Evidence-based Software Engineering

Barbara A. Kitchenham[1,3]
barbara@cs.keele.ac.uk

Tore Dybå[2,4]
tore.dyba@sintef.no

Magne Jørgensen[2]
magnej@simula.no

[1]*National ICT Australia, Locked Bag 9013 Alexandria, NSW 1435, Australia*
[2]*Simula Research Laboratory, P.O. Box 134, NO-1325 Lysaker, Norway*
[3]*Dept. of Computer Science, Keele University, Staffordshire ST5 5BG, UK*
[4]*Dept. of Software Engineering, SINTEF ICT, NO-7465 Trondheim, Norway*

Abstract

***Objective**: Our objective is to describe how software engineering might benefit from an evidence-based approach and to identify the potential difficulties associated with the approach.*

***Method**: We compared the organisation and technical infrastructure supporting evidence-based medicine (EBM) with the situation in software engineering. We considered the impact that factors peculiar to software engineering (i.e. the skill factor and the lifecycle factor) would have on our ability to practice evidence-based software engineering (EBSE).*

***Results**: EBSE promises a number of benefits by encouraging integration of research results with a view to supporting the needs of many different stakeholder groups. However, we do not currently have the infrastructure needed for widespread adoption of EBSE. The skill factor means software engineering experiments are vulnerable to subject and experimenter bias. The lifecycle factor means it is difficult to determine how technologies will behave once deployed.*

***Conclusions**: Software engineering would benefit from adopting what it can of the evidence approach provided that it deals with the specific problems that arise from the nature of software engineering.*

1. Introduction

In the last decade, medical research had changed dramatically as a result of adopting an evidence-based paradigm. In the late 80s and early 90s, studies showed on the one hand that failure to organise medical research in systematic review could cost lives [5] and on the other hand that the clinical judgement of experts compared unfavourably with the results of systematic reviews [1]. Since the publication of these influential papers, many medical researchers have adopted the evidence-based paradigm. Sackett *et al*. [14] point out that since 1992, the number of articles about evidence-based practice has grown from 1 publication in 1992 to about a thousand in 1998 and international interest has led to the development of 6 evidence-based journals specialising in systematic reviews.

The success of evidence-based medicine has prompted many other disciplines that provide services to, or for, members of the public to attempt to adopt a similar approach, including for example psychiatry[1], nursing[2], social policy[3], and education[4]. We do not suggest that software engineers should adopt a new practice just because "everyone else is doing it", particularly since the evidence-based movement has its critics. For example, Hammersley points out that research is fallible, relies on generalisations that may be difficult to interpret, and is often insufficient for determining appropriate means for delivering best practice [6]. However, we believe that a successful innovation in a discipline that, like software engineering, attempts to harness scientific advances for the benefit of society, is worth investigating.

Thus, in this paper we discuss the possibility of evidence-based software engineering using an analogy with medical practice. We describe the scientific and technical infrastructure needed to support EBSE. We also identify two areas where the analogy with medicine breaks down. This allows us to identify a number of serious problems that need to be resolved if EBSE is to become a reality.

[1] www.med.nagoya-cu.ac.jp/psych.dir/ebpcenter.htm
[2] www.york.ac.uk/healthsciences/centres/evidence/cebn.htm
[3] www.evidencenetwork.org
[4] cem.dur.ac.uk/ebeuk/EBEN.htm

Proceedings of the 26th International Conference on Software Engineering (ICSE'04)
0270-5257/04 $20.00 © 2004 **IEEE**

2. Why evidence is important in software engineering

Initially it is worthwhile considering why evidence would be beneficial to software developers, users and other stakeholders e.g. public purchasing bodies, certification bodies and the general public. EBSE is potentially important because of the central place software intensive systems are starting to take in everyday life. For example, current plans for advanced life-critical systems such as drive-by-wire applications for cars and wearable medical devices have the potential for immense economic and social benefit but can also pose a major threat to industry, to society, and to individuals. If systems are reliable, usable and useful, the quality of life of individual citizens will be enhanced. However, there are far too many examples of systems that have not only wasted large amounts of public money but have also caused harm to individual citizens (e.g. the automated command and control system for the London Ambulance Service). Individual citizens have a right to expect their governments to properly administer tax revenues used to commission new software systems and put in place controls to minimise the risk of such systems causing harm.

