tripleC 4(2): 316-327, 2006
ISSN 1726-670X
http://tripleC.uti.at

What does it mean to Know Computer Science? Perspectives from Gender Research

Christina Björkman and Lena Trojer

Technoscience Studies
Blekinge Institute of Technology
P.O. Box 520, S-372 25 Ronneby
Christina.Bjorkman@bth.se
Tel. +46 457 38 55 69

Abstract: The epistemological basis for computer science (CS), on which research and education as well as development of applications are founded, are fundamental for its production of knowledge. In this paper we raise the issue of how gender research developed within science and technology can be used within computer science, to approach and discuss foundations of the discipline, and what the implications of this reflection are for CS education. After an introduction, which serves to motivate the questions raised, we discuss issues concerning the foundations of computer science. We then introduce gender research, as we use it, and present some points where this type of research can contribute to the question "What does it mean to know CS?".

Keywords: Gender research, computer science foundations, epistemology

1 Introduction

"From its inception just half a century ago, computing has become the defining technology of our age." [1]

Computer science (CS)[2], as one of the core disciplines within the broad area of information technology, has become one of today's most important disciplines by virtue of its influence on the shaping of technology and thus also society. There is little technical research, development and production done today that does not, in one way or another, involve results from (mostly in the form of applications of) CS. Computer science thus strongly influences the direction and content of technical research and development

It is reasonable to assume that the influence of CS on the current and future developments of technology will continue to grow, and that the discipline will remain at the centre of information technology.

[1] ACM and IEEE-CS Computing Curricula 2001, Computer Science Volume, chapter 3. http://www.sigcse.org/cc2001/

[2] We use the term 'computer science' (CS) in a broad sense, including software engineering and all relevant parts of computer engineering. The context for the article is computer science as it is commonly defined and delimited at Swedish universities which in all important matters conforms to a a fairly traditional and Anglo–American definition of CS as a discipline in the faculty of natural science and/or engineering. For a discussion concerning the usage of 'computing' and 'computer science', see the section 'Computer Science' below.

Thus, CS as a field of knowledge and technology holds a dominant position. In order to take responsibility for this dominant position, we argue that there is a need for CS to be a multifaceted discipline with many angles of approach. It is not enough, as we see it, to merely include more areas into CS. Rather, we want to take a more radical step, and introduce the notion of epistemological pluralism (i.e. different ways of knowing and learning) (e.g. Wagner 1994).

Equally important as a multifaceted discipline is a broad representation of developers of knowledge and technology within CS. A more diverse understanding of CS is likely to result in a more diverse representation of people being attracted to the field. However, in a Swedish and anglo-american context, this is not the case today, when the dominant picture is that CS mainly attracts a fairly narrow group of students, mostly young males with a passion for, and often experience with, computers and programming[3]. Maria Klawe eloquently expresses the need for a diverser representation among computer scientists (Klawe 2001: 67-68):

"*The point here is that computer science also needs to attract students with broader interests and abilities than the traditional computer scientists—nerds.[…] But nerds are not enough. We need more computer scientists whose passions are art, language, literature, education, entertainment, psychology, biology, music, history, or political science. We need them because computers have an impact on all areas in our world. We need people with passion and vision from every area to drive the development of computer technology as well as the applications. […] We need non-nerds in computer science, so let's figure out the proper approaches to integrate their talents and perspectives into our field*."

We claim that this narrow recruitment basis is one indication of a too limited understanding of what it means to "know CS". We argue that the issue of narrow recruitment to the discipline not only concerns the image of computer science, nor educational structure and culture (though these are certainly important issues to address), but that it is also necessary to look at the disciplinary foundations and approaches to knowledge in CS. How knowledge and learning processes are formed, mediated and reflected, e.g. in education, is a large, but so far mostly overlooked, part of the complex problem of narrow representation patterns in CS.

We thus argue that there is a growing need for a more complex and integrating understanding of knowledge processes, by which we mean integration and acceptance of different approaches to knowledge, epistemologies, methodologies etc, i.e. epistemological pluralism.

In this paper, we approach and discuss the foundations of computer science. We also discuss how gender research within CS enables possibilities to develop more complex understandings and interpretations of CS. We ask many questions, which are meant to invite to reflections and discussions. We strongly believe in the contribution from such discussions to facilitate discursive space for transformation trials[4] within computer science, with a particular focus on education.

2 Computer Science

What 'is' computer science? Or rather, how is it constructed and perceived? What constitutes the 'core' of the discipline? Is CS a mixture of other disciplines or does it have its own unique core? What fundamental 'paradigms'[5] guide knowledge processes within the discipline?

CS is fairly young as a discipline in its own right and is still being formed and the subject of many discussions regarding its core character and content. The boundaries of CS are constantly debated: what is to be considered to be within the discipline and what is to be considered to be outside (but connected to) it, for example where does software engineering and human-computer interaction belong? The

[3] The issue of under-representation of women in CS has been extensively discussed in the literature, for an overview and critical discussion see Björkman (2002).

[4] By discourse we mean a pattern of understanding that counts as meaningful in a certain normative context. By a discursive space for transformation trials we mean an environment, where what counts as meaningful for transformation, is broadened and developed. For a discussion of discursive space / discursive practice see Barad (2003).

