Seeds of Evidence: Integrating Evidence-Based Software Engineering

David S. Janzen
Computer Science Department
California Polytechnic State University
San Luis Obispo, CA USA
djanzen@calpoly.edu

Jungwoo Ryoo
Information Sciences and Technology
The Pennsylvania State University-Altoona
Altoona, PA USA
jryoo@psu.edu

Abstract

With increasing interest in Evidence-Based Software Engineering (EBSE), software engineering faculty face the challenge of educating future researchers and industry practitioners regarding the generation and use of EBSE results. We propose development and population of a community-driven web database containing summaries of EBSE studies. We present motivations for inclusion of these activities in a software engineering course, and address the particular appeal of a community-driven web database to students who have grown up in the Internet generation. We present our experience with integrating these activities into a graduate software engineering course, and report student and industry practitioner assessments of the resulting artifacts.

1: Introduction to EBSE

Software developers are known for adopting new technologies and practices based solely on their novelty, promise, or anecdotal evidence. Evidence-based software engineering (EBSE), on the other hand, endeavors to produce a body of documented experiences that might inform software practice adoption decisions.

The goal of 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" [12]. This "best evidence from research" results from experimentation. Experimentation in software engineering involves the empirical study of human activities [3] to aid decisions on what are the best practices, processes, methods or tools for developing software, and when (in what context) they should be applied.

978-0-7695-3144-1/08 $25.00 © 2008 IEEE
DOI 10.1109/CSEET.2008.29

Experimental validation in software engineering ranges from informal assertions ("I used X and it worked great!") to formal theoretical analysis. Commonly recognized EBSE experimentation involves observational studies (e.g. case studies) and controlled experiments, both of which can be conducted in the field with professional software developers or in labs, typically with students [16].

1.1: Growing Interest in EBSE

Interest in EBSE appears to be increasing, beginning perhaps in the mid-1990's [13]. In 1994, Gibbs wrote that "after 25 years of disappointment with apparent innovations that turned out to be irreproducible or unscalable, many researchers concede that computer science needs an experimental branch to separate the general results from the accidental" [14]. Zelkowitz and Wallace observed that almost half of the papers in *IEEE Software*, *IEEE Transactions on Software Engineering*, and *ICSE* conferences (from 1985, 1990, and 1995) included only assertions or no evaluation at all [15].

More recently, however, many conferences and journals such as *IEEE Transactions on Software Engineering* specifically request empirical studies and experimental validation of new ideas. Springer publishes a dedicated journal entitled *Empirical Software Engineering: An International Journal*. Reporting guidelines have been proposed and compared [11, 8]. Empirical software engineering projects have received significant government and corporate funding. Research centers have been founded such as the "NSF Center for Empirically-Based Software Engineeringu" (www.cebase.org).

1.2: EBSE Challenges and Opportunities

Despite increased interest, the growth of EBSE research is somewhat slow [5] and difficult. Many factors contribute to the challenges of EBSE. First, we suggest that barriers limit access to conduct EBSE studies. While laboratory experiments are often conducted in academic environments, there are many inherent threats to validity. Students are rarely as mature as professional software developers. Application domains are often contrived. Software projects are rarely as large and complex as "real-world" projects.

Unfortunately companies and organizations are often reluctant to participate in field experiments. This seems to be particularly true in the United States. Many may be unwilling to try new, perhaps unproven approaches. Others may be concerned that they might reveal poor metrics or performance. Or they may be unwilling to allow researchers in for fear of losing proprietary information or simply that they may slow down the team.

Second, we believe that professional software practitioners struggle to acquire, analyze, and apply EBSE results. EBSE studies are commonly reported in academic journals and conference proceedings. Many practitioners rarely read these or even have access to them. Furthermore EBSE studies can be difficult to find among the mix of other papers. Even when they are found, most practitioners lack any formal education on how to analyze the studies.

We suggest that EBSE education and easy access to EBSE data may address some of the challenges of EBSE. In the next sections we propose a system and a pedagogical approach that take advantage of these opportunities.