There are many strategies to improve the dependability of software involving the adoption of "better" software development procedures and practices. At a high level, the Capability Maturity Model and SPICE suggest procedures for improving the software production process. In addition, the professional bodies are establishing procedures for certification of individual software engineers. However, the high level process and the individual engineers are constrained by the specific technologies (methods, tools and procedures) they use. In most cases software is built with technologies for which we have insufficient evidence to confirm their suitability, limits, qualities, costs, and inherent risks. Thus, it is difficult to be sure that changing software practices will necessarily be a change for the better. It is possible that EBSE can provide the mechanisms needed to assist practitioners to adopt appropriate technologies and to avoid inappropriate technologies.

3. The goal of EBSE

The goal of evidence-based medicine (EBM) is "the integration of best research evidence with clinical expertise and patient values" [14]. By analogy, we suggest that the goal of evidence-based software engineering (EBSE) should be:

> *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.*

Thus EBSE would provide:

- A common goal for individual researchers and research groups to ensure that their research is directed to the requirements of industry and other stakeholder groups.
- A means by which industry practitioners can make rational decisions about technology adoption.
- A means to improve the dependability of software intensive systems, as a result of better choice of development technologies.
- A means to increase the acceptability of software-intensive systems that interface with individual citizens.
- An input to certification processes.

4. Practising EBSE

Sackett *et al.* [14] identify 5 steps that are needed to practice evidence-based medicine. These steps are shown in the second column of Table 1. Although there are some detailed medical references, it is easy to reformulate the steps to address evidence-based software engineering (see column 3 of Table 1). In fact, we would hazard a guess that at least part of the attraction that evidence-based medicine has for other disciplines is the ease with which the basic steps can be adapted to other fields. However, it is important to remember that even if high-level process steps for evidence-based practice appear to be similar for medicine and software engineering, this does not guarantee that the underlying scientific, technological and organisational mechanisms that support evidence-based medicine apply to evidence-based software engineering. For this reason we consider each step in more detail below.

The first point to note is that Sackett *et al.* [14] consider EBM from the viewpoint of an individual medical practitioner who needs to decide how to treat a particular patient exhibiting a particular set of symptoms. For EBSE our viewpoint is likely to be somewhat different. In software engineering organizations, individual developers seldom have the option to pick and choose the technologies they are going to use. Technology adoption is often decided either by project managers on a project by project basis, or by senior managers on a departmental or organizational basis. Furthermore, in software engineering, our concern is not usually the specific task to which the technology is applied but the outcome of the project of which the task is a part.

Table 1. Five steps used in Evidence-based Medicine and (by analogy) in Evidence-based Software Engineering.

Step	Evidence-based Medicine	Evidence-based Software Engineering
1	Converting the need for information (about prevention, diagnosis, prognosis, therapy, causation, etc) into an answerable question.	Converting the need for information (about development and maintenance methods, management procedures etc.) into an answerable question.
2	Tracking down the best evidence with which to answer that question.	Tracking down the best evidence with which to answer that question.
3	Critically appraising that evidence for its validity (closeness to the truth), impact (size of the effect), and applicability (usefulness in our clinical practice).	Critically appraising that evidence for its validity (closeness to the truth), impact (size of the effect), and applicability (usefulness in software development practice).
4	Integrating the critical appraisal with our clinical expertise and with our patient's unique biology, values and circumstances.	Integrating the critical appraisal with our software engineering expertise and with our stakeholders' values and circumstances.
5	Evaluating our effectiveness and efficiency in executing Steps 1-4 and seeking ways to improve them both for next time.	Evaluating our effectiveness and efficiency in executing Steps 1-4 and seeking ways to improve them both for next time.

