Investigating the Interaction Between Inspection Process Specificity and Software Development Experience

Jeffrey Carver[2]
carver@cs.umd.edu

John Van Voorhis[2]
jvv@cs.umd.edu

Victor Basili[1,2,3]
basili@cs.umd.edu

[1]University of Maryland
Institute for Advanced
Computer Studies
University of Maryland
College Park, MD 20742
USA

[2]Department of Computer
Science
University of Maryland
A.V. Williams Building
College Park, MD 20742
USA

[3] Fraunhofer Center - Maryland
University of Maryland
4321 Hartwick Road
Suite 500
College Park, MD 20742
USA

ABSTRACT

This paper describes a study conducted to compare the interaction of experience and specificity in a requirements inspection technique. Two versions of a requirements inspection technique, PBR, were generated. One version had a high level of specificity and the other had a low level of specify. These techniques were used by subjects of varying experience levels to determine if experience and specificity were related. The results of the study indicated very little difference among the treatment groups. As a result, we examined any assumptions that we made about the environment. In doing so, we uncovered some issues that must be addressed in future studies that focus on people. This paper provides a complete description of the results obtained and describes the assumptions that we made and their impact on the reliability of the results.

1. INTRODUCTION

Software inspections have been shown to be effective for identifying defects. One of the weaknesses in the inspection process is the lack of guidance given to individual inspectors. *Software Reading Techniques* have been helpful to inspectors. Software reading is the analysis of a textual software product (e.g., requirements, design, code, test plans) by an individual inspector to achieve the understanding needed for a particular task (e.g., defect detection, reuse, maintenance). A software reading technique is a specific set of instructions showing an inspector how to read an artifact to accomplish a particular task (e.g. defect detection) [Shull02].

In current practice, a software document is often analyzed in an ad hoc manner rather than by using a clear, defined approach. Reading techniques can be used to improve inspections that are done in an ad hoc manner. A series of studies have demonstrated the benefits of procedurally defined reading techniques in different domains and inspection types: natural language requirements [Basili96], formal notation [Porter95], high-level designs [Laitenberger00,Shull01], code [Basili87, Wood97, Laitenberger01], and user interfaces [Zhang99].

Previous studies of requirements inspection techniques have generally focused on the details of the technique itself, e.g. [Basili96, Porter97b]. Another major factor in the inspection process, the inspectors who are using the technique, is just as important and has for the most part been neglected. The inspection process is human based, so the variations among individual inspectors will likely have an impact on the outcome of the inspection; therefore study of the characteristics of the inspectors is an important task. In this study, the interaction between one specific characteristic, the software development experience of the inspector, and the specificity they require in an inspection process was studied.

As we shifted the focus of the study from the process (the inspection technique) to the people (the characteristics of the inspectors themselves), some new issues arose. In particular, many of the assumptions that we had made about our environment proved to have a different impact on this study than they had on previous studies. After describing the details of the study itself, we will discuss these assumptions and their impact along with the reasons why the impact was different in this study. There is a tree.

2. BACKGROUND

2.1 Inspections

Because there are many styles of inspections, any discussion of inspection research should clearly identify the type that is used. In the traditional team-based inspection paradigm [Fagan76], a group of inspectors, each assigned a specific role, works together to detect defects. The focus of this style of inspection is the team meeting. Prior to that meeting, the team members do some individual preparation, based on their role, to familiarize themselves with the document to be inspected. The inspectors are generally given little guidance on how to effectively prepare for the team meeting. After the individual preparation, the team members meet to discuss the artifact and detect defects. A series of studies have been conducted that show that the team meeting may not be a necessary part of the inspection process in terms of defects detected [Votta93]. In fact, the individual inspectors who are involved may have as great or greater an impact than the overall inspection process itself [Siy96]. Because the research has shown that the effectiveness of the individual inspectors is as important as the team meeting, and because they are often given little guidance for individually reviewing a software artifact, we focus on improving the effectiveness of the individual inspector.

2.2 Individual Techniques

Due to the general lack of guidance normally provided to the individual inspectors, researchers have sought to create techniques to provide that guidance. Various techniques have been created ranging from simple checklists to more complex and structured procedures. While any software artifact can be inspected, it has been shown that it is cheaper to find and repair defects early in the software lifecycle than it is later in the software lifecycle [Kelly92, Boehm01]. Therefore, in this work we have chosen to focus on the inspection of software requirements documents. Researchers have shown that one particular type of structured, procedural requirements inspection technique, the scenario-based technique, provides benefits over a less procedural checklist in some situations [Basili96]. A specific scenario-based technique is called Perspective Based Reading (PBR). PBR asserts that for a requirements document to be correct, it must satisfy the needs of all its potential stakeholders. PBR consists of a set of techniques, specific to the needs of each stakeholder. PBR aims to ensure that the needs of all relevant stakeholders are considered during an inspection. Each PBR technique consists of three parts as described below:

First, to ensure that the needs of each stakeholder are met, each PBR technique asks the inspector to assume the perspective of one of the stakeholders. By focusing on the needs of one stakeholder rather than the needs of all stakeholders at once, the inspector can do a more thorough job of ensuring that the information present is sufficient to meet the needs of that stakeholder. The idea being that if each member of an inspection team assumes a different perspective, all of the stakeholders have their needs addressed. The most common perspectives are *tester, designer,* and *user*.

The second aspect of PBR is a step-by-step procedure that is followed for each perspective. Each stakeholder of the requirements typically uses an abstraction or model of to think about the requirements. For example, the *tester* might abstract the requirements into a set of test cases, the *user* to a user manual, and the *designer* to a design that provides high-level details of potential classes, attributes and methods. The PBR procedure provides the inspector with a series of steps to follow to create the appropriate model for the perspective they are using. The creation of the model helps the inspector ensure that all of the necessary information is present.

The third aspect of the PBR procedure is the defect taxonomy that defines the important classes of defects. After identifying the classes of defects (i.e., omitted information, incorrect information, inconsistent information, ambiguous information, and extraneous information), a series of questions is inserted into the above procedure to help the inspector identify each relevant defect type. After each model creation step, the inspector is asked to look for defects from the specific defect classes. Based on these three aspects a set of techniques is created for reviewers to follow. For more information on PBR see [Shull00].

2.3 Variance among Individual Inspectors

Previous studies have also shown a wide variation in the performance of different inspectors even when using the same technique [Schneider92, Basili96, Laitenberger00]. These studies showed that there are influences other than the techniques that affect an inspector's performance. Researchers have suggested that the selection of inspectors based on their characteristics can affect the number of defects found during the inspection process [Parnas85], [Porter97a], [Sauer00]. An inspector's background and experience in various areas, e.g. software development or application domain, can have an impact on the number of defects they find during an inspection. Some of the characteristics that have been investigated in the past relate to the experience of the inspectors in different tasks, such as writing requirements, using requirements, testing software, writing use cases, and so forth [Carver03].

A series of studies have been conducted by researchers in the Reader's project, a collaboration between American researchers at the University of Maryland and at the Fraunhofer Center – Maryland and Brazilian researchers at the University of Sao Paulo, the Federal University of Sao Carlos, the Federal University of Rio de Janeiro/COPPE, and UNIFACS [Shull02]. Results from these studies have indicated some conflicts between the experience level of the subjects and the inspection steps of the inspection technique they were asked to use. The evidence of these conflicts appeared both in the quantitative results of the studies as well as in qualitative results gathered through discussions the researchers had with the subjects after the studies.

