Journal of Science and Technology
ISSN: 2456-5660 Volume 7, Issue 02 (March - April 2022)
www.jst.org.in *DOI:https://doi.org/10.46243/jst.2022.v7.i02.pp384 - 394*

Experimental validation of new software engineering

N. Mownika, J.VijaySree, P.Hema sai
Assistant Professor[1,2,3],
Dept. of CSE,
mail-id:mounika.nekkanti@gmail.com, mail-id:jvijaya.pavani@gmail.com,
mail-id:hemasai531@gmail.com
Anurag Engineering College,Anatagiri(V&M),Suryapet(Dt),Telangana-508206

To Cite this Article
N. Mownika, J.VijaySree, P.Hema sai, **Experimental validation of new software engineering"** *Journal of Science and Technology, Vol. 07, Issue 02,-March – April 2022, pp384-394*
Article Info
Received: 21-03-2022 *Revised: 19-03-2022* *Accepted: 10-04-2022* *Published: 27-04-2022*

Abstract
When to apply a new technology in an organization is a critical decision for every software development organization. Earlier work defines a set of methods that the research community uses when a new technology is developed. This chapter presents a discussion of the set of methods that industrial organizations use before adopting a new technology. First there is a brief definition of the earlier research methods and then a definition of the set of industrial methods. A comparison of the two sets leads into the perspectives of these methods of experts in the research and industrial community via surveys made to those communities.

Keywords: Experimentation, Models, Software engineering, Survey, Technology transfer

1 Introduction

Although the need to transition new technology to improve the process of developing quality software products is well understood, the computer software industry has done a poor job of carrying out that need. Often new software technology is touted as the next "silver bullet" to be adopted, only to fail and disappear within a short period. New technologies are often adopted without any convincing evidence that they will be effective, yet other technologies are ignored despite the published data that they will be useful. One problem is that two distinct communities are involved in the technology transition process:

The purpose of this work is to enable understanding of these communities and understand their differences. Just as Snow described the literary and scientific communities having difficulties because each lacked respect for and mistrusted the other [Snow63], as time passed, these communities did accept the other because they began to understand the role of each. In the two communities of research and industry, understanding each other better should help in developing research programs that are better able to meet the needs of both of these communities.

The methods used by each community to evaluate new technology differ, and this "disconnect" between these two communities is part of the difficulty in moving new ideas from the research laboratory into industry. Researchers have studied the role of experimentation in computer science research, for example [Fenton94]. However, most of these have looked at the relatively narrow scope of how to conduct valid replicated scientific experiments within this domain. The problems of the role of experimentation as an agent in transferring new technology into industry are the focus of this chapter.

Researchers, whether in academia or industry, have a desire to develop new concepts and are rewarded when they produce new designs, algorithms, theorems, and models. The "work product" in these cases is often a published paper demonstrating the value of their new technology. Researchers often select their research topics according to their own interest; the topics may or may not be directly related to a specific problem faced by industry. After achieving a result that they consider interesting, they have a great desire to get that result in print. Providing a good scientific validation of the technology is often not necessary for publication, and studies have shown that experimental validation of computer technology is particularly weak, e.g., [Tichy95] [Zelkowitz98].

Development professionals, however, have a desire and are paid to produce a product using whatever technology seems appropriate for the problem at hand. The end result is that produces revenue for their employer. In industry, producing and getting a profit out of a product is most important, and the "elegance" of the process used to produce that product is less important than achieving a quality product on time as a result of the process. Being "state of the art" in industry often means doing things as well (or as poorly) as the competition, so there is considerable risk aversion in trying a new technology unless the competition is also using it.

As a consequence, researchers produce papers outlining the values of new technology, yet industry often ignores that advice since there has been no empirical justification that the technology will be effective in making their job easier. On the other hand, "gatekeepers" in industry adopt other assorted "silver bullets" proposed as solutions to the "software crisis" without any good justification that they may be effective [Brooks87]. They are used for a time by large segments of the community and then discarded when they indeed turn out not to be *the* solution. Clearly the research community is not generating results that are in tune with what industry needs to hear, and industry is making decisions without the benefit of good scientific developments. The two communities are severely out of touch with one another.

This chapter describes two classifications of techniques that have been used to support the introduction of new technologies by each of these communities. Not surprisingly, these two taxonomies differ. But there are basic similarities that are often overlooked by the two communities. The chapter describes both classifications and presents results from several studies that show how these methods are used in practice to demonstrate the effectiveness of a new method.

Section 2 of this chapter will discuss general methods for scientific experimentation, whereas Section 3 defines an experimentation model applicable for software technology. In Section 4 a model for industry validation of new technology is presented, while Section 5 presents a survey that compares the two models – research and industrial. Section 6 summarizes this work and presents our conclusions.

Experimentation

Software engineering is concerned with techniques useful for the development of effective software programs, where "effective" depends upon specific problem domains. Effective software can mean software that either is low cost, reliable, rapidly developed, safe, or has some other relevant attribute. To make the assumption that to answer the question "Is this technique effective?" requires some measurement of the relevant attribute. Saying only that a technique is "good" conveys no real information. Instead, a measurement applied to each attribute is necessary so that a judgment can be made that one technique is more or less effective than another.

For some attributes, this mapping of an effective attribute to a measurement scale is fairly straightforward. If effective for an attribute means low cost, then cost of development is such a measure. For other attributes (e.g., reliability, safety, and security), measures may be harder to derive. Measures like number of failures in using the product per day, errors found during development, or MTBF (Mean Time Between Failure) indicate reliability of a product in hardware domains. But for software, a count of the number of errors found during testing does not, by itself, indicate if there are or are not further errors remaining to be found. While safety is related to reliability, it is not the same attribute. A very unreliable program can be very safe if it can turn itself off each time the software fails. Does security mean the time it takes to penetrate the software to bypass its security protection, how many "security vulnerabilities" are present in the system, or what level of information the program is allowed to process?