Thus, in addition to the viewpoint of the individual practitioner, there are two other viewpoints that are important for EBSE in practice:

1. That of a project manager who wants to achieve a favourable outcome for a particular project.
2. That of a senior manager who wants to improve the performance of a department or organization as a whole.

In addition, we believe EBSE (and indeed EBM) place requirements on researchers:

- To improve the standard of individual empirical studies and systematic reviews of such studies.
- To identify outcome measures that are meaningful to practitioners.
- To report their results in a manner that is accessible to practitioners.
- To perform and report replication studies.

4.1. Defining an answerable question

Sackett *et al.* [14] suggests that a well-formulated question has three parts:

1. The study factor (e.g. the intervention, diagnostic test or exposure).
2. The population (the disease group or spectrum of the well population).
3. The outcomes.

Medical practitioners are usually interested in all forms of outcomes, so they usually concentrate on the first two parts. This would be the same for EBSE.

EBM researchers point out that it is important that the question is broad enough to allow examination of variation in the study factor and across populations. In EBSE, the study factor would be the technology of interest. The technology should not be specified at too high a level of abstraction e.g. design methods, software lifecycles, or management methods, but must be general enough to identify the majority of relevant empirical studies, for example OO methods, Agile methods, or Cost estimation methods. For some questions it may be necessary to be even more precise e.g. Contract-based specifications, Pair-programming, or Statistically-derived estimation models. It is even more difficult to determine the correct level of abstraction for specifying the population of interest. The population of interest may be categorised in many dimensions based on experience of technology users, types of problem addressed by the technology, application area. However, even fairly broad categories may be counter-productive if useful empirical evidence is lost by restrictions imposed by such categorisation.

4.2. Finding the best evidence

One of the reasons for formulating the question precisely is to help researchers and practitioners to find all relevant studies. According to the Australian National Health and Medical Research Council, there are over 20,000 journals in the biomedical field. A major problem for EBM is finding relevant papers from the massive amount of published work. Medical researchers and practitioners use a two-stage process:

1. They look for already published systematic reviews, i.e. papers that have already assembled all relevant reports on a particular topic.
2. They use the question of interest to construct search strings aimed at finding relevant individual studies.

However, they have a large amount of technological and scientific infrastructure to support them:

- There are several organisations (in particular the international Cochrane Collaboration, see www.cochrane.com) that assemble systematic reviews of studies of drug and medical procedures. To provide a central information source for evidence, the Cochrane Collaboration publishes systematic reviews in successive issues of *The Cochrane Database of Systematic Reviews*. These reviews are continually revised both as new experimental results become available and as a result of valid criticisms of the reports. The Cochrane Collaboration actively solicits comments on their reports (subject to published house rules).
- Some countries have established central abstracting services for medical research papers. The largest and most well-known is the Medline data base (www.nlm.nih.gov), which provides references and abstracts from 4600 biomedical journals.
- To reduce the problem of "publication bias", the Cochrane Collaboration provides a database for researchers to register that they are intending to perform a controlled trial. Publication bias is the phenomena that more "positive" results are published than "negative" results. This can lead to an overestimation of the effect size in systematic reviews and an under-reporting of risks. The Cochrane Collaboration Groups use the register to follow up all trials whether or not they are published.

Although we have no equivalent to the Cochrane Collaboration, there are many abstracting services that provide access to software engineering articles. Organizations such as the IEEE, with its database *IEEE Xplore*, and the ACM, with its *Digital Library* provide access to databases of articles. The articles are indexed by author names, and keywords and usually have links to abstracts and sometimes access to the original articles. Such indexing makes it easier to search for information regarding a problem area or find an answer to a specific question.