2.4 High level goals of the study

In addition to showing a variation in the performance of individual inspectors, the data from earlier studies also indicated that the presence of an inspection technique seemed to neutralize some of the effects of inspector experience [Basili96, Carver03]. This neutralization was due to the increase in performance of inexperienced inspectors and/or the decrease in performance of more experienced inspectors. Based on discussions with our Brazilian colleagues, we believe that the negative effect of the technique on experienced inspectors occurred because the specificity (amount of detail) of the technique interfered with the innate process that an experienced inspector would normally use. Conversely, the specificity of the technique seemed to provide inexperienced inspectors needed guidance to improve their performance. But, there were still some open questions related to the interaction between the level of specificity in the technique and the experience level of the inspector:

1) Does the level of process specificity make a difference?
2) Does an inspector's software development experience have an effect on the amount of detail they need in an inspection process?

This study was designed to address these questions on the importance of the specificity of a technique. The answers to these questions can provide advice to organizations on which techniques to choose based on the experience levels of their inspectors.

The primary goal of the study was to begin understanding the impact that software development experience has on the use of a specific requirements inspection process, Perspective Based Reading (PBR). The metrics of interest in this study were effectiveness, the number of defects detected, and efficiency, the amount of effort required in finding those defects. The specific software development experience metric studied here was the inspector's experience in the PBR perspective. The experimental hypothesis was:

> *To be effective, the level of detail in a technique must be tailored based on the inspector's experience. More experienced inspectors need less detail while less experienced inspectors need more details.*

The secondary goal of this study was to pilot a method for evaluating the effectiveness of the training and the expertise of the subjects. Because the results of most studies are based on the assumption that the training session provides adequate opportunity for the subjects to gain necessary skills to use the technique, evaluating the effectiveness of the training is an important task. The method chosen was that of a pretest/posttest, used to evaluate the subjects' knowledge before and after the training session. In addition to being useful in the evaluation of the training session, the pretest/posttest can also be useful in allowing researchers to move away from more subjective self-reported experience levels towards a more objective measure.

2.5 Impact of the Results on practice

The results of this study can give advice to an inspection team leader on the types of techniques he or she should use depending on the level of experience of the inspectors on the team. If it is shown that inspectors with different levels of experience benefit from techniques with different levels of specificity, then an inspection planner will have an idea of some concrete metrics he or she can collect in order to assign the right technique to each team member.

3. MEASUREMENT GOALS

The high level goal for this study can be broken down into some more specific measurement goals, with their associated questions and metrics using the Goals, Question, Metrics (GQM) framework [Basili94]. The goals, questions and metric for this study are enumerated below.

Goals

G1 To analyze the **PBR techniques** for the purpose of evaluating the tailoring in terms of level of detail with respect to effectiveness and efficiency from the point of view of the inspector.

G2 To analyze the **list of defects** for the purpose of characterizing it with respect to completeness from the point of view of the subjects.

G3 To analyze the **pre/post tests** for the purpose of characterizing and evaluating them with respect to effectiveness from the point of view of the researcher.

Questions

Q1 -- What percentage of defects did the high experience inspectors find?

Q2 -- What percentage of defects did the low experience inspectors who used a low detail technique find?

Q3 -- What percentage of defects did the low experience inspectors who used a high detail technique find?

Q4 -- How much effort did the high experience inspectors use to find defects?

Q5 -- How much effort did the low experience inspectors who used a low detail technique use to find defects?

Q6 -- How much effort did the low experience inspectors who used a high detail technique use to find defects?

Q7 -- What was the subjects' opinion of the technique they used?

Q8 -- Do the subjects agree with the defect list? Why or why not?

Q9 -- Were the pre/post test questions equivalent?

Q10 -- Were the pre/post test questions categorized properly?

Q11 -- How many unique responses were given by the subjects for each pre/post test question?

Metrics

M1 -- True defects found by the subjects

M2 -- Effort expended by the subjects

M3 -- Post-study questionnaire answers

M4 -- Defect feedback question answers

M5 -- Pre-test question answers

M6 -- Post-test question answers

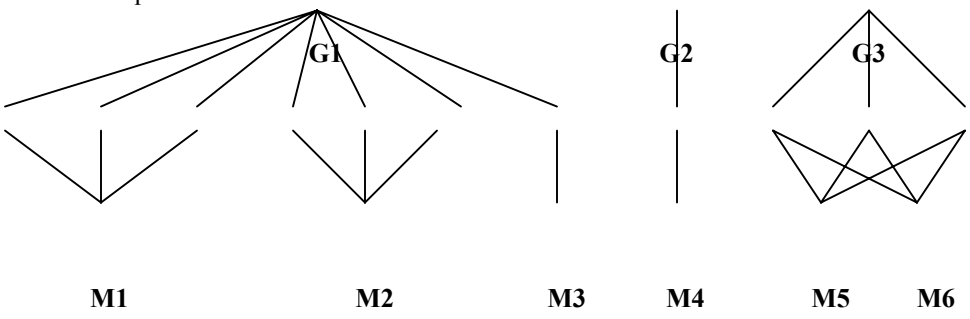

Figure 1 – Goals, Questions, and Metrics

4. THE EXPERIMENT

4.1 Experimenters

Researchers at the University of Maryland and the Fraunhofer Center-Maryland conducted this experiment. This same group of researchers created the PBR techniques used in the study, and therefore had a high level of expertise in their use.

4.2 Subjects

The subjects were students in a graduate level software engineering class in the Fall 2002 semester at the University of Maryland. Approximately 1/3 of the students were classified as experienced testers, based on their self-assessment of having experience testing on more than one industrial project. The other 2/3 of the students were classified as inexperienced testers.

4.3 Materials

The subjects inspected a requirements document for a Parking Garage Control System (PGCS). The PGCS requirements document described a system that managed a parking garage. The system was responsible for keeping track of monthly and daily (pay for each use) parking tickets as well as ensuring that cars were only allowed to enter the garage if there was space available. The PGCS requirements document had 17 pages, which included 21 functional and 9 non-functional requirements. This requirements document contained thirty-two defects, some of which were seeded by the experimenters and some occurred naturally. Each subjects used one of two versions of the PBR tester perspective to perform their inspection.

Version 1 of the PBR technique was the *high-level version* (low specificity) that did not contain a step-by-step procedure for creating an abstraction model of the requirements. Rather, the technique instructed the inspector to create the abstraction model, a set of test cases, using a method of their choosing. The technique provided a series of questions for the inspector to consider before, during and after creating the test cases, to help them uncover defects.

Version 2 of the PBR technique was the *step-by-step version* (high specificity) that included a detailed, (step-by-step) procedure for building the abstraction model, a set of test cases, based on a specific acceptance testing technique, Category Partition Testing (CPT) [Ostrand88]. To employ CPT, the tester groups requirements into *functional units*, which are either high-level user commands, or large functions that will be called by other functions.

Table 1 – PBR Techniques

	Version 1	Version 2
Name	High-level Version	Step-by-step Version
Process Detail	Low Process Specificity	High Process Specificity
Model	None Included	Category Partition Testing

For each functional unit, the tester must identify necessary parameters and environmental variables. Based on these parameters, the tester creates a set of test cases. This testing technique was assumed to be simple enough that that the inspectors would be able to easily understand it and be able to focus more attention and effort on uncovering defects. The two versions of PBR are summarized in Table 1 and can be found in Appendix B.

