Validating the Benefit of New Software Technology

Marvin V. Zelkowitz
Information Technology Laboratory
National Institute of Standards and Technology
Gaithersburg, MD 20899
and Department of Computer Science
University of Maryland
College Park, MD 20742

Dolores Wallace
Information Technology Laboratory
National Institute of Standards and Technology
Gaithersburg, MD 20899

Abstract

The software engineering research community has developed methods for showing that new development technologies are effective and the corporate world investigates various techniques before applying them on new developments. Unfortunately, these two communities do not often work together to make technology transfer a smooth process. In this paper, methods for validating research claims in the software development domain are described and methods used by industry to investigate new technologies are given. Guidelines that allow research papers to address the needs for industrial investigations are proposed that should greatly improve the transfer of new technology to industry.

Keywords: Classification, Experimentation, Industry, Technology transfer, Validation models

1. Introduction

There is continual tension in the industrial world as worldwide competition and technological changes force organizations to rapidly adapt to new ways of conducting business. Because of these factors, the term ``software crisis,'' referring to our inability to deliver quality software on time and within budget, has been a staple of our industry for over 30 years. Companies are continually looking for technologies that will give them a competitive advantage, and research organizations and universities have been funded to develop new tools, techniques, and methods which can be applied to solve these problems.

While research has produced many approaches to deliver quality products, industry is reacting slowly to accept these new methods [9]. This problem is caused by the lack of interaction between the corporate world and the research world. Each is working on very difficult problems. But also, each side is often perceived as not understanding the needs, problems, or constraints of the other. While many issues contribute to this problem, two major ones are the following:

Corporate world. Many in industry are still looking for the ``silver bullet,'' the one technology that will give them the 10-fold improvement in productivity and quality [5]. However, no such magic is likely to be found. While many industries have adopted the essence of Deming's 14 points that address *continual* quality improvement [7], much of the software industry is still looking for the *instant* fix. Although inroads are being made in addressing incremental quality improvement within specific application domains, such as Basili's Quality Improvement Paradigm [2], an industry desperate for rapid improvement is an easy target for smooth talking vendors promising vast increases in productivity and quality, at least until the check clears.

Research world. The research community has similarly "kept its head in the sand" about the needs of industry. Research funding in the computer sciences has been relatively easy to obtain, and building a new technology is easier, and more fun, than showing that such a technology is effective. While other sciences have adopted the scientific method of theory building, hypothesis testing by experimentation, data collection, followed by analysis and theory reformulation to show that new methods are effective, this paradigm only weakly applies to the computing community. In a recent study, Tichy [10] showed that 50% of all relevant papers failed to validate the models presented in those papers. We obtained similar conclusions [12] in a study that will be explained later. It is no wonder that the corporate world does not know which technologies to apply since the research world[1] has done such a poor job of explaining its results.

This lack of interaction between these two communities is called the *methodology gap* by Bach [1]. His solution to the problem requires *heroes* in each organization who have the knowledge and insights to determine which technologies can be applied in spite of the lack of experimental validation. Unfortunately, our industry has a shortage of "heroes." Instead, we need to improve communication between these two communities in order to decrease this methodology gap. This provides the theme for this paper. We have studied the methods used to validate research results within the computing community and have developed a taxonomy of 12 experimental methods and one theoretical method for validating results on new tools, techniques and methods. If published papers more consistently validated research claims using these 13 methods, industry would have a better foundation upon which to base its technology transfer decisions when they transfer a new technology into practice.

In Section 2 we give a synopsis of this taxonomy, and we survey how industry currently decides to use a new technology, both good and bad techniques. In Section 3 we present our approach to validating process improvement within the software development domain. We will offer guidance on what research papers should contain in terms of validating the claims in those papers, and present the corporate world with an identification of the risks that apply if a given validation strategy was used to justify a new technology.

It is our hope that this paper can play a role in developing a bridge between these two communities. For the research community it provides some guidelines on what validation methods can be employed to justify using a new technology and for the

[1] In this paper, "research world" includes those who conduct research on new technology, develop new technology, experiment with new technology, or apply new technology and write about their efforts, whether they reside in academia, government, or the corporate world.

industrial community it provides guidance on what questions to ask when a new technology is proposed.