4.3. Critically appraising the evidence

The work of the Cochrane Collaboration and other national medical organisations (e.g. the Australian National Health and Medical Research Council) has radically changed the nature of medical research. Medical research has recognised that single studies (even the most rigorous double-blinded randomised controlled trials, RCTs) are insufficient to properly qualify a medical treatment. The emphasis now is on the accumulation of evidence from many independent experiments.

Critical appraisal in EBM has been supported by improved methodology both for systematic reviews and individual studies:

- Several organisations have produced guidelines for systematic reviews and evaluating evidence. The Cochrane collaboration publishes a handbook (*The Cochrane Reviewers Handbook*, March 2003). The Australian National Health and Medical Research Council publish a series of more general handbooks that consider experimental methods other than just RCTs (see www.health.gov.au/mrc). Importantly the Australian NHMRC makes a distinction between collating experimental evidence and packaging the evidence into tailored guidelines for various stakeholders.
- Medical journals have pressed for improvements in the conduct and reporting of individual experiments. A particular example is the CONSORT statement, which defines the standards for randomised, controlled trials (RCTs), see [13]. This statement has been adopted as the standard for reporting RCTs by all the most important medical journals.

This can be contrasted with the situation in empirical software engineering. Currently evidence related to software engineering technologies that is available is:

- *Fragmented and limited*. Many individual research groups undertake valuable empirical studies. However, because the goal of such work is either individual publications and/or post-graduate theses, there is sometimes little sense of overall purpose to such studies. Without having a research culture that strongly advocates systematic reviews and replication, it is easy for researchers to undertake research in their own areas of interest rather than contribute to a wider research agenda.
- *Not properly integrated*. Currently, there are no agreed standards for systematic reviews. Thus, although most PhD students undertake reviews of the "State of the Art" in their topic of interest, the quality of such reviews is variable, and they do not as a rule lead to published papers. There is little appreciation of the value of systematic reviews, for example, there is only one Computing journal that solicits reviews (*ACM Surveys*). Furthermore, if we consider "meta-analysis", which is a more statistically rigorous form of systematic review, there have been few attempts to apply meta-analytic techniques to software engineering not least because

of the limited number of replications. In general there are few incentives to undertake replication studies in spite of their importance in terms of the scientific method [11].

- *Without agreed standards*. There are no generally accepted guidelines or standard protocols for conducting individual experiments. The recent dispute between Berry and Tichy [4] and Sobel and Clarkson [17] over the conduct of an experiment into formal methods [16] makes it clear that empirical software engineering is badly in need of guidelines and protocols. Kitchenham et al. [9] proposed some preliminary guidelines for formal experiments and surveys. However, they do not address observational, and investigatory studies. Furthermore, because they attempt to address several different types of empirical study, the guidelines are not as specific, nor as detailed as the CONSORT statement.

4.4. Integrating the critical appraisal with software engineering expertise

In EBM, a doctor is expected to relate evidence to the needs of the specific patient. For example, the advice given to a patient with a particular disease may differ according to his/her age and gender and the severity of the symptoms he/she displays. Although there are opportunities for individual software engineers and managers to adopt EBSE principles, the decision to adopt a technology is often an organisational issue that is influenced by factors such as the organizational culture, the experience and skill of the individual software developers, the requirements of clients, project constraints, and the extent of training required. Thus, to use EBSE in practice may be more demanding than EBM because the decision-making structure and adoption process is often more complex.

We believe that EBSE would work well in an organization that has a strong commitment to process improvement, e.g. based on the recommendations in [3]. However, currently this does not appear to be happening. Research results are:

- *Not in wide-spread use in industry*. In our opinion, researchers often address issues that are not perceived to be of relevance to industry or present their results in a way that is virtually incomprehensible to decision makers in industry.
- *Not of perceived value to stakeholders*. Certification bodies, public purchasing bodies, and consumer groups should all be concerned about the quality of the techniques used to build software products. It is likely that any trust such groups have in the quality of software intensive products would be substantially undermined if they were aware that the choice of development techniques is based on fashion and hype rather than scientific evidence.

