Design Doctorate in Computing: a defence of "doing cool stuff"

Edwin Blake
University of Cape Town
Department of Computer Science
edwin@cs.uct.ac.za

ABSTRACT

This position paper argues that it is time to extend the notion of worthwhile scholarship in Computer Science to embrace Design and to award doctorates in the field of Design.

Categories and Subject Descriptors

K.3.2 [Computer and Information Science Education] Computer science education.

Keywords

Design.

1. COMPUTER AND SCIENCE

The problem lies with the combination of *Computer* with *Science*. Many of us have tried long and hard to put science into Computer Science. This originated as a resistance to the bad old days when there was a tendency to venerate and pursue "cool stuff" for its own sake and its own reward, culminating, in the case of Computer Graphics, with the infamous "proof by pretty picture" papers. At the time it seemed the only other option was mathematical formalisms which appeared to be neither really useful nor interesting.

This effort to put Computer Science on a firm empirical footing was vindicated by our discovery that there was a field called *Experimental Computer Science* (ECS). It was propounded, for example, in the book "Academic Careers for Experimental Computer Scientists and Engineers" [1]. This certainly showed how to do good science in the pursuit of interesting and useful applied Computer Science. The essential method of ECS is to built artefacts and then evaluate them experimentally [2][3]. Such research differs from that conducted in related fields such as Information Systems (where the building of artefacts plays almost no role) or Theoretical Computer Science (which is simply mathematics, in spite of the name).

There has always been a slight niggle though: many students (and some colleagues) instinctively and atavistically revert to wanting to do "cool stuff"! I have to confess that it sometimes seems a bit artificial to require them to design a proper experimental investigation of the artefact, when in fact the artefact in itself is the thing of creative interest and passion. It seems to me the problem with an experimental view of Computer Science is that it does not pay enough attention to the "Design" side of the subject[1].

In this I am merely echoing Fred Brook's Newell Award lecture "the scientist *builds in order to study*; the engineer *studies in order to build*"[4]. In this paper I do not wish to define or restrict what Computer Science is, as long as we can agree that it is a broad church. My interest is in recognizing and legitimizing the whole range of scholarly activities that Computer Scientists engage in.

1.1 Proper Experimental Computer Science

In this section I would like to explore the specific case of Experimental Computer Science (ECS). As we tell our honours students, the process of ECS involves:

- Form a hypothesis
- Construct a model and make a prediction
- Design an experiment and collect data
- Analyse results

The reason to do this is that this approach is generally regarded as the basis for the whole scientific and technological revolution that shapes much of our society today. It is the dominant way of getting to the truth.

When such research is presented in a "doctoral thesis" an examiner tends to look for a number of specific things. Primarily, a doctoral dissertation must present a unified proposition that is defended by argument. The dissertation is expected to contain a coherent theory or thesis that is defended by means of facts and reasoned argument. In short, there must be a "thesis" as such.

The stated hypothesis in a dissertation serves to formalize the claims that there is such an original point of view advanced as the result of research. A doctoral dissertation cannot hang together purely on the basis of the application or artefact. This is not normally enough to constitute a defended thesis.

So what does a researcher, who has created a new artefact, have do if they want to justify it as novel under the broad methodology of ECS?

If a doctorate is sought for an artefact, as such, then the only respectable argument that can be made is a "proof of existence". That is, the artefact in question conveys the essence of entirely

[1] Actually there are *two* niggles. The one I deal with in this document is related to "design", the other niggle is that we have still not satisfied the prime imperative of science which is to discover "new stuff", the sort of thing that gets into Nature and Science. This is a different issue.

new phenomenon that was never before imagined. This would be the case (one imagines) with the computer mouse as demonstrated by Doug Engelbart. This is very rare and not what most artefacts achieve.

A much more "normal" ECS contribution is one of improved performance or else proof of concept. The proof of concept type of artefacts are those where the complexity of the system overwhelms our ability to predict performance analytically. Such overwhelming complexity is quite normal in most computer systems, particularly so when they involve users and usability. The argument is that a particular artefact allows better performance, and the researcher performs a series of experiments to verify the improvements.

For a doctorate one would normally expect more than one series of experiments with one artefact. In general there is an expectation that some underlying thesis is being defended by means of the written dissertation. Such an underlying thesis will have a theoretical insight which is validated by artefacts which play the role of exemplars or testbed for the real contribution: which is the theoretical insight. The artefacts and accompanying empirical evidence serve to triangulate the concept (hence the joking reference to the fact that a research masters requires only two experiments but for a doctorate you need three!).

1.2 What's Wrong?

The problem is that Computer Science itself derives from at least three different disciplines each with a very different epistemology and methodology, namely, mathematics, experimental science and design or engineering [5]. As Matti Tedre [6] succinctly puts it: "It is notoriously difficult to conduct research in the intersection of research traditions without making a mess of it". We cannot simply deny the importance of the various aspects of our discipline.

