Performing Systematic Literature Reviews in Software Engineering

David Budgen
Department of Computer Science
Durham University
Durham DH1 3LE
U.K.
David.Budgen@durham.ac.uk

Pearl Brereton
School of Computing & Mathematics
Keele University
Staffordshire ST5 5BG
U.K.
o.p.brereton@cs.keele.ac.uk

ABSTRACT

Context: Making best use of the growing number of empirical studies in Software Engineering, for making decisions and formulating research questions, requires the ability to construct an objective summary of available research evidence. Adopting a systematic approach to assessing and aggregating the outcomes from a set of empirical studies is also particularly important in Software Engineering, given that such studies may employ very different experimental forms and be undertaken in very different experimental contexts.
Objectives: To provide an introduction to the role, form and processes involved in performing Systematic Literature Reviews. After the tutorial, participants should be able to read and use such reviews, and have gained the knowledge needed to conduct systematic reviews of their own.
Method: We will use a blend of information presentation (including some experiences of the problems that can arise in the Software Engineering domain), and also of interactive working, using review material prepared in advance.

Categories and Subject Descriptors

D.2.0 [**Software Engineering**]: General

General Terms

Experimentation, Measurement

Keywords

Systematic Literature Review, Evidence

1. WHAT THIS TUTORIAL IS ABOUT

The *goal* of this tutorial is to introduce the practice of *Systematic Literature Review*, with the *objectives* of:

1. Introducing the concepts involved in systematic literature review and providing a set of guidelines that can be used for conducting (and reading) them.
2. Reporting on our experiences with conducting a number of systematic literature reviews.
3. Providing some 'hands-on' experience of reading a review and of some of the activities involved in conducting a review.

The tutorial is intended for anyone who is involved in research or in the evaluation of Software Engineering technologies and products/services. It is also appropriate for postgraduate students, since it addresses the question of how a candidate can provide a balanced *Background* chapter for their thesis, and ensure that their research is firmly grounded in an objective assessment of the current state of the art.

2. WHY IS THIS IMPORTANT?

Research papers and PhD theses in Software Engineering normally include sections or chapters that are intended to provide background, giving the context within which the particular work is set. In reviewing the existing literature that addresses a particular topic, the review process is rarely underpinned by any clear and systematic procedures for ensuring that the literature is surveyed in an objective manner and that all relevant material is included–not just the material that supports the arguments being put forward!

Appropriate procedures and frameworks do exist, including in other disciplines or branches of computing, but they are not widely known or employed in Software Engineering. Indeed, a survey of publications in Software Engineering journals has indicated that the research methods employed by Software Engineers are not drawn from a particularly wide set of reference disciplines [3, 2].

The introduction of the use of systematic evaluation processes, and in particular, of Systematic Literature Review [5], will help to improve the situation for both Software Engineering and Computer Science in a number of ways. Firstly, as a means of obtaining an objective summary of research evidence concerning a topic or phenomenon [6]. Secondly, authors and students will benefit by having a clear set of procedures to follow in reviewing background material for their thesis, and for identifying where this supports or conflicts with their own work. Thirdly, by producing better quality reviews and evaluations it will improve the quality of papers and theses and hence of the review and examination processes. Finally, the experience so gained will provide an element of transferable skill for a student to employ in their later research careers.

ICSE'06, May 20–28, 2006, Shanghai, China.
ACM 1-59593-085-X/06/0005.

3. SYSTEMATIC LITERATURE REVIEWS

One of the major tools used to support an evidence-based paradigm in other domains is the generation of systematic literature reviews, used to aggregate the experiences gained from a range of different studies in order to answer a specific research question [4]. (Henceforth we will term this a *systematic review* for brevity.) Such reviews employ carefully-defined *protocols* to determine which studies are to be included, as well as for analysing their contribution in as unbiased a form as possible [5, 7]. While performing systematic reviews in Software Engineering does present some difficulties, our own experience suggests that the outcomes are far more convincing than the more informal approach to generating background chapters commonly encountered in many research papers as well as PhD theses [1]. Indeed, work currently in progress suggests that the outcomes from such a review may well contradict accepted wisdom–as indeed has happened in Medicine.

