Issues in Using Students in Empirical Studies in Software Engineering Education

Jeffrey Carver[1], Letizia Jaccheri[2], Sandro Morasca[3], and Forrest Shull[4]

[1] Experimental Software Engineering Group, Dept. of Computer Science
A.V. Williams Bldg., University of Maryland, College Park, College Park MD 20742
carver@cs.umd.edu

[2] Department of Computer and Information Science, Norwegian University of Science and Technology, Sem Sælands vei 7-9, 7491 Trondheim, Norway
letizia@idi.ntnu.no

[3] Dipartimento di Scienze Chimiche, Fisiche e Matematiche, Università degli Studi dell'Insubria
Via Valleggio 11, I-22100, Como, Italy
sandro.morasca@uninsubria.it

[4] Fraunhofer USA Center for Experimental Software Engineering Maryland
4321 Hartwick Road, Suite 500, College Park MD 20742, USA
fshull@fc-md.umd.edu

Abstract

Several empirical studies have been carried out with college students as subjects in the last few years. These studies are often used by researchers as pilot experiments before they are carried out in industrial environments. Reports on these studies usually focus on the results obtained and issues such as their external validity. However, the effects and value of empirical studies with students may go beyond the contribution to scientific literature. For instance, the pedagogical challenges and value of these studies is hardly ever stressed.

In this paper, we identify four primary actors that are involved in these empirical studies, i.e., researchers, students, instructors, and the industry. We discuss the costs and benefits for these actors, which are different because of the actors' different goals, expectations, and constraints, which must be recognized to fully exploit empirical studies with students. We also provide some advice on how to carry out empirical studies with students based on our experiences

Keywords
Empirical Studies, Pilot Studies, Software Engineering Education

1. Introduction

Evidence-driven management of software processes and products may help plan, monitor, control, evaluate, and improve them based on solid information. To accomplish this, measurement-related activities, such as data collection and knowledge extraction from data, should become a part of software engineering practice in software organizations and should be integrated with the other software development activities.

Whether based on more quantitative or qualitative information, evidence-driven assessments should be applied to the different processes, methods, techniques, and tools already used by software organizations. In addition, evidence-driven assessments should be applied to the many new processes, methods, techniques, and tools that are constantly proposed for possible use in the software engineering practice. These new proposals should be carefully evaluated based on solid evidence before they are actually deployed in industrial software environments, because of their impact on software quality, costs, and development time. Also, it is important to evaluate the costs and benefits of these new proposals and understand how they should be used to take full advantage of their strengths and reduce the risk that their introduction may fail or be less effective than it could be.

Although reliable evidence is sorely needed on processes, methods, techniques, and tools, few software organizations use measurement-related activities. To this end, empirical studies may be used to

- quantitatively assess the specific objects of study (processes, methods, products, etc.) at hand;
- show the advantages of empirical software engineering and open the way for larger-scale data collection activities.

Different kinds of empirical studies may be carried out (e.g., full-fledged experiment, quasi-experiments, correlational studies, case studies, surveys), depending on the goals and constraints of the application at hand. However, empirical studies in industrial settings in many cases require a good deal of time, effort, and resources, so they need to be planned and carried out carefully. Before running an empirical study at a software company, it is therefore useful to carry out a pilot study with students in an academic setting. Pilot studies with students are often used for several goals. Some of those goals are technical ones, e.g., obtaining some preliminary evidence that supports research hypotheses. Others are of an organizational kind, e.g., fine-tuning the details of the empirical study.

Though technical and organizational goals are usually the main reasons behind empirical studies with students, it should be clear that the researcher's viewpoint is only one of the viewpoints that should be taken into account, and not necessarily the most important one.

For instance, carrying out pilot studies with students should be a two-way street, i.e., both the researchers and the students should perceive value from the study. Pilot studies should have a pedagogical value, and the students' viewpoint must be taken into account. As a consequence, the instructor's viewpoint must be considered too, since the instructor needs to be able to ensure the educational value of these pilot studies.

It is true that, in this context, often the researcher and the instructor are the same person. Even so, the same person is actually playing two different roles, with possibly conflicting goals, at the same time. Suppose that this researcher/instructor would like to carry out a pilot study to empirically validate a new measure of software cohesion as a researcher. However, suppose also that this pilot study has little educational value. This situation would make it difficult for the pilot study to match the instructor's goals. Thus the researcher's and the instructor's viewpoints may truly be different.

In addition, one of the instructor's (but not the researcher's) goals, would be to teach students about measurement-related activities. To this end, carefully planning and performing, empirical studies during software engineering classes may help the instructors reach this educational goal. In software classes, students learn how to program, design, test, specify, etc. with the idea that these are the activities in which they are going to be involved when they become professionals in software organizations. By the same token, students should also be involved in other activities in which they may participate in their job, such as participating in data collections. Data collection is an activity that is routinely carried out in many work environments outside software engineering, so it should be no surprise to the students when they graduate and start working that data collection should also be carried out in software organizations. Just like in other engineering environments, measurement and data collection involve the process, the people, and the products. Since software is a very human-intensive business, getting used to these activities may even be more important in software than in other business areas.

