Research Methods in Computer Science

Serge Demeyer
University of Antwerp
Department of Mathematics and Computer Science
Middelheimlaan 1 B-2020 ANTWERPEN
http://www.win.ua.ac.be/∼sdemey/

I. Origins of Computer Science Research

Computer Science as a research discipline has always struggled with its identity. On the one hand, it is a field deeply rooted in mathematics which resulted in strong theories.[1] For example, there is computational complexity theory (turing machines, the halting problem), database theory (the relational model, expresive power of query languages), formal language theory (the chomsky hierarchy, well-formedness, formal semantics). On the other hand, it is a field deeply rooted in engineering which resulted in machines that have completely warped our society: the von Neumann architecture (the basis for digital computers), parallel processors (the new generation of multi-core machines), distributed computers (a prerequisite for the success of the internet and recent phenomena like grid computing). Consequently, computer science has inherited its research methods from the same disciplines: on the one hand, the mathematical approach with axioms, postulates and proofs; on the other hand the engineering approach with quantification, measurements and comparison.

Software Engineering research in particular has suffered from this identity crisis, and several authors have argued the need for stronger research methods [1], [2], [3]. Moreover, software engineering research —with its emphasis on processes and team work— must also take into account group dynamics and cognitive factors, hence borrows research methods from sociology and psychology as well [4], [5]. Certainly, with innovations like distributed development and open source release, software engineering is at the forefront of introducing new communication paradigms, hence is itself a testbed for experiments in social sciences.

Consequently, case studies are a dominant research method within software engineering [6]. This should come as no surprise, since case studies are particularly useful to *"investigate a contemporary phenomenon within its real-life context; when the boundaries between phenomenon and context are not clearly evident"* [7]. Nevertheless, the term "case study" is used quite liberally, ranging from small toy examples for illustrative purposes to full blown benchmarks with all the required threats to validity. Consequently, a PhD student should understand the full spectrum of what computer scientists refer to as a "case study".

[1] "theory" should be interpreted here in the scientific sense of the word, i.e. an important feature of nature supported by facts gathered over time.

II. Publication Culture

The computing science field has a long tradition of using conference publications as the primary unit of dissemination, which is in contrast with other scientific disciplines (physics, biology, ...) where emphasis is on journal publications. This is a hotly debated issue within the community (see among others [8], [9], [10]) and will not change in the foreseeable future.

Indeed, scientific publications drive today's academic research, yet scientific publications are *not* a goal in themselves! Rather they are a means towards a goal, namely recording and disseminating the contributions to human knowledge — the proverbial "standing on the shoulders of giants". Nevertheless, quantifying scientific output is a common practice these days with scientific publications being the most visible aspect. Hence there is a tremendous pressure on PhD students to organise a PhD around a collection of peer reviewed articles. A common template is to have (a) several workshop papers outlining research idea(s) in the beginning of the process; (b) two conference publications halfway in the PhD process detailing some of the findings; (c) a journal publications summarizing the main contribution for archival reference.

Given this template, a (computer) scientist is expected to write a significant number of peer reviewed papers. Consequently, a PhD student should understand the peer review process, in order to increase the chances of success.

III. Goals for the Tutorial

This tutorial is aimed at PhD students who want to have a better grasp on what exactly is "good" research. We explore the role of research methods in computer science, drawing upon practical examples from empirical approaches in software engineering. Given the need for stronger research methods (Section I) and the ongoing pressure on publication output (Section II) the tutorial wants PhD students to ...

- Name and explain different approaches to conduct computer science research (i.e. feasibility study, case study, comparative study, literature survey).
- Understand the peer reviewing process inherent in academic research, including the implications it has for their own research (i.e. writing papers).

Acknowledgment

This work has been sponsored by the Research Foundation Flanders (FWO) sponsoring a sabbatical leave of Prof. Serge Demeyer.

REFERENCES

[1] M. V. Zelkowitz and D. R. Wallace, "Experimental models for validating technology," *Computer*, vol. 31, pp. 23–31, May 1998.

[2] A. Höfer and W. F. Tichy, "Status of empirical research in software engineering," in *Empirical Software Engineering Issues. Critical Assessment and Future Directions*, ser. Lecture Notes in Computer Science, V. Basili, D. Rombach, K. Schneider, B. Kitchenham, D. Pfahl, and R. Selby, Eds. Springer Berlin / Heidelberg, 2007, vol. 4336, pp. 10–19.

[3] M. Shaw, "What makes good research in software engineering?" *International Journal on Software Tools for Technology Transfer (STTT)*, vol. 4, pp. 1–7, 2002.

[4] D. E. Perry, A. A. Porter, and L. G. Votta, "Empirical studies of software engineering: a roadmap," in *Proceedings of the Conference on The Future of Software Engineering*, ser. ICSE '00. New York, NY, USA: ACM, 2000, pp. 345–355.

[5] S. Hanenberg, "Faith, hope, and love: an essay on software science's neglect of human factors," *SIGPLAN Notices*, vol. 45, pp. 933–946, 2010.

[6] P. Runeson and M. Höst, "Guidelines for conducting and reporting case study research in software engineering," *Empirical Software Engineering*, vol. 14, pp. 131–164, 2009.

[7] R. K. Yin, *Case Study Research: Design and Methods, 3 edition*. Sage Publications, 2002.

[8] R. Andonie and I. Dzitac, "How to write a good paper in computer science and how will it be measured by ISI web of knowledge," *International Journal of Computers, Communications and Control*, vol. V, no. 4, pp. 432–446, 2010.

[9] J. Grudin, "Technology, conferences, and community," *Commun. ACM*, vol. 54, pp. 41–43, February 2011. [Online]. Available: http://doi.acm.org/10.1145/1897816.1897834

[10] C. Ghezzi, "Reflections on 40+years of software engineering research observed through ICSE: an insider's view," Keynote at ICSE 1999 Conference, 1999.