'*Follow the moon*' development: writing a systematic literature review on Global Software Engineering Education

Tony Clear
Faculty of Design & Creative Technologies
Auckland University of Technology
Private Bag 92006
Auckland 1010
+64-9-921-9999
tony.clear@aut.ac.nz

ABSTRACT

This presentation reflects on method and practice in Computer Science Education Research, through introducing the process of conducting a Systematic Literature Review. While Systematic Literature Reviews are an established research method within the Software Engineering discipline, they are a relatively unfamiliar research approach within Computer Science Education. Yet research disciplines can be strengthened by borrowing and adapting methods from other fields. I reflect on the rationale and underlying philosophy behind Systematic Reviews, and the implications for conducting a rigorous study and the quality of the resulting outputs. This chronicle of the journey of an ITiCSE working group, outlines the process we adopted and reflects on the methodological and logistical challenges we had to overcome in producing a review titled *Challenges and Recommendations for the Design and Conduct of Global Software Engineering Courses*. I conclude by discussing how systematic literature reviews can be adapted to an undergraduate teaching setting.

CCS Concepts

• Social and professional topics • Social and professional topics~Computing education • Software and its engineering

Keywords

Global Software Engineering Education; International Collaboration; Open Ended Group Project; Capstone; Teaching and Learning; Global Software Development; Evidence-Based Software Engineering; Systematic Literature Review; Research Methodology.

1. INTRODUCTION

This presentation reflects on approaches to conducting Computer Science Education Research, and introduces the Systematic Literature Review as one methodological option. Systematic Literature Reviews (SLRs) are an established research method within the Software Engineering discipline, but they have been less commonly adopted within Computer Science Education. Yet research disciplines can be strengthened by borrowing and adapting methods from other fields [14]. So this discussion of the underpinnings, the merits and issues of SLRs and the practicalities of conducting an SLR, indicates their potential for adoption within Computer Science Education research and practice.

Koli Calling 2015, November 19-22, 2015, Koli, Finland
ACM 978-1-4503-4020-5/15/11.
http://dx.doi.org/10.1145/2828959.28350190

2. UNDERPINNINGS

SLRs have their origins within the 'evidence-based' movement, initiated through Evidence Based Medicine [12]. Evidence-based research and practice was developed initially in medicine because research indicated that medical advice based solely on expert opinion was not as reliable as advice based on the accumulation of results from scientific experiments, and could lead to adverse patient outcomes. Moreover, remaining current with the literature in over 20,000 medical journals presented unsurmountable challenges for medical practitioners.

Evidence based approaches have been actively adopted within the software engineering discipline, by Kitchenham and colleagues [12], who assert that the goal of Evidence-based Software Engineering [EBSE] is:

> *''To provide the means by which current best evidence from research can be integrated with practical experience and human values in the decision making process regarding the development and maintenance of software''* [9].

A systematic literature review (SLR), is a methodologically rigorous review of research results [13] which contrasts with more ad-hoc processes.

> *"The aim of an SLR is not just to aggregate all existing evidence on a research question; it is also intended to support the development of evidence-based guidelines for practitioners.*
>
> *The end point of EBSE is for practitioners to use the guidelines to provide appropriate software engineering solutions in a specific context"* [9].

A good example of such a Software Engineering study reviewing the state of practice *Global Software Development and Collaboration: Barriers and Solutions* [16] served as the inspiration for this educationally focused study.

3. ITiCSE 2015 WORKING GROUP REPORT

We duly submitted, and had accepted, a proposal for an ITiCSE working group in 2015, initially titled "*Approaches to the Design and Conduct of Global Software Engineering Courses*". Working groups (WGs) are a distinctive and popular feature of the annual ITiCSE conference, whereby WG participants meet and work with likeminded international colleagues aiming to create a substantial paper published later in the year as a supplement to the

conference proceedings. Groups work before, during and after the conference and reports undergo a full peer review process.

Our starting perspective was that work in the area was fragmented and had not cumulatively built on prior work, so guidelines for practitioners were few. After reviewing prior systematic reviews in the area [15, 7], which covered aspects of the 'state of the art', we tightened our focus to address not only the challenges, but also (consistent with Kitchenham [9], and Noll [16]) to produce a set of recommendations for GSE-Ed practitioners.

Accordingly we aimed to review the literature to answer two research questions:

RQ1: What are the challenges in delivering GSE courses to SE Students?

RQ2: What are the recommendations for delivering GSE courses to SE Students?

The aim was to produce a broad ranging resource for global software engineering educators, which would support efforts to design and conduct successful courses between globally dispersed institutions and student teams.