2. Validating a Technology

Validating new information technology is a recent development within the software community [8]. However, the research community and the software development community both use different approaches toward addressing this issue. We first survey these two communities and then discuss why they find it difficult to communicate among one another.

2.1 Research Validation

In an earlier paper [12], we explored 13 methods for classifying software development research. Twelve of these were experimental methods that can be used to validate research claims (11 of the 12 followed the scientific method and one was a weak form of validation) and a thirteenth method provides for a theoretical demonstration of validation. We then applied this taxonomy to classify over 600 published software engineering papers appearing between 1985 and 1995.

The classifications, briefly described, are the following:

(1) Project Monitoring. This is the normal collection and storage of data that occurs during project development. The most common data is personnel accounting data (e.g., weekly time cards) and sometimes error reporting data. Results from such studies tend to be weak.

(2) Case Study. A project is monitored and specific data collected over time in order to collect information about some specific attribute under study.

(3) Theoretical Analysis. Some papers represent a theoretical contribution whose validation consists of logical proofs derived from a specific set of axioms.

(4) Field Study. It is often desirable to compare several projects simultaneously. This is a cross between the project monitoring method, where any data is collected, and the case study, where specific data is collected. An outside group will monitor the subject groups to collect the relevant information.

(5) Literature Search. An investigator analyzes the results of previously produced papers and other documents that are publicly available and may use meta-analysis techniques to combine several such studies in order to derive new conclusions.

(6) Legacy Data. A completed project leaves a legacy of artifacts. These artifacts include the source program, specification document, design, and a test plan, as well as data collected during product development. A study of this quantitative data is a legacy data validation. Often data is lacking and the following lessons-learned study must be conducted.

(7) Lessons-learned. A lessons-learned study is a qualitative study of a project after its completion.

(8) Static Analysis. We can often obtain information by looking at the completed product and analyze the structure of the product to determine characteristics about it.

(9) Replicated Experiment. Several projects are staffed to perform a task using alternative treatments in an industrial setting. However, duplication of projects is very expensive, so the following synthetic experiment is often used instead.

(10) Synthetic Environment Experiments. This classification represents what most think of as a ``typical'' experiment. These are replications performed in a smaller artificial setting, which only approximates the environment of larger projects. These are the typical university experiments, often involving multiple instances of programmers performing some task.

(11) Dynamic Analysis. The given product is either modified or executed under carefully controlled situations in order to extract information on using the product.

(12) Simulation. We can evaluate a technology by executing the product using data based upon a model of the real environment. We predict how the real environment will react to the new technology.

We were able to place every paper that contained a proper validation into one of these 12 categories. However, there was a large group (about one-third) of papers that contained a rather weak form of validation. The authors of those studies knew that some validation was necessary, yet the validation was more a justification that the method worked, rather than an impartial evaluation comparing that method to other similar methods. We created the following *assertion* method to account for these papers:

(13) Assertion. The validation of a claim is a simple experiment favoring the proposed technology over alternatives. As skeptical scientists, we would have to view these as potentially biased since the goal is not to understand the difference between two treatments, but to show that one particular treatment (the newly developed technology) works.

Each validation from the 612 evaluated papers fits into one of the 13 categories listed above. However, some of the 612 papers we reviewed contained no validation. In order to classify every paper, we had to add the following two additional categories:

(14) Not applicable. The paper presented an idea that was not amenable to validation. For example, a report on a conference or a tutorial describing some technology generally is not expected to have any validation in it.

(15) No validation. A reader of this paper should reasonably expect a validation of the claims made within the paper, but none was present.

Figure 1 shows the distribution of the 562 evaluated papers that did or should contain validations according to 14 of the 15 validation categories. (This chart does not include the 50 papers classified as “Not applicable.”) Approximately one-third of those papers were assertions, or a weak form of validation, and another 20% had no validation of any kind. This means that over 50% (similar to Tichy's figures [10]) of all published papers

did not present good scientific evidence that the claims in the paper were valid. It is no wonder that industry does not know whether to trust the results of such studies.

