Replicating Software Engineering Experiments: Addressing the Tacit Knowledge Problem

Forrest Shull,
Fraunhofer Center – Maryland
fshull@fc-md.umd.edu

Victor Basili
Fraunhofer Center – Maryland and University of Maryland
basili@fc-md.umd.edu

Jeffrey Carver
University of Maryland
carver@cs.umd.edu

José C. Maldonado
ICMC-USP
jcmaldon@icmc.sc.usp.br

Guilherme Horta Travassos
COPPE/UFRJ
ght@cos.ufrj.br

Manoel Mendonça
UNIFACS
mgmn@unifacs.br

Sandra Fabbri
UFSCar
sfabbri@dc.ufscar.br

Abstract

Recently the awareness of the importance of replicating studies has been growing in the empirical software engineering community. The results of any one study cannot simply be extrapolated to all environments because there are many uncontrollable sources of variation between different environments.

In our work, we have reasoned that the availability of laboratory packages for experiments can encourage better replications and complementary studies. However, even with effectively specified laboratory packages, transfer of experimental know-how can still be difficult. A cooperation between Brazilian and American researchers addressing effective running of replications was formed in 1999. One of the specific issues being addressed is the problem of transferring tacit knowledge.

We discuss what we learned about the tacit knowledge transfer problem and the evolution of laboratory packages in the description of a replication performed in Brazil using a PBR laboratory package.; also how further issues will be addressed.

1. Introduction

In the past few years, there has been a growing awareness in the empirical software engineering community of the importance of replicating studies (e.g. [5], [11], [9], [2], [13], [15]). Researchers realize that the true goal of empirical research should be not the running of individual studies but developing a better understanding of software development, the cost and benefits of various techniques and at the very end consolidating a body of knowledge and establishing software development models. Too many uncontrollable sources of variation exist from one environment to another for the results of any study, no matter how well run, to be extrapolated to all possible software development environments. Most researchers accept that no one study on a technology should be considered definitive.

A result of this realization is an increased commitment to run more studies in a variety of environments. Replication in different environments is an important characteristic of any laboratory science. It is the basis for credibility and learning. Complementary, replicated studies allow researchers to combine knowledge directly or via some form of meta-analysis. Since intervening factors and threats to validity can almost never be completely ruled out of a study, complementary studies also allow more robust conclusions to be drawn when related studies can address one another's weak points. In software engineering, this enables us to build a body of knowledge about families of related techniques and basic principles of software development.

For clarity a brief working definition of a replication will be given here. While in many contexts the term *replication* implies repeating a study without making any changes, this definition is too narrow for our purposes. In this work we will consider a replication to be a study that is run, based on the design and results of a previous study, whose goal is to either verify or broaden the applicability of the results of the initial study. For example, the type of replication where the same exact study is run could be used to verify results of an original study. On the other hand, if a researcher wished to explore the applicability of the results in a different context, then the design of the original study may be slightly modified but still considered a replication.

In our own work, we have reasoned that better replications and complementary studies can be encouraged by the availability of laboratory packages that document an experiment. A **laboratory package** describes the experiment in specific terms and provides materials for replication, highlights opportunities for variation, and builds a context for combining results of different types of experimental treatments. Laboratory packages build an experimental infrastructure for supporting future replications. They establish a basis for confirming/denying original results, complementing the original experiment, and tailoring the object of study to the specific experimental context.

However, despite our high hopes, our experience has shown that replication is difficult and lab packages are not the solution by themselves. Even when both the *original experimenters* and the *replicating researchers* are experienced experimentalists, there are so many sources of variation and implicit assumptions about the experimental context that composing a static lab package to describe all relevant aspects of the experiment, in such a way that unexpected sources of variation are not introduced into the replication, is nearly impossible. As examples, consider the following cases from our own experience:

- A study of a software review technique unintentionally introduced a source of variation when a limit was placed on the time available for performing the review. (In the original experiment, the time was open-ended.) The results were very different (and incomparable) between the two studies because subjects in the replication altered their behavior to try to prioritize aspects of the review and could not check their work, while subjects in the original study were allowed to work under more realistic conditions.
- Another study of software review techniques introduced a possible variation in results when the wrong time estimate was given to reviewers. When the review took much longer than the reviewers had expected, they reported feeling frustrated and de-motivated with the technique and quite possibly reviewed less effectively as a result.