In evaluating a new method, the researcher needs to know if the various attributes result in an effective measurement. Experimentation determines whether these methods results in the relevant software attributes being as effective as necessary. Should the underlying theory upon which the technique is based be modified? What predictions can be made upon future developments based upon using these techniques?

Pseudo Experimentation

Experimentation is one of those terms frequently used incorrectly in the computer science community. Papers are written that explain some new technology and then "experiments" are performed to show the technology is effective. In almost all of these cases, this means that the creator of the technology has implemented the technology and shown that it seems to work. Here, "experiment" really means an example that the technology exists or an existence proof that the technique can be employed. Very rarely does it involve any collection of data to show that the technology adheres to some underlying model or theory of software development, or that it is effective, as "effective" was defined previously, to show that application of that technology leads to a measurable improvement in some relevant attribute.

A typical example could be the design of a new programming language where the "experiment" would be the development of a compiler for the new language with sample programs compiled using this compiler. The designer may claim this language is better than others. However, the "success" for the experiment may be the demonstration that the compiler successfully compiles the sample programs, instead of providing data that shows the value or effectiveness of this new language. A confirming experiment would have demonstrated attributes proving utility of the language.

Without a confirming experiment, why should industry select a new method or tool? On what basis should another researcher enhance the language (or extend a method) and develop supporting tools? A scientific discipline requires more than to simply say, "I tried it, and I like it."

How To Experiment?

When one thinks of an "experiment," one often thinks of a roomful of subjects, each being asked to perform some task, followed by the collection of data from each subject for later analysis. However, there are four approaches toward experimentation [Adrion93]:

Scientific method. A theory to explain an observable phenomenon is developed. A given hypothesis is proposed and then alternative variations of the hypothesis are tested and data collected to verify or refute the claims of the hypothesis.

Engineering method. A solution to a hypothesis is developed and tested. Based upon the results of the test, the solution is improved, until no further improvement is required.

Empirical method. A statistical method is proposed as a means to validate a given hypothesis. Unlike the scientific method, there may not be a formal model or theory describing the hypothesis. Data is collected to statistically verify the hypothesis.

Analytical method. A formal theory is developed, and results derived from that theory could be compared with empirical observations.

The common thread of these methods is the collection of data on either the development process or the product itself.

When researchers conduct an experiment, more properly an experiment using the scientific method described above, they are interested in the effect that a method or tool, called a *factor*, has on an attribute of interest. The running of an experiment with a specific assignment to the factors is called a *treatment*. Each agent that the researchers are studying and collecting data on (e.g., programmer, team, source program module) is called a *subject* or an *experimental unit*. The goal of an experiment is to collect enough data from a sufficient number of subjects, all adhering to the same treatment, in order to obtain a statistically significant result on the attribute of concern compared to some other treatment. (For more on experimentation, see, for example [Campbell63].)

In developing an experiment to collect data on an attribute, researchers have to be concerned with several aspects of data collection [Kitchenham96]:

Replication—A researcher must be able to replicate the results of an experiment to permit other researchers to reproduce the findings. To ensure this, the researcher must not *confound* two effects. That is, the researcher must make sure that unanticipated variables are not affecting the results. If there is not a homogeneous sample of subjects for all treatments, paradoxically, this confounding effect can be counteracted by *randomizing* the factors that are not of concern.

Local control—Local control refers to the degree to which the treatment applied to each subject can be modified (e.g., the researcher usually has little control over the treatment in a case study, as defined in Section 3.1). Local control is a major problem in computer science research since many of the treatments incur significant costs or expenditures of
time. In a *blocking* experiment, the researcher assumes each subject of a treatment group comes from a homogeneous population. Thus if the researcher randomly select subjects from a population of students that represents a blocked experiment of students.

In a *factorial design* the researchers apply every possible treatment for each factor. Thus if there are three factors to evaluate, and each has 3 possible values, then they need to run 9 experiments, with subjects randomly chosen from among the blocked factors.

Experimental validity—Researchers must also be concerned with the validity of the experimental results. They want the experiment to have *internal* validity. That is, the factor being measured should indeed be the factor responsible for the effect the researchers are seeking. In addition researchers want the experiment to have *external* validity. That is the results of the experiment should be generalizable to other similar environments so that the results obtained are useful in other contexts. The process of randomizing the subjects, mentioned previously, is one way to ensure that researchers haven't accidentally included some other factor.

With software development, there are two additional aspects to consider:

Influence. In developing experiments involving large, complex, and expensive methods, such as software development, researchers need to know the impact that a given experimental design has on the results of that experiment. The authors will call this *influence* and classify the various methods as *passive* (viewing the artifacts of study as inorganic objects that can be studied with no effects on the object itself) or *active* (interacting with the artifacts under study often affecting the behavior of the objects, as in the case of the well-known "Hawthorne" effect).

Temporal properties. Data collection may be historical (e.g., archaeological) or current (e.g., monitoring a current project). Historical data will certainly be passive, but may be missing just the information needed to come to a conclusion.