4.4 Procedure

4.4.1 Overview

Because, the overall goal of this study was to understand the interaction between an inspector's experience in their PBR perspective and the amount of detail needed, the subjects were grouped based on their experience in the Tester perspective of PBR. Some of the subjects were given the step-by-step version of the PBR and the other subjects were given the high level version of PBR as described in the next section. Using both data from this study and historical data from studies that used versions of the PBR tester technique, we were able to compare the performance of subjects using different versions of the technique.

4.4.2 Experimenter's Procedure

The first step of this study was for each subject to complete background and experience questionnaire, a copy of which can be found in Appendix A. This questionnaire collected information about the subjects' experience in various aspects of software development, including aspects represented by the PBR perspectives. Based on the number of subjects in this study, it was feasible to use only one of the PBR perspectives. So, the responses from the

questionnaire were used to select the tester perspective because it had the best balance of experienced and inexperienced subjects.

The classification criterion was determined prior to the examination of questionnaire responses so that the division of subjects in to the high experience group and the low experience group would not be not biased by the profile of the subjects in this study.

Subjects with industrial testing experience on more than one project were classified as high experience and the rest as low experience. When this criterion was applied, approximately 1/3 of the subjects were in the high experience group and 2/3 in the low experience group. This was the most even split of the three perspectives, so the tester perspective was chosen.

The low experience subjects were split between the two versions of the techniques. Four of the low experience subjects were concurrently enrolled in a testing course, in which they were exposed to, but did not use, the CPT technique, so those subjects were assigned to use Version 2 of PBR, which contained the CPT details. The remaining twelve subjects from the low experience group were randomly assigned to one of the two versions of the PBR technique.

Table 2 – Experimental Design

	Group 1	~~Group 2 (not enough subjects to use)~~	Group 3	Group 4
Perspective Experience	High	~~High~~	Low	Low
Process Detail	No Model	~~Model included~~	Model included	No Model
Model	Own	~~Abstract~~	Category Partition Testing	Own
Number of Subjects	6		10	6

The high experience subjects were all assigned to use Version 1 (the high-level version) of PBR. Because there were half as many high-experience subjects, both versions of PBR could not be used. Had there been more subjects in the high-experience group, they would also have been split in half. The complete design can be seen in Table 2.

4.4.3 Training

The subjects were trained according to the version of the PBR technique that they were assigned. The first part of the training was the same for both versions. As shown in Table 3, the training covered two class periods. Each activity in the training will be described in more detail below.

Pretest

The pretest at the beginning of the training session was the first part of the pilot study. It was used to evaluate the subjects' level of knowledge about detecting requirements defects and creating test cases prior to the training. For the pretest, two requirements excerpts were created, one for an Automated Teller Machine (ATM) and the other for ABC Video Store (ABC) and seeded with defects. The two excerpts were independently analyzed prior to the pilot study to try to ensure that their relative difficulty was similar. Half of the subjects were randomly assigned to each artifact. In the pretest, the subjects were asked to 1) determine the presence or absence of a defect in each of three requirements from the requirements excerpt, and 2) create test cases for three other requirements from the same excerpt.

Training

The first part of the training was the same for all subjects regardless of the technique that they had been assigned. First, the subjects were given some background on the current research and practice in inspections. Next, the subjects were trained in software defects so that they could become more familiar with defects in general and with requirements defects. The first class period finished with a general introduction to reading techniques. At the beginning of class period 2, the subjects were trained in PBR. The common part of the training covered the background and theory behind PBR as well as an explanation of the high level version of PBR. (The subjects were not informed that there were two different techniques and the training for the step-by-step version built on the

Table 3 – Training Sessions

Class Period 1					
Activity		**Time**	**Topic**	**Handout**	**Slides**
Pretest		9:30-9:45	--	ABC / ATM	2
Training		9:45-10:00	Inspection Background	--	3-13
		10:00-10:30	Software Defects	Gas Station	14-21
		10:30-10:45	Reading Techniques	--	22-27
Class Period 2					
Activity		**Time**	**Topic**	**Handout**	**Slides**
Training		9:30-10:00	PBR	--	28-42
Assignment		10:00-10:05	Explanation	--	43
Split Class		10:05-10:10	--	--	44
Group 3	Training	10:10-10:35	Category Partition Testing	PBR Technique	45-60
	Posttest	10:35-10:45	--	ABC / ATM	61
	Assignment	10:45	Pass out Assignment	Assignment / PG / Defect Form	--
Groups 1 & 4	Posttest	10:10-10:20	--	ABC / ATM	--
	Assignment	10:20	Pass out Assignment	Assignment / PBR Technique / PG / Defect Form	--

training for the high level version, so this time was not wasted for those subjects who had been assigned the step-by-step version.)

Split Class
Next, the class was split based on the version of the technique they were to use, with each group going to a separate room. (The subjects were not informed why they were split, so they did not know there were two techniques).

Extra Training
The subjects, in Group 3, who had been assigned the step-by-step version of PBR, received an extra training session, to cover the details of the step-by-step version not in the high level version. This training session included an explanation of the Category Partition Testing technique and its use. After this explanation, the subjects were shown how the Category Partition Testing technique was used in the PBR technique.

Posttest
At the completion of the training session (prior to the use of PBR) the second half of the pilot study was run. To evaluate what they had learned from the training, each subject was given a posttest, with the same instructions as the pretest. For the posttest, the subjects were given the requirements excerpt (ABC or ATM) they did not use during the pretest.

Assignment
Finally, the subjects were given an assignment description that contained the details necessary to execute the inspection and report the defects. The subjects were given a chance to ask the experimenters questions to clarify the assignment.

4.4.4 Execution
After the posttest, each subjects was given the appropriate version of PBR and the requirements document to be inspected. The subjects performed the inspection of the requirements document, on their own, and recorded any defects detected. For each defect, they also recorded the time the defect was found and its type (from a predetermined scheme). The subjects were given a week and a half to complete the inspection.

4.4.5 Data Collection

Both quantitative and qualitative data were collected in this study. The quantitative data included the time required to perform the inspection, the number and type of defects detected, and the time each defect was found. The qualitative data was collected via post-experiment questionnaires and through subject feedback, as described below.

After completion of the inspection, the subjects filled out two questionnaires to discuss their experience with PBR. On the first questionnaire, which can be found in Appendix C, the subjects were asked for their thoughts on the amount of detail contained in their version of the technique, and how their background knowledge impacted their performance. Because the two versions of PBR differed slightly, there were two versions of the questionnaire, one for each version of the PBR technique. On the second questionnaire, which can be found in Appendix E, the subjects were asked about what they learned by participating in the inspection. The goal was to get the subjects' opinions of the types of learning that occurred while inspecting a requirements document in order to find defects.

After the subjects submitted their defects lists, the experimenters "scored" the subject defect lists by determining which of the reported defects were true defects based on the master defect list prepared prior to the study. These defects were then recorded in a spreadsheet. After a preliminary analysis of this data, the experimenters gathered more information about the defects from the subjects in order to ensure that the scoring was done properly. Based on this information, the defect lists were slightly adjusted; that is the defect count was increased by one for three subjects and decreased by one for two subjects. Details on this feedback process are in Section 4.6.2.

Finally, once all of the data had been collected, and some preliminary analysis done, a presentation was made to the subjects. The purpose of this presentation was to explain the experimental design and discuss the rationales behind decisions made on various tradeoffs, so that the subjects could begin to learn more about experimentation. In addition, the subjects were given a chance to provide feedback to the researchers about the preliminary results. This feedback session provided a sanity check on the data and the results. Some of the subjects provided extra information during the discussion that aided in the final analysis of the data.