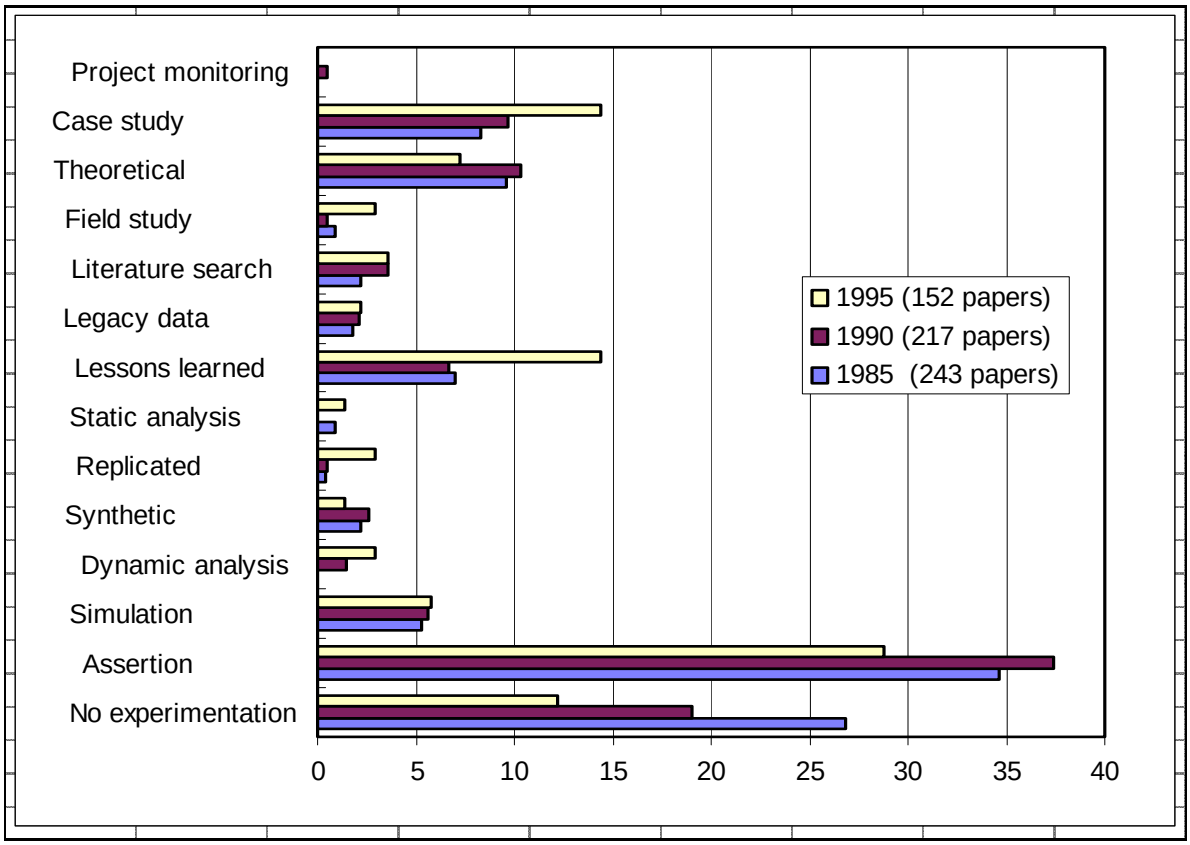

Figure 1. Use of Validation Methods in Published Papers.

The goals for the research community must be to eliminate the assertion category (13) by making those weak studies more thorough and to eliminate the no validation category (15) by adding a validation phase to the research.

2.2 Current Industrial Improvement Practices

Industry uses several investigative methods before adopting a new technology. While many industries are attuned to the costs of developing new technology (e.g., the aircraft and automobile industries spend billions of dollars in new product design and development), the need to spend up-front money for software process design is rarely recognized. It is often said that software costs too much. But costs too much compared to what? With software being such a critical component of most products today, many companies still appear to view it as an arbitrary expense, rather than viewing it as a valuable corporate asset that can give them a competitive advantage.

Therefore, when industry addresses a new technology, an overriding consideration in performing this evaluation are techniques that are low cost to evaluate. In the list of 7 practices below we identify those techniques that are most frequently used and relate them to the methods described in Section 2 (Table 1). The first two have a negligible or relatively low cost in order to evaluate a new technology. The remainder of the list are

methods involving some form of experimentation. For these we use a variant of the classification developed by Brown and Wallnau from their technology framework evaluation [6].