[5] We use the word 'paradigm' here in a loose sense. By using it, we want to point to foundational ideas of importance for knowledge in CS. In this meaning, it is also used by for example Denning et al (see below).

foundations for computer science, on which all education and research, as well as development of applications, are based, are fundamental for the production of knowledge. Methodology and epistemology are intertwined with what we do and how we do it, thus underlying all research and knowledge production (Harding 1987).

One dilemma we face is related to terminology. The term 'computing' is often used in a more inclusive sense than computer science. Some researchers use the term computing to mean (more or less) the whole field of IT, some use it to disconnect the discipline from the physical computer (Dijkstra, see McGuffee 2000), and others use it to mean "all of computer science and computer engineering" (Denning et al 1989: 10). In the works referred to below, we have taken pains to identify what the authors mean when they use the word 'computing', and unless otherwise stated, it can be understood as synonymous to our use of the term 'computer science' in all relevant matters[6].

One of the most well known contributions and attempts to define computing was made in 1989 by the ACM Task Force on the Core of Computer Science (Denning et al 1989). They identify three major paradigms or "cultural styles": theory, rooted in mathematics; abstraction (modelling), rooted in the experimental scientific method; and design, rooted in engineering. These processes are seen as closely intertwined; they cannot be separated but are nevertheless distinct, since they each represent different competences. Thus, the task force concludes: "Computing sits at the crossroads among the central processes of applied mathematics, science and engineering" (Denning et al 1989: 11). A short definition of computing is given as:

> *"The discipline of computing is the systematic study of algorithmic processes that describe and transform information: their theory, analysis, design, efficiency, implementation and application. The fundamental question underlying all of computing is, "What can be (efficiently) automated?""* (Ibid: 12)

In this definition, the notion of 'algorithm' is seen as a central concept in CS.

The algorithmic side of computer science is emphasized by Judith Gal-Ezer and David Harel in their discussion "What is CS" (Gal-Ezer and Harel 1998: 78):

> *"The point is that CS is not only the scientific basis of a major technological revolution, but has at its heart a special and powerful way of thinking—algorithmically—which is required in dealing with the ever-complex modern world, and which is becoming crucial in many other scientific and engineering disciplines, too."*

So then, what does an algorithmic definition of CS entail for the understanding of knowledge within the discipline? Abelson and Sussman directly address this (Abelson and Sussman 1985[7], quoted in Denning et al. 1989: 11-12):

> *"The computer revolution is a revolution in the way we think and in the way we express what we think. The essence of this change is the emergence of what might best be called procedural epistemology – the study of the structure of knowledge from an imperative point of view, as opposed to the more declarative point of view taken by classical mathematical subjects. Mathematics provides a framework for dealing precisely with notions of 'what is'. Computation[8] provides a framework for dealing precisely with notions of 'how to'".*

What are the implications of this "procedural epistemology" for knowing within the discipline?

Judith Gal-Ezer and David Harel recognise two sides of CS: the algorithmic side and the systems side, and claim that "CS itself is an unusually dichotomic subject – one facet is more mathematical and the other is a type of engineering." (Gal-Ezer and Harel 1998: 79). They argue that there are also dichotomies within these facets: mathematics encompasses computability, complexity and logic on the one hand as well as numerical analysis on the other, while engineering encompasses the design and construction of hardware as well as the development of software.

[6] We use CS to emphasise the discipline aspect, an aspect that is not always clear from the word 'computing', and at the same time argue for a broadened understanding of the discipline of CS.

[7] Abelson, Harold and Sussman, Gerald Jay (1985) *Structure and Interpretation of Computer Programs*. Cambridge, Mass: MIT Press.

[8] 'Computation' in this quote should be understood in the same sense as 'computing'.

So, what are the implications of a discipline based on inherent dichotomies, and what tensions, useful as well as restrictive, exist because of this dichotomic nature? Is it possible to deconstruct and go beyond these dichotomies, and if so, what would that entail for the discipline?

One side of the dichotomy discussed above refers to mathematics. Abelson and Sussman bring up issues of knowledge in connection to the discussion of mathematics vs. computing, and it is interesting to note that they see computing as different from mathematics. The role of mathematics in and for computer science is a cause of much dissent within the community of computer scientists. A fairly strong and influential group within CS defines the discipline as closely related to mathematics. In a famous paper from 1989: "On the cruelty of really teaching computing science", Edsger Dijkstra claims that "computing science is—and will always be—concerned with the interplay between mechanized and human symbol manipulation usually referred to as 'computing' and 'programming', respectively" (Dijkstra 1989: 1401), and that computing should be localised in "the direction of formal mathematics and applied logic" (Ibid: 1402). He even goes so far so as to propose that computing science be called 'VLSAL' (Very Large Scale Application of Logic) (Ibid: 1402). The discussion about mathematics is far more complex than a mere discipline issue; to a large extent it is historically conditioned, but it is also about power, about 'who is best/right', and about what counts as 'superior' knowledge.