4.5 Data Analysis

Table 4 shows six potential experimental treatments based on two experience levels (high & low), the level of detail in the process (high & low) and for the high detail process, the type of model present in the technique (simple & complex). The table shows where G1, G3, and G4 from this study fit. Additionally, to fill in the gaps for two of the treatments, data from an earlier study was used. The CMSC735 (1997) study was conducted in a graduate level software engineering class at the University of Maryland. In this study, there were both high experience and low experience testers. The subjects used a step-by-step version of PBR that included a complex testing technique (Equivalence Partition Testing) to inspect the PGCS requirements. In the CMSC735 Fall 1997 study, both high and low experience inspectors used the high detail-complex model version of the PBR tester perspective. G2 would have filled in the sixth treatment had there been enough high experience subjects. Figure 2 gives a visual presentation of this information.

Table 4 – Breakdown of Variables and Data Sources

Treatment	Perspective Experience	Process Detail	Underlying Model	Studies
T-H1	High	High	Complex	CMSC735 (1997)
T-H2	High	High	Simple	~~G2~~
T-H3	High	Low	N/A	G1
T-L1	Low	High	Complex	CMSC735 (1997)
T-L2	Low	High	Simple	G3
T-L3	Low	Low	N/A	G4

Based on Table 4 and Figure 2, the following statistical analysis was done. First, for the high experience subjects, the results from the subjects in G1 were compared with the results from the high experience subjects in the CMSC735 (1997) experiment using a parametric t-test. Next, for the low experience subjects, the results from G3 were compared with the results from G4 using a parametric t-test. Finally, the results from G3 and G4 were individually compared with the results from the low experience subjects in the CMSC735 (1997) experiment using a parametric t-test for each of the two comparisons.

4.6 Results

The results of this study are grouped according to the measurement goals from Section 3.1.

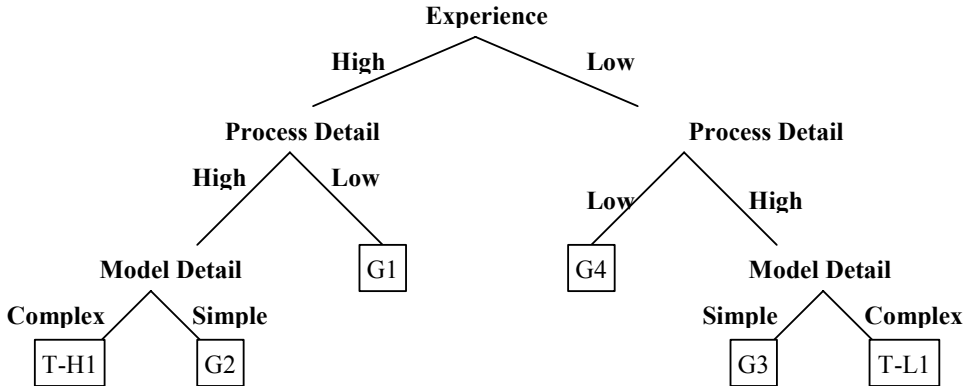

Figure 2 – Variables/Treatments Tree

4.6.1 Goal 1 – Effectiveness and Efficiency of the PBR Techniques

The defect lists submitted by each subject was analyzed to determine the percentage of the known defects found. Table 5 shows the average percentage of defects found by the subjects in each treatment group and the efficiency, defects found per hour. Table 6 shows the same information for the CMSC 735 Fall 1997 study discussed earlier.

Table 5 – Defect Rates

	Group 1	Group 3	Group 4
Perspective Experience	High	Low	Low
Process Detail	No Model	Model included	No Model
Model	Own	Category Partition Testing	Own
Average Defect Rate	22.1%	20.6%	26.5%
Efficiency (defects/hour)	4.9	2.6	4.2

Table 6 – Historical Defect Rates

	CMSC735 Fall 1997	CMSC735 Fall 1997
Perspective Experience	High	Low
Process Detail	Model Included	Model Included
Model	Equivalence Partition Testing	Equivalence Partition Testing
Average Defect Rate	26.7%	27.3%
Efficiency (defects/hour)	1.9	3.7

The results for the **high experience subjects** were:

- Subjects from Group 1 in this study, found 22.1% of the defects at the rate of 4.9 defects/hour.
- High experience subjects from CMSC735 1997 who used the PBR tester perspective technique that included a very specific method (Equivalence Partition Testing) for creating test cases found 26.7% of the defects at the rate of 1.9 defects/hour.

The statistical tests described in Section 4.5 were run to compare the results from these two groups of subjects. The subjects from the CMSC735 Fall 1997 study found more defects, but the difference was not statistically significant ($p < .05$). The subjects from Group 1 were significantly more efficient than those from CMSC735 Fall 1997 ($p < .05$).

Based on these results, the more complex model helped subjects find more defects, but significantly reduced their efficiency. Upon further analysis of the post-experiment questionnaire (discussed in Section 4.6.3), it appears that the high-experience subjects may not have been experienced enough to benefit from the low specificity version of PBR.

The results for the **low experience subjects** were:

- Subjects from Group 4 in this study found 26.5% of the defects at a rate of 4.2 defects/hour.

- Subjects from Group 3 in this study found 20.6% of the defects at a rate of 2.6 defects/hour.
- Low experience subjects from CMSC735 1997 who used a PBR technique with a complex model (Equivalence Partition Testing) for creating test cases found 27.3% of the defects at a rate of 3.7 defects/hour.

The statistical tests described in Section 4.5 were run. The subjects from Group 3, who used the simple model, found fewer defects than the subjects from Group 4 and significantly fewer defects ($p < .05$) than the subjects from CMSC Fall 1997. There were no statistically significant differences in terms of efficiency.

Based on these results, using a simpler model for creating the test cases provides little or no benefit to the subjects. In fact, because that group of subjects was both the least effective and the least efficient, it may have hurt their performance. Based on the results of the post-experiment questionnaire discussed in the next section, in general, the subjects did not think they completely understood the concepts necessary to effectively use Category Partition Testing as a method for creating test cases. The fact that the subjects who were given a technique that provided no guidance on creating test cases found almost as many defects as those with the more complex model indicates that the subjects did not benefit from the guidance provided by the technique. Furthermore, we can hypothesize that the more complex testing technique, Equivalence Partition Testing, is simply a better tool for finding defects than the less complex testing technique, Category Partition Testing. In order to address these issues, further study will be done allowing the subjects more time to become familiar with and practice the model building technique before using it in a defect detection technique.

4.6.2 Goal 2 - Quality of the Requirements Defects

Because this study used an artifact that was seeded with defects, this goal has two purposes:

1) To determine whether the defects seeded in the artifact were representative of artifacts of that type
2) To determine if the list of defects being used by the experimenters was accurate

During the study, we did not collect data to specifically address the first issue. However, after the study, some subjects suggested that the real power of PBR could not be seen because the defects present in the requirements were too easy to find and would have likely been found without using PBR. While this argument is plausible in general, in this case the subjects found an average of less than 25% of the defects, so this assertion does not seem plausible in this case.

To address the second issue, we asked the subjects to provide feedback on our defect list. After the conclusion of the inspection, the subjects were given a copy of the experimenters' master list that of known defects. The subjects were asked the following questions about each defect on the list.

1) Do you agree that this item is really a defect? If no, then why not?
2) Regardless of whether you agreed that this item was a defect, did you see this issue while performing your inspection?
3) Do you think you reported this item on your defect list?