(1) Expert opinion. A major factor in using a new technology is the opinion of experts. This can take the form of hired consultants brought in to teach a new technology or attendance of a trade show where various vendors demonstrate their products. The major weaknesses for this method are an inability to distinguish between the true expert and others, and validation that the method proposed is applicable to your own environment. All too often time and money are spent listening and then applying this advice only to be disillusioned later when the technology turns out to be ineffective in the new environment. Expert opinion investigations play a role in industry similar to the assertion classification for research – such investigations may indicate a potential solution to a problem, but there still must be a form of experimentation to determine if the proposed solution is applicable. An important lesson that many in industry do not yet realize is that new technologies must often be tailored to work effectively within new development organizations [3]. The “one size fits all'' model is often a recipe for failure.

For larger organizations, there may be a full-time employed expert whose job is to keep abreast of new technology and suggest what techniques to apply. Within the technology transfer community, this person is often referred to as the *gatekeeper*. This can often work effectively if the expert is so viewed as one by the development organization. However, all too often advice from inside experts is often ignored because others within the company consider that an outside consultant “obviously'' knows more than a company employee, or the inside expert is not part of development project and is viewed as not really understanding the developer's problems.

(2) Edicts. Often companies resist any changes unless forced to by an outside agent. For example, use of Ada as a programming language, certification at a “level 3'' software capability evaluation, or ISO 9000 certification may be imposed as a condition to contract award, even though validation from the research community that such methods are effective in the given domain are weak. As with the expert opinion category, this is a form of assertion validation by the outside agent ordering the change.

A driving force in most of these organizations is to maintain the status quo. Managers are allowed to fail (e.g., come in late or over budget) if they are using “standard practices,'' but are penalized if they fail using a new technology. Many organizations seem content to apply the same technologies as their competitors. This defines the industry standard upon which they will be judged. It is no wonder that the tendency is to avoid trying anything new.

For companies that do evaluations, we can break them down into the following classes based on the framework developed by Brown and Wallnau [6]:

(3) Feature analysis. A feature analysis is a study of the features of the new technology and a subjective evaluation of its impact on the development process. It is the Literature search (Method 5 of Section 2) research validation method. For comparing a new technology with an older technology, previously completed projects (e.g., Legacy data (Method 6) and Lessons-learned (Method 7)) may provide a baseline that can be used to provide some estimates for using the relevant features previously.

This technique can provide insights into using the new technology. One typically chooses competing technologies and produces a chart listing all relevant features and how well each competing technology addresses that feature. Using a standard (e.g., a language standard or an interface standard) from groups like ANSI or ISO often provides a checklist of features that can be investigated in the new technologies. One then estimates the impact of the total set of features for each competing technology on the development process.

As a “paper study'' the cost of feature analysis is lower than if an actual implementation were attempted. But it does have the drawback that actual use of the technology may differ from how it is theorized that it will be used. The human factors of how easy the new technology is to use and how well it integrates into an existing development process is hard to estimate simply by looking at the features in a document. Hands-on use of the technology is generally needed for this.

(4) Compatibility studies. Such studies are used to test whether various technologies can be combined or if they interfere with one another. A company will try some new method as an adjunct to their normal development process. If several such studies are conducted, this represents the Replicated experiment (Method 9) or Synthetic environment (Method 10) validation method of Section 2.

Compatibility studies are often accomplished through *model problems*, narrowly defined problems that the technology can address. These relate to the Synthetic environment method (9) of the previous section. This method obviously has the advantage of being relatively inexpensive since the scope of the problem is rather narrow. However, its greatest risk is scalability -- can the method be used on a full-scale implementation, such as a pilot study?

A more realistic compatibility study is the *demonstrator study*, which represents a real-life scaled-up application development, perhaps with some attributes (e.g., performance, documentation) reduced in order to limit costs or development time. But there are several problems with evaluating such studies. A demonstrator study is usually validated as a Case study (Method 2). It is well known that software development projects are highly variable and it would be unclear whether a certain improvement (say 5% or 10%) is due to ease of problem, better staff, or the influences of the new technology. Using a single data point leads to uncertainty of the results. Multiple compatibility studies would be an improvement, but that ultimately greatly increases the cost of the evaluation.