The point of these examples is not to imply that either the original or replicating researchers were somehow deficient in their preparation[1], but to illustrate the major changes to experimental results that can occur from seemingly miniscule changes to experimental conditions. Thus the issue of *process conformance* between seemingly identical experiments becomes a real issue that may affect the level of confidence in results.

In this paper, we review our experiences in experimental replication leading us to identify an important *tacit knowledge* problem affecting the process conformance and hence the comparability of results between replications. (Here, tacit knowledge refers to information that is important to the experiment but is not coded into the lab package. It lies in the head of the original experimenters and has a difficult time being made explicit, for a variety of reasons.) As argued in this paper, we now believe that effective replications require not only lab packages but also a *process* of replication involving the original researchers to support their instantiation, evolution, and use. The need for such a process implies an associated need for an effective *collaboration structure* between the original and replicating researchers to convey the tacit knowledge that is important for process conformance.

Section 2 provides a brief introduction to the Readers' Project, a bilateral project supported by the Brazilian and American national science agencies that is investigating replications and tacit knowledge issues. Section 3 describes the tacit knowledge problem more fully, giving examples to help demonstrate the issues and dangers involved. Section 4 presents a brief overview of a particular replication performed by the Readers' Project that has been designed to address these issues. Special attention is paid to how the tacit knowledge problems instantiated themselves in this replication, despite the existence of a relatively complete lab package for supporting such replications. The contributions of this replication toward a better understanding of tacit knowledge issues and the development of an experimental process for avoiding them are described in Section 5. Finally, Section 6 ends the paper with a summary of the points raised and a set of general recommendations for combating the problem.

2. Introduction to the Readers' Project and Its Goals

To address issues of running effective replications, a cooperation between Brazilian and American researchers, named *"Readers: A Collaborative Research to Develop, Validate and Package Reading Techniques for Software Defect Detection,"* was initiated in 1999. This cooperation, supported by the Brazilian (CNPq) and American (NSF) national science agencies, has the general purpose of investigating techniques for analyzing software documents in diverse cultural settings. The long-term objective in this research line is to tailor these "analysis techniques" to accomplish a specific software-related task (defect detection) using specific kinds of software documents (requirements, specification, and code documents) in specific environments or cultural settings (Brazilian and American industries and academia) based on empirical evaluation. The members of the project contribute crucial experience in the areas of both

[1] To avoid this danger we have deliberately avoided giving names or references.

developing and refining software technologies, and in empirically evaluating technology effectiveness.

The high level goals of the Readers' Project are to facilitate the running of more and better replications to support more useful and robust conclusions concerning the effectiveness of software development technologies. To achieve these goals, based on our experiences on the project to date, we have identified several sub-goals that will be necessary to overcome the practical difficulties, and are experimenting with an organizational structure to serve as a template for facilitating effective replications.

2.1 Research Goals

Running more studies does not simply require more researchers interested in empirical studies. Empirical studies are complex and very difficult to run in isolation without a community of like-minded researchers to give feedback and suggest new directions. Particular issues to be addressed to facilitate the running of more studies include the following:

- **Mechanisms for creating a community of researchers.** To be effective, empirical work in software engineering requires not only more empirical researchers, who can run studies to evaluate development technologies, but also researchers in other technical areas who are willing to undertake empirical studies of the technologies they develop. Accomplishing this has a non-technical aspect, in that empirical researchers need to better articulate and "advertise" the values of collaboration and how empirical results can feed into and improve the development of other technologies. But there is also a technical side, in that this diverse and possibly cross-cultural community needs to be supported in terms of developing a common understanding of the problems, a common jargon for talking about the issues, and appropriate tools for supporting collaboration.
- **Fostering of specific collaborations.** We are trying to help define an optimal method for collaborations to occur between replicators and an experiment's original creators that takes the needs of both parties into account. We need more experience with the impact of different collaboration types; for example, a project on a specific topic such as the Readers' Project may feedback or may provide focused information to a more general network such as the International Software Engineering Research Network, whose members are independent researchers with no common funding source, but with common interests, a common terminology and a common set of research issues.
- **Transfer of experimentation know-how.** To build up a community of researchers capable of producing more good-quality studies, new ways of transferring the requisite know-how must be found. If this cannot be done, there is the threat that studies will be run with undetermined flaws, or that other sources of variation will be introduced that could affect the results. In short, enough knowledge has to be transferred that replicated experiments can be run with a certain minimum amount of process conformance, where variation in results can be traced with confidence not to variations among experimenters but variations among important causal factors.