Many other definitions of CS have been suggested too, some quite simple: "computer science is the study of computers" (Newell, Perlis and Simon[9] quoted in McGuffee 2000: 74), or the Computing Sciences Accreditation Board: CS is "a discipline that involves the understanding and design of computers and computational processes" (McGuffee 2000: 74).

Another interesting question is whether any recent changes can be seen in the view of computer science. In the ACM Computing Curricula 2001, Computer Science Volume (ACM CC2001), the rapid evolution of the discipline is discussed. There is no attempt to define CS in this document[10], since the report is focused on curricula for CS education, but the report observes that technology has undergone radical changes during the last decade, not the least with the development of networking and the WWW. It also recognises that cultural factors affect computer science and CS education. What this report calls computing has become broader, encompassing more areas. However, the acceptance and inclusion of more areas does not necessarily by itself entail a fundamental change in the understanding of the knowledge processes within CS.

One of the central changes that can be seen in CC2001 is the inclusion of professional practice as an integral component in the CS curricula. Where the 1989 report identified three paradigms, this could now be seen as expanded: "All computer science students must learn to integrate theory and practice, to recognize the importance of abstraction, and to appreciate the value of good engineering design" (ACM CC2001, chapter 4, our italics). However, whether the integration of practice in the curricula should be interpreted as a change in the view of the discipline is not clear, and many computer scientists are likely to argue that practice might be part of the profession, but not part of the discipline. What would it mean if practice were actually regarded as part of the discipline? Such a change could be fundamental, if it were really incorporated into the core, opening up new views of what is important knowledge for a computer scientist. What to include in 'professional practice' is still an open question, but in our opinion this should include knowledge of the area of use as well as of users and how society and technology are intertwined.

Peter Denning is one of the prime movers in the ongoing discussion of "the profession of IT" and the related topic of practice within computing[11]. He argues for accepting the importance of professional practice: "Practices are as important a part of knowledge as discourses, mental models, conceptual frameworks, processes and rules" (Denning 1999: 2). He claims that "applications domains are the front lines of the profession" (Ibid: 2) and that "Value skills connect a professional's technical performance with the customer." (Denning and Dunham 2001: 24). Peter Denning regards computing as the discipline and

[9] Newell,Allen, Perlis, Alan and Simon, Herbert (1967*) What is computer science?* In: Science, no 157, pp. 1373-1374.

[10] In chapter 4 of the report, the committee lists what they see as the areas encompassing the body of knowledge within CS. This list includes for example Software Engineering, Human–Computer Interaction and Information Management.

[11] Peter Denning uses the term 'computing' in the same sense as defined on page 3, i. e. as equivalent to our use of computer science.

IT as the profession, and he claims that there is currently a gap between the two. Computing is no longer the driving force, controlling the field, and he advocates that it should cross the chasm and seek leadership within the new profession, by for example accommodating "embodied professional knowledge" (Denning 2001: 24). A similar argument is made by Steve Cunningham: "Any computing education that does not pay attention to the user's role in computing is missing the most vibrant and exciting part of computing today." (Cunningham 1998: 4a).

Another noteworthy point in CC2001 concerns what the committee regards as important for a curriculum, in the sentence: "Development of a computer science curriculum must be sensitive to changes in technology, new developments in pedagogy, and the importance of lifelong learning" (ACM CC2001, chapter four). This puts focus on technology and knowledge, but no reference is made to society or issues such as risk, sustainability, accountability etc. The latter are issues that we claim to be of crucial importance for society on the whole, and they should thus also be considered within CS.

A discipline does not exist on its own; it is defined and held together by its practitioners. Computer science and computer scientists are constructing, and are constructed by, each other in a mutual and constantly ongoing process. What then is a computer scientist? How is a computer scientist 'formed'? How do computer scientists understand CS, what ideas and concepts do they find central to the discipline, howdo they understand and create knowledge and images of concepts? How is CS 'thought' and 'talked'?

James McGuffee (McGuffee 2000) argues that a good alternative to defining CS is to describe what a computer scientist does. He quotes Dirk Siefkes: "As computer scientists we discuss problems, describe solutions, design and use computers and formalisms" (Siefkes 1997[12], quoted in McGuffee 2000: 76). It is interesting to note the concept of 'problems' in this, indicating an engineering relationship, as well as to note the absence of 'use' and 'users' of products of CS. How generally accepted is this definition within the community? There is a tendency to discuss CS as something separate from computer scientists, existing on its own. This becomes especially clear when looking at how issues of women and computer science are commonly discussed (Björkman 2002). In these discussions, focus is almost always and solely on the first word: women, and the discipline itself is usually taken for granted. From this kind of perspective, adaptation comes solely from the side of the (prospective) computer scientist, and the mutually constructed character of the relationship is obscured. This creates the image of CS as existing on its own, independent of people.

New paradigms or metaphors for computing are surfacing; the most important one today seems to be interactivity or interactionism. This concept has been discussed by a number of researchers. Lynn Andrea Stein argues the need for a shift in the underlying metaphor of computing, from the traditional metaphor, "computation as calculation", towards a metaphor of "computation as interaction" (Stein 1999). Such a change, Stein argues, would affect how CS is viewed and thus also what is taught and how, as well as how computer scientists think. Peter Wegner writes about "why interaction is more powerful than algorithms" (Wegner 1997). Frances Grundy discusses a new conception of computing that she terms "interactionism" (Grundy 2001) and Heidi Schelhowe sees interaction as a successful approach to development of software (Schelhowe 2004). What could the effects of these emerging paradigms be? In what ways could they support 'epistemological pluralism' (Turkle and Papert 1990, Wagner 1994), or other ways of knowing? Can different metaphors or paradigms for computation affect the learning processes in CS?