3. Research Models

To understand the differences between the research and industrial communities, the authors examined *experimentation models* for computer technology research and developed a simple taxonomy to classify that research. It was possible to identify 14 methods used by researchers to develop new technology that have been used in the computer field (Table 3.1) and verified their usage by studying 612 papers appearing in three professional publications at 5-year intervals [Zelkowitz98] from 1985 through 1995. About 15% of the papers contained no validation at all and another third contained a weak ineffective form of validation (called an *assertion* in this study). The figure for other scientific disciplines was more like 10% to 15% with no validation [Zelkowitz97].

collected over time. The data that collected is derived from a specific goal for the project. A certain attribute is monitored (e.g., reliability, cost) and data is collected to measure that attribute. This is the method that is most common in studying large projects. The impact on a given project is relatively low, but the data can be significant. The downside to this method is that the effectiveness of the new technique being studied cannot be compared to other projects not using the new technique. However, if many such projects are studied over time assuming they represent a blocking experimental group, the relative value of a new technique can be determined.

Field study - Data is collected from several projects simultaneously. Typically, data are collected from each activity in order to determine the effectiveness of that activity. Often an outside group will monitor the actions of each subject group, whereas in the case study model, the subjects themselves perform the data collection activities. This has an advantage over the case study in that several projects, some using the new technique and some not using it, can be studied at the same time. However, the data is usually relatively meager, so only broad generalizations can be determined. Since there is little control over the environment for each of the given projects, the results from each study are not directly comparable. In contrast, the controlled studies, described below, offer more control over the environment to allow for more precision in the observed results.

3.2 Historical methods

Literature search - In this method, previously published studies are examined. It requires the investigator to analyze the results of papers and other documents that are publicly available (e.g., conference and journal articles). Meta- analysis, a technique where the results from several studies are

Formal methods – These methods involve using a formal model to describe a process. Ultimate validation depends upon using another validation method to determine whether the formal model agrees with reality.

Informal methods – These methods are generally ad hoc and do not provide significant results that the technique under study provides the benefits that are claimed.

The 14 validation models can be grouped according to the above 5 general areas as follows.

Observational methods

Project monitoring – In this method, the researcher collects and stores development data during project development. The available data will be whatever the project generates with no attempt to influence or redirect the development process or methods that are being used. After the project is finished, the data will be analyzed to determine if there is anything of interest. This method rarely produces significant results.

Case study - A project is monitored and data is

combined to increase the amount of data that is available, is sometimes used to allow for results to be obtained from experiments where each one individually cannot be used to specify a conclusion [Miller99]. The problem here is *selection bias*. It is not clear if the published literature represents representative use of a technology. In particular, failures in the use of a technology are rarely reported and even if reported are rarely accepted for publication. This is also called *publication bias* or the *file drawer problem*, the probability that a study reaches the literature and is thus available for combined analysis, depends on the results of that study [Scagle99]. Since positive results are more likely to be published, this can have the effect of skewing the results.

Legacy data - Data from previous projects is examined for understanding in order to apply that information on a new project under development. Available data includes all artifacts involved in the product, (e.g., the source program, specification, design, and testing documentation, as well as data collected in its development).

Lessons-learned - Qualitative data from completed projects is examined. Lessons-learned documents are often produced after a large industrial project is completed. A study of these documents often reveals qualitative aspects, which can be used to improve future developments.

Static analysis - This is similar to the above two methods, except that it centers on an examination of the structure of the developed product. Since the developed product is implemented in some programming language (whether C, C++, or HTML for a web product), it is defined by some formal syntax, which allows for an automated tool to access the source files and perform the analysis.

Controlled methods

Replicated experiment – The researcher monitors multiple versions of product. In a replicated experiment several projects are staffed to perform a task in multiple ways. Control variables are set (e.g., duration, staff level, methods used) and statistical validity can be more applied. This is the "classical" scientific experiment where similar process is altered repeatedly to see the effects of that change. However, within software development, this rarely applies due to the cost of replication. The simpler synthetic environment method is more often used.

Synthetic environment – A researcher replicates one or more factors in a laboratory setting. In software development, projects are usually large and the staffing of multiple projects (e.g., the replicated experiment) in a realistic setting is usually prohibitively expensive. For this reason, most software engineering replications are performed in a smaller artificial setting, which only approximates the environment of the larger projects. For example, multiple instances of a technique (e.g., code reading) are duplicated (e.g., using students in a classroom). This provides insights into the effectiveness of the method. But since the method is applied in isolation, the impact of this method relative to the other methods used in a project is not immediately apparent.

This form of experimentation leads to evolutionary changes in a development method since only one or two factors are under study for change at any one time. Major shifts in technology cannot be tested in this manner since the proposed changes are too extensive to be tested in isolation.

Dynamic analysis - A product is executed for certain runtime information (e.g., performance). Software is often instrumented by adding debugging or testing code in such a way that features of the product can be demonstrated and evaluated when the product is executed.

Simulation – Related to dynamic analysis is the concept of simulation where a researcher executes the product with artificial data often in a model of the real environment. The limitation of simulation is how well the model corresponds to the real environment.

Formal methods

Theoretical analysis – The researcher uses mathematical logic or some other formal theory to validate a technique. Validation consists of logical proofs derived from a specific set of axioms. This method also requires one of the 11 previous methods to be applied to show that the model that was developed agrees with reality; that the concrete realization of the abstract model is correct.

Informal methods

Assertion – This is a weak form of validation. It is usually presented as an example use of the new technology where the developer of the technology demonstrates its value, rather than to objectively assess its relevance compared to competing technologies. In the study of 612 published papers, almost one-third of the papers fell into this category.

No validation – In about 15% of the papers that studied by the authors, there was no validation at all. The authors simply explained a new technology and claimed success. Although some of these technologies were validated in later publications, the high percentage of no validation and assertion validations (almost half of the total number of papers) is disturbing. Table
briefly summarizes the strengths and weaknesses to each method.

