Supporting Evidence-Based Software Engineering with Collaborative Information Retrieval

(Invited Paper)

Heri Ramampiaro, Daniela Cruzes, Reidar Conradi
Department of Computer and Information Science
Norwegian University of Science and Technology (NTNU), Norway
Email: {heri,dcruzes,conradi}@idi.ntnu.no

Manoel Mendona
Department of Computer Science/UFBA
Salvador, BA - Brazil
Email: manoel.g.mendonca@gmail.com

***Abstract*— The number of scientific publications is constantly increasing, and the results published on Empirical Software Engineering are growing even faster. Some software engineering publishers have began to collaborate with research groups to make available repositories of software engineering empirical data. However, these initiatives are limited due to issues related to the available search tools. As a result, many researchers in the area have adopted a semi-automated approach for performing searches for systematic reviews as a mean to extract empirical evidence from published material. This makes this activity labor intensive and error prone. In this paper, we argue that the use of techniques from information retrieval, as well as text mining, can support systematic reviews and improve the creation of repositories of SE empirical evidence.**

Keywords: Collaborative Information Retrieval, Empirical Software Engineering, and Systematic Review.

I. Introduction

The number of scientific publications is continuously increasing, and the number of journals reporting on results from Empirical Software Engineering is also growing. As a consequence, it is important to have approaches to execute secondary studies, i.e., studies that draw conclusions over the evidence collected from previous studies. Systematic Review has quickly become the approach of choice to integrate evidence from Software Engineering literature [1]. A systematic review is a summary of the literature that uses explicit methods to perform a repeatable and thorough literature search. Systematic reviews execute critical appraisals of individual studies and uses appropriate techniques to combine results of valid studies.

Although a systematic review seeks to accurately capture evidences, the process is costly, taking several months from conception to publication [2] and tremendous effort [3]. The systematic review process requires that a user identify a comprehensive collection of articles, extract information from those articles, verify the accuracy of those extracted facts, and analyze the extracted facts using either qualitative or quantitative techniques [4].

To reduce the risk of bias, a systematic review requires explicit methods to search for literature for inclusion in the review [4]. An explicit search strategy needs to be developed and systematically applied to a range of resources, such as ACM, IEEE, SpringerLink, and Elsevier repositories. In most approaches to systematic reviewing, the aim is to produce a comprehensive and unbiased set of relevant papers to the research question. It is then, necessary to find a balance between sensitivity - i.e., finding all the papers in a topic area, and specificity - i.e., finding relevant papers.

Dieste et al. [5] investigated the sensitivity problem in one specific systematic review in software engineering. They found important limitations that can affect search strategy sensitivity:

1) Limited bibliographic resources. Some databases do not cover a broad spectrum of publications but are confined to just publications by one publisher. This applies to IEEEXplore, Springerlink, ScienceDirect and ACM Digital Library. This is an impediment for searching because each search strategy has to be applied over again on different search engines for combination at a later date. The combination has to be made apart usually by hand.
2) Problems with the search algorithm. In both IEEEXplore and ACM DL, the search cannot be run on certain fields.
3) Failure to recognize plurals, synonyms and roots. In some cases, like IEEEXplore (Basic Search option), the search engine did not search both the singular and plural of the specified term. This meant that the plural form of each term had to be entered separately.
4) Incomplete article abstracts or full texts. In some cases, like IEEEXplore and ACM DL, the abstract or the full text cannot always be accessed because they are sometimes not linked to the article title. This is an important weakness, as it makes it difficult to retrieve articles and for researchers to check abstracts and texts.

As a result of the above limitations, a researcher performing (the) systematic review has to rely on his experience in the field to judge if the underlying search was done properly. There is also a risk that the retrieved references may lead to bias due to subjectiveness of the researcher with respect to relevance judgments. And there is an increased chance that not all the relevant papers are retrieved in these searches.