4.5. Evaluation of the process

Sackett *et al*. [14] recommend that individual doctors review the way in which they practice and teach EBM in order to improve their individual performance. For EBSE, this would involve propagating successful technologies throughout a company and preventing the spread of technologies that are unsuccessful. This concept fits well with the goals of software process improvement.

However, there is a broader level of feedback in medicine. For example, individual doctors are responsible for reporting unanticipated side-effects of drugs. This contributes to the evidence associated with a particular treatment and may lead to further basic research. It would be useful if this model could be applied in software engineering. However, in a competitive industry, there is little incentive for individual companies to assist their competitors by reporting good and bad experiences with new technologies. In addition, it is difficult for experiences with a technology to be disentangled from the particular context in which it was used.

4.6. Implications for EBSE

It is clear that a full-scale implementation of EBSE is an extremely ambitious goal. It cannot be achieved without extensive collaboration and long-term commitment among individual research groups world-wide, and active support from other stakeholders such as practitioners in industry, certification bodies etc. Furthermore it cannot be achieved without initial financial support from research funding agencies to enable the basic technological and methodological infrastructure to be established.

It is clear that individual practitioners and researchers can use some of the ideas of EBSE without extensive technical support. However, Sackett *et al*. suggest that the support infrastructure is a major reason for the widespread adoption of the evidence-based paradigm in medicine [14].

5. Scientific foundations

With enough resources, technological and organisational support for evidence-based software engineering could be put in place. However, such support would be of little value if there were fundamental differences between medicine and software engineering that would make evidence-based software engineering difficult or impossible.

5.1. The skill factor

One area where there is a major difference between medicine and software engineering is that most software engineering methods and techniques must be performed by skilled software practitioners. In contrast, although medical practitioners are skilled individuals, the treatments they prescribe (e.g. medicines and other therapeutic remedies) do not usually require skill to administer or to receive. Furthermore, it is noticeable that evidence-based surgery, which is far more analogous to software engineering than other types of medical practice, is far less advanced than evidence-based medicine [18].

The reason why skill presents a problem is because it prevents adequate *blinding*. In medical experiments (particularly drug-based experiments), the *gold standard* experiment is a double-blind randomised controlled trial (RCT). In a double-blind experimental trial neither the doctor nor the patient knows which treatment the patient is receiving. The reason double-blinded trials are required is to prevent patient and doctors expectations biasing the results. Such experimental protocols are impossible in software engineering experiments that rely on a subject performing a human-intensive task.

There are two complementary approaches we can adopt to address this issue:

1. We can develop and adopt experimental protocols that reduce experimenter and subject bias.
2. We can accept that our experiments are bound to be less rigorous than medical trials and attempt to qualify our experiments appropriately.

5.1.1. Experimental protocols. Although we cannot usually blind experimenters or subjects, we can use blinding in a number of ways to reduce the opportunity for bias by reducing the direct interaction between subjects and experimenters during the course of an experiment [8]:

- *Blind allocation to treatment groups*. When we run experiments to compare different techniques, computerised methods can be used to automate the random allocation of subjects to each technique..
- *Blind distribution of material*. Linked to blind allocation, computers can be used to distribute experimental materials to subjects.
- *Blind or automated marking*. If the task results cannot be linked directly to the treatment e.g. where the subjects are asked to answer some questions that test their understanding of a software document, the marker(s) should be blind to which treatment was used by the subjects (i.e. the task response should not identify the subject or the treatment to the marker). Occasionally marking can be computerised.
- *Blind analysis*. The results should be coded so the analyst does not know which treatment group is which.
- *Blind data collection*. Computerised systems can be used when information is required from subjects. This can also improve the accuracy of such data. Subjects will usually be more accurate in reporting information to a computer system, particularly if they are guaranteed anonymity.