2: SEEDS: Software Engineering Evidence Database System

We propose a web-based, community-driven database for collecting, surveying, analyzing, and disseminating EBSE results. The evidence database is tentatively named "SEEDS: Software Engineering Evidence Database System." This database would serve as a comprehensive, constantly evolving focus point for summarizing and analyzing EBSE results.

SEEDS would be designed and constructed as a kind of structured wiki. SEEDS would be organized by topic. Each topic would include "how-to" references that would summarize the purpose and mechanism for applying the practice/tool/method. Each topic would include a listing of all related EBSE studies. The EBSE study summaries would conform to a common format for reporting design, context, and result summaries.

Researchers would be able to add EBSE studies with links to official publications. Community members would be encouraged to rate studies in terms of study quality, validity, and importance of results. The rating system would be designed to ensure that the most reputable and pertinent studies "bubble up" so they become the most widely read by industry practitioners. Community members would also be able to write summaries of the EBSE study reports. The summaries would accomplish the goals of both condensing the information to the most salient points, but also to provide a critical analysis. Multiple EBSE study summaries would be allowed for each EBSE study report. These summaries could also be rated, with the most useful/accurate summaries also "bubbling up" for improved content quality. We imagine summary comments to address issues such as threats to validity, or praise for particularly well executed studies.

As an added feature, we propose that the system include user-driven comparison grids that compare a set of EBSE studies on a related topic using a set of dynamic attributes. Such grids are inspired by those presented in two test-driven development summary articles [6, 9].

The database would provide a single point of reference for researchers and practitioners. Unlike traditional survey publications, the database promises to be constantly up-to-date, rather than providing a snapshot of results at a single point in time. Traffic to the database would be tracked and reported, highlighting trends and interest by the community.

We propose SEEDS as an asynchronous, global portal for increasing EBSE education and activities through simple access to comprehensive EBSE data and community participation in EBSE analysis. We believe the community-driven nature of SEEDS will be embraced by the young software engineers who have grown up with the Internet, social networking, and Wikipedia. By improving EBSE visibility and access, this approach may help satisfy the need for more new and replicated studies [2], as well as satisfy an ethical duty of software professionals to assist colleagues and develop the field.

2.1: Related EBSE Databases

The NSF-funded Center for Empirically-Based Software Engineering (CeBASE) provides a forum for researchers and practitioners who are participating in EBSE studies to share data and results. However, CeBASE does not provide comprehensive summary EBSE results for the general software engineering community. The Empirical Research Repository (ERR) [1] is hosted by Durham University and is assessed in section 4. The ERR appears to have similar goals to SEEDS. Differences include the fact that SEEDS is community-driven

whereas ERR appears to have only a select set of contributors. ERR also has established strict EBSE study selection criteria for inclusion in the repository.

3: Pedagogical Approach

Given the increased interest in EBSE, software engineering faculty must find effective ways to integrate EBSE topics into their curriculum. Software engineering students who intend to conduct research will find it necessary to design and perform experimental validations. Students who proceed primarily to professional practice will need to be able to find, interpret, and analyze EBSE reports in order to make informed adoption decisions.

Education on EBSE is proposed as a possible strategy to improve EBSE awareness, access, and analysis. An increased awareness should reduce entry barriers for conducting field experiments. Improved access should encourage practitioner participation and use of EBSE results. Enhanced analysis should also improve practitioner understanding and application of EBSE results.

3.1: Related Courses

In 2003, Jorgensen et al. took the approach of developing an entire course dedicated to EBSE at Hedmark University College in Rena, Norway [10]. Their course involved teaching modules on EBSE background, theory, argumentation, and prerequisite statistics. Students then completed a course project that involved selection of an EBSE topic, relevant background research, and a significant report "that marshals the available evidence to support a conclusion."

Steve Easterbrook notes nine university courses focused on EBSE [4] including his own. All but one of these courses appear to be at the graduate level, and nearly all are completely devoted to the topic of EBSE.

We have proposed mechanisms for incorporating EBSE information in professional training [7]. This approach, however, is slow and localized.

3.2: Learn By Doing Approach to EBSE

We desired to instill an understanding and appreciation for EBSE topics, while staying within the confines of the two existing graduate software engineering courses at California Polytechnic State University, San Luis Obispo (Cal Poly). Cal Poly is a primarily undergraduate institution located on the central coast of California, about half way between San Jose and Los Angeles.