Therefore, it is unquestionable that the area would profit from tools and methods that could help to locate, organize, and summarize information for systematic reviews, as well as to synthesize it into usable knowledge [6], [7]. A sensible question would be: Can such tools be built?

This paper investigates the challenges on the search activity

COLLABORATECOM 2010, October 9-12, Chicago, USA

DOI 10.4108/icst.collaboratecom.2010.9

of the systematic review and the use of information retrieval techniques to accomplish some of these tasks. Information retrieval (IR) is a term that has a broad meaning and is used for almost all aspects that involve making information available to users. The main emphasis is on users need for information. As such, IR is concerned with the representation, storage, organization of, and access to information item [8], [9].

The reminder of this paper is organized as follows. Section II introduces some examples of searches for relevant papers for systematic reviews in software engineering. Section 3 describes some approaches we foresee to approach some of the challenges in the area. Section 4 summarizes our paper and describes our plans for future research.

II. EBSE Approaches

A systematic review is a defined and methodical way of identifying, assessing, and analyzing published primary studies in order to investigate a specific research question. A systematic review can also discover the structure and patterns of existing research, and so identify gaps that can be filled by future research [4]. Systematic reviews differ from ordinary literature surveys in being formally planned and methodically executed. A good systematic review should be independently replicable and so will have much greater scientific value than an ordinary literature survey. However, systematic reviews require much more effort than ordinary literature surveys. There are many reasons for undertaking a systematic review. The most common reasons are [4]:

- To summarize the existing evidence concerning a treatment or technology e.g. to summarize the empirical evidence of the benefits and limitations of a specific agile method.
- To identify any gaps in current research in order to suggest areas for further investigation.
- To provide a framework/background in order to appropriately position new research activities.

The following features differentiate a systematic review from a conventional literature review [4]:

- Definition and documentation of a systematic review protocol in advance of conducting the review, to specify the research questions and the procedures to be used to perform the review.
- Definition and documentation a search strategy as part of the protocol, to find as much of the relevant literature as possible;
- Description of the explicit inclusion and exclusion criteria as part of the protocol, to be used to assess each potential study;
- Description of quality assessment mechanisms as part of the protocol, to evaluate each study;
- Description of review and crosschecking processes as part of the protocol, and involving multiple independent researchers, in order to control researcher bias.

Kitchenham [4] published guidelines for software engineering researchers performing systematic reviews. Although procedures and systems for systematic reviews are well established in other disciplines (particularly in medicine), software engineering researchers have yet to come to a well-understood consensus about the conduct and value of systematic reviews. The guideline was derived from three existing guidelines used by medical researchers.

Kitchenham describes the three main phases of a systematic review process: planning the review, conducting the review, and reporting the review. Each of these phases contains a sequence of stages, but the execution of the overall process involves iteration, feedback, and refinement of the defined process [4], the process is described in Figure 1.

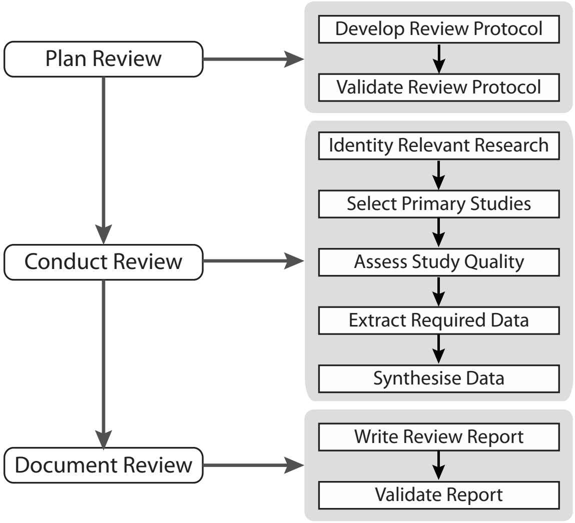

Fig. 1. Systematic Review Process [4]

Recently, Kitchenham et al. [1] performed a review on the current status of EBSE since 2004, in particular they focused on articles related to EBSE literature reviews (SLRs). This review that was published this year identified 20 relevant studies in 10 journals and 4 conference proceedings. The search approaches done in five of these studies are described in Table I. This also shows how searches are performed in these studies. It is worth to note that the search process, the sources of papers and how many papers were reviewed for the selection were not clearly stated in some of the papers from the original review. Nevertheless, we decided to select some of these approaches to illustrate how the reviews have been performed.