Finally, one should always keep in mind that the researcher's, the student's, and the instructor's goals are related to some industrial goals, so the industry's viewpoint should always be taken into account. The researcher's goal is to run the empirical study as a pilot study that is later replicated in the industry. The student's and the instructor's goals are to, respectively, learn and teach skills that will be used in the industry.

Therefore, there are a number of viewpoints that need to be taken into account when carrying out empirical studies with students, in addition to the researcher's viewpoint, which is the one that has been traditionally considered in the scientific literature. Scientific literature has usually examined only the benefits of pilot studies from the researchers' point of view.

The goal of this paper is to discuss the issues related to empirical studies with students, from the viewpoint of the various stakeholders that either explicitly or implicitly play a role: researchers, students, instructors, and the industry. Each of these roles has its goals, which may be conflicting, and constraints, which may hinder the achievement of the goals. Based on our experiences in carrying out empirical studies with students, we identify and discuss a number of benefits and costs from the viewpoint of each stakeholder. These costs and benefits are therefore technical (related to scientific research), pedagogical (related to the quality of the students' educa-

tion and instructors' teaching), and industrial (related to the possible industrial benefits of the data collected). We also discuss the ethical issues related to empirical studies with students, whose right to reach their educational goals should be guaranteed during pilot studies carried out during software engineering classes as much as it should during any other kind of educational activity. Based on our experience, we also offer concrete advice for carrying out empirical studies with students.

The paper is organized as follows. Since a relevant part of this paper is about educational issues, Section 2 concisely reports on the research and debate on software engineering education. Sections 3 – 6 focus on the costs and benefits for the major stakeholders involved in carrying out pilot empirical studies during software engineering classes, i.e., the researchers (Section 3), the students (Section 4), the instructors (Section 5), and the industry (Section 6). Ethical issues are discussed in Section 7. Conclusions (including lessons learned) and an outline of future work are in Section 8.

2. Background

This paper has its background in two fields. First, the software engineering education community is the reference point for the educational goals and methods. Then, the empirical studies with students as subjects show that significant work can be done in academic settings. These provide the context and motivate the need for our work.

2.1 Software Engineering Education

The computer science education community has been active for more than three decades and the SIGCSE Technical Symposium on Computer Science Education has now reached its 33rd edition. The Conference on Software Engineering Education and Training (CSEE&T) has reached its 16 edition. There are specific journals devoted to the field (like Kluwer Education and Information technologies). IEEE Software has recently devoted a special issue to the theme.

The general goal of the software engineering education community is to provide a pedagogically sound framework for educators [1] [2] [3]. Some of the issues which are under discussion in the community are computing programs and curricula, specific courses [4], laboratories, and alternative ways of teaching. Among other proposals, the importance of practice-based software engineering education is well accepted in the community and some good examples are reported in the literature [5].

The importance of project based software engineering learning is well recognized. Moreover, as pointed out for example in [6], there are still too few academic programs which offer curricula that focus on this type of education.

Another issue, which is getting more and more attention in the community, is industrial relevance [7]. Lethbridge reports about the results of a study that had the goal of extracting the software engineering topics that should be emphasized more or less in the software engineering curricula based on professionals requirements.

The software education community has exploited empirical methods to evaluate the quality of the provided education from its beginning. This method of evaluation is in line with the pedagogy and psychology research that has traditionally been based on student based investigations.

Empirical software engineering methods in education have received interest in the software engineering education community. In [8], the author reports on a student project in which students act as both experiment designers (under the teacher supervision) and as subjects of the software engineering experiment. The pedagogical goal of this kind of course is to teach students about empirical methods. Students will need this knowledge when they will be in the position of evaluating and proposing technology and processes.

2.2 Empirical Studies with Students as Subjects

There are many empirical studies with students as subjects that have been reported in the software engineering literature. Here we cite those reported in [9-12]. The explicit goal of validating one or several research hypotheses is the common factor among these studies. Pedagogical considerations are of secondary or little importance. For example, the study reported in [11] is about four studies in the context of software estimation. This study is a combination of industrial and student experiments. Students are regarded as less experienced than professionals, but pedagogical considerations about the course and its main goals, the year of study, and the background of the students are omitted.

The work reported in [13] examines the differences between students and industry people in empirical investigations. The conclusions are that there are significant differences from a research point of view, when comparing undergraduate and graduate students. On the other

hand, the differences are small between industry people and graduate students.

The same issue is also investigated in [14]. One contribution of this paper is that it identifies a set of conditions under which student experiments should be carried. Moreover, the relationship between pedagogical and empirical goals is explicitly addressed. The conclusions are that the learning goals of the courses and the research goals of the studies should be harmonized.

The authors of this paper have been involved as researchers/instructors in several studies with students [15 - 17]. These experiences have been the basis for the research described in the paper.

3. Benefits and Costs for the Researchers

Special care is required in the planning and execution of empirical studies in industrial settings, because they may require a good deal of time, effort, and resources. Empirical studies with students are often a way for reducing technical and organizational risks. This reduction of risk is one of the typical benefits that researchers have in mind when carrying out empirical studies with students. These and other ***benefits*** are discussed next.