In addition, we need to ensure that experimental designs allow for systematic subject difference due to skill, gender, and race by blocking, covariate analysis, or cross-over designs (where each subject acts as their own control).

Last but not least, we should encourage replication studies by experimenters who have no vested interest in the outcome of the study. However, we need to make sure replications are not too similar. It is important to vary experimental designs and experimental materials to avoid the risk of any common cause bias in replications [8].

5.1.2. Evaluating experiment quality. Even in EBM, it is recognised that it is sometimes impossible to perform randomised trials and evidence from other types of experiment may need to be considered. The Australian National Health and Medical Research Council have published guidelines for evaluating the quality of evidence [2]. They consider:

- *The strength of the evidence*. This has three elements: Level, Quality, and Statistical Precision. Level relates to the choice of study design and is used as an indicator to which bias has been eliminated by design. Quality refers to the methods used by the investigators to minimize bias within the study design. Statistical Precision refers to the P-value or the confidence interval.
- *Size of effect*. The distance of the estimated treatment effect from the null value and the inclusion of clinically important effects in the confidence interval.
- *Relevance of evidence*. The usefulness of the evidence in clinical practice, particularly the appropriateness of the outcome measures used.

These criteria appear to be equally valid for software engineering evidence.

5.2. The lifecycle issue

The other major difference between software engineering and medicine is that most software engineering techniques impact a part of the lifecycle in a way that makes the individual effect of a technique difficult to isolate:

- They interact with many other development techniques and procedures. For example a design method depends on a preceding requirements analysis. It must consider constraints imposed by the software and hardware platform and programming languages, timescales, and budget. It must be integrated with appropriate coding and testing techniques. Thus, it would be difficult to confirm that a design technique had a significant impact on final product reliability. In general, it is difficult to determine a causal link between a particular technique and a desired project outcome when the application of the technique and the final outcome are temporally removed from one another, and there are many other tasks and activities that could also affect the final outcome.
- The immediate outcomes of a software engineering technique will not necessarily have a strong relationship with final project outcomes. E.g. if you are interested in the effect design techniques have on application reliability (i.e. probability of failure in a given time period under defined operational conditions), measures of the design product (or design process) have no obvious relationship with the desired outcome. There are no good surrogate measures of product reliability that can be measured at the end of the design process.

There seem to be two major approaches to this issue:

1. We can experiment with individual techniques isolated from other techniques.
2. We can undertake large-scale empirical studies.

5.2.1. Experimenting with individual techniques. One approach is to experiment with individual techniques isolated from other techniques. However, this does not address the problem that the outcomes may be poor surrogates for the project outcomes practitioners are interested in. Furthermore, it leaves open the issue of how the technique will behave when it is integrated into a full development process.

5.2.2. Large-scale empirical studies. Another approach is to undertake large-scale empirical studies for example industrial case studies. The problem with case studies is that they cannot be performed with the rigor of proper experiments e.g. there will be limited opportunities for replication. Furthermore industrial studies suffer from the problem that case studies are performed within the context of a particular company. This means they are affected by the specific process standards, application area, staffing practices, software tools and other context specific factors. Thus, results from case studies cannot usually be generalised outside their specific context. This makes it difficult to extract meaningful evidence. One approach is to attempt to categorise context factors better [7]. However, this can lead to a combinatorial explosion of contextual information and the conclusion that each project is unique and non-comparable.

An alternative approach is to conduct a field (or quasi-) experiment. Such experiments lack random assignment of units to conditions but otherwise have similar purposes and structural attributes as randomised experiments [15]. Like all empirical studies, a causal inference from a quasi-experiment must meet the basic requirements for causal relationships: that cause precedes effect, that cause covaries with effect and that alternative explanations for the causal relationship are implausible. The first two of these are easily accomplished in all experiments. In randomised experiments, the third requirement is met by ensuring that alternative explanations are randomly distributed over the experimental conditions. Since quasi-experiments lack randomisation, the third requirement must be met by alternative principles, such as the identification and study of plausible threats to internal validity, by design controls, or by coherent pattern matching [15].

