Empirical Software Engineering, 6, 105–110, 2001.

Ethical Issues in Empirical Software Engineering: The Limits of Policy

ANNELIESE AMSCHLER ANDREWS aaa@cs.colostate.edu
Computer Science Department, Colorado State University, 601, South Howes Lane, Fort Collins, CO 80523

ARUNDEEP S. PRADHAN apradhan@research.colostate.edu
Technology Transfer Office, Colorado State University Research Foundation, 601, South Howes Lane, Fort Collins, CO 80523

Abstract. Empirical studies in software engineering can involve a variety of organizations, each with their own set of policies and procedures geared at safeguarding the interests and responsibilities of the researchers, students, the collaborating company, the university, and possibly national funding agencies like the National Science Foundation and the National Institute of Health. Each of these organizations have differing goals for participating in these studies and bring widely different cultures and expectations to the table. While policies, procedures, contracts, and agreements set expectations, they by themselves cannot ensure ethical behavior. This position paper describes some of the common approaches to encourage ethical behavior and their limits for enforcing ethical behavior.

Keywords: Ethics, empirical studies

1. Introduction

Empirical studies in software engineering use software product or process data to understand better how software is built and behaves, to assess the quality of the processes used throughout the software life cycle, and to assess software products and production techniques. Some studies work on data furnished by the company, others observe people at work.

This means on the one hand, that companies have to trust researchers and their students with a variety of data that can be vital to a company's interest. Examples include defect data, product source code, morale of their software developers, plans for product lines, to name a few.

On the other hand, researchers and students trust the company that any data they get is truthful and valid. The university must trust the research team that it conscientiously evaluates the data and neither overinterprets results, nor brushes unpleasant results under the table. Lastly, if there are human subjects involved, they trust the researchers and their employer that data collected about them and their behavior will not be used in an adverse way against them.

Collectively, these expectations could be called expectations of ethical behavior. To be sure, ethical behavior cannot be legislated. However, all parties involved put safeguards in place to protect their interests and limit liabilities.

In the following, we analyze some of these safeguards and point out limitations in their ability to enforce ethical behavior.

2. Trust with Information

Information here means any and all information that is disclosed by the company to further a research project. How much information is disclosed by the collaborating company varies widely. Therefore, the information needs to be defined as concisely as possible and generally limited to the actual information provided by the company. This is often distinguished from the information generated under the study because academic institutions typically have different guidelines for these two types of information. This provides a comfort level for both parties because the information is easily documented and tracked. However, open ended definitions of information often lead to misunderstanding, confusion and may ultimately result in adverse outcomes. Some companies provide data that has already been "scrubbed" and made anonymous. Others disclose fully the raw data and enable not merely high level analysis, but full root cause investigation. Information can include all metrics collected about process or product. In may cases, though, this also includes access to algorithms, product architecture, details about how software is built, long term plans. Examples of situations where many such details are disclosed as a natural part of the work are

- Empirical evaluation of reverse engineering techniques, code analysis methods, code decay analysis.
- Assessment of the utility of development methods (e.g., use of patterns or object oriented concepts).
- Assessment of the quality of source code.

In such situations, the researchers and the students by necessity acquire detailed knowledge about the software they are studying. The standard approach to safeguard such information is to execute a non-disclosure agreement. This type of an agreement establishes the boundaries within which the faculty and students operate. However, the following issues still need to be considered:

1. Monitoring and Enforceability. Students graduate. At this point it becomes difficult to monitor, much less enforce, a non-disclosure agreement. The knowledge acquired about a software product could be used in inappropriate ways (e.g., building a competing product) and it often would be very difficult to even identify the culprit as a former team member.

A similar point can be made about researchers themselves. This is one reason why some companies ask higher level representatives of a researcher's institution to sign a non-disclosure agreement. It is neither feasible nor cost effective for the research administration at academic institutions to monitor and review all research projects to ensure that non-disclosure agreements are followed. The primary responsibility for monitoring appropriate use of the information falls directly on the researchers.

Obviously then, this still leaves a collaborating company at some risk with respect to unethical use or disclosure that may be difficult to trace to its source.

One of the reasons that companies and universities collaborate on such studies is the collective and diverse knowledge and expertise that is available at a university. The result of any study is an education experience for the students and researchers and adds collectively to their knowledge base. In software engineering, "industry is our laboratory", as one researcher once put it. This is especially true for empirical software engineering research. We validate software engineering development and analysis methods "in vivo". Further, working with industry helps us to identify what the real practical issues are for which we, as researchers, should be finding solutions. Lastly, adoption of methods and research outcomes is the ultimate validation of our work. While agreements and university policies can encourage ethical behavior in these activities, it is ultimately the mutual trust of researchers and industry that makes these relationships work.

2. Responsibility. The other side of the issue is the need to properly disclose all relevant information gained in the study, some of which may represent desirable results, some of which may not (e.g., when comparing the efficacy of several debugging tools, the sponsor's may come in last. In this case there may be pressure to not disclose negative information). Given that the research is usually funded by the collaborating company, this creates a potential conflict of interest that must be disclosed and properly managed.

3. Publications. A major objective of academic institutions is to educate, to perform research and to publish the results of research activities. This is in contrast to many industrial software development environments where most information is considered proprietary and is seldom published. Any restrictions on a university's ability to publish severely compromises its principles and adversely affects graduate students, post doctoral fellows and faculty. The university and the researchers in their role as advisors need to safeguard the right to publish student work (theses) and scholarly articles. This is counterbalanced against the need to protect company interests. It has happened in longer term collaborations that competitors analyzed a series of publications, interpreted them for their own interests and turned this interpretation into negative publicity. Surely, this is potentially chilling for the collaborative climate in empirical software engineering.