A systematic review is *a means of identifying, evaluating and interpreting all available research relevant to a particular research question, or topic area, or phenomenon of interest* [5]. The studies that form the basis for the systematic review are termed *primary* studies, and the systematic review is itself a form of *secondary* study.

As with other disciplines that employ systematic reviews, there are a number of different reasons why these should be undertaken. Some common ones are:

- to summarise existing evidence concerning a practice or technology;
- to identify where there are gaps in current research, in order to help determine where further investigation might be needed;
- to help position new research activities;
- to examine how far a given hypothesis is supported or contradicted by the available empirical evidence.

One obvious question is what differentiates a *systematic* review from any other form of literature review? There are a number of characteristics, including the following.

- The start point for a systematic review is a *review protocol* that specifies the research question being addressed and the methods that will be employed in the review process.
- Systematic reviews employ a defined *search strategy* for identifying as much as possible of the relevant literature.
- The search strategy employed is *documented*, enabling the reader of a review to assess how rigorous and complete this is.
- A systematic review involves specifying explicit *inclusion* and *exclusion* criteria to determine whether each potential primary study should be included.
- The review process specifies the information that is to be obtained from each primary study, including any quality criteria to be used when evaluating a primary study.

In addition, the use of a systematic review is a pre-requisite for employing any form of quantitative meta-analysis to the aggregated results.

4. THE REVIEW PROCESS

The review process as proposed for Software Engineering has three phases [5]: *planning* the review; *conducting* the review; *reporting* the outcomes from the review.

A key element is the development of the *review protocol* that specifies the methods to be used in undertaking a specific review. By defining this at the outset, the possibility of researcher bias is reduced, since without a protocol, such aspects as the selection of primary studies or the form of analysis employed may be driven by researcher expectations or preconceptions. (In medicine, a review protocol is usually subjected to a peer review process.) The tutorial will address the design of a protocol and will study some that have already been employed in Software Engineering.

5. SOME EXPERIENCES

We have undertaken, and are undertaking, a number of systematic reviews, partly to validate our ideas about how these should be conducted in Software Engineering. While our experiences are generally very positive about the use of systematic review, they have revealed some rather domain-specific issues. Two that we (and others) have noted are:

- the poor quality of search engines available for use in finding primary sources, and the difficulty of getting consistent results from these;
- the variable quality of the abstracts available for Software Engineering papers, with analysts often needing to consult other parts of a paper in order to determine whether or not a paper should be included.

We will address these experiences in the tutorial, and suggest some of the ways in which they can be overcome or otherwise mitigated.

6. REFERENCES

[1] P. Brereton, B. Kitchenham, D. Budgen, M. Turner, and M. Khalil. Employing Systematic Literature Review: An Experience Report. Technical Report TR 05/01, School of Computing & Mathematics, Keele University, 2005.

[2] R. Glass, V. Ramesh, and I. Vessey. An Analysis of Research in Computing Disciplines. *Communications of the ACM*, 47:89–94, June 2004.

[3] R. Glass, I. Vessey, and V. Ramesh. Research in software engineering: An analysis of the literature. *Information & Software Technology*, 44:491–506, 2002.

[4] K. Khan, R. Kunz, J. Kleijnen, and G. Antes. *Systematic Reviews to Support Evidence-Based Medicine: How to review and apply findings of healthcare research.* Royal Society of Medicine Press Ltd., 2003.

[5] B. Kitchenham. Procedures for undertaking systematic reviews. Technical Report TR/SE-0401, Department of Computer Science, Keele University and National ICT, Australia Ltd, 2004. Joint Technical Report.

[6] B. Kitchenham, T. Dybå, and M. Jørgensen. Evidence-based software engineering. In *Proceedings of ICSE 2004*, pages 273–281. IEEE Computer Society Press, 2004.

[7] J. Webster and R. Watson. Analysing the Past to Prepare for the Future: Writing a Literature Review. *MIS Quarterly*, 26:xiii–xxiii, 2002.