The mathematicians will not concern us further at this point as their research is normally written up by the doctoral student for the supervisor and one other person (but we'll return to theoretical analysis below in Section 2.3). What we would like to do here is recognize the importance of both empirical science and design within the discipline.

I think it is fairly clear by now how one ought to do good science[2]. The question is how to do good design and get recognition for it. It is also becoming apparent that Computer Scientists are simply not accepting the call for empirical justification in their papers. Wainer *et al.* [7] repeated the survey by Tichy *et al.* [3] of papers published by the ACM. The first survey was a sample of papers from 1993 while the repeat study was of papers from twelve years later. Their main conclusion is that not much has changed! Tichy *et al.* pointed out that in their sample, according to their criteria, 40% of the papers which should have had experimental validation had none at all. In the repeat study of papers from 2005 this number was 33%. Not much improvement there. The obvious question is which were the papers that lacked empirical validation? They turned out to be the ones classified as "Design and Modelling"[3] by the authors of these studies.

What we can do now is to question the (unreasonably) high regard that empiricism and (positivist) research has in academia where it is pretty much equated with scholarship, and that is the topic of the next section.

2. THE SCHOLARSHIP OF COMPUTER SCIENCE

The question that is really raised is where does true scholarship of Computer Science lie? We first need to accept that it is worth-while to regard a University is as a place for engaged intellectuals. If we accept that we need such institutions, and I do not see how we can possibly get by without them in this country, then we are ready to for a broader view of scholarship.

Ernest Boyer made this distinction in his book "Scholarship Reconsidered: Priorities of the Professoriate" [8]. This book and his papers point to some interesting and pertinent views. Amongst other things he argues that academia has come to overvalue the Scholarship of Discovery (meaning Science) and undervalue other forms of scholarship including, amongst others, the Scholarship of Practice.

2.1 Alternatives to Experimental Science — the Scholarship of Practice

The form of scholarship being argued for here is one of design (I speak as someone who does not hold a qualification in the field of design in what follows). Design doctorates are professional doctorates that demonstrate the opening up of a new field to designed solutions. In such a case the novel design significance has to be shown. Design is usually embedded in an enquiry of the context of the use of the design. Such doctorates are not yet awarded in Computer Science within the South African research setting[4]. I think that it is time that we addressed this methodological issue explicitly.

If what is being aimed at is a design doctorate then the emphasis is rightly on the artefact being produced. In this it would be joining a new trend of "Professional Doctorates". One can argue that Computer Science is a synthetic discipline and has all along been concerned with design. It is however unusual to attempt to obtain a doctorate for design work as such, that is, work where the synthetic artefact produced is the sole object of elucidation and investigation.

Design doctorates are based on qualitative research, case studies, contextual enquiry, and the like. They acknowledge their lack of objectivity, arguing that it is an unachievable idealization, and

[2] Actually it isn't completely clear since good science can include qualitative research. Qualitative research is far from a settled question; especially in a multidisciplinary setting.

[3] Design & Modeling. The main contributions of articles in this category are systems, techniques, or models, whose claimed properties cannot be proven formally. Examples include software tools, performance prediction models, and complex hard- and software systems of all kinds.

[4] Although apparently some of our colleagues in Information Systems are under the impression that Design Research is accepted in Computer Science. In fact they probably mistake the lack of methodological rigour in our discipline for adherence to design science.

replace it with various techniques for supporting the design claims by means of triangulation. Such a doctorate would depend on a deep examination of issues of design in a specific domain. It would go beyond improvement of solutions or even solving new problems in an established domain. Doctoral design work would establish a new context within which we can solve a new class of problems. Of necessity it may involve solving several ancillary problems: something that is clearly a feature of many attempts to do "cool stuff".

The impact of doctoral work in design is to extend the boundaries of what can be solved, and the strategies which are used to look at problems. Such work should have generalizable impact beyond the examples chosen for the project.

If a study is aiming to extend the scholarship of practice then it has to show the generalizable use of the design artefact, and I would imagine, the impact it has had on the typical users of such a device.

2.2 Of Course There is Science

In emphasizing design I am not downplaying the scientific nature of Computer Science. Clearly it is a science, aspects of it can even be called a natural science as more and more information processing aspects of Nature itself are uncovered [9]. That is not all there is to it however, and it is a mistake to insist on it.

The reason it is a mistake is that statements such as "Computing is no longer a science of just the artificial. It is the study of information processes, natural and artificial" ([9], p 18) look as if they are broadening the scope of the field but in fact narrow it. They miss out the area of design — it never was merely the *science* of the artificial, it included the *art* of creating the effective artificial.