4. GLOBAL SOFTWARE ENGINEERING EDUCATION

In this working group we had to wrestle with scoping the study and defining terms. A working definition for GSD/GSE is given below:

> *In GSD, stakeholders from different national and organizational cultures and time zones are involved in developing software…and tasks at various stages of the software lifecycle may be separated and implemented at different geographic locations coordinated through the use of information and communication technologies*…[8]

Defining Software Engineering itself is contentious, but we settled on a working definition of GSE-Ed for the purposes of this paper.

> *GSE-Ed represents a combination of learning and teaching strategies that prepare students for GSE/GSD.*

where GSD adopts the definition from Holmström above [8].

5. METHOD - Systematic Literature Review (SLR) Procedure

We used the SLR procedure defined by Kitchenham and Charters [10] to identify, evaluate and interpret the available published studies relating to our research questions.

In accordance with systematic review guidelines [10] we took the following steps:

1. Identify the need for a systematic literature review
2. Formulate review research question(s)
3. Source selection - carry out a comprehensive, exhaustive search for primary studies
4. Assess and record the quality of included studies
5. Classify data needed to answer the research question(s)
6. Extract data from each included study
7. Summarise and synthesise study results (meta-analysis)
8. Interpret results to determine their applicability
9. Write-up study as a report

These steps are detailed in our protocol [3], which is based on the process used by Beecham and colleagues [2]. We developed our protocol by piloting the process with three researchers who performed searches based on rules given in the protocol.

5.1 Document retrieval

Table 1 shows the number of papers selected from our source databases. Having removed duplicates across databases, we were left with 649 unique papers to consider including in our study. The table shows the several filtering phases used to establish our final set of 82 papers.

Table 1: Paper selection process

Selection Process	# papers	Validation process
Database papers found	762	Check for known papers
Duplicates removed (-136)	626	Agreement across researcher
Direct searches (+23)	649	n/a
Sift based on Title and Abstract (478 rejected)	171	All 647 papers assessed by 2 researchers
Full papers reviewed (63 rejected) during data extraction process	108	9 papers reviewed by 2 researchers to check agreement
Replicated studies removed (26 removed) to produce final set of 82 papers.	82	8 papers discussed by group to agree which paper to retain

5.1.1 Developing a Coding Scheme

We synthesised the data extracted as text snippets into themes using content analysis. Content analysis aims to identify the meaning of text by assigning a code that conveys that meaning. As such, it is essential that the coding scheme used to convey meaning is accurate. Also, it must be repeatable: different researchers should assign the same code to a given text fragment, and the same researcher should assign the same code to a given fragment when analyzed a week or a month later.

But a good coding scheme is not only accurate and repeatable; if the number of codes is small, and their definitions are clear, the coding process becomes straightforward and can be completed easily and quickly. Our coding method is adapted from Noll [17,18] and comprises the following steps:

1) Create Initial Type Set:

This initial code set attempted to capture the wide variety of meanings, and comprised a total of 110 minor codes that were grouped in 18 major categories that reflected both research questions.

2) Aggregate into Type Categories: Next, the list of codes was examined to discover broader categories. Codes with similar meaning were grouped together, and coalesced into a single category. The goal was to refine the list into a handful of categories with distinct meanings, so that it was easy to decide to which category a given text fragment belonged. The categories were given names which became the codes that were assigned to text fragments.

The initial set of 18 major codes was refined to seven categories, shown in Table 2, that capture meaning appropriate to the research questions for this study. These seven major categories were used to code the remaining data extractions.

Table 2: Final Set of 7 codes, plus their associated Minor classifications

1. GLOBAL DISTANCE Increased complexity Cultural Temporal Linguistic General Organisational Skills	4. INFRASTRUCTURE Tools Technical issues Version Control
2.STAKEHOLDER/ ROLE Client Instructor Student University representative Role conflict	5. DEVELOPMENT PROCESS S/w Development Process Requirements Design Coding Testing System/code integration
3.PEOPLE/SOFT ISSUES Motivation Trust Stress Self-awareness	6. CURRICULUM/PEDAGOGY Course design Learning Outcomes 7. TEAMWORK/TEAM CREATION Synergy Task allocation

5.2 Validation

5.2.1 Validation 1 - Paper Selection based on Title and Abstract.

Our paper selection followed a repeatable, auditable and reliable process as outlined in our protocol [3]. The initial list of papers was derived from several sources (see Section 3.3). After eliminating papers that were duplicated across sources, 649 primary papers were identified as potential sources for this study (see Table 1).

Three authors performed the initial screening of this list of papers in three stages. The aim was to only include those papers that met our inclusion/exclusion criteria, and in each stage refine the criteria and confirm the coding consistency. .