Cal Poly's motto is "Learn by Doing." In order to apply the "Learn by Doing" approach to EBSE education, the lead author incorporated the following assignments into a fall 2007 graduate software engineering course:

1. Students will work alone or in pairs to write surveys of empirical studies on a particular software engineering topic. The surveys will be added to a common database. Students are expected to contribute a minimum of seventeen study reviews per person, along with one "how-to" summary of the topic being surveyed.
2. Students will participate in a team to develop, document, and present the requirements, architecture, and prototypes for the SEEDS system described earlier.

The selected course is the first of two software engineering courses in a computer science masters program. The first course typically focuses on requirements and architecture topics, while the second course typically focuses on design, construction, and maintenance topics. An alternative hands-on approach would have been to have students plan and/or participate in an empirical software engineering study. Such an activity typically occurs in a year-long senior capstone project for the undergraduate software engineering major at Cal Poly. We elected not to duplicate that experience in the graduate course, but opted rather for assignments that require more analytical thinking on the part of the students.

The pedagogical goals of these assignments were as follows:

- Increase student awareness and competence in evaluating EBSE studies
- Integrate EBSE material in existing graduate software engineering courses at Cal Poly
- Engage Net generation students through a collaborative/community-driven Web model
- Create a repository of useful EBSE information for widespread use
- Extend the life of student work beyond the limits of a ten-week course

Students completed the collection and analysis of their EBSE studies by week eight of the ten week quarter. Because the student SEEDS projects were under development, we implemented a rapid prototype using the Drupal content management system to house the student EBSE summaries. The prototype is missing many of the features of the student projects, but it provided a simple mechanism to create EBSE topic areas, and allowed community members (in this case students) to register and contribute summaries of EBSE studies. This prototype is available at http://www.evidencebasedse.com.

4: Results

The course activities were assessed in three ways. In order to assess the quality and usefulness of the student-written EBSE summaries, a survey was conducted with professional software engineers. A second survey was conducted with the students in order to assess their experience and perspective on both the EBSE summaries and the team project. Finally, observations from the course instructor will be reported.

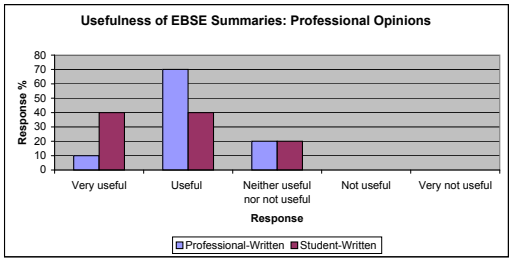
4.1: Survey of Software Professionals

A survey of ten questions was sent to software professionals in four companies: Amgen, Google, Intuit, and LSI. A total of ten people responded by the cutoff date. The professional status of the respondents is reported in Table 1. Although the survey did not request degree information, based on hiring practices it is believed that all of the respondents have at least an undergraduate degree in a computing field. One respondent is known to hold a Ph.D., and one an MBA.

Of the ten respondents, half indicated that they had used a digital library such as those provided by ACM and IEEE. Again half indicated that they currently had access to such a library. Seventy percent indicated that they had never read a report of an evidence-based software engineering study. When asked "I understand how evidence-based techniques are applied to software engineering," seventy percent responded less than favorably. Interestingly, seventy percent responded favorably (very likely or likely) to the question: "How

Professional Status	Response %
full-time software developer	20
part-time software developer	10
full-time software manager	40
part-time software manager	0
administrator or executive in a software-related business	0
other (sys admin, analyst, etc.) in a software-related business	30

Table 1. Professional Survey Results: Professional Status



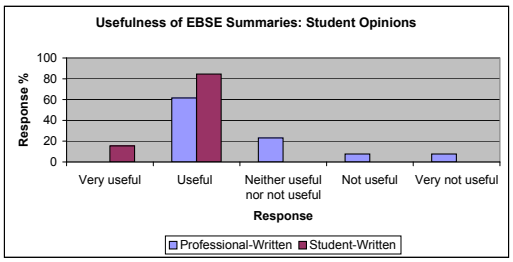


Figure 4.1. Usefulness of EBSE Summaries:
(a) Professional Survey Results (b) Student Survey Results

likely are you to find and read evidence-based studies prior to adopting a particular software engineering practice, process, method, or tool."