(5) Pilot study. This is a full-scale implementation using the new technology and may in some instances resemble the Case study of Section 2. This method, if not carefully evaluated via an appropriate compatibility study, poses a great risk to the development organization due to the lack of knowledge of the impact of the new technology on the development process. All too often this is the next step after choosing a new technique by the expert opinion (1) or edict (2) method. If successful, it may make a significant impact on the development organization. But if a failure, an effective method may be discarded simply because it was not applied correctly and the costs of failure make a repeat of this study highly unlikely.

(6) Synthetic benchmarks. Synthetic benchmarks represent the Dynamic analysis (Method 11) and Simulation (Method 12) techniques of Section 2. Such experimentation methods are best used to evaluate products by creating an operating environment that can be tested without the risk of applying the product with large numbers of users. Its weakness is that the environment so tested may differ from the actual operating environment.

(7) Other lab techniques. Various other techniques are sometimes employed. Techniques such as the Field study (Method 4) or Static analysis (Method 8) can be used to evaluate a new technology.

In this analysis we did not refer to validation method 1 (project monitoring) or method 3 (theoretical analysis). Neither is an important component of technology transfer. Normal project monitoring generally includes accounting data keeping track of who is working on the project and when. It provides information on too coarse a level to generally be useful for the detailed analysis needed for technology evaluation. As for theoretical analysis, which can indicate which methods may work, the assumptions needed to specify any real system make it very hard to accurately apply a formal model. One still must apply one of the experimentation techniques described above before adopting that technique for general use.

INDUSTRY INVESTIGATIONS	RESEARCH VALIDATION
1. Expert opinion	Assertion
2. Edicts	Assertion
3. Feature analysis	Literature search; Legacy data; Lessons-learned
4. Compatibility Studies (Demonstrator study; Model problems)	Replicated experiment; Synthetic environment; Case study
5. Pilot study	Case study
6. Synthetic benchmarks	Dynamic analysis; Simulation
7. Other lab techniques	Field study; Static analysis
Generally not used by industry	Project monitoring; Theoretical analysis

Table 1. Similar validation practices of industry and research methods

For organizations that do a poor job of technology investigation, decisions are generally made through expert opinion or edict decisions (which we catalogue as being in the weak classification of assertion methods). Pilot studies provide a high risk method for testing the new technology if feature analysis or a compatibility study is not performed first. At the other end of the spectrum, organizations that do a good job of technology evaluation resort to multiple experiments and a continuing series of decisions, often covering three or four years, before a change is made to endorse a new technology [11].

2.3 A Culture Clash

Both the research community and the industrial community have very different expectations and constraints in which to operate. Unfortunately for both, the current reward structure does not encourage good science.

In the research world publishing early and often is a major key to success. An appropriately designed experiment can only hinder that goal. And worse, if the experiment fails, there is little left to publish – negative results are not accepted as interesting in computer science. Unlike in other disciplines (e.g., physics) where experimental validation of results is expected, no such approach is generally required in computer science.

Within industry, a major goal is to complete a project on time and within budget. Since this rarely happens, a secondary goal is to use state-of-the-practice technology. It is acceptable to fail as long as the failure is typical of other organizations. Managers are severely censured for using a new technology and then failing. There is a strong disincentive to try anything new.

Because of this aversion to try anything new, industry typically adopts the "silver bullet" approach, where some new technology is proposed to provide "orders of magnitude" improvement so that it would be obvious that the new technology is an improvement. Over the years many have been proposed, have not been appropriately validated, so have failed to live up to their promises (e.g., structured programming, the Ada language, formal methods, expert systems). Each can be effective in an appropriate setting, but none of these have revolutionized software development.

One of the few organizations not to react in this manner is the NASA Goddard Space Flight Center's Software Engineering Laboratory, where continual process improvement has been a goal for over 20 years [4]. Although improvement has only been a modest 4% a year using various technologies appropriately validated, through compounding this has become a several hundred per cent improvement over the last 20 years [4].