Running better studies does not require simply fine-tuning existing techniques that can improve any study of a given type. Rather, more sophisticated techniques are needed and their relevance for different types of situations must be better understood. And, since it is probably impossible to run a "perfect" study that removes every conceivable threat to validity, we must learn new techniques of combining studies so that more robust results may be achieved across several empirical sources. Several issues must be addressed before it is possible to do the more detailed analysis that is necessary to achieve this larger goal:

- **Techniques for abstracting higher-level conclusions about a technology.** As indicated above, drawing "better" conclusions in the sense that they are more robust, justifiably inspire more confidence, or address useful levels of generality requires conclusions to be drawn from across families of studies, rather than individual ones. One area of research we will explore is how to compose an effective family of studies: that is, given the weaknesses of existing studies, which new studies would be most optimal?
- **Guidelines for packaging the experimental artifacts.** An important tool for supporting replications is the laboratory package. Not only do lab packages have a role to play in supporting more replications (the presence of easily-available designs and materials can facilitate replications by reducing the amount of effort required of other researchers) but *well-designed* packages are crucial for facilitating better, uniform replications. We will investigate important open questions in the area of creating effective lab packages, such as:
  - Defining what the package should contain.
  - Defining quality factors & criteria for evaluating packages.
- **Procedures and tools for evolving lab packages.** A major difficulty in running replications is artifact evolution. Given the large numbers of documents that comprise any lab package and the interrelationships between those documents, maintaining consistency over time between experiments and the comparability of results imposes a lot of overhead effort. (For example, if an inspection experiment uses a

requirements document with seeded defects, that document can evolve but has always to be matched up with the correct list of seeded defects, the correct version of the system design, and the appropriate results that were calculated based on a particular version of the defect list.) Finding ways to maintain these complex relationships between different versions of documents will help decrease the effort required of the researcher and increase the level of confidence in results.

- **Understanding and defining the knowledge transfer process.** Effective replications also require more research into the knowledge transfer process itself in order to provide more organizational support. For example, what kinds of collaboration mechanisms are necessary and sufficient for the information about a single experiment to be transferred? What are the main steps in running an experiment and what information is necessary at each step?

2.2 Collaboration Structure

The Readers' Project represents a close collaboration between researchers, in which replicators work directly with the original experimenters to understand a technology, plan an evaluative study, and analyze the results.

The primary mechanism for information exchange is a planned series of bi-yearly meetings. Although the principles are in close contact via email, we have found this arrangement to be insufficient for the needs of undertaking replications. Email correspondence is sufficient for answers to focused questions but does not facilitate both parties grasping the overview of the experiment, which is necessary precisely because of the complexity and interrelated nature of experimental design. It has been our experience that building a shared understanding of an experimental replication requires a workshop environment that is exploratory, not confirmatory, in nature. That is, there is a need for a wide-ranging discussion to permit detailed descriptions of the experimental design details as necessary and to correct misconceptions immediately.

It is this last point – misconceptions caused by bad assumptions about experiment design – that seems to make replication so difficult. Focused working groups, physically co-located with all relevant experimental materials on-hand, seem best suited to combating this tendency. Meetings support discovery of the many inter-related facets of an experiment in a way that shorter, focused communication exchanges do not. But, it is not always feasible to have a close collaboration, so defining the experimentation process and assessing process conformance during the replication become mandatory activities to ensure successful results. Defining the experimentation process aims at making clearer and explicit the tacit knowledge - the knowledge and assumptions underlying the techniques and the related Laboratory Package - as discussed in the next section.