This first filtering based on abstract and title, resulted in 171 accepted papers, to go to the next phase of analysis which was to read the full paper and complete data extraction forms.

5.2.2 Validation 2 - Paper selection (Full Paper)

A generic data extraction form was developed to record the context of the paper, and how each paper addressed our research questions.

When going through the papers in detail, we rejected 62 more papers as they failed to meet our inclusion/exclusion criteria. We updated the protocol to reflect issues in the criteria. For example we realized that we should exclude secondary reviews from our study, since we had captured many of the primary papers within the reviews, and the inclusion of these papers would have resulted in duplicate findings.

5.2.3 Validation 3 -Data synthesis

In order to test our synthesised codes multiple authors independently mapped the text to the codes. Every code snippet was coded by at least two reviewers.

6. REFLECTIONS

It is not the purpose of this presentation to present the findings of the working group, although the elaboration above has given an indication of the broad process adopted, but the WG has produced a draft paper [6] (currently under review), and we are hopeful that it will stand scrutiny.

The process has involved a mixed emotional journey, at times exhilarating, exhausting, frustrating, and confusing. Academically for me it has been one of those peak experiences, working with intelligent, knowledgeable and committed academic colleagues on an ambitiously scoped and intensely challenging project. The project has presented challenges conceptually and methodologically, not to mention substantively. In those pressure stages of finalising the analysis and report, characterised by working in a globally distributed team, (at times in a 24 by 7 mode 'following the moon', depending on who was awake) and to a tight deadline across several different time-zones. As with all teams we have also had members juggling other commitments, work and travel, holidays and significant bouts of sickness.

We have had points of significant disagreement, apparently occasioned by the practicalities of process, but actually underpinned by epistemology and the conceptual basis of the method, but we have managed to navigate our arguments respectfully and collegially towards stronger outcomes, whether working face-to-face or as a globally distributed team "eating our own dogfood" on the topic under investigation.

7. EDUCATIONAL RECOMMENDATIONS

Kitchenham and colleagues [11] have reported a number of "educational benefits of mapping studies:

- They teach students how to search the literature and organize the results of searches.
- For PhD students, they provide a valuable means of initiating their research.
- They provide students with reusable research skills.
- They give a good overview of the literature.
- They are challenging but enjoyable.

The main problem was that they require considerable effort" [ ].

As one practical educational outcome of the working group, following his exposure to the process, John Barr has decided to introduce his undergraduate students at Ithaca College to SLRs, a practice that I have applied at Auckland University of Technology with some success [5], so we wish him luck!

8. CONCLUSION

This presentation reflects upon the origins of evidence based research approaches in particular within the software engineering discipline. It reveals the challenging process of conducting a systematic literature review on global software engineering education, itself within a globally distributed setting. Approaches to ensuring the quality of the process and resulting outcomes are discussed. The potential for the wider adoption of SLRs as a research and teaching method in CS Education is posed.

However in presenting this perspective, I do acknowledge that evidence based approaches and SLRs have their critics. There are other approaches to being systematic in the conduct of reviews [19, 1], which could equally be considered.

In the context of education in particular, evidence based approaches have been criticised as being excessively utilitarian and narrow. Proponents have been cast as ill-informed through privileging positivist approaches, which are highly restrictive and indeed inappropriate in their stipulation of medically informed controlled experimental research methods as a gold standard. Moreover they have been criticized for setting a top-down agenda lacking in a democratic value base [4]

So while research disciplines can be strengthened by borrowing and adapting methods from other fields [14], they can also be damaged if such borrowing is wholesale and indiscriminate. Researchers need to consider the context of origin, the goals and the underpinning value systems of methods translated from other disciplines. For researchers in the educational disciplines, the scope of evidence based research needs to be set within the moral and democratic dimensions of education, which must not be lost in a simplistic quest for 'what works' [4]. Yet sharing knowledge of effective teaching and learning practices in a changing and emerging discipline clearly serves a valuable function for CS educators. As ever, research methods must be appropriately chosen and tailored to the purpose of any investigation. But, for all that, the message of this presentation is that systematic literature reviews definitely have value as a method for research and teaching in CS Education.

ACKNOWLEDGEMENTS

My working group colleagues: Sarah Beecham, John Barr, Mats Daniels, Roger McDermott, Michael Oudshoorn, Airina Savickaite, John Noll.

This work was supported, in part, by Science Foundation Ireland grants 10/CE/I1855 and 13/RC/2094 to Lero - the Irish Software Research Centre (www.lero.ie), and by contract CF 2014 4348 from the European Regional Development Fund and Enterprise Ireland.