2.3 Why Analysis Does Not Offer an Alternative

I need to deal with a distraction, and that is the role of Mathematics in Computer Science. Formal analysis offers many deep insights in Computer Science. In fact it is even more "unreasonably effective" in Computer Science than it is in the case of the natural sciences (see for example, Knuth on the "surprising" usefulness of discrete mathematics [10]). So much so, that those parts of mathematics that are useful in reasoning about computers is called "Theoretical Computer Science". However this is not the theory of Computer Science in the same sense that theoretical physics relates to physics [5].

It does not set the agenda for Experimental Computer Science. On the contrary we resort to ECS precisely when theoretical computer science is no help, which is to say, most of the time when dealing with real systems. B. C. Smith made this point some time ago [11]. His main argument is against notions that one can prove correctness of programs "This is why I think it is somewhere between misleading and immoral for computer scientists to call this 'correctness'. What is called a proof of correctness is really a proof of the compatibility or consistency between two formal objects of an extremely similar sort: program and specification" ([11], p 23). He deals with the relation of a model to a real computer: "The point is that computers, like us, participate in the real world: they take real actions. One of the most important facts about computers, to put this another way, is that we plug them in. They are not, as some theoreticians seem to suppose, pure mathematical abstractions, living in a pure detached heaven" ([11], p 21). He clearly implies that similar arguments can be made in the case of dealing with complexity, human-computer interaction, reasoning at different levels of abstraction and action research.

We can successfully argue that every program is a member of a type and proves a proposition (in the sense of Per Martin-Löf's type theory). We still have not shown that the proposition relates to the real world.

Brooks eloquently argues for the engagement with the world of practice and away from the "deadly trend" that "already curses American mathematics" [4].

3. CONCLUSION

I have one major disagreement with most of the authorities I have quoted: it seems they tried to define what computer science *is*. I am arguing for recognizing what Computer Scientists *do*. I am not trying to discover if Computer Science is a Science or is not. My view is that as practicing Computer Scientists we engage in a broad range of activities; all of which have a beauty and can excite our creative interest.

These activities include mathematical analysis and wonderment at the applicability Intuitionism to reasoning about programs. We can appreciate a good experimental investigation of the effects of visual displays in creating a sense of Presence in a virtual world. We should also have a way of recognizing the effectiveness of a new masterfully designed tool.

At present we seem unable to give recognition to a brilliant new design, in its own terms, in our doctorates, I think that has to be remedied.

4. REFERENCES

[1] National Research Council (U.S.) 1994. *Academic Careers for Experimental Computer Scientists and Engineers*. National Academy Press. www.nap.edu/openbook.php?record_id=2236

[2] Tichy, Walter, Lukowicz, P., Prechelt, L., & Heinz, E.A. Experimental evaluation in Computer Science: a quantitative study, *Journal of Systems and Software* 28 (1) (1995) 9–18.

[3] Tichy, Walter 1998. "Should Computer Scientists Experiment More?", *IEEE Computer*, 31(5), 32-40. doi.ieeecomputersociety.org/10.1109/2.675631

[4] Brooks, F. P. 1996. The computer scientist as toolsmith II. *Commun. ACM* 39, 3 (Mar. 1996), 61-68. DOI= doi.acm.org/10.1145/227234.227243

[5] Denning, Peter J., Comer, D. E., Gries, D., Mulder, M. C., Tucker, A., Turner, A. J., and Young, P. R. 1989. Computing as a discipline. *Commun. ACM* 32, 1 (Jan. 1989), 9-23. DOI= doi.acm.org/10.1145/63238.63239

[6] Tedre, Matti 2007. "Know Your Discipline: Teaching the Philosophy of Computer Science" *J. Information Technology Education*, 6, 105–122. URL= jite.org/documents/Vol6/JITEv6p105-122Tedre266.pdf

[7] Wainer, Jacques, Barsottini, C., Lacerda, D., de Marco, L. 2009. Empirical evaluation in Computer Science research

published by ACM. *Information and Software Technology* 51 (2009) 1081–1085

[8] Boyer, Ernest 1990. *Scholarship Reconsidered: Priorities of the Professoriate.* Princeton. Scanned microfiche copy at eric.ed.gov/ERICWebPortal/contentdelivery/servlet/ERICServlet?accno=ED326149

[9] Denning, P. J. 2007. Computing is a natural science. *Commun. ACM* 50, 7 (Jul. 2007), 13-18. DOI= doi.acm.org/10.1145/1272516.1272529 .

[10] Knuth, D. E. 1974. Computer Science and Its Relation to Mathematics. *American Mathematical Monthly*, 81, 4 (Apr., 1974), 323-343. www.jstor.org/stable/2318994

[11] Smith, B. C. 1985. The limits of correctness. *SIGCAS Comput. Soc*. 14,15, 1,2,3,4 (Jan. 1985), 18-26. DOI= doi.acm.org/10.1145/379486.379512