3. The Tacit Knowledge Problem

Even when effectively specified packages exist, researchers face difficulty understanding and reviewing the available laboratory packages to select an appropriate one. The main difficulty (found also on the replication described later in this paper) is to understand the concepts underlying the techniques under study and to master the knowledge involved in running the experiment. This problem can be thought of as the difficulty in transferring experimental know-how between the original experimenters – the knowledge providers – and the replicators – the knowledge users. Two kinds of knowledge must be transferred in this process: tacit knowledge and explicit knowledge. The explicit knowledge refers to what is written down somewhere in the laboratory package. The replicators can absorb this kind of knowledge by reading and using the lab package. The tacit knowledge refers to information that is important to the experiment but is not coded into the lab package.

While the goal of the original experimenter is to make as much tacit knowledge explicit as possible, it is important to observe that some tacit knowledge is not coded because the experimenter does not anticipate its relevance to other researchers, or simply because it seems such a minor point that he or she did not think it an important enough component of the overall experimental design to describe. For example, consider the two replications described in Section 1. Another point, just to illustrate, is the process to decide on updating the list of defects related to a specific artifact used in an experiment. Is the list of defects to be updated? If so, how? Who would be the people involved? However, there are other kinds of tacit knowledge that are too difficult to code or are not supposed to be coded at all. This includes tips, simple procedures and decisions that are too detailed to write into a lab package. For capturing this kind of information, it is very important for the replicators to interact with researchers that have run the experiment before.

Absorbing tacit knowledge has been a major source of difficulty during the replications done so far in the Readers' Project. The lack of tacit knowledge can affect the comparability of a replication's results to the original experiment, since when different and independent researchers carry out replications, there is danger that experimental steps (such as training) are executed in a slightly different way that introduce a new source of variability. Systematic procedures must be stated and followed for the creation and use of lab packages. These procedures must establish, step by step, all the tasks that one must accomplish, its timing and deliverables. Even so, it is our opinion that interaction with the group that executed the original experiment is fundamental to ensure the transference of tacit knowledge, knowledge that is not – and could not be – written anywhere in the lab package.

4. Experiences with a Family of Empirical Studies

To enable the investigation of the desired research questions on replication, a specific technology was necessary to serve as a testbed of replication issues, over a series of specific evaluative studies. Based on the background of the principles we elected to work in the area of software defect reduction, or more specifically, review and testing technologies.

Software review and testing are validation activities that have been used to achieve high quality software. Software review and testing techniques aim at detecting faults/defects in the software products at different phases of the software development process. Although many empirical studies have been carried out providing evidence of the effectiveness of these activities (e.g. on software inspections: [14], [1], [6], [10], [15]), many open questions still remain to be answered and investigated:

- Are reviews and testing complementary? If so, in what ways can they be used together?
- What is the effectiveness of a particular technique?
- Which technique is more effective and efficient within a particular context? In other words, which technique reveals more faults/defects?
- Do the cultural and environmental aspects influence the results obtained by the underlying techniques?

All these questions should be investigated by carrying out well-planned, replicable empirical studies. Thus, the establishment of lab packages that support the replication of experiments leading to the creation of a database to support a broader analysis and conclusive results constitutes a relevant task and contribution to software engineering researchers.

The experiment described in this paper belongs to a family of experiments being undertaken by the Readers' Project in this area, aimed at replicating previous experiments on "reading techniques" for human-based review of software requirements for finding defects [1].

This experiment addresses the following issues related to our high-level goals:

- Transfer of know-how. The core technology being evaluated, the PBR reading techniques, has to be taught to subjects in a new environment before they are adequately prepared to apply it and can provide good data. This study afforded us a chance to explore how to bring subjects in a different environment up to speed in a new technology.
- Packaging the experimental artifacts. A lab package already exists for supporting a controlled study to evaluate PBR. This first replication provided a chance to evaluate whether the package contained sufficient materials and was organized appropriately to transfer the necessary information. As work is currently ongoing at both locations, we also have had an opportunity to jointly examine how to package and further evolve the required materials.

This replicated experiment also served as a way to test our mode of collaboration, which we identified as another problem to be investigated in general for supporting replications. As we attempt to explore these research areas we are examining ourselves to see what complications and difficulties we face and whether these might be resolved with another mode of collaboration.