III. Filling the gap

As can be inferred from our discussion above there are several challenges that we need to address.

First, finding a sensible way to build uniform interface to existing research repositories is crucial but is by far a challenging task. As can be inferred from our discussion above, existing repositories such as the ACM DL, IEEExplore, Science Direct, and Springer Digital Library all provide their proprietary interfaces. This makes it less straightforward to

TABLE I
SEARCH APPROACHES IN EBSE

Ref	Source List			Search String	Approach
	IEEE	ACM	Others		
Evaluation Approaches for Software Architectural Documents [10]	X	X	X	IEEE String: (software <and> <not> (hardware, synthesize, circuit) <and> architecture <phrase> (<or> (evaluate, assurance, review, inspection, verification))) <in> metadata	80 scientific studies were identified and after the analysis, only 54 described at least one evaluation approach. Based on these studies, 20 software architectures evaluation approaches were identified.
Software effort estimation terminology: The tower of Babel [11]			X	Google: software engineering course books Amazon: top 100 bestselling. BestWeb: Manual Selection	Google: They manually investigated the first 100 URLs that appeared relevant, and counted the frequency of each textbook used in university courses on software engineering. Amazon: books were selected by using Amazons list of the top 100 bestselling computer science books. BESTweb: The journal papers were selected by manually scanning potentially relevant journals, while conference papers were identified by a search in the INSPEC-library. They ended up with 8 books and 23 research papers.
A Systematic Review of Software Development Cost Estimation Studies [12]			X	Manual Selection: The main criterion for including a journal paper in the review was that the paper describes research on software development effort or cost estimation.	The search for papers was based on an issue-by-issue, manual reading of titles and abstracts of all published papers, in more than 100 potentially relevant, peer-reviewed journals with papers written in English. Papers that were potential candidates for inclusion in their review were read more thoroughly to decide whether to include them or not. In total, 304 relevant papers were found in 76 of these journals.
A systematic review of web Engineering Research [13]	X	X	X	(Web engineering OR WWW engineering OR World?Wide Web engineering OR Internet engineering)	They selected 173 primary studies published in scholarly literature (e.g. Web engineering track of the WWW conference, IEEE Multimedia special issues on Web engineering, Web engineering conferences).
A survey of Controlled Experiments in Software Engineering [14]	X	X	X	Manual Search issue-by-issue in each of the selected sources	One researcher systematically read the titles and abstracts of 5,453 scientific articles published in the selected journals and conference proceedings for the period 1993-2002. Excluded from the search were editorials, prefaces, article summaries, interviews, news, reviews, correspondence, discussions, comments, readers letters, and summaries of tutorials, workshops, panels, and poster sessions. In the end, 103 articles were selected.

provide users uniform access. Still, we believe it is possible to address this challenge by providing a Meta-Searcher as shown in Figure 2 that allows the user to search through these repositories using a unified user interface. Such an interface will also collects and merge the results as if he/she were using a single search engine.