3.6 Study results

A study based upon of 612 papers from IEEE Transactions on Software Engineering (TSE), IEEE Software magazine (Soft.) and the International Conference on Software Engineering (ICSE) for the years 1985, 1990 and 1995 [Zelkowitz98], enabled a classification of all

according to the 14 methods presented here. Of the 612 papers, 50 were considered not applicable since they were not a research contribution (e.g., a tutorial, a report about some new activity or some political or social issue affecting the software engineering community). The results of the remaining 562 papers from that study are summarized in Table 3.3.
The authors' results were consistent with those found by Tichy in his 1995 study of 400 research papers [Tichy95]. He found that over 50% of the design papers did not have any validation in them. In a more recent paper [Tichy98], he makes a strong argument that more experimentation is needed and refutes several myths deprecating the value of experimentation.

Industrial Models
While Table 3.1 defines a taxonomy for evaluating research results, a better taxonomy needs to represent the efforts used by industry in its technology adoption process. A few industrial interviews and some earlier work by Brown and Wallnau [Brown96], provided a basis for defining an industrial transition taxonomy for technology evaluation, as used by industry (Table 4.1). While the transition models include some similar to those used by researchers, many are different, a total of 15 different models used by industrial organizations to evaluate a new technology.
The major difference between the two sets of models is that the ultimate goal of the research community is to determine the effectiveness of the new technology to be tested compared to competing technologies. However, the ultimate goal of industry is to develop a product and realize revenue by delivering the product to customers. Thus achieving the best technology is not always of uppermost concern. The goal is to be good enough, [Basili02].
yet better than the competition. Cost is a factor, and industrial methods are usually skewed to those that cost less to accomplish. The 15 methods, using the same five categories of observational, historical, controlled, formal and informal methods, are defined as Project monitoring - Data is continually collected on development practices. This data can be investigated when a new technology is proposed. This is also called measurement. By building a baseline of data that describes a development environment, it provides data useful to comparing new projects. It is an important adjunct if techniques such as case studies or field studies are also employed, since it provides a yardstick that can be used to compare a project that uses the new technology with other projects developed by that organization. This is the basic method used by the NASA Goddard Space Flight Center's Software Engineering Laboratory (SEL) in developing the large body of knowledge on development practices that have characterized SEL research from 1975 through 2001**Case study** - Sample projects, typical of other developments for that organization, are developed, where some new technology is applied and the results of using that technology are observed. This is viewed as an initial experiment to see if the new method is an improvement over past practices. However, since this is a solitary project, there is no definitive test to determine what the results would have been if the new technology were not used. If multiple case studies are performed testing different technologies, this process is similar to the research case study model.

Field study - An assessment is made by observing the behavior of several other development groups over a relatively short time. There is less control over development environment, and it has similar characteristics as the field study research method.

Historical methods
Literature search - Information is obtained from professional conferences, journals, and other academic sources of information are used to make a determination of the effectiveness of a new technology. The advantage to this method is that there is less risk of misusing a poorly conceived new technology without positive experiences from others. The disadvantage is that it is not always clear that the tested environment is similar to the new industrial environment, so that the experiences may not be the same.

Legacy data - Completed projects are studied in order to find new information about the technologies to develop those projects. The technique of *data mining* is often used to see if any relationships are hidden within the data collected from a completed project in order to be able to generalize the use of this new technology [Berry00]. This is similar to the legacy data research method.

Expert opinion - Experts in other areas (e.g., other companies, academia, other projects) are queried for their expert
opinion of the probable effects of some new technology. This informal method is most similar to the lessons learned method used in the research community. This method uses individuals outside of the organization. On the other hand, the Education method (below) refers to using employees that are part of the organization.

Feature benchmark - Alternative technologies are evaluated and comparable data are collected. This is usually a "desk study" using documentation on those features present in the new technology. It is most similar to the static analysis research method where the structure of a new technology is evaluated.

Education - The quickest way to install a new technology within an organization is to train the staff to use the new technology. The advantages are that the new technology does not need to be tested, thus saving much money and that transfer of the technology to the new organization is relatively rapid. The real downside is that it is not clear if the new technology is applicable to the new organization's goals and processes. This method can be divided into two subcategories:

People - Hire the experts in a technology to help learn about it. This immediately provides the organization with the necessary expertise to use the new technology[1].

Training - Course materials to teach a new technology are given to current employees. Often it is not practical to hire new employees expert in the new technology and this method provides a larger group of individuals knowledgeable about a new technology. However, each newly trained employee cannot be considered an expert in the use of that technology.

Controlled methods
Replicated project - One or more projects duplicate another project in order to test different alternative technologies on the same application. Although is the same as the replicated project of the research methods, it is an expensive method since multiple instances of a project are developed, only one of which is necessary. It is also called a *shadow project*.

Demonstrator projects - Multiple instances of an application, with essential features deleted, are built in order to observe behavior of the new system. This has some of the same characteristics as the synthetic environment of the research method, where the new technology is tested multiple times in isolation of its interaction with other aspects of the system.

Synthetic benchmarks - A benchmark, or executing a program using a predefined data set, is often used to

[1] A comment once heard by one of the authors (source unknown): "The best technology transfer vehicles is often the moving van transporting a new PhD to his first job."

compare one product against others. If the benchmark is truly representative of the class of problems for which the product will be used, it is an effective evaluation tool. The major difficulty with benchmarks is that vendors often try and skew benchmarks to make their product appear effective and may not truly represent its use in actual production. But if the benchmark is truly representative, then it is one of the few methods for determining objective comparisons between competing products.