These questions were chosen based on the reasoning process that an inspector might go through while inspecting a requirements document. Figure 3 shows a decision tree that an inspector might follow in deciding whether or not to report a defect. Reporting a defect is a complex action, so several issues, including those included in Figure 3, are confounded in defect counts. By following up with these questions, our goal was to more accurately understand the thinking process of the subjects and get a better idea of what the subjects did or did not see during the inspection.

After receiving the subject's answers, there were some discrepancies between the defects that the subjects said they reported and the "scoring" done by the experimenters. There were some cases where the experimenters gave the subject credit for reporting a defect while the subject did not think they reported it. There were also some cases where the subject thought that they had reported a defect, but the experimenters had not given them credit for that defect.

Because of the discrepancies between the subjects answers and the experimenters' scoring, the subjects' defect lists were returned to them with the experimenters' scoring included. They were asked to correct any problems they saw, including:

1) Defects for which the experimenters had given them credit but they did not think they found.
2) Defects that they thought they found, but for which the experimenters had not given them credit.

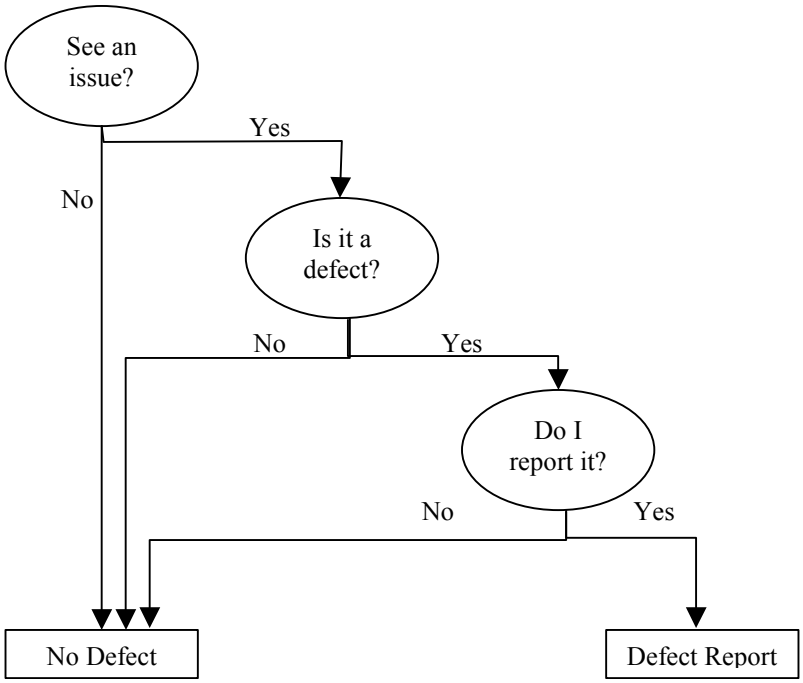

Figure 3 – Defect Reporting Decision Tree

After receiving this feedback from the subjects, the defect lists were reexamined and adjusted as necessary to reflect the true defect counts as accurately as possible. Table 7 contains the details of the adjustments that were made to the initial data, based on the subjects feedback. In the table, "Added" refers to a defect on the subject list for which they were not originally given credit, but after the feedback were given credit. Likewise, "Removed" refers to a defect for which the student was originally given credit, but after feedback was not given credit. "Changed" refers to a defect from the experimenters list for which a subject was given correctly credit, but was mapped to the wrong item on the subject defect list. Therefore, the net change for an "Added" defect is '+1', for a "Removed" defect is '-1', and for a "Changed" defect, is '0'.

Table 7 – Defects Added/Removed/Changed

	Changed	Added	Removed	Net Change
Subject 1	0	1	2	-1
Subject 5	1	1	0	+1
Subject 8	1	0	0	0
Subject 9	0	1	0	+1
Subject 16	0	0	1	-1
Subject 22	0	1	0	+1

Table 7 shows that 2 subjects had their count reduced by one, 3 subjects had their count increased by one, and the other 17 subjects had the number of defects found unchanged. This small amount of change indicates that the "scoring" done by the experimenters gave an accurate count of the number of defects found.

Conversely, after analyzing the results of the subjects' feedback on the defect list, it was clear that there was some disagreement about what constituted a true defect. If a relatively large number of subjects think that a defect on the master defect list is not really a defect, then that defect is, at the very least, confusing and at the worst, not really a true defect. On average, 3 subjects disagreed with each defect, with a minimum of 0 disagreements for 3 defects and a maximum of 9 disagreements for 1 defect.

Future work in this area will be to perform a qualitative analysis of the subject's explanations of their disagreements to determine if their responses can be categorized in anyway. In addition, the specific defects that had the highest amount of disagreement will be further analyzed to understand any similarities among those requirements.

4.6.3 Goal 3 - Pilot Study of Pre/Post-Test

For each of the two requirements excerpts, there were three requirements, one of which had an obvious defect, one had a subtler defect and the third was correct (i.e. it had no defect). The expectation was that more subjects would find the obvious defects than the subtler defects. Furthermore, it was expected that more subjects would correctly identify the defects after the training than before the training.

In addition, there were three other requirements for which the subjects were to create test cases. Due to the time restrictions placed on the pretest and posttest, many subjects did not have time to fully address these questions. Therefore, this analysis is only for the defect identification questions.

Because this investigation was a pilot study, it was important to understand whether the instruments used (the ATM excerpt and the Video store excerpt) were of equal difficulty. It is necessary for the pretest and the posttest instruments to be of equal difficulty to ensure that the effect of learning is measured, rather than being masked by differences in the measurement instruments.

Each requirements excerpt had an easy to find defect, a subtler defect, and a correct requirement, so one way to compare the requirements excerpts was to compare the performance of the subjects from ATM to the performance of the subjects on the Video store for each type of question. The subject was given credit for getting the question right if they either correctly identified the defect (for the obvious and subtle defects) or stated that the requirement was correct (for the requirement that contained no defect).

Table 8 presents the percentage of subjects who correctly identified each type of defect or lack of defect for each of the requirements excerpts (regardless whether the instrument was used in the pretest or the posttest). Based on this measure the two instruments do not appear to be equivalent. The obvious defect in the Video store excerpt was much easier to find, with 90% of the subjects finding it and only 30% of the subjects finding the obvious defect in the ATM excerpt. Furthermore, because less than 25% of the subjects correctly identified the requirement that had no defect, those requirements were less clear than assumed.

Table 8 – Percentage Correct by Defect Category

Defect Category	ATM	Video
No Defect	24%	20%
Obvious Defect	30%	90%
Subtle Defect	44%	37%

Understanding the problems with the excerpts, we analyzed whether the subjects did better on the post-test than on the pre-test, regardless of the ordering of the requirements excerpts. Table 9 presents the percentage of subjects who correctly identified each type of defect or lack of defect for the pretest and posttest. Based on the data, the only improvement from pretest to posttest was for the subtle defect. Conversely, fewer subjects correctly identified the requirement that had no defect.

Table 9 – Percentage Correct by Test

Defect Category	Pretest	Posttest
No Defect	27%	17%
Obvious Defect	52%	56%
Subtle Defect	24%	57%

An alternative measure of difficulty is the relative number of unique responses generated by each requirement. A response is either "no defect" or the explanation of a defect present in the requirement. The rationale behind this measure is that a clear requirement has less room for alternate explanations. Thus, a requirement with many unique responses may have been more difficult to understand than one with fewer unique responses. Table 10 presents the number of unique responses for each requirement in the pilot study. There were more unique responses for the correct requirement in the Video store than in the ATM. Conversely, there were more responses for the obvious defect in the ATM than in the Video store. One conclusion that can be drawn from this data is that the two requirements excerpts were not equivalent.