Paradigms or metaphors of importance within CS will take on a significant role in education. We see the teaching of programming as being of particular importance. What are the paradigms and views of knowledge of CS and programming in programming courses? Is this visible in the courses or not recognised but taken for granted? The concept of programming is one of the first things that students learn. How has the knowledge foundation in programming (theories, methodologies, methods and languages) evolved? What constitutes the fundamental knowledge base, and what assumptions and choices have been made during the course of time? Is there support for different styles of approaching programming (see for example Turkle 1984, Turkle and Papert 1990), and what would be the implications

[12] Siefkes, Dirk (1997) *Computer science as cultural development: Toward a broader theory*. In: *Foundations of Computer Science: Potential-Theory-Cognition*. Berlin and New York: Springer.

of that? What role does for example skills knowledge, “knowing how” (Adam 1998), play in the learning of programming?

Does object-orientation in any substantive way constitute a ‘paradigm shift’? Or is it just a minor change in methodology, neatly incorporated into existing paradigms? And if it is something entirely new, what would that mean for the discipline and its practices? For example, Abelson and Sussman talked about the “procedural epistemology” within CS. Does object-orientation have an effect on this? Is the procedural thinking still valid in times of OO? Sherry Turkle and Seymour Papert argue that a shift towards object-orientation might potentially mean a shift in thinking and the legitimising of alternative methods of programming (what they term ‘bricolage’ as contrasting with the commonly taught ‘planning’ approach, Turkle and Papert 1990). As it is now, it seems as if the potential power of object-orientation has not brought on significant changes within the teaching of programming, but has rather been incorporated into existing methodologies. If and how a different paradigm or metaphor can promote learning of programming is a question that ought to be of great interest to the whole computer science community.

What constitutes the core and the foundations of a discipline can always be the focus of study, it can be debated and perhaps reformulated and changed, since production of knowledge and our understanding of it are ongoing processes. There is nothing ’naturally’ inevitable about how computing is constructed. As argued in the introduction, we claim the need to consider the possibility of creating new, additional approaches to knowledge within the core of CS.

3 Gender/Feminist Research in Science and Technology and its Relevance to CS

Gender/feminist[13] research concerning computer science has to a large extent focused on issues of gender in relation to computer science, for example the lack of women within computing, and gender equality aspects (see the overview and discussion in Björkman 2002). In these studies, CS is often seen as firmly defined, and the underlying perceptions of development and knowledge in CS are seldom brought into focus. We want to show in this article how gender research can be a resource within CS, for discussions concerning the discipline. Time might be ripe for us, “as partakers in the modern research complex, to develop a readiness to think and feel ourselves as part of the problem, and learn how to use our implicatedness as a resource for transformatory projects.” (Trojer and Guldbrandsen 1996: 131).

Gender research represents many theoretical and methodological approaches, and the meaning and focus of the research is different within different disciplines. We here want to give a brief introduction to gender research as it has developed within science and technology.

Gender research can have two general focuses: sex/gender on the one hand, and feminist frameworks for science itself on the other. Gender research within natural science and technology mainly concentrates on the second of these, focusing science itself, its theories, methodologies and other knowledge processes. This type of gender research discusses and studies the bases of the disciplines and broadens the epistemic point of departure in order to help approaching the foundations of the discipline and its knowledge production.

The emphasis on transformation, out of identified needs, as a prime goal for gender research, is essential. From the very beginning it was perceived inadequacies and imbalances in established research that motivated a growing feminist critique of science. This science critique developed from issues concerning women, to realising and focusing on problems concerning how science is constructed and practiced. Sandra Harding formulated this in her groundbreaking book “The Science Question in Feminism” (Harding 1986). Harding argued for a shift of focus, from “the woman question in science”, by which she meant, “What is to be done about the situation of women in science?” (Harding 1986: 9) and towards what is often called “the science question in feminism”, where she argued for and pointed to a reflexive turn, where feminists’ transformation work also includes ourselves, as part of the problem and part of the solution.

[13] We use the term ‘gender research’, which is the most commonly used term in Sweden. However, many researchers, mainly from Anglo-Saxon countries, use the term ‘feminist research’. An older term is ’women’s studies’.

Knowledge and knowledge processes within science are of particular interest for gender research. A number of questions are relevant to ask around knowledge issues, such as: what knowledge is valid and why? Who can have knowledge? Who has the preferential right of interpretation and why? And "Whose science? Whose knowledge?" (Harding 1991). Finally, but not the least: How could it be different? Such questions can throw light on implicit scientific practices of importance for our understanding of what it means to know CS.