3. Documenting Research Contributions

We have summarized 12 methods that can be used to validate new software engineering technology (by ignoring the weak assertion category), and we have surveyed methods that are used by industry to investigate new technologies (Table 1). For an effective technology transfer model that will reduce the previously mentioned methodology gap, we want to be sure that the 12 validation methods are consistent with the industrial investigation methods. Or stated another way, what should research papers contain to make them applicable for industrial use?

As an initial response to this question, we offer the following criteria that research papers must address. If so stated in the paper, the results of that research should be more amenable to industrial practices and hasten the transfer of that technology, if effective, to industrial practices. In effect, all we are doing is applying the scientific method clearly to the software engineering domain.

1. *What is the new technology (e.g., tool, technique, standard, metric) that the paper addresses?* Surprisingly, in our 612 paper study, this was not always clear. How

does this technology relate to other technologies and why is it of value? “Different" is not the same as “useful."

2. *What validation methods (from Section 2) are used to validate the effectiveness of the technology?* Important for the scientific method is that criteria for success of the research need to be identified before performing the experiment. Simply collecting data and stating afterwards that “users seem to like it" leads to a weak assertion validation. Do you have baseline data so that you can compare your new technology to this baseline in order to validate “success?"

3. *What tailoring of the technology was done to perform your experiment?* If a pilot study, model problem, or compatibility study, how has the technology been altered due to time or cost constraints? What resources were used to conduct your experiment? It should be possible for the reader of the paper to be able to duplicate the conditions used by the researcher. It is only then that the reader can discover if the technology may be applicable to a new environment.

4. *What are the results of your experimentation?* What data was collected? How do the results compare with the criteria for success that were developed before experimentation began? What were the benefits and disadvantages in using the new technology that you found in your validation?

5. *What is needed to transfer this technology to industry?* Initial experimentation on a new technology is usually performed by experts who develop the technology and thoroughly understand it. What training is needed in other organizations to use the technology? What hardware and software resources must be procured? What risks are most apparent if this technology should be used in a larger project or in a different domain? What problems are adopters of the technology most likely to have?

4. Conclusions

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 become a general practice in industry [9]. This time was consistent with other engineering technologies. However, the generally poor validation methods used by the software research community (e.g., [10], [12]) make the technology transition problem even harder. In addition, with the rapidly changing hardware base, we simply cannot afford to wait almost 20 years for new technology to be applicable. (Remember that 17 years ago, in 1981, the desktop computer standard was the Apple 2e, and the 64K IBM PC was just announced.)

In this paper we have examined both the research world and corporate world in an attempt to provide guidance on methods researchers should employ to validate their claims. We also provide guidelines on the risks involved by industry in investigating many of these methods and the need for industry to approach technology evaluation carefully as a complex and costly process. Technology transfer is an important requirement for today's corporate world, and the need to focus on the methods to achieve this effectively should be recognized by all.

And, finally, the inclusion of information useful to industry in research papers in professional journals may accelerate technology transfer. This should both increase the professionalism of the journals themselves and lower the need for heroes and other extraordinary techniques by industry in order to get their job done.

References

[1] Bach, J., The hard road from methods to practice, *IEEE Computer*, (February, 1997) 129-130.

[2] Basili V. R. and H. D. Rombach, The TAME project: Towards improvement-oriented software environments*, IEEE Trans. on Soft. Eng.* 14, 6 (1988) 758-773.

[3] Basili V. R., G. Caldiera and G. Cantone, A Reference Architecture for the Component Factory, *ACM Trans. on Software Engineering and Methodology*, 1, 1 (1992) 53-80.

[4] 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.

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

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

[7] Deming W. E., *Out of the Crisis*, MIT Press, Cambridge, MA (1986).

[8] Kitchenham B., L. Pickard, and S. L. Pfleeger, Case studies for method and tool evaluation, *IEEE Software*, 12, 4 (1995) 52-62.

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

[10] Tichy W. F., P. Lukowicz, L. Prechelt, and E. A. Heinz, Experimental evaluation in computer science: A quantitative study*, J. of Systems and Software* 28, 1 (1995) 9-18.

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

[12] Zelkowitz M. V. and D. R. Wallace, Experimental validation in software engineering, *Information and Software Technology*, (November, 1997).