- **Obtain preliminary evidence to confirm or refute hypotheses.** As a part of any engineering discipline, new ideas and hypotheses should always undergo an empirical validation before they are used in the industrial practice. Experiences obtained via empirical studies with the students as subjects may be a first source of information on the research hypotheses, which may be changed or refined if needed.
- **Control factors that may affect the study.** This control may not be present in all empirical studies, but researchers may have a greater control in an *in vitro* empirical study than an *in vivo* empirical study. Being able to control factors other than the phenomenon under observation may help the researcher ascertain whether a phenomenon of interest is statistically significant, for instance.
- **Show software companies the relevance of the research.** Like in any engineering discipline, positive evidence gives more strength to new ideas and helps their adoption by the industry when that evidence shows the practical usefulness of the ideas and the extent to which their application might contribute to the achievement of a software company's goals.
- **Show software companies the usefulness of carrying out empirical studies in their own environments.** Positive evidence obtained through empirical studies with students encourages software companies to carry out in further studies in their own environments.
- **Show software companies the feasibility of carrying out full-fledged empirical studies in industrial environments.** Through studies carried out with students, both the researchers and software companies may better evaluate the amount of resources needed and the amount of time required for a full-scale empirical study.
- **Fine-tune the organization and details of an empirical study, before it is carried out in an industrial environment.** Planning an empirical study in the industry is hardly ever a simple task. Many details can impair the execution of the empirical study, even when the problem and the hypotheses are well understood and spelled out. Researchers may use a preliminary empirical study with students (pilot study) to test the execution of the actual study and detect and remove many possible problems before running the empirical study in the industry, when the cost of failure would be very high. In addition, researchers try to prevent problems due to even trivial mistakes because failure in the execution of an empirical study may make it much more difficult for them to work with an industrial partner in the future.
- **Produce an experimental "kit."** Materials, guidelines, and data collection procedures must be carefully prepared and tested before running an empirical study in the industry. Thus, researchers have a complete and tested experimental "kit" before they go to the industry. This "kit" also helps in future replications of the empirical studies, in both academic and industrial settings.
- **Train junior researchers in the empirical research field.** Though this is obviously possible in empirical studies in the industry as well, the impact of empirical research on junior researchers may be "softer" if the empirical research is carried out in an academic environment instead of an industrial one. This is due to the closeness of junior researchers to the environment and the lesser responsibilities that are related to running an empirical study in an academic setting.

On the other hand, any study, whether in classroom or industrial contexts, entails ***costs*** for the researchers, two of which are listed next.

- **Effort.** Effort is needed on the researcher's side to create the design, prepare the materials, run the ex-

periment and analyze results. Although it is sometimes thought that classroom studies have fewer constraints than studies in other contexts, we have sometimes found that the unique constraints of the classroom can make it harder to plan appropriate study designs, and require at least as much if not more time to plan. For example, opportunities for training or conducting the experiment are limited, as students meet only at a certain number of pre-defined class times per week. Furthermore, the instructor has a pedagogical duty to teach all students, so having control groups is extremely difficult. Designing around such constraints can cost more effort than in other contexts.

- **Threats to validity.** Such compromises as to the "ideal" study design also cost researchers in a less easily measured fashion by producing results with additional threats to validity. For example, students have different motivations than software professionals (especially if participation is reflected in their grades), and so the experimental design has to guard against the possibility of exchanging answers. Or, data may be biased because inexperienced students can learn at a different rate across multiple treatments than software professionals would. For these reasons, researchers may have difficulty using the results in publications.

  The scientific literature has often focused on the possible risks associated with drawing general conclusions from empirical studies carried out with students, i.e., on the external validity of the results, and therefore their industrial relevance. Empirical studies with students have sometimes been criticized because of their supposed lower degree of external validity. While this is certainly an important problem, we would like to point out that:
  - External validity is an issue that needs to be taken into account in *any* empirical study, not only in empirical studies with students; many different variables (e.g., cultural and technological ones) are involved in a study, and those may be more influential than the subjects' experience;
  - Obvious as it may be, having an empirical validation with students of some phenomenon of interest, hypothesis, or new idea is better than no validation at all; too many Software Engineering techniques are all too often introduced in work environments without any kind of preliminary empirical assessment;
  - In many contexts, especially in the current US educational climate, the line between students and novice professionals is being blurred. More and more students are working over the summers or as interns in industrial environments, so they bring an expanded set of skills to many upper-level courses.

  As a consequence, while it would be scientifically incorrect to unduly overemphasize the general significance of the results obtained via empirical studies with students, it is also not true that those results are hardly relevant to the progress of the field.

Still, these costs must be measured against the benefits of using a readily available subject pool (who can in fact benefit pedagogically from their participation). Furthermore, if results have to be balanced against associated threats to validity, the problems and difficulties students have with the experimental protocols are usually a fairly accurate predictor of where difficulties would arise with a pool of professional subjects.

4. Benefits and Costs for the Students

We believe that empirical activities in software engineering classes will also provide benefits to the students, as we now describe.