Important work concerning theory, epistemology and methodology for this type of gender research has been advanced by for example Sandra Harding (e g Harding 1986, 1991), Evelyn Fox Keller (e g Fox Keller 1985, 1992) and in particular Donna Haraway (e g Haraway 1991, 1997). Epistemological pluralism within feminist methodological development contributes with expanding the notions of knowing, accepting other and different ways of knowing than the dominating propositional view of knowing ("knowing that", e.g. Turkle and Papert 1990, Adam 1998). Important issues in feminist epistemologies are for example situated knowledge, partial translations (Haraway 1991), and embodied knowledge. Focusing situated knowledges is a base we strive after for our knowledge claims. We don't believe in universal claims of truth. Included in this is the notion of situatedness as part of an epistemological consciousness. Situated knowledge increases possibilities for relevant knowledge claims however partial interpretations they must be. Haraway (1991: 196) stresses that what we can reasonably bring about in our knowledge production can never be more than partial translations. Translations are always interpretative, critical and just partial.

> *"We do not seek partiality for its own sake, but for the sake of the connections and unexpected openings situated knowledges make possible. The only way to find a larger vision is to be somewhere in particular.*" (Haraway 1991: 196).

For a thorough account of feminist/gender research within science and technology, see Trojer (2002, in Swedish) and Mörtberg (1999).

4 Gender Research within Computer Science

> *"The interaction of women's studies and CS should expand and improve our information revolution."* (Thelma Estrin, professor in CS, in Estrin 1996: 46).

How can gender research in CS contribute to the goals outlined in the Introduction, such as broadening the meaning of "knowing CS"? In this section, we want to give examples of issues where gender research in CS contributes significant work, as well as point to issues that need to be further investigated. Using and developing gender research within CS opens up possibilities for new approaches, which we will give examples of below. Theories and methodologies from gender research offer new opportunities to explore issues around knowledge in CS. We believe that especially feminist epistemological thinking has the potential to enrich computer science. In this way, gender research can become an active participant, in particular within CS education. This is supported by other gender researchers in CS, for example Norwegian informaticians[14] Tone Bratteteig and Guri Verne, who "see epistemological inquiries to establish alternative understandings of knowledge" as being the most challenging and having the greatest potential for contributing to change in CS (Bratteteig and Verne 1997: 60).

4.1 Paradigms of computer science

As discussed above, paradigms of importance within CS will take on a significant role in education. A rethinking of these could likely have considerable impacts on *what* is taught as well as *how* it is taught.

Frances Grundy raises questions concerning the 'fundamental nature' of the discipline of computer science. She challenges the three major paradigms identified within CS: mathematics, science and engineering (Grundy 2000a, 2000b, 1998). She discusses the role of mathematics in computing, and in particular what role mathematics actually plays for abstraction. Her argument is that mathematics is only

[14] "Informatics is the term for computer science departments in universities in Norway, indicating that the discipline is defined more broadly than in traditional computer science departments." (Bratteteig and Verne 1997: 59).

one type of abstraction involved in computing, and she further claims that mathematics is a status symbol and has been used as an argument for making CS into a science.

Abstraction is considered very important for CS. However, the products of CS are very concrete. Why is abstract, formal and logical thinking and knowing seen as superior within CS? Sue Clegg (Clegg 2001) argues that computing is neither an extension of mathematical thinking nor an applied science. She sees the reasons for these views of CS as historically conditioned. Instead, she suggests that computing should be seen as a concrete science, concerned with materiality and social practices. We argue that the implications of such a change in perception of the discipline could potentially have almost revolutionary effects within CS education, and open up for different approaches than those that dominate today.

Frances Grundy has developed a concept around what she terms *interactionism.* In her version, this is a cluster of ideas, involving for example a blurring of the distinction between the subject and object (Grundy 2000b). "Interactionism emphasises the practicality of computing; it also recognises that much computing is about communication and it recognises the importance of pluralism." (Grundy 2001).

4.2 Integration of use and practice

In a preceding section, we pointed to the discussion concerning integration of practice into CS education. The practice of many computer scientists concerns production of software, including design. An important focus for gender researchers has been issues of design and use (e.g Bratteteig 2004). Software design and development is a complex activity, requiring knowledge not only of the technology involved but also knowledge of the area of use. Gender research, with a foundation in situated knowledge, may contribute to the discussion about use and design, and to develop other theories and methodologies, for example to account for complexity and for heterogeneity among users, in order to develop responsible and sustainable technology (Mörtberg 2003).

Tone Bratteteig and Guri Verne (Bratteteig and Verne 1997) argue that use of technology and applications ought to be included as an integrated part of computer science and that alternative understandings of knowledge are developed through the experience of application. Different "models of the world" will result in different computer systems – and thus also different consequences for the users. How systems are constructed depend on who construct them, and what world-view and understandings of knowledge, experience, values and needs they integrate in the development and the final products. Who influences development is thus important to take into consideration (Mörtberg 1997).

What is excluded from CS? Referring to excluded issues such as missing accountability, the absence of subjectivity and the excluded views of the system users, Ulrike Erb argues that "in particular if we do feminist research inside the discipline of computer science, one main purpose of this research might be [….] to reveal the excluded and to integrate the excluded in order to enrich computer science by means of the forgotten perspectives" (Erb 1997: 206). What must not be dismissed, however, is that the actual integration processes should be transformative rather than merely additive.