Table 10 – Unique Response Count by Defect Category

Defect Class	ATM	Video
No Defect	4	11
Obvious Defect	8	3
Subtle Defect	9	9

5. DISCUSSION OF ASSUMPTIONS

In addition to studying the process detail, this focused on the people involved in an inspection. The domain of social science is therefore relevant and insights from social science researchers can be helpful. In this study, we were concerned with the experience of the subjects and how that experience related to the use of a technique. Experience characteristics are not directly observable. In psychology and sociology, an object of study that is not directly observable or measurable is called a *construct*. This term is used because the object of study is constructed as part of the theory, rather than being directly observable itself. In the social sciences, researchers focus on understanding the relationship between a mental construct and a concrete representation of that construct, such as a document or process.

Figure 4 describes the relationships between theoretical constructs and physical representations, or operationalizations, of those constructs. Theories represent the predicted causal relations between objects. In an experiment, an operationalization of the theory into a program or treatment is created and then applied by a group of subjects. Observations are then made from the resulting data.

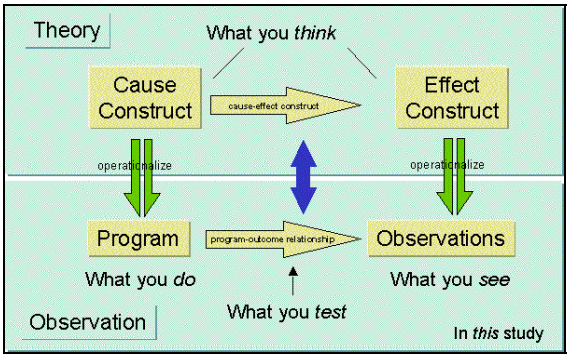

Figure 4 –Theoretical Constructs Used in Experimentation [Trochim00]

In Figure 4, arrows represent inferences by the researcher that may or may not be valid in a given study. Arrows from theory to observation (experiment) represent the construct validity. That is, did the program (treatment) and the observations (measurements) operationalize an accurate representation of the theory? The arrow from program to observation represents the internal validity. That is, were the observations really produced by the program (treatment)?

Software Engineering experiments have some similarities to experiments in both the physical sciences (e.g. physics) and the social sciences (e.g. psychology). In the physical sciences, experiments tend to focus on studying the effects of a process on a product. In the social sciences, experiments tend to focus on studying a person using a process to accomplish some task. Physical science and social science experiments typically have to account for only one class of object in addition to the process being studied (either the product or the person).

In contrast, software engineering researchers often study the effects of a person who is using a process on a product. Therefore, software engineering researchers have to take into account people, products and processes. In order to design valid experiments, software engineering researchers should be informed by both physical science and social science experimentation methods.

In any given study, researchers make a series of assumptions related to the people, products and processes. Some of the assumptions will vary depending on the focus of the study, but many of the assumptions will be common across all types of studies.

Figure 4 can be used to organize assumptions made about the people, processes, and products. In terms of people, the assumption that the characterizing measures are valid is really an assumption that the theoretical construct of experience has been accurately operationalized into a set of metrics. In terms of processes, the assumption that the training was adequate is an assumption that the training construct was operational in the class and that the expertise was truly developed by the subjects. In terms of products, the assumption that the format was useful is an assumption that the specific operationalization of the author's mental model allows the reader of the artifact to process information easier than another operationalization would.

For each of the three objects, an important consideration is balancing the various threats to validity. Increasing the internal validity often reduces the external validity. Controlling internal validity too much, at the expense of external validity, can affect the construct validity of the experiment. For example, if a researcher constrains the domain of the artifacts too much, the subjects' prior knowledge of the world may conflict with the arbitrary bounds and constraints of domain of the artifact. In this situation, construct validity is challenged because the mental models formed by the subjects may be different from those assumed by the researcher.

Therefore, there is a tradeoff among internal validity, external validity, and construct validity. A researcher must typically choose to focus on one of the three types, but has to also consider the other two types during the planning of a study. As the study focus changes, and with it the impact of the assumptions, these tradeoffs must be reassessed to optimize overall validity.

The results of this study were inconsistent with our expectations. Based on previous results, those expectations seemed to be rational. In order to understand the results and what occurred during the study, we examined the assumptions, explicit and implicit, that we made during the experiment. Those assumptions, related to people, products and processes, have different impacts on the validity of the results obtained from the experiment.

Many of the human experience characteristics that we were interested in measuring were not directly observable, so it was necessary to measure them indirectly. Because of this indirect measurement, several of the assumptions depended on the reliability or validity of measures used in the study. *Reliability* is defined as the consistency or repeatability of a measurement (i.e. the likelihood that the same results will be obtained if the measure is repeated). *Validity* is defined as the truthfulness of a measurement, (i.e. how close to the true value is the measurement that was taken) [Trochim00].

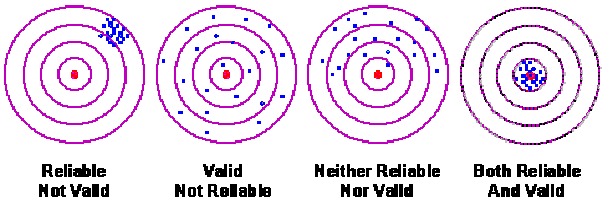

Figure 5 – Measure Validity and Reliability [Trochim00]

Figure 5 illustrates the difference between reliability and validity. The center of the bull's-eye is the true value of measurement. The dots are data points that represent actual measurements collected. When the data points form a cluster, then the measure is reliable, because as the measure is repeated, the values obtained are close together. When the average of the data points is close to the true value, then the measure is valid indicating that the measure will likely provide the true value [Trochim00].

The number of defects found by a subject is likely a valid measure of defect detection ability, but it is not very reliable. The number of defects found will fluctuate depending on several factors other than a subject's ability. Alternatively, self-reported data on the number of years of experience in software development and testing is probably a reliable measure – subjects will likely give the same answer if asked the question again. Yet, it is not clear that it is a valid measure of true skill in development or testing. The amount of experience (number of years) may not necessarily translate into the quality of experience (expertise).

In other earlier experiments, we focused on the process [Basili96, Carver03, Shull98]. The veracity of the assumptions related to people and products had an impact only on the external validity of those studies. Mistaken assumptions in those areas would affect the generalizability of the results but not the internal validity of the experiment. As the focus of our experiments shifted from processes to people, specifically the individual characteristics of people, the assumptions impacted different classes of validity threats.

People

The first set of assumptions related to the background and experience of the subjects. We assumed that:

1) Our measure of experience and the criterion for grouping were valid.

2) The amount of experience reported by the subjects was reliable and valid.

These assumptions relate to issues of measurement that frequently arise with human subjects. The first assumption affects the construct validity of the study. We theorized that a more experienced tester would be more effectiveness. Testing experience is not observable directly, so we had to operationalize it into specific observable measures for the study. Our operational measure of experience included two factors: location (industry vs. school) and amount (number of projects). Specifically, we assumed that experience in industry was more beneficial than experience in the classroom and that more industrial experience was more beneficial than less industrial experience.

The results of the study showed little difference in effectiveness between the subjects with low testing experience and the subjects with high testing experience. In fact, the subjects from Group 4 were the most effective of all the subjects. This result cast doubt on the veracity of our assumption about experience.