4.1 Object of study

The technology being evaluated in this study is Perspective Based Reading (PBR), a technique developed by the Experimental Software Engineering Group at the University of Maryland – one of the current partners of the Readers' Project. PBR works to improve the effectiveness of software requirements inspections by providing inspectors with procedural guidance for finding defects during their individual review of a requirements document. PBR requires the reader to first create an abstraction of the product, and then to answer questions based on the analysis of this abstraction from a particular perspective that he assumes. The readers are asked to assume the perspective of some stakeholder of the requirements document while they do the review; a "basic set" of perspectives including a software designer (D), a tester (T), and an end-user (U) has been identified and was used for this review. For each perspective, the readers address a set of questions that allow them to discover the defects in the software artifacts (e.g., the questions for the tester perspective lead the reader to discover those requirement defects that would be found by testing the final product.). The set of questions are driven by the taxonomy of requirements defects.

4.2 Overview of the Design

The original experiment took place at the University of Maryland in the US, with professional software developers from NASA's Goddard Space Flight Center. The replication occurred in Brazil at the University of São Paulo with undergraduate students who were relatively inexperienced in software development (slightly more than one year of experience on average).

The basic design of the study was a within-subjects comparison, in which subjects first used a checklist approach to review a requirements document, were trained in PBR, and then applied PBR to review a different document.

Two different requirements documents were used, both containing seeded defects. The order of the documents was assigned randomly to subjects, so that half of the subjects performed the checklist review on Document A and the PBR review on Document B, while the other half did the Checklist on Document B and PBR on Document A. The effectiveness of each review was measured as the percentage of the seeded defects uncovered during the review.

Thus for each subject, the effectiveness during each of the two reviews could be compared to discover whether there was any net improvement due to PBR.

4.3 Running a Replication as Part of the Series

The first aspect in replicating the PBR experiment was for the replicating researchers to gain access to the laboratory package with its associated artifacts. That was an easy task due to the framework of the Readers' Project, however one should keep in mind that this framework is not always present. This process may involve negotiations regarding artifacts and results ownership between the packages providers and the replicators.

A difficulty that we had was to assemble a complete and consistent lab package for the replication. The original lab package consisted of several artifacts that due to several replications evolved over time. The identification of compatible and/or consistent artifacts was not an easy task. This was due to the fact that the artifact repository at the University of Maryland was kept in a simple file system structure. We had to identify, for example, what was the best list of defects for the most current requirements specifications. This is a version control and configuration problem that has arisen from the growth of the number of artifacts available in the UMD experience base. It indicates that we need more sophisticated tools to keep track of experimental artifacts, which in a sense is a good sign. It shows that we have a growing body of knowledge in software engineering experimentation. These issues are being addressed by the CeBASE project [8], which is building a large body of knowledge on empirical software engineering and is using knowledge management information systems to assemble it into experience bases.

After gathering all the artifacts, the replicating researchers still had to adapt some of them. We had to include, for example, questions regarding English language expertise in the Subject Characterization Questionnaire. This situation always happens when one has variations between the original and the replicated experimental settings. In this case, the replication had different cultural settings – Brazilians have different levels of expertise on the artifact language (English), a problem that was not present in the original US experiments. Other types of variations may appear in experimental replications of this kind, for example, variations between replications done in academic and industrial settings.

Because of the level of effort involved in running even a replicated experiment, the replicators were not willing to undertake a full experiment without testing the material and concepts in their own environment. As a result, they decided to run a pilot study to better understand how the experimental process should be conducted, paying attention especially to experimental mechanics such as the timing issues, tasks to be executed and documents to be delivered to the subjects. Although the pilot study also required an investment of effort and resources, it was considered appropriate and necessary since this was the first time this experiment was undertaken outside the Maryland umbrella and it seemed to afford the best opportunity to identify and master the tacit knowledge issues. The pilot study was intended to ensure the quality and conformance of the experimental process.

The pilot study was in fact very useful for identifying tactic knowledge issues. For example, the replicators could not really understand why the same amount of training time was specified for both the checklist and PBR approaches, since they provide different levels of detail and require different levels of background knowledge. For each perspective of PBR the operational aspects have to be taught in addition to the understanding of the questions and of the requirements defect taxonomy. The replicators made some adjustments to the training time but kept it equal for both techniques. It is important to point out that this aspect should be seen as a threat to validity, because it probably was not fair to use equal training time for two techniques of varying complexity. This point should be addressed in further experiments.