4.3 Knowledge and learning

> *"Knowing is not necessarily a matter of saying and representing what is the case but can also be a kind of practical involvement with the world*." (Belenky et al.1997[15]).

Questions concerning 'what knowledge?' and 'whose knowledge?' are among the most central issues for gender research to focus on. Alison Adam has extensively discussed epistemological issues in her work on artificial intelligence (AI)[16] (e.g. Adam 1995, 1998). She discusses issues of knowledge, such as 'whose knowledge' and 'what knowledge' is represented in AI systems. Among other topics, she

[15] Belenky, Mary, Clinchy, Blythe, Goldberger, Nancy and Tarule, Jill (1997) *Women's Ways of Knowing: The Development of Self, Voice and Mind*. New York: Basic Books.

[16] AI is often regarded as a sub-discipline of CS – or at least some aspects of AI are. It can be argued that AI is a separate discipline, with its own epistemology. However, the issues concerning knowledge are highly relevant in CS.

discusses the differences between propositional knowledge ('knowing that') and skills knowledge ('knowing how'), or mental vs. embodied knowledge, and how the former has been seen as superior to the latter (Adam 1995).

Computer science does require a certain amount of abstract thinking. However, there is no doubt also need and room for what can be called concrete thinking, and not least concrete learning. Thelma Estrin, professor in CS, sees "concrete thinking" as one way where feminist epistemologies can influence CS, and she takes her examples from programming education. By concrete she means practical involvement:

> *"Every science is incomplete and always in the process of extension and expansion from new ideas. Feminist epistemology, with its dedication to concrete learning introduces new ideas for gaining knowledge that may make CS more relevant…"* (Estrin 1996: 46).

This could introduce new ideas for gaining knowledge that may make CS more relevant to a more diverse group of people. Knowledge and acceptance of different types of knowledge construction (see e.g. Alsbjer 2001) is essential for extending the view of knowledge within CS, and thus potentially accommodating greater diversity in its practices and among its practitioners. We strongly believe that CS education would gain from cherishing "epistemological pluralism".

4.4 Programming and the Object-Ooriented paradigm

We see the teaching of programming as being of particular importance. Maria Alsbjer has used gender research and feminist epistemological theory to discuss programming education, in particular the processes involved in learning to program (Alsbjer 2001).

Whose knowledge is built into objects in object-oriented design? Cecile Crutzen and Jack Gerrissen have analysed the ontology and epistemology of the object oriented paradigm (OO) (Crutzen and Gerrissen 2000). They argue that OO enhances the idea of the controllable and deterministic:

> *"[It is] based on the same illusions of objectivity and neutrality of representation; the negating of power and dominance by translating it into 'natural and obvious', and on the existence of truth by transforming it into progress."* (Crutzen and Gerrissen 2000: 132-133).

They claim that object orientation is based on the idea that everything and everybody can be represented in terms of objects, an idea that they object strongly against. They argue that OO should not be used for the analysis of human worlds, but only for what it was originally intended: the realisation of software[17].

It is interesting to compare this analysis of OO with the views expressed by Sherry Turkle and Seymour Papert ten years earlier (Turkle and Papert 1990) where they see OO as potentially revolutionising programming methods and also as challenging traditional ways of thinking and knowing.

Both views expressed above are relevant for a discussion within the CS community. They are furthermore, as we see it, not mutually exclusive. OO could challenge traditional ways of thinking within CS (but has not really done it so far), at the same time as there has to be an awareness of the risks of OO representations as pointed out by Crutzen and Gerrissen.

4.5 Representation and metaphors

Computer science builds competences on consensus-marked classifications, standardisations and formalisations. Christina Mörtberg discusses representation in a way that can serve to illustrate the reasoning (Mörtberg 2000: 58 our translation):

> *"Formal representations are created in processes that entail abstractions, quantifications, hierarchisations, classifications, standardisations and simplifications* […]. *In these processes, there are negotiations about borders and content and in these negotiations, technology and gender are shaped."*

[17] Note that what they criticise is the paradigm of object-orientation at a fairly high level, for example for making analysis of "human worlds", *not* the low level object-oriented programming, used for "realisation of software".

Categorisation is not only a means of structuring the outside world – it also limits and affects our way of thinking. By leaving established categories, new forms of understanding can be created.

What kind of presumptions, choices, standardisations, classifications etc. are involved in the knowledge processes? So far, for example gender-marked representations and metaphors are neutralised, made implicit and integrated in the development of models, computer systems, etc. The use of language has proved to be very important in our understanding of ideas and the images they call to mind. The presence of clearly gender-marked metaphors can be a factor in supporting the gender structure within the discipline. Metaphors create images that will be of importance in the knowledge processes (Fox Keller 1995).

Furthermore, design of computer products are not value or gender neutral. The knowledge and experience of the designer influences his or her design, and in our society gender is one factor influencing experience. What is integrated into design, and perhaps even more important, what is not integrated? However, it is not easy to isolate the gendered aspects of technology, since they are integrated socio-cultural phenomena (Bratteteig 2003). Uncovering cultural aspects in software is an important issue, which is likely to require knowledge about design and construction of software.