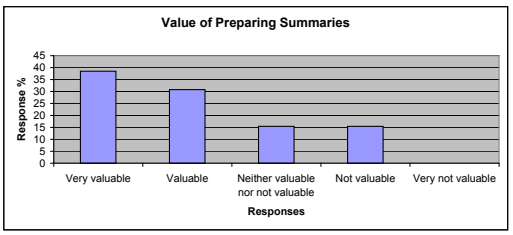
These results, although limited by their sample size, are consistent with our intuition that software professionals are interested in EBSE results, but they lack education in the area. In fact, they even lack the access to research EBSE studies through digital libraries.

In order to assess the quality and usefulness of the student-written EBSE summaries, the software professionals were asked to read a few of the study summaries in two EBSE repositories. The first is hosted at Durham University (see [1]). The second is the repository prototype created for this study and populated with student-written surveys. As Figure 4.1(a) indicates, the software professionals found the student-written summaries to be at least as useful as the professional-written summaries. In fact 30% more of the respondents found the student-written summaries to be very useful.

4.2: Survey of Graduate Students

A similar survey of ten questions was sent to the thirteen students in the graduate software engineering course. All thirteen students responded. Eleven of the thirteen indicated that they had used a digital library prior to enrolling in this course. However, twelve of the thirteen students indicated that they had never read a report of an evidence-based software engineering study prior to enrolling in this course.

When asked "I understand how evidence-based techniques are applied to software engineering," all students responded favorably. Similar to the professional respondents, 83% responded favorably (very likely or likely) to the question: "How likely are you to find and read evidence-based studies prior to adopting a particular software engineering practice, process, method, or tool."



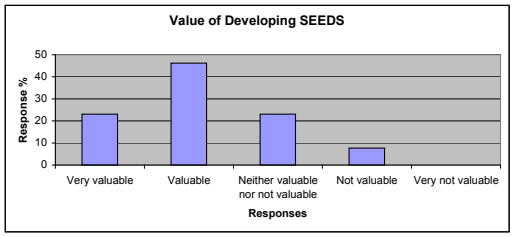


Figure 4.2. Student Survey Results:
(a) Value of writing summaries (b) Value of developing SEEDS

Prior to the survey, the students had not been introduced to the Durham University EBSE repository. Figure 4.1(b) indicates that after being asked to read summaries from both the Durham repository and the repository of their own summaries, nearly forty percent more students found their own summaries to be useful or very useful.

Figure 4.2 reports the student responses to the questions: "In terms of preparing you for a career as a software engineer, how valuable was the experience of preparing the EBSE summaries in this course?" and "In terms of preparing you for a career as a software engineer, how valuable was the experience of developing requirements, architectures, and prototypes for the EBSE summary database in this course?" Student comments on the former question indicated that most students enjoyed learning how to critically analyze an EBSE study, and they enjoyed learning about a particular software engineering topic in depth. A couple of students however reported that they would have preferred studying a wider range of topics themselves, rather than hearing and reading the reports of their peers. On the latter question, most students reported that they had completed numerous team projects in their undergraduate courses and, although they enjoyed learning some new technologies, most thought their time could have been more usefully spent on other tasks.

4.3: Instructor Observations

Students initially struggled to understand the loosely defined system proposal. However, once they were exposed to a variety of empirical studies, they were successful in finding and summarizing studies on their own topics. This personal experience demonstrated the value of the SEEDS system to the students, who also proposed interesting and viable alternative requirements, architectures, and prototypes. Two of the teams elected to apply eXtreme Programming as their process model. Both XP teams documented detailed use cases, and then proceeded to implement several of the highest priority use cases. Both teams implemented rich internet applications using Google Web Toolkit and Adobe's Flex respectively. The third team elected to follow a traditional plan-driven waterfall/linear process. They completed a more detailed software requirements specification and a horizontal (UI) prototype, but did not deliver a functional prototype.