Based on our experience, we recommend including in any replication process a pilot study aiming at mastering the underlying concepts and tacit knowledge and also providing a mechanism to assess the process conformance before any significant replication effort is undertaken.

One additional important note on the replication in Brazil is that the subjects found several new defects in the seeded documents in addition to those already documented in previous experiments. Thus, in addition to the existing version control problem, there was an update to the defect lists occurring as the project was running. This update required a decision as to whether to re-analyze previous experimental data to ensure consistency among all results.

4.4 Results of the Replication

The specific, technology-related results of this replication included:

- PBR reviews were more efficient for both documents;
- PBR reviews were more effective for one document and as effective as checklist reviews for the other.

Although this points to benefits of PBR, an interesting point to be further investigated is the complementary aspect of the two inspection approaches, since both approaches found defects that the other did not. That is, neither approach could be said to have found a superset of the other's defects.

Another important result concerns the uniformity of the results provided by each approach. One goal for software process improvement activities is to make effective software development results more repeatable and less dependent on the characteristics of individual developers. Neither Checklist nor PBR led to complete uniformity of reporting defects, but PBR had a higher percentage of subjects achieving the same highest performance (within each perspective).

It is important to point out that to analyze the results of such experiment it is not an easy task. The number of ways to analyze the information from a typical experiment is really fantastic, and it is often the case that simply testing the experimental hypotheses neglects a significant percentage of the data collected. For example, once a technology has been evaluated by comparison to another technology, it is necessary to evaluate the data from different points of view to better understand the reasons for the success or failure of the technology. For instance, given a specific defect, who are the subjects that found it? Given a specific perspective, which defects were more likely found by using it rather than another? For a specific class of defects, which subjects did better? For this reason we recommend that hypothesis testing be complemented by a discovery-driven approach, with the aim of hypothesis generation for guiding further studies. This process, comparable to data mining over the data collected by an experiment, is again difficult to do in isolation and benefits greatly from a close collaboration where hypotheses can be brainstormed, debated, and the data quickly checked.

5. Contributions to project goals

To facilitate the replications, it becomes necessary to transfer knowledge from the original experimenters to the replicators. This knowledge transfer task is not as simple and straightforward as one might expect. This section discusses some of the issues with tacit knowledge and formalizing an experiment into a lab package.

5.1 Communicating Tacit Knowledge

In order to minimize the tacit knowledge problem, the procedures for experimental replication were detailed as much as possible determining timing and documents to be delivered at each step of the experimental process for both training and execution phases, based on the Pilot Study. Thus, to the original lab package were added new documents like the one presented in Figure 1. This figure describes some of the steps that compose the PBR replication. Each major activity (included both in training and process execution) for the experimenters was represented as a separate step, and has the timing associated with it. Each step is described in terms of the following information:

- Tasks to be executed, such as teach the topic *Reading Techniques Theory*;
- Time constraints, broken down by *Total Time* (amount of time required for the entire step) and *Subject Time* (amount of time required for independent work by subjects).

A similar process was also defined for the Checklist technique. In our replication the Checklist is the first technique to be applied, additionally its process also includes the timing required to fill out the initial forms of the lab package. The *Analyst Survey* is an important document that characterizes the subjects involved in the replication.

The tacit knowledge problem exists in part because it is difficult for the original experimenter to know what the replicators are going to need. This is one positive outcome of doing the replications. That is, we begin to understand what kinds of tacit knowledge need to be made explicit to allow for a successful replication.

Based on our experience, we have found that a more effective way of communicating tacit knowledge about an experiment is to include more than just a description of what is included in the experiment. By also including a list of issues that are invalid or outside the scope, along with rationales for these decisions, the experiment can be delimited more clearly. For example, another of the major contributions of this PBR replication was the creation of a list of "frequent false positives," that is, a list of issues in the documents inspected that were frequently reported by subjects as defects but not accepted by the experimenters[2]. For each item on the list, a rationale is also included as to why this issue should not be treated as a real defect. This new artifact helps experimenters to reduce the data analysis effort and helps to recognize when new issues reported by the subjects can be removed from the defect analysis and when they signify that evolution of the lab package is indeed necessary.