5 Summary and concluding discussion

In the introduction, we argued for a growing need to develop a more complex and integrating understanding of knowledge processes within computer science. This need is based on the fact that CS has an increasing influence on current and future technological development. We recognize practices of society and practices of CS as interlinked in progressively more sophisticated ways.

We believe that using gender research based within science and technology to study and transform CS and its knowledge processes provides potentials for the development of new conceivable understandings and interpretations of CS, and what it means to "know CS". In this article we have discussed and pointed to issues where gender research can contribute:

- By asking questions per se, decisions and assumptions underlying technology can be made visible, thus avoiding 'black-boxing' (Latour 1999)
- By showing how technology and science are closely intertwined
- By pointing to how CS does not exist in itself, it is constructed by people and can thus be reconstructed
- By paying attention to how views of knowledge implicitly exist in syllabi and curricula, and question these assumptions
- By querying the role of different paradigms and metaphors in the discipline, and show possibilities for, in specific contexts, more functional alternatives
- By unveiling how software is laden with cultural values and choices, including gendered aspects
- By promoting reflexivity concerning issues of knowledge and their implications for practice, e.g. the assumptions implied in teaching
- By concretely fostering and working with integration of epistemological pluralism into CS.

We believe that reflection around issues of knowledge is important for every discipline, especially for teaching and for meeting potentially new groups of students. Can computer scientists[18], by becoming aware of their own views of knowledge and understanding, also become aware of, respect and accommodate for, greater diversity among students and their backgrounds, interests, motives and understandings? Can we, as feminists and computer scientists, thus in the long run, change the discipline into one that is more attractive to a broader range of students, for example women?

[18] Christina includes herself in 'computer scientists'.

References

ACM and IEEE-CS (2001*) Computing Curricula 2001, Computer Science Volume*. http:, , www.sigcse.org, cc2001, [2005-04-05].

Adam, Alison (1995) *Artificial intelligence and women's knowledge: What can feminist epistemologies tell us?,* In: Women's Studies International Forum, Vol. 18, No. 4, pp. 407-415.

Adam, A. (1998) *Artificial Knowing: Gender and the Thinking Machine*. London & New York: Routledge.

Alsbjer, M. (2001) *Att hitta ingångar i formandet av programmeringskunskap* [To find entrances in gaining programming knowledge]. Karlskrona: B Sc thesis, Blekinge Institute of Technology. http:, , www.bth.se, fou, cuppsats.nsf, allabeslut, 038630f4efec13e5c1256a6a006be4a0, $file, kandarbMA.pdf [2004-09-29]

Barad, K. (2003) *Posthumanist performativity: toward an understanding of how matter comes to matter*. In: Signs, Vol. 28, No. 3, pp. 801-831.

Björkman, C. (2002) *Challenging Canon: the Gender Question in Computer Science*. Licentiate thesis. Karlskrona: Blekinge Institute of Technology.

Bratteteig, T., Verne, G. (1997) *Feminist or merely critical?* In: Moser, Ingunn and Aas, Gro Hanne (eds) *Technology and Democracy: Gender, Technology and Politics in transition*? Oslo: Centre for Technology and Culture, University of Oslo, pp. 59-74.

Bratteteig, T. (2003*) Kjönnet design av IT* [Gendered design of IT]. In: NIKK Magasin, No. 2, pp. 15-17.

Bratteteig, T. (2004) *Making Change: Dealing with Relations Between Design and Use*. PhD Thesis. Oslo: University of Oslo, Faculty of Mathematics and Natural Sciences, Department of Informatics.

Clegg, S. (2001*) Theorising the machine: gender, education and computing*. In: Gender and Education, Vol. 13, No. 3, pp. 307-324.

Crutzen, C., Gerrissen, J. (2000) *Doubting the OBJECT world* In: Balka, Ellen and Smith, Richard (eds*) Women, Work and Computerization: Charting a Course to the Future.* Boston: Kluwer Academic Publishers, pp. 127-136.

Cunningham, S. (1998) *Outside the box: the changing shape of the computing world*. In: SIGCSE Bulletin, Vol. 30, No. 4, pp. 4a-7a.

Denning, P. (1999*) Computing the profession*. In: *Proceedings of the Thirtieth SIGCSE Technical Symposium on Computer Science Education*. New Orleans, Louisiana, pp. 1-2.

Denning, P. (2001) *Crossing the chasm*. In: Communications of the ACM, Vol. 44, No. 4, pp. 21-25.

Denning, P., Dunham, R. (2001) *The core of the third-wave professional*. In: Communications of the ACM, Vol. 44, No. 11, pp. 21-25.

Denning, P., Comer, D., Gries,D. (1989*) Computing as a discipline*. In: Communications of the ACM, Vol. 32, No. 1, pp.9-23.

Dijkstra, E. W. (1989) *On the cruelty of really teaching computer science.* In: Communications of the ACM, Vol. 32, No. 12, pp. 1398-1415.

Erb, U. (1997) *Exploring the excluded. A feminist approach to opening new perspectives in computer science.* In: Grundy, Frances, Kohler, Doris, Oechtering, Veronika, Petersen, Ulrike (eds) *Spinning a web from Past to Future, Proceedings of the 6th International IFIP [TC9, WG9.1] Women, Work and Computerization Conference.* Berlin: Springer, pp. 201-207.