The qualitative data from the study indicated that our measure of testing experience and our criterion for grouping did not seem to correspond to our theoretical concept of experience:

1) Of the six subjects classified as having high testing experience, two stated that the training needed more examples of creating test cases. Four of the six stated that there were not enough details in the technique.

2) All of the high experience subjects reported that they needed additional background, which was not included in the lecture, in order to understand and use the technique.

Based on these and other data, we concluded that the subjects who we classified as having high experience in reality did not have a high level of testing expertise. There are many possible reasons for the misclassification of subjects, such as the metric "number of projects" not being a valid measure of the theoretical concept of experience, our criterion for grouping subjects was not a valid break point along the theoretical continuum of experience, or we simply did not capture the right kind of experience.

Another issue associated with this assumption was whether we were measuring the right type of experience. We believe that experience is important, but it is not always clear how to operationalize experience into a metric. For example, which experiences are truly important? Is number of projects really what we want to measure? We are interested in a subject's ability to effectively perform a particular task, not necessarily how long they have been doing that task. Thus, one subject might become proficient in a day, while another subject might not become proficient even after years of experience. In this study, it is clear that amount of experience was not a good proxy for proficiency in the technique.

The second assumption was that the experience levels reported by the subjects were reliable and valid. The potential for lack of accuracy in self-reported data arises from two sources. First, when asked to rate themselves, people often overestimate their experience. In a recent study, the correlation between self-reported ability and tested ability was less then 20% [Powers02]. Second, some questions can be interpreted in multiple ways. Thus, two subjects with the same actual experience level might rate themselves differently.

The subjects were asked to report experience they had in various aspects of software development at the classroom and industrial level. Because we did not specifically define experience, the term was ambiguous and open to interpretation. For example, does testing experience on one project in industry mean that the subject was the main tester for that project or that the subject performed some minor testing activities on one project? These two interpretations are very different.

Another example of this self-reporting problem related to the application domain knowledge of the subjects. On a post-experiment questionnaire, the subjects were asked to discuss cases where the requirements contained information that was new or different from what they expected. Two items seemed to make a difference on defect detection effectiveness. The six subjects who indicated that their mental model of a parking garage was in conflict with the model in the requirements document had an average detection rate of only 18.8%; while the seven subjects who said the requirements document helped them clarify their mental model had an average detection rate of 25.5%. This result was not statistically significant, but it does suggest that having a mental model of the domain that is in conflict with the model presented in the requirements has an adverse effect on defect detection ability.

When the artifact under inspection operationalizes the information differently from the subject's theoretical model, there seems to be difficulty during the analysis. If the subject's theoretical model is not contradicted but rather is simply clarified by the operationalization, then the subject seems to do well. This result may reflect either better synthesis of the new information by the subjects due to innate ability, or an inherent problem with new or different domains that inhibits detection ability.

To address these assumptions we recommend using more objective measures of subject experience. One potential method for measuring the subject's experience is use of a pre-test and/or post-test, such as the one piloted in this study. Instead of asking the subjects to report their experience, a pretest or posttest would allow a more objective evaluation of the subjects. Based on the results of such an evaluation, they could then be grouped into high experience and low experience groups. Furthermore, this objective test could also help account for the secondary variables. Better validation of the background survey would be another approach to the problem [Messick95]. With

an accurate characterization of the subject population, researchers can determine how likely it is that any other confounding factors are present.

Overall, the assumptions about people impacted the internal and construct validity of our study focused on people. False assumptions about the experience of the subjects have forced us to question the accuracy of our study results. We need to develop better methods for collecting the true experience of subjects.

Processes

The second major area of assumptions concerned the processes used by the subjects during the study. The primary object of study was the performance of the subjects using the techniques in which they had received training. In terms of processes we assumed:

1) The training was adequate for the subjects to become competent in using PBR.
2) The subjects followed the PBR process.

For these assumptions, there are two high level issues. The first deals with the threat to internal validity presented by assuming the training was effective. One technique might be ten times more effective than another might, but if the training is ineffective, and the technique is not used properly, that difference may not be seen. The second issue is the construct validity of the process. A training lecture can present the subjects with knowledge of the technique, but it is more difficult to guarantee that they have actually acquired skill in the technique so that they can use it effectively. The training is the operationalization of the technique as a theoretical construct. It is also difficult to ensure that subjects who were adequately trained actually follow the process.

The subjects in this study subjectively indicated that the training was adequate, with only one subject from each of the three treatment groups indicating that the training was insufficient. However, the subjects did indicate some desired improvements in the training. Half of the subjects who were trained in the step-by-step version of the technique asked for more examples to help them better understand the details of the method. One-third of the subjects trained on the high level version of the technique said they wanted more details about creating test cases. Furthermore, as reported earlier, many of the subjects indicated that there was necessary knowledge that they did not possess, e.g. about testing in general, about the specific model used in PBR, and about the application domain.

The researcher must decide on the overall goal of the training. Is the goal for the subjects to understand the theory behind a technique and the times in which it is applicable; or is the goal for the subjects to effectively use the technique? Those two contrasting goals require different training approaches. If the goal is that the subjects become effective in the use of the technique, then more laboratory sessions should be provided so the subjects can practice using the technique and receive feedback from an expert. If the goal is simply for the subjects to acquire knowledge about a technique, then classroom lectures including theory and proofs of correctness are more appropriate.

Once a researcher has identified the goal of the training session, then the assumption that the training was adequate can be addressed. One method for addressing this type of assumption is use of a pre-test/post-test design. The knowledge and skill of the subjects can be measured prior to training and then again after training to determine what they have learned. Conversely, an evaluation test could be created that would rate the knowledge and skill level of each subject based on an objective measure. Finally, subjects could be kept in the training process until they reach a certain level of knowledge and skill. Whatever method is chosen, knowing, rather than assuming, the impact of the training is the goal.

In this study, we piloted the use of a more objective measure to look at the effectiveness of our training. We developed a pretest and a posttest for the subjects to complete before and after the training session. The goal was to measure the improvement of the subjects from the pretest to the posttest. Despite our efforts to make the pretest and posttest artifacts comparable, the results of the study showed that they were not. Our experiences point out the difficulties involved in creating valid pretests and posttests for measuring the effectiveness of the training.

Products

The final set of assumptions was related to the products used during the study. We assumed that:

1) The format of the artifact was useful.
2) The scope of the problem was realistic.
3) The defects were reasonably independent and a good representation of actual defects.

These assumptions are related to construct validity. The format of the artifact is assumed to be a rational operationalization of the author's theoretical model of the domain. A defect is a construct of a theoretical belief of what could be wrong in a software artifact. The scope of the problem is related to the external validity of the study.

The first assumption was that the formatting of the requirements document was useful. The requirements in this study were formatted such that the steps necessary to achieve a user level function were broken up into smaller functional units. Therefore, several numbered requirements must be read together to understand a single function. Furthermore, the artifact did not provide a description of the system structure. The results of the study indicated that those subjects who felt that the format was helpful were less effective in finding defects than those who saw the format as problematic. These results echo the divergence between subjective assessment and objective reality found in the Cleanroom experiments [Selby87].

The second assumption was that the scope of the problem was realistic. Often the scope of the "toy" problems that must be used in this type of study is artificially smaller than it would be in the real world. Qualitative data was collected to understand the subjects' opinion of the scope of the inspected artifact. Over half of the subjects found something new or unfamiliar in the document. It was not as familiar a domain as we had thought. In contrast with people and their mental models, we rarely think about unobserved attributes of a textual document. Yet, defects actually are unobservable attributes of a software artifact. The reviewer must make an inference using their existing knowledge and the text to find a defect.