- **Education on state-of-the-art topics.** Researchers usually carry out empirical studies on topics with close connections to the state of the art, since they investigate problems that still need to be solved. Thus, students are exposed to topics that may be more modern than those usually taught in software engineering courses.
- **Industrial relevance.** Students can get better insights on specific industrial problems, since empirical studies are usually carried out with an industrial goal in mind.
- **Hands-on practice.** An empirical study, which may be carried out in a situation that simulates an industrial application, gives students a better opportunity to assess for themselves their knowledge about a specific topic than "theoretical" classes would. In addition, students get acquainted with specific, practical problems that are constantly encountered by practitioners during their work and that can be unknown to the students before they are forced to face them.
- **Empirical methods.** Empirical studies show students the advantages of using quantitative methods even in a human-intensive business such as software engineering. By involving them in an empirical study, students may realize that there is a need to base the improve-

ment of at least some software development activities on firmer, evidence-based grounds.

- **Third party assessment.** Empirical studies may show the students that they should not be afraid of being the subjects of empirical studies and data collection activities in general. Students need to realize that during their work life they will be continuously subject to assessments and they will have to turn in reports, questionnaires and data for various purposes.

The flip side of the coin is the costs that are incurred by the students, as we detail next.

- **Potential lost education time.** In any class, there is always more information to teach than there is time to teach it. If students have to give up one or more classes of instruction in order to receive training and participate in a study, then they might be missing the chance to learn about a topic that could be particularly interesting or helpful to them. So, there is an opportunity cost for the students when empirical studies are carried out. This cost can be measured using a post-study questionnaire where the students are asked to rate their perceived value in the new technology as well as to describe any benefits they think they gained from learning it. We have used this method of evaluating the appropriateness of a study for a classroom and found that if the students understand why the study material fits the curriculum of the class, then they see more benefit and less cost, while if the students do not see the fit, they see less benefit and more cost.
- **Training in an ineffective technology.** Sometimes, the results of classroom studies may show that the technology being evaluated is less effective than a comparison technology or benchmark data. If this is the case then it could be argued that students have now spent time learning something that will be of no benefit to them. This risk needs to be mitigated by careful planning on the part of the instructor, so as not to gamble students' time on evaluations of technologies that lack a minimum of existing objective data giving some confidence in their effectiveness. However, when even carefully planned evaluations of technology show a lack of benefit, one potential benefit is that the students can be learning a new, cutting-edge technology that might provide them with a much-desired skill, even if it is not applicable to the current software engineering problem. Additionally, by being a participant in an empirical study, the students learn that new technologies can be evaluated empirically and that they should not naively accept any new methodology or technology simply because it is new.

5. Benefits and Costs for the Instructors

When carrying out empirical studies with students the researcher and the instructor are often the same person. However, this is not necessarily the case, but even so, the same person is playing two different roles, with two different sets of goals and responsibilities. As a researcher, one needs to provide scientifically sound results to the research community and the industry. As an instructor, one needs to provide his or her students with the best education possible for their future activities. The instructor's goals may conflict with the researcher's goals, so a sensible trade-off should be sought. Thus, it is important that the instructor's goals in carrying out empirical studies with students be made explicit and contribute towards the general goal of improving software engineering teaching and learning. Here is a list of teaching benefits that empirical studies with students may help achieve.

- **Stimulate the instructor to use less conventional ways of teaching.** All too often, teaching is carried out in a traditional way. The instructor gives his or her lectures in the same way as many other disciplines, with little or no room for students' participation. There are several reasons for this, including the limited amount of time instructors have for preparing their classes, the tendency to reuse materials used in previous years, etc. Especially if the instructor is the same person as the researcher, he or she will find new incentives in using a different way of teaching the subject.
- **Introduce problem based software engineering education.** The design of an empirical study can help instructors design problem based education initiatives. For instance, our experience is that a process modeling exercise could be modified to become a process modeling empirical study. This point makes sense in contexts in which such exercises do not exist from before. In addition, instructors may be willing to change the way they teach a subject only if they can see the advantages from an educational and a research point of view.
- **Stimulate teamwork.** Empirical studies often require that teams of students be formed, to replicate the working practice in the industry. This situation helps students work in groups and understand the advantages and disadvantages of teamwork, especially in a

cultural environment in which individual achievements are usually encouraged and rewarded.