Another approach is to assemble data sets of project information, either from a single company or from many different companies in order to establish benchmarks for quality or productivity. In theory, such data sets could be used to assess the impact of software engineering technologies on quality or productivity. The major problem with this approach is the difficulty of obtaining a valid sample of project. In order to draw valid conclusions projects should be a random sample from a defined population. However, both definition of the population and obtaining a random sample are problematic for software projects [7].

6. Discussion and conclusions

We have suggested that evidence-based software engineering might deliver a variety of benefits to software practitioners and their clients and users. In particular, the adoption and use of techniques supported by evidence should both improve the quality of software-intensive systems, and reassure stakeholder groups that practitioners are using best practice.

However, there are a number of problems associated with EBSE. In order to be effective in terms of influencing software practitioners, EBSE needs substantial infrastructure support, particularly with respect to making systematic reviews available to practitioners. For example, an initiative similar to the Cochrane Collaboration would require an international collaboration with funding from national and multi-national agencies.

However, there are also scientific problems that may be more difficult to address. The problem of evaluating technologies that rely on human skill means that our experiments will always be vulnerable to subject and experimenter bias. There are approaches that can be used to reduce the scale of the problem. More difficult to resolve is the problem of the complexity of the software lifecycle. It will always be difficult to obtain reliable evidence about the behaviour of technologies in large-scale projects.

In our view, evidence-based software engineering is a worthy goal for researchers interested in empirical software engineering and practitioners faced with decisions about the adoption of new software engineering technologies. However, there are undoubtedly problems associated with EBSE that arise from the nature of software engineering and that require the development of new procedures and practices for empirical studies.

There are some types of study, particularly those related to software testing that appear to be promising candidates for EBSE. It is likely to be more difficult to apply EBSE to the software construction technologies (i.e. analysis and design technologies).

Furthermore, some aspects of EBSE are essentially low risk and should be adopted as soon as possible such as the development and adoption of guidelines for systematic reviews. For example, Jasperson et al. [11] provide a very good example of a systematic review in Information Systems research. Their paper was not aimed at gathering evidence. It was aimed at gaining an understanding of research area, and identifying topics for further research. Nonetheless, it illustrates the rigour required of any systematic review. For example, an important issue for any systematic review is to use an appropriate search methodology with the aim of achieving as complete (and, therefore, unbiased) a survey as possible. Furthermore, it is an important part of a systematic review to describe the search method. Jasperson et al. describe the method of search they used (i.e. hand search of 12 named journals between 1980 and 1999), justifying the method of search, the choice of journals, the selection of papers, and discussing the risks associated with their search method. Furthermore, they describe how they synthesised the individual studies, and provide an annotated summary of each paper they reviewed. We would like to see this type of rigour more often in software engineering survey articles and PhD theses.

Thus, our recommendation is for researchers to adopt as much of the evidence-based approach as is possible, to target systematic reviews (or summaries of such reviews) at practitioner publications, and to treat initial attempts at using EBSE as a means of further assessing the viability of the evidence-based approach.

7. References

[1] ANTMAN EM, LAU J, KUPELNICK B, MOSTELLER F, CHALMERS TC. A comparison of results of meta-analysis of randomized controlled trials and recommendations of clinical experts, *JAMA-Journal of the American Medical Association*, 268(2):240-248, July 1992.

[2] AUSTRALIAN NATIONAL HEALTH AND MEDICAL RESEARCH COUNCIL, *How to use the evidence: assessment and application of scientific evidence*, Handbook Series on Preparing Clinical Practice Guidelines, NHMRC: Canberra, February 2000.

[3] BASILI, V.R. AND CALDIERA, G. Improve Software Quality by Reusing Knowledge and Experience, *Sloan Management Review*, 37(1):55-64, Fall 1995.