A common way around this is for companies to disclose "scrubbed" data to researchers, or to keep the identity of the company a secret. In the first case, the researchers lose the ability to check whether the data is factual. In the second, the resulting publications can no longer be traced to the company where they originated and the company may lose the ability to check whether the published results are factual or doctored. A third approach is to agree to delay publications for a reasonable period of time so that the company can establish a product position and/or appropriate intellectual property protection. It is often easy to arrive at a pragmatic solution to issues regarding publications without compromising the needs of the company or the university.

The integrity of research data is not a new problem. In the life sciences, more comprehensive safeguards are in place than we currently apply. Yet, ethical research misconduct still happens.

Related to this issue is the question of properly interpreting data and results. In all too many papers, small results are overinterpreted. With enough detail disclosed in the paper, reviewers and readers will be able to assess the true value of the results. With heavily scrubbed data, or when key information cannot be disclosed, this is no longer possible. Lack of opportunity for independent evaluation opens doors to and invites unethical behavior. Empirical software engineering still has a way to go in defining what amount of data and attribution should accompany publications of results so they are credible. For the most part, we believe the data and results we see in publications and trust the peer review process.

3. Conflict of Interest

A conflict of interest or the appearance of a conflict of interest can broadly be defined as a situation under which the researcher or student benefit materially under a relationship with a company. A majority of US universities have conflict of interest policies that are modeled on the guidelines provided by the National Science Foundation and/or the Public Health Services. In addition to these guidelines there are situation specific issues that determine whether there is an actual or a perceived conflict. Any industry sponsored research in an academic institution has the potential to create a conflict of interest, perceived or actual. Examples include researchers who own stock in the collaborating company, or students who perform studies on software or software engineering methods developed by their research advisors, research funding that comes from the company whose software or software development processes are studied. Each of these situations can lead to the perception of pressure to make a project produce certain results that are favorable to the company, or, in the case of students, to the research advisor. Any of these situations may represent perceived or actual conflicts of interest. Their existence does not mean that the study should not be performed, but rather, it calls for appropriate disclosure and management guidelines to be established by the institution and the company to permit the study to go forward and to ensure credible results. One way to ensure credibility is to disclose, in publications, data that allows other researchers to evaluate the work independently. Of course, this may be at cross-purposes with the company's need for confidentiality of certain information.

4. Regulatory Compliance

Most universities require special approval for research that involves people. In software engineering, studies that involve human subjects include experiments with students (on a variety of topics including inspections, learning languages, value of

certain techniques and processes), as well as professionals. US universities are required by federal agencies to evaluate proposed studies for any adverse effects on the subjects. While usually there aren't any, this is not true for all studies. For example, if the objective is to evaluate programmer behavior under heavy time pressure, this may well lead to stress related problems. Usually all potential adverse effects must be disclosed to the subjects ahead of time and they must be given the opportunity to cease participation without adverse effects (such as a lesser grade in the case of student subjects).

Issues of ethical behavior arise when companies try to use data on software or process assessment surreptitiously for performance evaluation of individuals. For example, a study analyzes a software system for fault-proneness. The supervisor uses the results to identify the individuals who are responsible for the most fault-prone parts of the software. Besides the negative effects on the individuals who have participated, it also leads to protective behavior on the parts of the individuals. Potentially negative data simply isn't disclosed. The credibility of the data suffers.

It is important that participants in empirical studies can trust the experimentor and the company with the design of the study and how the results will be used.

5. Conclusions

When all is said and done, one has to ask the question "What are the implications of these guidelines for the university and the company?" The University benefits from gaining additional knowledge, providing students and researchers with industrial experiences and providing a mechanism for validating new methods. The company receives valuable information regarding its software, development processes, and ways to make them more efficient and effective. There can be any number of guidelines, policies, and procedures in place to manage situations and to encourage ethical behavior. But the bottom line is that none of these can guarantee ethical behavior. Thus, in the end, credible results and a strong discipline of empirical software engineering are based on mutual trust that everyone will behave ethically.

Dr. Anneliese Amschler Andrews is a Professor at Colorado State University. She is also Director of the Colorado Advanced Software Institute, a consortium of businesses, and Colorado research universities that supports collaborative Technology Transfer Research projects. She is the author of a text book and

numerous articles in the area of Software Engineering, particularly software testing and maintenance. She holds an MS and PhD from Duke University and a Dipl.-Inf. from the Technical University of Karlsruhe. She is currently Editor in Chief of the IEEE Transactions on Software Engineering. She has also served on several other editorial boards including the IEEE Transactions on Reliability, the Empirical Software Engineering Journal and the Journal of Software Maintenance. She contributes to many conferences in various capacities including general or program chair, or as a member of the steering or program committee.

Arundeep S. Pradhan received a Bachelor in Pharmacy from the Birla Institute of Technology & Science (India) in 1985 and MS in Pharmacy Administration from the University of Utah in 1989. He worked at the Technology Transfer Office from 1987 to 1999. In 1999, he joined the Colorado State University Research Foundation as Director for Technology Transfer. He has negotiated numerous contracts (over 200) between companies and the institution for the transfer of technology developed at the institution and for industry sponsorship of research at a university. Mr Pradhan has also participated in the development of policy regarding commercialization and use of intellectual property created at a university.