- **Establish a more direct and closer channel of communication between the students and the instructor.** By monitoring the empirical study, the instructor obtains much better feedback about what the students have learned. This feedback helps the instructor in the short term and in the long term. In the short term, the instructor may change the remaining classes of the course if necessary to reach the pedagogical goals of the course. In the long term, this continuous feedback helps the instructor rethink and modify the overall organization of the course, its contents, the time allocated to each subtopic, the materials used, the teaching techniques used, etc.
- **Obtain an evaluation of students that is not based on episodes.** Students are often graded based on the result of two tests (one halfway through the classes and one at the end) or even a single test carried out at the end of the classes. Though this grading method may be an economical way for evaluating the students' proficiency in the subject, the question remains: is it also an effective evaluation way? It is well known that many factors may influence the result of a test in addition to the subject's knowledge: the subject's tendency to get emotional under acute stress conditions, as a test may be perceived by some students; the subject's mental and/or physical conditions during the test; the subject's familiarity with the materials used during the test, which may be different from those used during the class (for instance, for organizational reasons, a test may be administered by using paper materials, while the classes have used computers); even small misunderstandings may greatly influence the results of a test. Continuous monitoring and observation of students will certainly provide a more accurate and effective evaluation of the students' skills and knowledge.
- **Keep the students' attention at a good level.** Sometimes, classes suffer from a "mortality" effect, and the attendance declines as the class progresses. There are several reasons for this phenomenon. The students may lose interest in the subject, prefer to attend other classes, or have problems understanding the concepts explained during the lectures. Again, a more direct instructor-to-student channel may help with these problems, by making the students feel that the instructor has a real interest in the student's learning.
- **Introduce empirical software engineering as a part of software engineering teaching.** Empirical software engineering is often a neglected topic in the teaching of software engineering. Software engineering classes often only contain a few references to software metrics. This is due to the background of many software engineering instructors, who have often had only a cursory introduction to empirical methods. In addition, it is true that software engineering as a discipline has proceeded on a mostly "ideological" basis. Software engineering methods and techniques are almost always introduced in the industrial practice with little or no evidence or data to back their introduction or to assess their strengths and weaknesses. In addition, the impact of the introduction of new software engineering methods and techniques is often assessed only on an anecdotal basis, if at all. Empirical, whether qualitative or quantitative, methods are hardly ever used. This is contrary to good scientific or engineering practices, which (1) recommend that new methods and techniques be introduced only after there is some kind of solid evidence that they will work, (2) require that their effects and side effects be studies after their introduction. Using empirical studies during classes may help instructors appreciate the usefulness of empirical software engineering during software development and the need for teaching empirical methods as a part of a software engineering curriculum, as is commonly done in many other engineering disciplines.
- **Development of a critical attitude in the students.** Instructors may be able to teach future software engineers that new methods and techniques *can* be studied empirically and that they should not just blindly accept the newest technique around just become someone has a nice sales pitch. The knowledge of empirical methods gives them some basis for asking for more evidence of the value of a technique.

It is true that there may be other ways of obtaining the above benefits without carrying out empirical studies with students. For instance, an instructor may introduce empirical software engineering in his or her classes after reading books or papers on empirical software engineering. Even so, it is more likely that his or her own teaching of empirical software engineering will be more effective after he or she has had a hands-on experience on observing the various phases of an experiment. In addition, like any other stakeholder in software engineering education, instructors must be motivated to change the way they have been teaching software engineering and they need to have evidence that the new way will be an improvement over their current way of teaching. As a conse-

quence, a continuous assessment must be done of the pedagogical effectiveness of teaching empirical software engineering through experimentation. This, however, should be the case for any other topic and teaching technique used during the classes.

We now summarize some of the costs for the instructors

- **Class meetings used for the empirical study.** The instructors must give up classes for the study to be run. These classes include both training sessions, and if applicable, time for running the study. This cost will vary depending on the other materials that could have been potentially taught during those classes. If the material contained in the training sessions covers a topic that was already planned for the course, then the cost to the instructor is not as great. These costs can be reduced by conducting the study as a homework assignment, i.e. by using students' time outside the classroom for performing the task. However, it should not be surprising that reduced costs can imply lower quality results, in this case due to lack of control over students' conformance to the experimental protocol and diminished confidence in the accuracy of any self-reported data.
- **Getting acquainted with the empirical study.** The instructors may also need to spend additional time to familiarize themselves with the empirical study, if the researcher and the instructor are two different persons. The reason is that the students expect the instructor to be able to answer their questions on this assignment too.

6. Benefits and Costs for the Industry

One important precondition for planning and starting empirical studies in industrial settings is that there should be reciprocal trust between industrial actors and researchers. Industrial actors need to believe in the importance of the empirical investigation and have strategic expectations that go beyond the economic compensations that they may get for the use of resources in the empirical studies.

Student courses can be seen as a cooperation context between industry and academia. And they may contribute to enforce the reciprocal trust between industrial actors and researchers/instructors. Examples of software engineering courses which rely on the cooperation with industry for educational goals are documented in the literature [18]. The interaction model described in this paper is to let students work to produce a software product for a real customer of a real organization. In this model, students play the role of analysts, designers, implementers, and testers.

We have also combined empirical studies and industry involvement in software engineering courses [16, 19]. Our experience of this kind of projects shows the following benefits for industrial actors:

- **Obtain preliminary evidence to confirm or refute hypotheses about new technology or method adoption.** Industrial actors, which are software engineering technology providers, tend to give students problems on technology or method evaluation. The goal is to use the student project as a kind of pilot projects in which, say, software development platforms (e.g., EJB or .net), or standards of various kind (e.g., RUP or XML) are assessed. Such assessment usually happens in an informal way.
- **Obtain preliminary evidence on methods and technology they have already adopted.** Industrial actors may obtain an evaluation and a comparison with other methods. For instance, hardly any industrial software developer can afford to develop the same product with two different technologies, to evaluate which one is more effective. Instead, this kind of study can be carried out in an empirical study with students.
- **Industrial actors can get ideas about new product development.** Industrial actors who are relying on an information system for their mission (e.g., public administration entities), assign to students problems that aim at requirements understanding and prototyping.
- **Industrial actors learn about new software methods.** Empirical studies foster the cooperation between the industry on one side and instructors/researchers/students on the other side. So, the industry may receive new perspectives on some the process, quality, and software technology, based on the framework of the course.
- **The industry gets better-quality graduates from education institutions.** Students with a more complete knowledge and more hands-on practice with technologies, who understand how to choose appropriate technologies for their current project, and who understand how measurement can be used to improve project management.
- **Obtain a better idea about costs and benefits of running an empirical study.** Industrial actors can be inspired and convinced to replicate the same study in house either in a pilot project or even in wider con-

texts when they know how expensive it would be and the extent of benefits it might provide. This is in fact a general goal of students based empirical studies and we believe that it is even more valid when the industrial actor has a deep practical knowledge of the study than when he/she has only read or been told about it.