Pilot study - A pilot study (also called a prototype) is a sample project that uses a new technology is developed. This is generally smaller than a case study before scaling up to full deployment, but is more complete than a demonstration project. It most closely relates to the simulation research method.

Formal methods
Theoretical analysis - Much like the theoretical analysis research method, an organization can base its acceptance of a new technology on an opinion based on the validity of the mathematical model of a new technology.

Informal methods
Vendor opinion - Vendors (e.g., through trade shows, trade press, advertising, sales meetings) promote a new technology that convinces an organization to adopt it. This can be a reasonable approach, but is missing a critical analysis of alternative methods, since vendors are interested in promoting their own products. It is quite similar to the assertion research method.
desires or government rules to only use the latest or best technology. Examples include converting to object oriented design technology in order to show customers that the organization is using the latest techniques for software design.
In general, the methods used by the research community can be considered as *exploratory*, in the researchers' attempts to understand and develop new technology. Industry, on the other hand, wants methods that work, so their techniques are more *confirmatory*, showing that a given method does indeed have the desired properties. As the explanations above clearly demonstrate, there is a strong agreement between the two sets of techniques. This relationship between the exploratory research methods of Table 3.1 and the confirmatory industrial techniques of Table 4.1 is given in Table 4.2. The two industrial methods - education and external edicts or state of art- do not have research analogues.
Some strengths and weaknesses of the research models are indicated in Table 3.2. Researchers principally use the methods from Table 3.1 in order to demonstrate the value of their technological improvements, and industry selects new technology to employ by using the methods in Table 4.1. How do these communities interact? How can their methods support forward growth in computer technology and its application in real systems? A better understanding of what each community understands and values could perhaps enable identification of commonalties and gaps, and from there, mechanisms to enable each community to benefit better from the other.
External - Sometimes the need to use a new technology is not up to the organization. Outside forces can dictate the use of a new technology. For these methods, there is often little evaluation of the effectiveness of the new technology. The organization is instructed to change their methods. This can happen in one of two ways:

Edicts - Occasionally an organization is told to use a new technology. This can be upper management (e.g., corporate headquarters of a company, a government rule or regulation). The mandated use of the Ada programming language during the early 1990s and the need for an organization to be rated at the Software Engineering Institute's Software Capability Maturing Model's (CMM) level 3 in order to secure certain government contracts are examples of the use of edicts to change the technology within organizations.

State of the art - An organization often will use a new technology that is based upon purchaser or client
Valuation of the Models
To understand the different perceptions between those who develop technology and those who use technology, the authors surveyed the

software engineering community to learn their views of the effectiveness of the various models of Tables 3.1 and 4.1. This section presents the development and results of this survey. Development of the Survey

The survey was intentionally kept simple in order to increase the likelihood of a higher than average response rate from the sample population. The survey did not ask for proprietary data, which while providing useful quantitative results would have further limited the response rate. Also, by keeping the questions simple, the results is a valid instrument that allows generalizing the results to other domains readily.

These tradeoffs are sometimes referred to as "Thorngate's clock" [Thorngate76]. This is the psychological equivalent of the Heisenberg uncertainty principle in quantum physics where location and momentum cannot both be measured precisely. In this case, the choice was among accuracy, generality, and simplicity for developing the survey instrument. Selecting two meant sacrificing the third. The result was selection of simplicity and generality, and thus accuracy suffers somewhat. The results described here indicate general trends, which more in depth surveys need to address.

Survey questions are based on a previous survey [Daly97], modified for current purposes. Each survey participant was to rank the difficulty of each of the 12 experimental models (or 13 original industrial transition models) according to 7 criteria, criteria 1 and 2 being new and 3 through 7 being the same as the Daly criteria. Having ordinal values between 1 and 20 for each criterion supported objectivity of scoring. A value of 1 for a criterion is considered an exact match between it and the experimental model, 10 being the maximum effort that a given company would apply in practice for that model, and 20 an impossible condition for that model.

Survey Questions

The survey consisted of the following eight questions:

How easy is it to use this method in practice? - What is the effort in using this method? The values of 1, 10, and 20, as described above, represent relative costs for using that experimental or validation model in practice. A value of 1 indicates the method is trivial to use, a 20 means that it is impossible, and a 10 indicates it requires the maximum effort practical in an industrial setting.

What is the cost of adding one extra subject to the study? - If the researcher wants to add an additional subject (another data point) to the sample, what is the relative cost of doing so? This would increase the precision of the evaluation process by having additional experiments to study.

What is the internal validity of the method? - What is the extent to which one can draw correct causal conclusions from the study? That is, to what extent can the observed results be shown to be caused by the manipulated dependent experimental variables and not by some other unobserved factor?

What is the external validity of the method? - What is the extent to which the results of the research can be generalized to the population under study and to other settings (e.g., from
student subjects to professional programmers, from one organization to others, from classroom exercises to real projects)?

What is the ease of replication? - What is the ease with which the same experimental conditions can be replicated (internally or externally) in subsequent studies? It is assumed that the variables that can be controlled (i.e., the dependent variables) are to be given the same value.

What is the potential for theory generation? - What is the potential of the study to lead to unanticipated a priori and new causal theories explaining a phenomenon? For example, exploratory studies tend to have a high potential for theory generation.

What is the potential for theory confirmation? - What is the potential of the study to test an a priori well-defined theory and provide strong evidence to support it?

For the eighth question, each participant was asked to rank the relative importance (again using the 1-20 ranking) of each of the 7 prior questions when making a decision on using a new technology. That is, on what basis (e.g., criterion) is a decision on technology utilization made? This determines for industry the major criteria for choosing to use a new technology.