[4] BERRY, D.M. AND TICHY, W.F. Comments on "Formal Methods Application: An Empirical Tale of Software Development", *IEEE Transactions on Software Engineering*, 29(6):567-571, June 2003.

[5] COCHRANE, AL. In Chalmers I, Enkin M, Keirse MJNC, eds. Effective care in pregnancy and childbirth. Oxford University Press, Oxford, 1989.

[6] HAMMERSLEY M. Some questions about evidence-based practice in education. Paper presented at the symposium on "*Evidence-based practice in education*" at the Annual Conference of the British Educational Research Association, University of Leeds, September 13-15, 2001. www.leeds.ac.uk/educol/documents/00001819.htm (accessed 16 September 2003)

[7] KITCHENHAM, B.A., The Case Against Software Benchmarking, Keynote Lecture, Proceedings of The European Software Measurement Conference FESMA-DASMA 2001, May 2001.

[8] KITCHENHAM, B.A, LINKMAN, S.G., AND FRY, J.S. Experimenter induced distortions in empirical software engineering. in Jedlitschka, A. & Ciolkowski, M. (Eds), "The Future of Empirical Studies in Software Engineering", Proc. of 2nd International Workshop on Empirical Software Engineering, WSESE 2003, Roman Castles, Italy, September 2003.

[9] KITCHENHAM, B.A., PLEEGER, S.L., PICKARD, L., JONES, P., HOAGLIN, D., EL EMAM, K., AND ROSENBERG, J. Preliminary Guidelines for Empirical Research in Software Engineering, *IEEE Transactions on Software Engineering*, 28(8):721-734, 2002.

[10] KITCHENHAM, B.A., TRAVASSOS, G.H., VON MAYRHAUSER, A., NIESSINK, F., SCHNIEDEWIND, N.F., SINGER, J., TAKADO, S., VEHVILAINEN, R., AND YANG, H. Towards an Ontology of Software Maintenance, *Journal of Software Maintenance: Research and Practice*, 11(6):365-389, 1999.

[11] JASPERSON, JON (SEAN), BUTLER, BRIAN S., CARTE, TRACI, A., CROES, HENRY J.P., SAUNDERS, CAROL, S., AND ZHEMG, WEIJUN. Review: Power and Information Technology Research: A Metatriangulation Review. MIS Quarterly, 26(4): 397-459, December 2002.

[12] LINDSAY, R.M. AND EHRENBERG, A.S.C., The Design of Replicated Studies, *The American Statistician*, 47(3):217-228, August 1993.

[13] MOHER D., SCHULZ K.F., ALTMAN D.G. The CONSORT Statement: Revised Recommendations for Improving the Quality of Reports of Parallel-Group Randomised Trials, The CONSORT Group, *Annals of Internal Medicine*, 134(8):657-662, April 2001.

[14] SACKETT, D.L., STRAUS, S.E., RICHARDSON, W.S., ROSENBERG, W., AND HAYNES, R.B. *Evidence-Based Medicine: How to Practice and Teach EBM*, Second Edition, Churchill Livingstone: Edinburgh, 2000.

[15] SHADISH, W.R., COOK, T.D, AND CAMPBELL, D.T. *Experimental and Quasi-Experimental Designs for Generalized Causal Inference*, Houghton Mifflin Company: Boston, 2002.

[16] SOBEL, A.E.K. AND CLARKSON, M.R. Formal Methods Application: An Empirical Tale of Software Development, *IEEE Transactions on Software Engineering*, 28(3):157-161, March 2002.

[17] SOBEL, A.E.K. AND CLARKSON, M.R. Response to "Comments on 'Formal Methods Application: An Empirical Tale of Software Development'", *IEEE Transactions on Software Engineering*, 29(6):572-575, June 2003.

[18] WENTE, M.N., SEILER, C.M., UHL, W., AND BÜCHLER, M.W. Perspectives of Evidence-based Surgery, *Digestive Surgery,* 20(4):263-269, 2003.