[2] The false positives are those defects reported by the readers that are not real defects; they are, normally, outside of the scope of the requirements documents. Because of the lack of standards and the degree of variation in scope of requirements documents in industry, it is a difficult task to define which issues are real defects and which are not.

Step 0

Perform training on Reading Techniques Theory (Topics covered include:

- Perspective-based Reading
- Design-based perspective
- Test-based perspective
- Use-based perspective)

Total time: 90 minutes
Subject time: 0 minutes

Step 1

a) Hand out the review document:
- Example Requirement Specification – ABC Video or Gas Station Control System

b) Hand out and explain the data collection documents:
- Defect Classification Scheme (Form E2)
- Execution Form (Form E4)
- Defect Report Form (Form E5)
- Reading Scenario for Designer
- Reading Scenario for Tester
- Reading Scenario for User

Total time: 15 minutes
Subject time: 0 minutes

Step 2 …

Figure 1 – Excerpts of experimental process dynamics, step descriptions – including pointers to other documents in the lab package that contain more details for each step.

5.2 Evolving the Lab Package

We believe that requirements documents for experimentation should ideally have two versions, an oracle version and a golden version. The golden version should be a correct or near correct version. Its usefulness would be two-fold: (1) to help build an oracle version and an error list through error seeding; and, (2) to ease the process of analyzing the errors reported during the experiment. The oracle version is the version used in the experiments that has been seeded with defects. We also believe there should exist a body of experimenters to decide on the evolution of the oracle version and its list of defects and false positives. This body of experimenters should be constituted of experienced experimenters [3], [4], [12].

As a concrete result of this replication, the Laboratory Package, which should serve as a baseline for further experimental replications, the training material in particular, has been evolved based on the input from the replications of the PBR experiment carried out in Brazil. Other changes include the definition of an explicit process for Defect Analysis, the elaboration of a list of defects and another one of frequent false positives, and the evolution of the Feedback Questionnaire to capture information on the cultural aspects of the subjects. In this questionnaire, we also included a defect assertion session so that the subjects can discuss whether the current list of defects is reasonable or not. This important information impacts the global analysis of the experiments, and will facilitate future efforts to analyze data across different replications.

5.3 Package Quality Goals

Based on this understanding of potential users, we have also outlined a series of goals that an effective lab package should be able to support, aiming at contributing to the definition of standard documents and procedures for an Experience Library in the perspective of Conradi *et al.´s* work [7]. These quality goals can be used as an informal evaluation checklist to assess the design of specific lab packages. As we continue to work with lab packages, we are trying to find useful metrics that can more accurately assess packages. The quality of a lab package can be assessed by how well it supports the following goals, many of which are involved with tacit knowledge transfer:

- Access to experimental artifacts

- Adequate and complete training materials (Complete materials include aspects such as the necessary background to comprehend the technology and take language issues into account.)
- Accurate estimates for the time required for subjects to execute the experiment
- Presence of oracles and golden artifacts (materials against which the quality of the systems being developed using the technology can be measured)
- Ease of package evolution and experimental feedback
- Complete descriptions of:
    - The analysis and goal of the experiment (so that other researchers can evaluate the feasibility of the experiment for their environment)
    - The experimental design, including threats to validity and strengths of experiment (so that other researchers can make changes to the design), along with the decision history (justifying decisions and reflecting on what works as well as what does not)
    - The context(s) in which the experiment was run
    - The process to run the experiment

6. Conclusions and Next Steps

Although the replication of experiments is a necessary component of research in software engineering and crucial to our ability as a field to understand fundamental principles of software development, running effective replications is not straightforward. We have discussed one important reason why this is so (the tacit knowledge problem) and argued that, due to the sensitivity of experimental designs involving human subjects, where small variations in the execution of the experiment can have large effects on results, stand-alone lab packages cannot be the entire answer to facilitating replications. We provided several examples of such difficulties in replicated experiments, both within the Readers' Project and without, to further motivate this argument. We wish to again emphasize that these difficulties are not the fault of inattentive researchers, but have occurred in spite of having experienced researchers as both the original experimenters and replicators, extensive preparation effort on both sides, mature materials, and lab packages explicitly designed to promote replication.