These 8 questions led to two different survey instruments —one for ranking each of the 14 research validation methods of Table
3.1 (i.e., the research survey) and one for ranking each of 13 evaluation methods of Table 4.1 (i.e., the industrial survey)[3].

Population samples

For the two survey instruments there were three random populations to sample. Sample 1 included U.S.-based authors with email addresses published in several recent software engineering conference proceedings[4]. These were mostly research professionals, with a few developers. Approximately 150 invitations to participate were sent to these individuals, and
45 accepted. The survey was not sent until the participant agreed to fill out the form, estimated to take about an hour to read and complete. About half of the individuals returned the completed form.

Sample 2 included U.S.-based authors with email addresses from several recent industry-oriented conferences. About 150 invitations to participate were sent and about 50 responded favorably to this invitation. They were then sent the industrial survey. Again, about half completed and returned the form.

Sample 3 were adult students in a graduate software engineering course at the University of Maryland taught by one of the authors. Almost all of the students were working professionals with experience ranging up to 24 years. This sample was given the research survey. Not surprisingly, the return rate of the form for this sample was high at 96% (44 of 46).

[3] Education, synthetic benchmark and expert opinion were added classifications after survey was conducted, and the technique *survey* was dropped as a redundant technique.

4 The survey was conducted via email.

It is important to realize that the responders would be giving their subjective opinions on the value of the respective validation techniques. Not everyone returning the survey had previously used all, or even any, of the listed methods. It was simply desirable to get their views on how important they thought the methods were. However, by choosing the sample populations from those writing papers for conferences or taking courses for career advancement, the author believe the sample populations are more knowledgeable, in general, about validation methods than the average software development professional. The invitations were sent early in 1998, and data was collected later that year. Table 5.1 summarizes the 3 sample populations.

Survey results

In order to understand the differences between the goals of the research community and the goals of the industrial community data were collected across all 7 criteria for all 62 who filled out one of the two research surveys.

Overall statistics

Figure 5.1 shows the average values for each technique using a "high-low-close" stock graph for this sample. The graph shows the average score for each of the 12 experimental methods over all 7 criteria as a small horizontal line. The vertical high and low "whiskers" show the confidence interval for α=0.05. The length of the whiskers and their relative positions provide an indication of the level and range of confidence in each average. The interest is in the bars, which do not overlap. These indicate a strong probability that they represent values from characteristically different sets. (The "7" in each criterion in the figure represents the midpoint (i.e., the literature search method) among the methods for ease in reading the figure.)

Practical and impractical techniques

Unfortunately, there is considerable overlap of the bars in Figure 5.1. While there are a few interesting bars (e.g., the average value of 12 for ease in performing a replicated experiment is much higher than the 4.3 value for project monitoring), almost every bar overlaps with another. Therefore it was necessary to use a weaker form of significance to get an indication of how these techniques compared. The methods for each criterion are split into three partitions: *practical*, *neutral*, and *impractical*, using the following procedure (recall that a low value indicates a more important technique):

Each method whose upper confidence interval is below the average value for all techniques was placed in the practical partition. These methods are all "better than average" according to the α=0.05 confidence criterion.

Each method whose lower confidence interval is above the average value for all methods was placed in the impractical partition. These methods are all "worse than average" according to the confidence criterion.

All other methods are in the neutral partition.

Table 5.2 presents the practical and impractical partitions. Looking at Question 1 from the survey (Ease of use), techniques that involve real projects (e.g., case study, legacy data, and project monitoring) are all considered practical techniques with respect to this criterion. Yet none of these are viewed as practical with respect to internal validity (e.g., measuring what one wants to measure) and only legacy data is viewed as practical with respect to external validity (e.g., the results can be generalized to an entire population). Not too surprising, the controlled experiments (replicated and synthetic) and the theoretical analysis were viewed as impractical techniques with respect to ease of use.

The data shown in Figure 5.1 is aggregate data. It is interesting to separate the two populations – the research workers and professional developers – that make up this aggregate. Separating the data from Figure 5.1 into the two sample populations shows how each group viewed the same criteria from a different background perspective. Tables 5.3 and 5.4 present the practical and impractical techniques for these two populations according to the same rules of significance used in Table 5.2.

In comparing Tables 5.3 and 5.4 three clear differences and one similarity between the two groups become evident. Consider first, techniques such as dynamic analysis and static analysis, which simply "exercise" the program in a laboratory setting. The research group (Table 5.3) considers such a "laboratory" validation as practical with respect to ease of use. This is not so with the industrial group. In a similar vein, replication is viewed as practical from the research community for several of the questions, but never with the industrial community.

Second consider how the two groups differed in their belief in the effectiveness of theoretical analysis with respect to internal and external validity (Questions 3 and 4). Whereas the research group considered a theoretical validation as likely to be used as much as any other technique (i.e., in the neutral partition), the industrial group considered it most difficult to use. The industrial group preferred instead the "hands on" techniques of case study and legacy data over the more formal arguments.

Finally, case study is an interesting technique (**Bold** in Tables 5.3 and 5.4) that clearly shows the different biases of the two populations. The research community considers it particularly impractical with respect to internal validity and ease of replication, two important criteria for determining repeatability of a phenomenon. Yet the industrial community considers it practical for these two criteria. The authors can only guess at why this is so, with 2 hypotheses:

The research community generally deals in formal theories and cause-effect relationships, and human subjects (the "objects" of study in a case study) are not precise. Measuring human performance is, therefore, viewed as suspect. On the other hand, the industrial community is generally wary of laboratory research results, so puts great faith in industrial experiences.

The research community has less access to developers and thus would find case study research hard to do. On the other hand, developers are well acquainted with other developers and would find such studies easier to accomplish.