The concept of meta-searcher is not new per se. It has been used already in Web search applications and in commercial library search applications such as Google Scholar[1]. However, the main difference with our approach is that while Google indexes books and papers from all fields in their repository, ours is restricted to a limited area. Another similar system is Scopus[2]

Like Google Scholar, Scopus is also a comprehensive system, indexing papers from many fields, consisting of 36 million records. This means that to be able to use Scopus for our approach, a search filter would be needed. We intend to focus our approach as an interface on top of existing repositories, and with capability of filtering out papers not in the Software Engineering field. In this way, we will be able to restrict our search results to software engineering papers only, and thus restricting the search space to a more manageable size. In addition, we will apply techniques that take into account the characteristics of EBSE and systematic review discussed in Section II. Furthermore, since more than one user or researcher will carry out systematic reviews, it can

[1] Google Scholar is a trademark of Google. See also http://scholar.google.com/

[2] Scopus is a trademark of Elsevier B.V. See also http://www.scopus.com/scopus/home.url

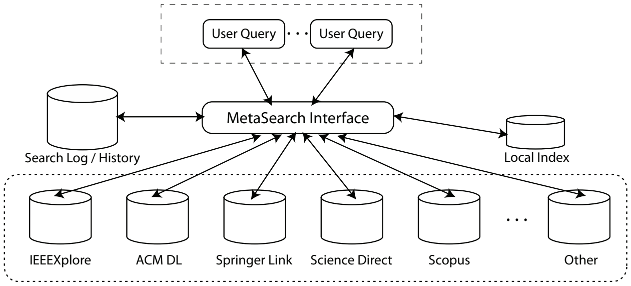

Fig. 2. Illustration of the idea of a meta-searcher

be seen as collaborative effort. As such, we can benefit from the collaborative information retrieval techniques. This means that results including user evaluations from several different searches can be extracted and used as a result for a specific systematic review process. This approach uses the search log or search history as a basis to analyze the search behavior, and synthesize the search result for a new search.

Second, finding suitable information retrieval (IR) techniques as well as text mining methods is an important challenge, and must be addressed carefully. One of the main issues with EBSE is that it is not only concerned with identification of relevant research papers as they are defined within the IR field, but the papers must also represent enough knowledge such that they can be used as means for evaluating and interpreting the available research. To address this challenge, we propose to use automatic text summarization in combination with the indexing techniques borrowed from the IR field. Automatic text summarization will extract most important sentences from the papers based on known natural language processing (NLP) techniques [15], [16]. Once the summarizations are available, they can be indexed and stored in a searchable repository. The main benefit is reduced labor effort in post processing the search results. Another approach is to group similar papers using known clustering method such as k-means or k-NN [17], before indexing and retrieval. In this way, one can reduce the search space before the actual search is initiated.

Third, according to [18] one of the main important steps in EBSE is to convert a relevant problem or information needed into an answerable question. This can then be used in the search of the best available evidence. Seen from IR perspective, an important challenge is to translate the information need to suitable search terms within a search domain, such that relevant papers representing the best available knowledge can be retrieved. One way to address this challenge is to collect relevant terms from the literature that has been used in previous manual search work (See Table I).

Finally, to be able to evaluate the applied IR techniques, there is necessity of having an effective evaluation strategy is important. With respect to EBSE, the main goal is to find all relevant papers without using too much effort. A possible way to evaluate is to measure recall and precision. However, as we mentioned before, the lack of consolidated corpus in combination of heterogeneous repositories makes this a challenge. Having a consolidated corpus with suitable and comprehensive judgment information, we can address this challenge. Thus, what we need is a test collection similar to the standard collections such as Test Retrieval Conference (TREC) and GOV2 [8], [9]. Such a test collection can be built based on currently available papers where the systematic reviews were carried out manually or semi-automatically. This, in turn, allows us to evaluate our approaches in a more systematic way using widely used measures such as mean average precision (MAP), as well as precision and recall.

IV. Concluding remarks

In this paper, we have motivated the necessity of having effective information retrieval tools to support evidence-based software engineering on performing systematic reviews. With growth of number of available papers, and the results published on Empirical Software Engineering field, the need for such a tool is widely recognized. However, there are number of challenges we face. This paper has addressed this challenges and outlined information retrieval approaches that we believe are sensible to use.

As a next step towards an investigation of the usefulness of these approaches, we will reproduce the search from the work we presented in this paper to evaluate the possibilities. Then, using the result from this study, we will further develop our approach, and evaluate the results based on a test collection and using end users from the Software Engineering field.