5: Conclusions and Future Work

Graduates from the Cal Poly program overwhelmingly enter careers as applied software engineers, primarily in the technologically rich Silicon Valley and Southern California markets. These efforts to increase EBSE awareness, along with the ability to find and analyze

EBSE results is expected to improve practitioner involvement and appreciation of EBSE.

The alternative requirements and designs for the EBSE database generated by student teams will serve as a starting point for a planned widely disseminated system. The student summaries of EBSE studies will provide seed data to the system that will offer immediate benefit once the system is deployed.

We have demonstrated the viability of incorporating EBSE topics into a graduate software engineering course, and provided initial evidence of its usefulness to professional software practitioners. We encourage software engineering educators to consider requiring that their students critically analyze EBSE studies and contribute their work to the prototype SEEDS repository located at http://www.evidencebasedse.com. SEEDS is expected to evolve and improve, but every effort will be made to ensure that study summaries be retained in all versions.

References

[1] Empirical research repository. http://www.dur.ac.uk/ebse/repository/evidencebasedlitreviews.php#whatlearnt.

[2] Victor R. Basili, Forrest Shull, and Filippo Lanubile. Building knowledge through families of experiments. *IEEE Trans. Softw. Eng.*, 25(4):456–473, 1999.

[3] Victor R. Basili and Marvin V. Zelkowitz. Empirical studies to build a science of computer science. *Commun. ACM*, 50(11):33–37, 2007.

[4] Steve Easterbrook. Csc2130s: Empirical research methods in software engineering. http://www.cs.toronto.edu/ sme/CSC2130/index.html.

[5] Robert L. Glass, V. Ramesh, and Iris Vessey. An analysis of research in computing disciplines. *Commun. ACM*, 47(6):89–94, 2004.

[6] D. Janzen and H. Saiedian. Test-driven development: concepts, taxonomy and future directions. *IEEE Computer*, 38(9):43–50, Sept 2005.

[7] David S. Janzen, Clark S. Turner, and Hossein Saiedian. Empirical software engineering in industry short courses. In *CSEET '07: Proceedings of the 20th Conference on Software Engineering Education & Training*, pages 89–96, Washington, DC, USA, 2007. IEEE Computer Society.

[8] Andreas Jedlitschka and Dietmar Pfahl. Reporting guidelines for controlled experiments in software engineering. In *2005 International Symposium on Empirical Software Engineering*, 2005.

[9] Ron Jeffries and Grigori Melnik. Tdd – the art of fearless programming. *IEEE Software*, 24(3):24–30, 2007.

[10] M. Jorgensen, B. Kitchenham, and T. Dyba. Teaching evidence-based software engineering to university students. In *11th IEEE International Software Metrics Symposium, Como, Italy, September 19-22*, 2005.

[11] Barbara Kitchenham, Hiyam Al-Khilidar, Muhammad Ali Babar, Mike Berry, Karl Cox, Jacky Keung, Felicia Kurniawati, Mark Staples, He Zhang, and Liming Zhu. Evaluating guidelines for empirical software engineering studies. In *ISESE '06: Proceedings of the 2006 ACM/IEEE International symposium on empirical software engineering*, pages 38–47, New York, NY, USA, 2006. ACM Press.

[12] Barbara A. Kitchenham, Tore Dyba, and Magne Jorgensen. Evidence-based software engineering. In *ICSE '04: Proceedings of the 26th International Conference on Software Engineering*, pages 273–281, Washington, DC, USA, 2004. IEEE Computer Society.

[13] Dag I. K. Sjoberg, Tore Dyba, and Magne Jorgensen. The future of empirical methods in software engineering research. In *FOSE '07: 2007 Future of Software Engineering*, pages 358–378, Washington, DC, USA, 2007. IEEE Computer Society.

[14] W. Wayt Gibbs. Software's chronic crisis. *Scientific American (International Edition) 271,*, 271(3):72–81, 1994.

[15] Marvin V. Zelkowitz and Dolores R. Wallace. Experimental validation in software engineering. *Information and Software Technology*, 39(11):735–743, 1997.

[16] Marvin V. Zelkowitz, Dolores R. Wallace, and David W. Binkley. Experimental validation of new software technology. pages 229–263, 2003.