More in depth studies would be needed to distinguish between these two (or any other) hypotheses.

None of the other criteria exhibited significant differences among the respondents. However, when combining the criteria into a single

composite number for each technique, other differences do become apparent, as demonstrated in the next section.
Several similarities exist within the first three questions. The strongest of these arises in the first two (Easy to use, and Additional cost) where both groups thought legacy data was useful. Furthermore, both also thought that replicated studies were impractical. This commonality provides a starting place to bridge the gap between the two groups.

A composite measure for each technique

A final eighth question of the survey was to rate the relative importance of each of the other 7 questions when making a decision on using a new technology. The purpose was to determine which of the criteria was viewed as most important when making such a decision. Figure 5.2 summarizes those answers on a single chart, with 3 columns for each question representing the three separate populations that were surveyed. (Remember that lower scores signify more important criteria.)
Figure 5.2 shows that the two samples made up mostly of industrial developers (Sample 2 and 3) agreed more closely with each other than with the research sample (Sample 1). This provides some internal validity to the rating scale for this study. Furthermore, Figure 5.2 shows that the participants in samples 2 and 3 viewed easy to do, internal validity (that the validation confirmed the effectiveness of the technique) and the ease of replicating the experiment as the most important criteria in choosing a new method. While internal validity was important, external validity was of less crucial concern. That can be interpreted as the self-interest of industry in choosing methods applicable to its own environment and of less concern if it also aided a competitor.
In contrast, for the research community, internal and external validity – the ability of the validation to demonstrate effectiveness of the technique in the experimental sample and also to be able to generalize to other samples – were the primary criteria. Confirming a theory was next, obviously influenced by the research community's orientation in developing new theoretical foundations for technology. At the other end of the scale, cost was of less concern; it rated as last.
Taken collectively, this addresses some of the problems addressed at the beginning of this chapter. The research community is more concerned with theory confirmation and validity of the experiment and less concerned about costs, whereas the industrial community is more concerned about costs and applicability in their own environment it was less concerned about general scientific results, which can aid the community at large.
One final view of the data is illustrative. A quantitative way to compare all of the validation techniques may show if any one of them was considered, in general, more important, than the others. Given the set of 7 criteria, the authors generated a composite measure for evaluating the effectiveness of the various validation methods. Since the respondents provided their impressions of the relative importance of each of the 7 criteria and the relative importance of each method for each criteria, it was easy to compute the weighted sum of all the criteria evaluations:
$\text{method}_i = \Sigma c_i * v_i$
where c_i is the average value of the i^{th} criterion and v_i is the importance of that criterion (from Figure 5.2). In this case, the lowest composite value would determine the most significant method. Table 5.5 presents these results analysis, and dynamic analysis are techniques that are easy to automate and can be handled in the laboratory. On the other hand, techniques that are labor intensive and require interacting with industrial groups (*Italics*) (e.g., replicated experiment, case study, field study, legacy data) are at the bottom of the list. This confirms anecdotal experiences of the authors over the past 25 years; working with industry on real projects certainly is harder to manage than building evaluation tools in the lab.
For the industrial community (the professional student population), almost the opposite seems true. Those techniques that can confirm a technique in the field using industry data (e.g., case study, field study, legacy data) dominate the rankings, while "artificial" environments (e.g., theoretical analysis, synthetic study) are at the bottom. Again, this seems to support the concept that industrial professionals are more concerned with effectiveness of the techniques in live situations than simply validating a concept.
The industrial group evaluating the industrial validation methods cannot be compared directly with the others two groups since the methods they evaluated were different; however, there are some interesting observations. For one, project monitoring (measurement - the continual collection of data on development practices) clearly dominates the ranking. The situation has apparently not changed much since a 1984 study conducted by the University of Maryland [Zelkowitz84]. In that earlier survey, the authors found that data was owned by individual project managers and was not available to the company as a whole in order to build corporate-wide experience bases.
This is surprising considering the difficulty the software engineering measurement community has been having in getting industry to recognize the need to measure development practices. With models like the Software Engineering Institute's Capability Maturity Model (CMM), the SEI's Personal Software Process (PSP) and Basili's Experience Factory [Basili88] promoting measurement, perhaps the word is finally getting out about the need to measure. But actual practice does not seem to agree with the desires of the professionals in the field. For example, , theoretical analysis came out fairly high in this composite score, but that does not seem to relate to experiences in the field and may be wishful thinking.
Finally, within the industrial group, the need to be state of the art (edict classification) came near the bottom of the list (11th out of 12) as not important. Basing decisions on vendor opinions was last. Yet image (being state-of-the-art) and vendors often influence the decision making process. Furthermore, vendor opinion was also judged to be least effective with respect to internal and external validity. Apparently since vendor opinion was judged to be one of the easiest to do, users rely on it even though they know the results are not to be trusted.

Discussion

Software engineering has been described as being in its alchemy stage. However, some widely recognized scientific principles are emerging. One of these is the importance of validating software-engineering techniques. This chapter identifies the experimental validation techniques in use by two communities: researchers and practitioners.

The study identified 14 research models found to be in use in the research community. These models are driven by the demands on the research community and reflect the biases and reward system of that community. Similarly, there are15 different models in use by industry. These reflect the ultimate goal of industrial software engineering, which is to solve a problem given a set of constraints. In this case, the problem is the production of a piece of software and the main constraint is funding.

It is clear from comparing these techniques that the research community is primarily focused on exploratory methods while industry focuses on confirmatory techniques. There are many similarities between the two sets of models. These provide a place to begin bridging the gap between the two communities. However, the need to better understand the relationships between models exists. Some of these relationships were explored in Section 5.