References

[1] B. Kitchenham, O. Pearl Brereton, D. Budgen, M. Turner, J. Bailey, and S. Linkman, “Systematic literature reviews in software engineering-A systematic literature review”, *Information and Software Technology*, vol. 51, no. 1, pp. 7–15, 2009.

[2] A. Petrosion, “Lead authors of cochrane reviews: Survey results”, Report to the Campbell Collaboration. Cambridge, MA: University of Pennsylvania., 1999.

[3] I.E. Allen and I. Olkin, “Estimating time to conduct a meta-analysis from number of citations retrieved”, *JAMA*, vol. 282, no. 7, pp. 634, 1999.

[4] B. Kitchenham, “Procedures for performing systematic reviews”, *Keele, UK, Keele University*, vol. 33, 2004.

[5] O. Dieste and OAG Padua, “Developing search strategies for detecting relevant experiments for systematic reviews”, in *First International Symposium on Empirical Software Engineering and Measurement, 2007. ESEM 2007*, 2007, pp. 215–224.

[6] C. Blake, “Information synthesis: A new approach to explore secondary information in scientific literature”, in *Proceedings of the 5th ACM/IEEE-CS joint conference on Digital libraries*. ACM, 2005, p. 64.

[7] D. Cruzes, M. Mendonça, V. Basili, F. Shull, and M. Jino, “Automated Information Extraction from Empirical Software Engineering Literature: Is that possible?”, in *First International Symposium on Empirical Software Engineering and Measurement, 2007. ESEM 2007*, 2007, pp. 491–493.

[8] Ricardo A. Baeza-Yates and Berthier Ribeiro-Neto, *Modern Information Retrieval*, Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 1999.

[9] Christopher D. Manning, Prabhakar Raghavan, and Hinrich Sch ”utze, *An introduction to information retrieval*, Cambridge University Press, 2008.

[10] R.F. Barcelos and G.H. Travassos, “Evaluation approaches for software architectural documents: a systematic review”, in *Ibero-American Workshop on Requirements Engineering and Software Environments (IDEAS), La Plata, Argentina*, 2006.

[11] S. Grimstad, M. Jørgensen, and K. Moløkken-Østvold, “Software effort estimation terminology: The tower of Babel”, *Information and Software Technology*, vol. 48, no. 4, pp. 302–310, 2006.

[12] M. Jorgensen and M. Shepperd, “A systematic review of software development cost estimation studies”, *IEEE Transactions on Software Engineering*, vol. 33, no. 1, pp. 33–53, 2007.

[13] E. Mendes, “A systematic review of Web engineering research”, in *2005 International Symposium on Empirical Software Engineering, 2005*, 2005, p. 10.

[14] DIK Sjøberg, JE Hannay, O. Hansen, VB Kampenes, A. Karahasanovic, N.K. Liborg, and AC Rekdal, “A survey of controlled experiments in software engineering”, *IEEE Transactions on Software Engineering*, vol. 31, no. 9, pp. 733–753, 2005.

[15] Y. Gong and X. Liu, “Generic text summarization using relevance measure and latent semantic analysis”, in *Proceedings of the 24th annual international ACM SIGIR conference on Research and development in information retrieval*. ACM, 2001, pp. 19–25.

[16] B. Liu, *Web data mining: exploring hyperlinks, contents, and usage data*, Springer Verlag, 2007.

[17] X. Wu, V. Kumar, J. Ross Quinlan, J. Ghosh, Q. Yang, H. Motoda, G.J. McLachlan, A. Ng, B. Liu, P.S. Yu, et al., “Top 10 algorithms in data mining”, *Knowledge and Information Systems*, vol. 14, no. 1, pp. 1–37, 2008.

[18] T. Dyba, B.A. Kitchenham, and M. Jorgensen, “Evidence-based software engineering for practitioners”, *IEEE software*, vol. 22, no. 1, pp. 58–65, 2005.