- **Obtain evidence needed to convince the upper management.** The industrial actors should not be considered all alike. They actually belong to different categories. Often, developers and managers need proof to convince the upper management that the effort required to implement a new technique will be worth it in the end.
- **Knowledge on empirical software engineering.** The industrial actors learn about empirical software engineering and about the possibility to run their technology and method assessment trials in a more efficient way. This applies mainly to software engineering technology providers. This may prompt the establishment of a measurement program, for instance.

An industrial organization's costs may be summarized as follows.

- **Extra effort required on the industry's side.** The person-hours of trained employees (every organization's most valuable resource) that need to be spent on preparation for the experiment are a cost for the software organization. Although experiments in classrooms are sometimes assumed to be "free" for the industrial organization that may benefit from the results, this is seldom completely true, especially if the organization has a specific question they want addressed. At a minimum, experts in the existing development process must be made available for interviews and feedback sessions with the researcher. These interviews can help the researcher to tailor the object of study to make sure it is appropriate for use in the industrial context, to become acquainted with the organization's development processes in order to recreate them as closely as possible in the students' task, etc.

Closer collaborations (requiring additional industrial person-hours) may be necessary in order to make available existing documents for use in the student context. Since documents from the industrial context can rarely be used "as is," additional effort is usually necessary in order to adapt the documents for the empirical study.

In short, costs to the organization can vary depending on the extent to which the organization feels comfortable in assigning its personnel to work with researchers. Even at a minimum, the organization can receive some benefit in receiving early results about a new technology, and a debugged experimental protocol if professionals are to be used in a later assessment of the technology. However, increasing the number of hours spent on collaboration and explaining the current state-of-the-practice yields increased value, in terms of results being better able to be extrapolated to the industrial context.

7. Ethical Issues

When carrying out empirical studies with students during a course, a number of ethical issues arise. The other teaching activities in the course should be obviously geared to the exclusive benefit of the students, while this is not the case in an empirical study where students are the subjects. While we do not have conclusive answers, it is important to discuss these ethical issues for a number of reasons, including the pragmatic reason that dealing with them in an adequate way may make empirical studies more successful for all the actors involved. Ethical issues in empirical studies in the industry have often been taken into account (e.g., see [20]), but they are also becoming to be recognized as important ones in pilot studies with students (see [21]). Here we outline some possible ethical issues that are especially relevant in empirical studies with students.

- Is it ethically correct to "use" students for the researcher's benefit in the context of a college course? For instance, would there be more productive ways for the students to spend the time devoted to participating in an empirical study? The same question may actually be asked about any topic taught during a course. For instance, during an introductory programming course, which language should be taught? Or, should be more than one language be taught, given that the students will probably use several languages when they work in the industry? Although there is no right answer to this question, it is clear that the students' own interest need to be a primary concern when designing the empirical study. The topic of the empirical study must be well within the main topics of the course, so the students may use the empirical study to learn something new on the topic or to deepen their knowledge of the topic.
- Is it ethically correct to base some of the final evaluation of a student on his or her performance in the empirical study? For instance, if the empirical study is about finding defects in an artifact, should the students be graded according to the number and criticality of defects they find? A point that may seem to support

this use of the results of an empirical study is the fact that workers are evaluated in work environments according to the outcome of their activities. However, one must also keep in mind that there is a difference between regular production activities and empirical studies, where, for instance, one would like to evaluate a new technique. Thus, the focus of the evaluation should theoretically be on the technique and not on the subjects, i.e., it would be conceptually incorrect to design an empirical study to evaluate the subjects. In some cases, however, to make sure that the students' participation will provide quality data and truly encourage the students to take the assignment seriously, it may be necessary to base at least some of the final grading on the results themselves. Thus, this is an open point on which it is not possible to provide advice that applies to all cases.