Section 5 also provides an example of how to setup and run an experiment. This chapter is in essence an example of data mining and a field study of software engineering validation techniques. The results of this experiment facilitate the understanding of the models currently used by the two communities and the connections between them. This, in turn, facilitates technology transfer, the ultimate goal.

Technology transfer is known to be a difficult process. A 1985 study by Redwine and Riddle showed that a typical software technology took up to 17 years to move from the research laboratory to general practice in industry [Redwine85]. (This time is consistent with other engineering technologies.) In fact, many new technologies do not even last 20 years! But once developed, it often takes up to 5 years for a new technology to be fully integrated into any one organization [Zelkowitz96]. Because of the short lifecycle of many critical technologies, we need to understand the transition process better in order to enable effective methods to be adopted more rapidly.

This chapter is a first step toward understanding the models in current use by the two communities. To formalize relationships between them and to better understand the universe of possible techniques, it is useful to have formalism in which to place the models. One such formalization is the three faceted approach of Shaw. Each method (an "experiment" in Shaw's terminology) fulfills three facets [Shaw01]:

Question - Why was the research done?
Strategy - How was it done?
Validation - How does one know it worked?
A case study that tries software inspections on an important new development could be classified according to this model as
Question: Feasibility - Do software inspections work?
Strategy - Technique - Apply it on a real project.
Validation - Experience - See if it has a positive effect.

On the other hand, determining whether inspections or code walkthroughs are more effective could be classified as
Question: Selection - Are inspections or walkthroughs more effective?
Strategy - Technique - Apply them on multiple real projects.
Validation - Evaluation - Use them and compare the results.

References

[Adrion93] Adrion W. R., Research methodology in software engineering, Summary of the Dagstuhl Workshop on Future Directions in Software Engineering, W. Tichy (Ed.), ACM SIGSOFT Software Engineering Notes, 18, 1, (1993).

[Basili02] Basili V., F. McGarry, R. Pajerski, M. Zelkowitz, Lessons learned from 25 years of process improvement: The rise and fall of the NASA Software Engineering Laboratory, IEEE Computer Society and ACM International Conf. on Soft. Eng., Orlando FL, May 2002.

[Basili95] Basili V., M. Zelkowitz, F. McGarry, J. Page, S. Waligora and R Pajerski, SEL's software process improvement program, IEEE Software 12, 6 (1995) 83-87.

[Berry00] Berry M. and G. Linoff, Mastering Data Mining, John Wiley & Sons, 2000.

[Brooks87] Brooks, F. No Silver Bullet: Essence and Accidents of Software Engineering, IEEE Computer (1987), 10-19.

[Brown96] Brown A. W. and K. C. Wallnau, A framework for evaluating software technology, IEEE Software, (September, 1996) 39-49.

[Campbell63] Campbell D. and J. Stanley, Experimental and quasi-experimental designs for research, Rand McNally, Chicago, (1963).

[Daly97] Daly, J., K. El Emam, and J. Miller, Multi-method research in software engineering, 1997 IEEE Workshop on Empirical Studies of Software Maintenance (WESS '97) Bari, Italy, October 3, 1997.
[Fenton94] Fenton N., S.L.Pfleeger, and R. L. Glass,Science and substance: A challenge to software engineering, IEEE Software, Vol. 11, No. 4, 1994, 86-95.

[Kitchenham96] Kitchenham B. A., Evaluating software engineering methods and tool, ACM SIGSOFT Software Engineering Notes, (January, 1996) 11-15.

[Miller99] Miller J., Can Software Engineering Experiments be safely combined?, IEEE Symposium on Software Metrics (METRICS'99), Bethesda, MD (November 1999).

[Redwine85] Redwine S. and W. Riddle, Software technology maturation, 8th IEEE/ACM International Conference on Software Engineering, London, UK, (August, 1985) 189-200.

[Scargle99] Scargle J.D., Publication Bias (The "File-Drawer Problem") in Scientific Inference, Sturrock Symposium, Stanford University, Stanford, CA, (March, 1999).

[Shaw01] Shaw M., Keynote presentation, International Conference on Software Engineering, Toronto, Canada, May 2001 (http://www.cs.cmu.edu/~shaw).

[Snow63] Snow, C.P., The two cultures and the scientific revolution, New York: Cambridge University Press, 1963.

[Thorngate76] Thorngate W., "In General" vs "It Depends": Some comments on the Gergen-Schlenker debate. Personality and Social Psychology Bulletin 2, 1976, 404-410.

[Tichy95] Tichy W. F., P. Lukowicz, L. Prechelt, and E. A. Heinz, Experimental evaluation in computer science: A quantitative study, J. of Systems and Software Vol. 28, No. 1, 1995 9-18.

[Tichy98] Tichy, W., Should computer scientists experiment more?, IEEE Computer, 31, 5, 1998, 32-40.

[Zelkowitz84] Zelkowitz M. V., Yeh R. T., Hamlet R. G., Gannon J. D., Basili V. R., Software engineering practices in the United States and Japan, IEEE Computer 17, 6 (1984) 57-66.

[Zelkowitz96] Zelkowitz M. V., Software Engineering technology infusion within NASA, IEEE Trans. on Eng. Mgmt. 43, 3 (August, 1996) 250-261.

[Zelkowitz97] Zelkowitz M. and D. Wallace, Experimental validation in software engineering, Information and Software Technology, Vol. 39, 1997, 735-743.

[Zelkowitz98] Zelkowitz M. and D. Wallace, Experimental models for validating technology, IEEE Computer, 31, 5, 1998, 23-31.