References

[1] Badampudi, D., Wohlin, C., and Petersen, K., 2015. Experiences from using snowballing and database searches in systematic literature studies. In *Proceedings of the 19th International Conference on Evaluation and Assessment in Software Engineering* ACM, 17.

[2] Beecham, S., Baddoo, N., Hall, T., Robinson, H., and Sharp, H., 2006. *Protocol for a systematic literature review of motivation in software engineering.* Technical Report UH-CS-TR-453 Report.

[3] Beecham, S., T Clear, J Barr, and Noll, J., 2015. *Protocol for a Systematic Literature Review on "Approaches to the Design and Conduct of Global Software Engineering Courses.".* (ITiCSE Working Group One: Technical Report No. Lero_TR_2015_01) Report.

[4] Biesta, G., 2007. Why "what works" won't work: Evidence-based practice and the democratic deficit in educational research. *Educational theory 57*, 1, 1-22.

[5] Clear, T., 2012. Systematic Literature Reviews and Undergraduate Research. *ACM Inroads 3*, 4 (Dec), 10-11. DOI= http://dx.doi.org/10.1145/2381083.2381087.

[6] Clear, T., Beecham, S., Barr, J., Daniels, M., Mcdermott, R., Oudshoorn, M., Savickaite, A., and Noll, J., 2015. Challenges and Recommendations for the Design and Conduct of Global Software Engineering Courses: A Systematic Review. In *Proceedings of the Working Group Reports of the 2015 on Innovation & Technology in Computer Science Education Conference*, N. Ragonis and P. Kinnunen Eds. ACM, New York, tba. DOI= http://dx.doi.org/tba.

[7] Fortaleza, L.L., Conte, T., Marczak, S., and Prikladnicki, R., 2012. Towards a GSE international teaching network: Mapping Global Software Engineering courses. In *Collaborative Teaching of Globally Distributed Software Development Workshop (CTGDSD)*, 1-5. DOI= http://dx.doi.org/10.1109/CTGDSD.2012.6226944.

[8] Holmström, H., Fitzgerald, B., Ågerfalk, P.J., and Conchúir, E.Ó., 2006. Agile practices reduce distance in global software development. *Information Systems Management 23*, 3, 7-18.

[9] Kitchenham, B., Brereton, O.P., Budgen, D., Turner, M., Bailey, J., and Linkman, S., 2009. Systematic literature reviews in software engineering–a systematic literature review. *Information and Software Technology 51*, 1, 7-15.

[10] Kitchenham, B. and Charters, S., 2007. Guidelines for performing systematic literature reviews in software engineering. In *Technical report, Ver. 2.3 EBSE Technical Report.* , Keele University, UK.

[11] Kitchenham, B.A., Budgen, D., and Brereton, O.P., 2011. Using mapping studies as the basis for further research–a participant-observer case study. *Information and Software Technology 53*, 6, 638-651.

[12] Kitchenham, B.A., Dyba, T., and Jorgensen, M., 2004. Evidence-Based Software Engineering. In *Proceedings of the 26th International Conference on Software Engineering* (2004), IEEE Computer Society, 999432, 273-281.

[13] MacDonell, S., Shepperd, M., Kitchenham, B., and Mendes, E., 2010. How reliable are systematic reviews in empirical software engineering? *Software Engineering, IEEE Transactions on 36*, 5, 676-687.

[14] McGrath, J., 1985. *Validity and the Research Process*. SAGE, Beverly Hills, California.

[15] Monasor, M.J., Vizcaino, A., Piattini, M., and Caballero, I., 2010. Preparing Students and Engineers for Global Software Development: A Systematic Review. In *5th IEEE International Conference on Global Software Engineering (ICGSE)*, 177-186. DOI= http://dx.doi.org/10.1109/ICGSE.2010.28.

[16] Noll, J., Beecham, S., and Richardson, I., 2010. Global Software Development and Collaboration: Barriers and Solutions *ACM Inroads 1*, 3 (Sept), 66-78.

[17] Noll, J., Seichter, D., and Beecham, S., 2012. A qualitative method for mining open source software repositories. In *Open Source Systems: Long-Term Sustainability* Springer, 256-261.

[18] Noll, J., Seichter, D., and Beecham, S., 2013. Can Automated Text Classification Improve Content Analysis of Software Project Data? In *Empirical Software Engineering and Measurement, 2013 ACM/IEEE International Symposium on* IEEE, 300-303.

[19] Wohlin, C., 2014. Guidelines for snowballing in systematic literature studies and a replication in software engineering. In *Proceedings of the 18th International Conference on Evaluation and Assessment in Software Engineering* ACM, 38.