We argue that lab packages must be augmented by a *replication process* and attention to *process conformance* within the experiment. In this paper, we touched upon the running of pilot studies, which was a successful technique for addressing such problems by providing a dry run of the entire experimental process "in the small". Although pilot studies require a significant investment of effort, this amount is small compared to the effort that would be wasted running an invalid experiment. We also concluded that the training of replicating researchers is a crucial aspect of ensuring process conformance between experiments. In this paper we mentioned that this training can be facilitated by modularizing the training materials available, breaking the materials down by topic and providing detailed timing estimates to convey information about the relative importance of each topic. In further work we will be investigating the use of other media, such as capturing previous training sessions using audio or video recordings for use by replicators.

7. References

[1] V.R. Basili, S. Green , O. Laitenberger, F. Lanubile, F. Shull, S. Soerumgaard, and M.V. Zelkowitz, "The Empirical Investigation of Perspective-Based Reading", *Empirical Software Engineering*, 1(2): 1996, pp. 133-164.

[2] V.R. Basili, F. Shull, and F. Lanubile, "Building Knowledge through Families of Experiments", *IEEE Trans. Software Engineering*, 25(4), July/August 1999, pp. 456-473.

[3] V.R.Basili, M. Lindvall, and P. Costa, "Implementing the Experience Factory Concepts as a Set of Experience Bases", In Proceedings of *The 13th International Conference on Software Engineering & Knowledge Engineering*, 2000, pp.102-109.

[4] V.R. Basili, R. Tesoriero, P. Costa, M. Lindvall, I. Rus, F. Shull and M.V. Zelkowitz, "Building an Experience Base for Software Engineering: A Report on the First CeBASE Workshop", *Profes (Product Focused Software Process Improvement)*, 2001, pp. 110-125.

[5] A. Brooks, J. Daly, J. Miller, M. Roper, and M. Wood, "Replication of experimental results in software engineering", *ISERN Technical Report ISERN-96-10*.

[6] C. Ciolkowski, C. Differding, O. Laitenberger, and J. Muench, "Empirical Investigation of Perspective-based Reading: A Replicated Experiment", International Software Engineering Research Network, Technical Report ISERN-97-13, 1997. http://www.iese.fhg.de/ISERN/technical_reports/isern-97-13.pdf

[7] R. Conradi, V.R. Basili, J. Carver, F. Shull,, and G.H. Travassos, "A Pragmatic Documents Standard for an Experience Library: Roles, Documents, Contents and Structure", University of Maryland Technical Report CS-TR-4235. April 2001.

[8] G. Goth, "New Center Will Help Software Development 'Grow Up'", *IEEE Software*, May/June 2001, pp. 99-102.

[9] P. Johnson. "BRIE: The Benchmark Inspection Experiment", http://csdl.ics.hawaii.edu/techreports/96-13/96-13.html.

[10] O. Laitenberger, K. El Emam, and T. Harbich, "An Internally Replicated Quasi-Experimental Comparison of Checklist and Perspective-based Reading of Code Documents", *IEEE Transactions on Software Engineering*, 27(5), 2001, pp. 387-421.

[11] C. Lott, and D. Rombach, “Repeatable software engineering experiments for comparing defect-detection techniques”, *Empirical Software Engineering*, 1(3), 1996, pp. 241-277.

[12] M.G. Mendonça Neto, C. Seaman, V.R. Basili, and Y.M. Kim, “A Prototype Experience Management System for a Software Consulting Organization”, *Knowledge Systems Institute, Thirteenth International Conference on Software Engineering and Knowledge Engineering*, 2001. pp. 29-36.

[13] J. Miller, “Applying meta-analytical procedures to software engineering experiments”, *Journal of Systems and Software,* 54, 2000, pp. 29-39.

[14] A. Porter, L. Votta, V.R. Basili, “Comparing detection methods for software requirements inspections: a replicated experiment”, *IEEE Trans. Software Engineering* 21(6), 1995, pp. 563-575.

[15] F. Shull, J. Carver, and G.H. Travassos, “An Empirical Methodology for Introducing Software Processes” In *Proceedings of the 8th ESEC*, 2001, Vienna, September, 2001, pp. 288-296.