study of algorithms, automation, programming, information, classes of computations, computers, usability, uses of computing, all of those, or something else? Different subject matters lead to different kinds of science or engineering. All the different subject matters of computers science give rise to different kinds of follow-up questions, yet some of those questions may not be topical to the practicing computer scientist anymore.

Philosophically interesting questions are, for instance, if algorithms are abstract or concrete things —both positions can be argued for (Smith, 1998:29-32); if essentially *everything* is a computer or if the universe is a computation (Searle, 1990; Wolfram, 2002); if the Church-Turing Thesis should be considered to be a *law* or something else (what is a *thesis*, anyway?); and what is the difference between data, information, knowledge, and wisdom and which of those do computers actually process. And perhaps the best-known philosophical issues that computer science has brought about are the question of whether machines can think, the limits of machine computability, and the epistemological question "*P=NP?*".

The philosophy of computer science is not a well-defined or well-developed field. The philosophy of computer science is *not* fully covered in the philosophy of science, in the philosophy of technology, or in the philosophy of mathematics. The philosophy of computer science is *not* fully covered in the philosophy of the mind or in the philosophy of AI (artificial intelligence). The philosophy of computer science is also more than the ethics of computing. And the philosophy of computer science is *not* equal to philosophy of computing, philosophy of information, or cyberphilosophy.

The philosophy of computer science is not generally about the truth or falsity about specific arguments in computer science, nor is it solely about the consequences that computers have on philosophy. The philosophy of computer science is not only about the impact of computers on society, either. The philosophy of computer science is essentially about the nature and justification of scientific activities of computer scientists (cf. Cottingham, 1996:300). It is the philosophy of a discipline.

If computer science is a unique field separate of mathematics, natural sciences, and engineering; if computer science poses unique questions, comes up with unique explanatory models, utilizes unique methods, and leads to unique ethical problems; then it is easy to argue that those unique things are probably not answered in the philosophy of mathematics, science, and engineering, and hence, there is a need for a field that addresses those unique things—the philosophy of computer science. The philosophy of computer science is an important part of understanding what computer science is, why computer science is what it is, why computer science has developed as it has, what are the methodologies of computer science, and why the methodologies of computer science (or lack thereof) are as they are.