- Is it ethically correct to base some of the final evaluation of a student on his or her degree of participation in a research study? This point is different from the previous one, since the instructor would not use the outcomes of the empirical study provided by the students, but the quality of the data they provide and their degree of involvement. Quality of data and degree of involvement may be used as indicators of the width and depth with which the students have learned during the empirical study. They should be used by the rewarding mechanism in place, since this will stimulate the students to widen and deepen their knowledge for their own good.
- How ethically correct is it to encourage the students to participate in an empirical study? It is clear, for the very benefit of all the stakeholders, that students should not be forced to participate in an empirical study. Therefore, they should be able to participate on a voluntary basis and withdraw from the study if they no longer want to be a part of it (as discussed in [21]). This implies that participation in an empirical study should not be used as an element to decide whether a student passes or fails the exam. However, incentives (in terms of their final grade, for instance) need to be given to the students who participate in the empirical study, based on the obvious principle that students who are willing to do additional work should obviously be rewarded. As a matter of fact, it is customary in several courses to provide *optional* assignments. For instance, students may be asked to present an essay on a specific subject. In some cases, the instructor/researcher is one of the beneficiaries of the assignment, since he or she may ask the students to do research on a subject he or she is not completely familiar with. The participation in an empirical study may very well be likened to one of these assignments. The instructor/researcher should also realize that it is obviously better to have well-motivated subject in the empirical study than subjects who participate only because they are afraid of possible negative repercussions if they do not participate even if participation is not mandatory. So, the instructor must make it clear that no form of punishment will be taken for the students who do not participate. It should be clear, however, that a voluntary basis also entails the risk of self-selection of the sample of subjects. For instance, only the best students may enroll in the empirical study. This would produce results that may not really representative from the researcher's point of view.
- Is it ethically correct to withhold information from the students as to the goals of the empirical study? It depends on the goals of the empirical study. Suppose that the goal of the empirical study is to prove that a certain theory is correct. Then, it would not be scientifically correct to tell the students what the goal is, since this may bias the results. This is especially true if the instructor/researcher is also the proponent of the theory, since the students may try to provide only data that support the theory. In these situations, it is advisable that researchers discuss the goals and experimental design and sometimes even preliminary results with the subjects at the conclusion of the study.In other cases, the goals of the empirical study may be safely disclosed to the students, for instance when the study is an exploratory one and not a confirmatory one.
- Is it ethically correct to withhold information from the students as to how the data they provide will be used? The students need to be informed about how the data will be used, so they can provide their informed consent to participate in the empirical study [21]. However, a formal acknowledgment is probably hardly ever necessary, and it might actually be counterproductive. A formal acknowledgment, e.g., a signed informed consent form, may make the process unnecessarily bureaucratic and it may actually scare the subjects, who may believe that by signing they give the researcher or the instructor permission to use the data against them. Specifically it must be made clear to the students that the data they provide will remain confidential and anonymous, i.e., they cannot be used against the students.

8. Lessons Learned and Conclusions

Empirical investigations with students as subjects have been beneficial for researchers for a long time. This is not peculiar of the software engineering field as many other fields such as pedagogy and psychology have been using students as subjects of experiments and empirical investigations in general.

In the software engineering field, empirical studies can be combined with project based education. The importance of project based education has been recognized by the software engineering education community and is well documented in the literature. However, project based education is costly in terms of educator, lab, and also students resources. Therefore, empirical studies may also be seen as a way to promote this kind of education.

To conclude, we would like to offer the following concrete advice for researchers running experiments in a classroom environment. This list can be a useful reminder of things for instructor/experimenters to double-check during course preparation:

- **Make sure the study is well-integrated with the goals and materials covered in the rest of the course. On the assignment sheet, explicitly state the anticipated educational benefits for students.** Instructors are more familiar with the course material than students, and often see the big picture of how the assignment fits into the class but do not communicate that knowledge in sufficient detail.
- **Give realistic time estimates. If necessary, run a pilot study or use a few volunteers to test the estimates before giving them to the class.** One of the frustrating things about learning software engineering for students is that there is no definitive right answer to many questions, against which their own answers can be compared. This also includes the time needed to complete software engineering tasks. Mistaken time estimates cause a great deal of frustration and even anger on the part of students, who might feel less intelligent for not being able to get a task done in an arbitrarily short period of time. If subjects become frustrated, the data provided may be of lower quality because subjects may either quit before completing their task, or may do a poor job once they have exceeded the estimated time.
- **Properly motivate the subjects but do not reveal the goals, measures and analysis to them *prior to executing the study*, unless absolutely necessary. Do make it clear *after executing the study* what the study design and analysis are all about.** Although the temptation to explain the specific hypotheses of the study to subjects is fed by good intentions (mainly a desire for students to understand the activity in which they are taking part), doing so can bias the running of the experiment and thus negatively affect the results.
- **Allow students a chance to give feedback. Make their opinions, if clearly based on empirical evidence, count.** An important way to motivate subjects is to make it clear to them that their feedback counts more than their simple ability to follow a process and generate data. Since software processes are interesting only in so far as they are able to be executed by human beings, the responses of subjects as to whether the given process is too onerous, too time-consuming, etc., should be interesting to researchers also.
- **Allow industry a chance to give input and feedback.** An important way to motivate subjects is to convince them that they are learning is relevant for the industrial world. At the same time, it may be important for industrial roles who are going to allocate resources to future investigations to influence the design of student based studies.
- **Avoid conflicts with other commitments.** Students may have to choose among several different commitments. They need to attend several different classes, do homework and classwork, and get ready for their tests. Thus, students have to make decisions on how to allocate their time and effort. These decision should be based on the students' own benefit, so students tend to make decision rationally, but allocating their time and effort on those activities that give them the highest return. The experimenters should therefore plan the experiments in such a way as to minimize possible conflicts with other activities; otherwise the students' response may not be adequate either in quantitative or qualitative terms.
- **Give the students feedback on the results of the experiment.** One of the challenges of project based software engineering education is that of providing feedback on the deliverables they produce. It is well known that there are no standard solutions to software engineering problems and the activity of reviewing software engineering artifacts is time consuming. Our experience is that the process of analyzing deliverables for research purposes, can give valuable feedback to students.