Estrin, T. (1996) *Women's studies and computer science: their intersection*. In*:* IEEE Annals of the History of Computing, Vol. 18, No. 3, pp. 43-46.

Fox Keller, E. (1985*) Reflections on Gender and Science*. New Haven: Yale University Press.

Fox Keller, E. (1992) *Secrets of Life, Secrets of Death*. London: Routledge.

Fox Keller, E. (1995) *Refiguring Life – Metaphors of Ttwentieth-century Biology*. New York: Columbia University Press.

Gal-Ezer, J., Harel, D. (1998) *What (else) should CS educators know?* In: Communications of the ACM, Vol. 41, No. 9, pp. 77-84.

Grundy, F. (1998) *Computer engineering: engineering what?* In: AISB Quarterly, No. 100, pp. 24-31.

Grundy, F. (2000a) *Mathematics in computing: a help or hindrance for women*? On CD-ROM from *Charting a Course to the Future, Proceedings of the 7th International IFIP [[TC9, WG9.1] Women, Work and Computerization Conference*, Vancouver, Canada.

Grundy, F. (2000b*) Where is the science in computer science*? On CD-ROM from *Charting a Course to the Future, Proceedings of the 7th International IFIP [[TC9, WG9.1] Women, Work and Computerization Conference*, Vancouver, Canada.

Grundy, F. (2001) *A new conception of computing: interactionism replaces objectivism*. Paper presented at GASAT 10 Conference, Copenhagen. http:, , www.keele.ac.uk, depts, cs, staff, a.f.grundy, home, interact.htm. [2004-05-01]

McGuffee, J. (2000) *Defining computer science*. In: SIGCSE Bulletin, Vol. 32, No. 2, pp. 74-76.

Haraway, D. (1991*) Simians, Cyborgs and Women: The Reinvention of Nature*. London: Free Association Books.

Haraway, D. (1997) *Modest_Witness@Second_Millenium.FemaleMan©_Meets_OncoMouse™. Feminism and Technoscience*. London: Routledge

Harding, S. (1986) *The Science Question in Feminism*. Ithaca: Cornell University Press.

Harding, S. (1987) *Feminism and Methodology*. Bloomington: Indiana University Press.

Harding, S. (1991) *Whose Science? Whose Knowledge?* Ithaca: Cornell University Press.

Klawe, M. (2001) *Refreshing the nerds*. In: Communications of the ACM, Vol. 44, No. 7, pp. 67-68.

Latour, B. (1999*) Pandora´s Hope: Essays on the Reality of Science Studies*. Cambridge, Mass. and London, England: Harvard University Press.

Mörtberg, C. (1997) *"Det beror på att man är kvinna...", Gränsvandrerskor formas och formar informationsteknologi* ["It's because one is a woman…", Transgressors are shaped and shape Information Technology]. Doctoral dissertation. Luleå: Luleå University of Technology.

Mörtberg, C. (1999) *Technoscientific challenges in feminism*. In: NORA (Nordic Journal of Women's Studies), Vol. 7, No. 1, pp. 47-62.

Mörtberg, C. (2000) *Teknikvetenskap och genusforskning* [Engineering science and gender research] In: Trojer, Lena (ed.) *Genusforskningens relevans, rapport från forskningsrådens expertgrupp för genusforskningens integrering.* Stockholm: Samverkansgruppen för Tvärvetenskap, Genusforskning och Jämställdhet, pp. 50-68.

Mörtberg, C. (2003*) In dreams begins responsibility – feminist alternatives to technoscience*. In Mörtberg, Christina, Elovaara, Pirjo, Lundgren, Agneta (eds*) How do we make a difference?*. Luleå: Luleå University of Technology, pp.57-69.

Schelhowe, H. (2004) *Computing science and software development: paradigms of mathematics, engineering, interaction.* Paper presented at Symposium Gender and ICT: Strategies of Inclusion, Brussels 20 Jan. 2004.

Stein, L. A. (1999) *Challenging the computational metaphor: implications for how we think* In: Cybernetics and Systems, Vol. 30, pp. 473-507.

Trojer, L. (2002) *Genusforskning inom teknikvetenskap - en drivbänk för forskningsförändring* [Gender research within Technoscience – a Hotbed for Research Transformation]. Stockholm: Swedish National Agency for Higher Education.

Trojer, L., Gulbrandsen, E. (1996) *Authority in transformation* In: The European Journal of Women's Studies, Vol. 3, No. 2, pp. 131-147.

Turkle, S. (1984) *The Second Self: Computers and the Human Spirit*. London: Granada.

Turkle, S., Papert, S. (1990) *Epistemological pluralism: styles and voices within the computer culture*. In: Signs, Vol. 16, No. 11, pp. 128-157.

Wagner, I. (1994*) Connecting communities of practice: feminism, science and technology*. In: Women's Studies International Forum, Vol. 17. No. 2, 3, pp. 257-265.

Wegner, P. (1997) *Why interaction is more powerful than algorithms*. In: Communications of the ACM, Vol. 40, No. 5, pp. 80-91