Our future work will comprise completing the empirical studies with students, which we are running and to plan

new studies that take into account the lessons we have learned.

In addition, our challenge is to define the quality of the provided education in quantitative terms to validate and improve our knowledge on the benefits and costs we have presented in this paper.

Acknowledgments

This work has been partially supported by the ESERNET (IST-2000-28754) EC-IST Project.

References

[1] P. Denning, "Educating a New Engineer," Communication of the ACM, vol. 35, pp. 83-97, 1992.

[2] B. Meyer, "Software Engineering in the Academy," IEEE Computer, vol. 34, pp. 28-35, 2001.

[3] T. B. Hilburn and W. S. Humphrey, "The Impending Changes in Software Engineering Education," IEEE Software, vol. 19, pp. 22/24, 2002.

[4] R. Cannon, J. Diaz-Herrera, and T. B. Hilburn, "Teaching a Software Project Course Using the Team Software Process," ACM SIGCSE Bulletin, 33rd SIGCSE Technical Symposium on Computer Science Education, vol. 34, pp. 369 - 370, 2002.

[5] R. Andersen, "Project Courses at the NTH: 20 Years of Experience," presented at Software Engineering Education, 7th SEI CSEE Conference, San Antonio, Texas, USA, 1994.

[6] D. J. Bagert, T. B. Hilburn, G. Hislop, M. Lutz, M. McCracken, and S. Mengel, "Guidelines for Software Engineering Education Version 1.0 (1999)," CMU/SEI CMU/SEI-99-TR-032, 1999.

[7] C. T. Lethbridge, "On the Relevance of software education: A survey and some Recommendations," Annals of Software Engineering, vol. 6, pp. 91/110, 1998.

[8] M. Höst, "Introducing Empirical Software Engineering Methods in Education," presented at Conference on Software Engineering Education and Training (CSEE&T), Covington, Northern Kentucky, USA., 2002.

[9] V. R. Basili, L. C. Briand, and W. L. Melo, "A validation of object-oriented design metrics as quality indicators," IEEE Transactions on Software Engineering, vol. 22, pp. 751 - 761, 1996.

[10] Porter A., Votta L., and B. V.R., "Comparing Detection Methods for Software Requirements Inspection: A Replicated Experiment," IEEE Transactions on Software Engineering, vol. 21, pp. 563-575, 1995.

[11] M. Jørgensen, K. H. Teigen, and K. Mølloken, "Better Sure than Safe? Overconfidence in Judgment Based Software Development Effort Prediction Intervals," Journal of Systems and Software, 2003.

[12] L. Prechelt and W. F. Tichy, "A controlled experiment to assess the benefits of procedure argument type checking," IEEE Transactions on Software Engineering, vol. 24, pp. 302 -312, 1998.

[13] P. Runeson, "Using Students as Experiment Subjects - An Analysis on Graduate and Freshmen PSP Student Data," EASE 03, 2003.

[14] M. Höst, B. Regnell, and C. Wohlin, "Using Students as Subjects - A Comparative Study of Students and Professionals in Lead-Time Impact Assessment," Empirical Software Engineering, vol. 5, pp. 201-214, 2000.

[15] L. Baresi, S. Morasca, and P. Paolini. "An Empirical Study on the Design Effort of Web Applications. In Proceedings of the 3rd International Conference on Web Information System Engineering (WISE2002), pages 345–354. IEEE-CS Press, 2002.

[16] M.L. Jaccheri, "A software quality and software process improvement course based on interaction with the local software software industry," in the Tutorial section of Computer Applications in Engineering Education Journal, John Wiley and Sons, Inc. page 265-272, Mar 2002.

[17] F. Shull, J. Carver, and G. H. Travassos, "An Empirical Methodology for Introducing Software Processes," in Proceedings of European Software Engineering Conference, Vienna, Austria, Sept. 10-14,2001. pp. 288-296.

[18] R. Andersen, J. Krogstie, G. Sindre, A. Sølvberg, "Project Courses at the NTH: 20 Years of Experience, " Software Engineering Education, 7th SEI CSEE Conference, San Antonio, Texas, USA, Springer, 1994.

[19] M.L. Jaccheri, R. Conradi, B. Dyrnes, "Software Process Technology and Software Organisations," EWSPT'00, Seventh European Workshop on Software Process Technology, Kaprun, Austria, 22-25 Feb. 2000, Springer Verlag LNCS 1780, p. 96-108

[20] R. B. Grady, "Practical Software Metrics for Project Management and Process Improvement," Prentice Hall, 1992.

[21] J. Singer and N. G. Vinson, "Ethical Issues in Empirical Studies of Software Engineering," IEEE Trans. on Software Engineering, Vol 28, N. 12, pp. 1171 - 1180, Dec. 2002.