The results showed that the subjects were evenly divided on their opinion of whether or not there was enough information in the document to continue with designing and building a system. The defect detection rates for each group reveals a more interesting result. Those subjects who felt the specification did not adequately describe the scope of the project found fewer defects than those who felt the scope was well defined. The issue with the scope suggests that those subjects who were less certain of the system boundaries assumed that information was left out because it was outside of the scope and therefore not reported as defects. More interestingly, those who felt that there was not enough information present to start the design found more defects than did those who felt there was enough information. It is not clear whether the subjects understanding of the problems with the requirements caused this result or if it was caused by some other factor.

The third assumption was that defects seeded into the requirements document were reasonable. Because the results of this study are based upon the percentage of those defects that the subjects find, it is important to understand the quality of the seeded defects. One method to understand the quality of the defects is to ask the subjects to provide feedback. At the conclusion of the study, the subjects were provided with the master defect list and asked to indicate whether they agreed or disagreed that each item on the list was a true defect, and to explain why. There were some disagreements as to what constituted a true defect. One of the interesting results was that we also asked the subjects to indicate which of the defects on the master list they believed they reported during their inspection. There were many cases where the subjects indicated they had found a defect that the experimenters had not given them credit for and vice versa. The results of this exercise further indicate the mismatch between the constructs of the experimenters and those of the subjects. Both the parking garage concepts and the requirements defects concepts were different among the subjects.

In addition, there are several other avenues to pursue in characterizing and evaluating artifacts. First, any experiment involving an artifact as complex as a requirements document, even for a "toy" problem, must be pilot tested. Second, if multiple artifacts are used in a study, then pilot studies should be run to ensure that the defects seeded in each artifact are comparable. Third, requirements defects are more subjective than code defects, so researchers should remember that a defect is an operational construct of a theoretical concept. Finally, other issues that can affect construct validity include overall document size, defect density, defect realism, and defect type distribution. Thus, a proper operational definition (or a correct defect seeding) is critical for producing valid conclusions about defect detection.

6. CONCLUSIONS AND FUTURE WORK

Conclusions

The results of this study have shown that there was little difference in performance among the three treatment groups. Using historical data to get a better understanding of the effect of the various levels of detail in the techniques on subjects of high experience and subjects of low experience did provide some more detailed results.

For high experience subjects, those from this study that used the high-level version of PBR, which had no specified model embedded, were less effective but more efficient than those from the previous study, who had used the step-by-step version of PBR that included a specified model. For low experience subjects, those that used the version of PBR that included a simple model found fewer defects than either the subjects who used the high-level version of PBR, with no specified model or those that used a step-by-step version of PBR with a more complex model.

In addition to these conclusions, some issues dealing with assumptions call into doubt the validity of the above results and help to explain the lack of strong support for the experimental hypothesis. First, there are some potential issues related to the experience level of the subjects. The first problem is that the subjects self-reported their experience level rather than having it collected objectively. The second problem is that the researchers might have incorrectly mapped the reported experience levels into high and low experience. Secondly, it appears that the subjects need more time to practice and learn PBR before they will be able to be effective users of PBR. Having one or two class periods of training with little or no laboratory time for the subjects to practice does not appear to be adequate.

Another issue that arose as a byproduct of this study was the lack of agreement among the subjects and the researchers as to what the nature of a requirements defect. When given the opportunity to comment on the defect list used by the researchers, many of the subjects in the study believed that one or more of items considered defects by the researchers were not really defects for one reason or another. The lack of agreement on the nature of a requirement defect presents an interesting issue for researchers, which must be addressed in future work.

Finally, this study has proposed a method of objectively evaluating both the level of expertise of the subjects, as well as the effectiveness of the training. A pilot study that was run to investigate the usefulness of pretest and posttests showed promising results. However, this method needs to be refined and studied further. Once pretests and posttests are studied further and matured, they can be used to address some of the shortcomings discussed earlier.

Future Work

In addition to the conclusions drawn above, the results of this work have provided many ideas for future study. Because the classification of subjects in to "high" experience and "low" experience appears *a posteriori* to have been inaccurate, the subjects should be reclassified as high or low experience based on their answers to the post-experiment questionnaire. A formula for mapping the answers to the post-experiment questionnaire to the appropriate experience level needs to be developed. Once the mapping is done, the data can be reanalyzed to determine if the experimental hypothesis holds with properly classified subjects.

A second area of future work concerns the ambiguous definition of a requirements defect. To further study this issues, we need to perform a deeper analysis of the subject's qualitative feedback on the researcher's defect list. The responses of the subjects should be analyzed to determine why they agreed or disagreed with the researcher's list of defects. Patterns in this data could reveal the types of requirements defects that are ambiguous. Also, the defects that received a high amount of disagreement from the subjects should be analyzed to look for any similarities that can provide further insight into the issue of ambiguity in requirements defects. Finally, historical studies can be reanalyzed for further insight into the requirements defects that received a high amount of disagreement.

A third area for future work is the pretest and posttest. The development of an objective measure of expertise can be beneficial for researchers in terms of lowering their dependence on subjective experience levels provided by the subjects of the study. This study provided some lessons learned about pretests and posttests. We naively assumed that creating equivalent pretests and posttests to measure the subjects would be a relatively easy task. The analysis of the results of this study showed that creating an effective measuring device is a difficult task.

Two issues must be considered during the creation of pretests and posttests. The first issue is the validity of the pretest and/or posttest in terms of its accuracy in measuring the experience that the researcher is trying to measure. The pretest and/or posttest should allow the researcher to accurately separate subjects who have low knowledge in an area from those that have high knowledge, with some degree of certainty. The second issue is the comparability of the pretest and the posttest in terms of the expected score of the subjects. Researchers should strive for pretests and posttests that are as similar as possible. The more certain the researcher is of the consistency between the pretest and the posttest, the less likely that results are influenced by the pretest or posttest that was taken.

One important method of preparing the pretest and posttest is the pilot study. After developing an initial version of a pretest and posttest, the researcher must spend considerable time piloting the instruments (something we did not do enough). The goals of the pilot studies should be to ensure the two qualities mentioned above, discrimination and comparability. In order to assure discrimination, researchers need to have pilot subjects from all experience levels.

By piloting the test on subjects of various experience levels, the researcher can determine if the performance of the subjects on the pretest and/or posttest is different and if the pretest/posttest correctly identifies the high experience and low experience subjects. On the other hand, to ensure the comparability of the pretest and the posttest, multiple subjects from various experience levels are needed. Each of the subjects should be given both the pretest and the posttest (with the order permuted among the subjects) to determine if each group of subjects achieves the same score for both tests.

Finally, a fourth area for future work concerns the effect of reading techniques on specific decision points in the defect detection process. Figure 6 shows the potential effect of PBR on the decision tree presented earlier in Figure 2. The primary goal of the PBR techniques is to increase the number of issues that are seen by several reviewers by having them concentrate on the important information for different stakeholders of the requirements (The top oval in Figure 6). One possible detrimental result, as evinced by the responses mentioned above, is that some defects may not be reported because the reviewer may feel it is not his responsibility (The bottom oval). In addition, by concentrating on one stakeholder's view of the system, items that may have been considered defects by another stakeholder may not be considered defects by the current stakeholder (The middle oval).

After analysis and discussion of the results of this study, there still remain a series of open questions.

1) Does the expertise of a subject affect the amount of detail he or she needs in a technique to be effective?
2) Which types of experience are the important ones to focus on?
3) How can subjects be accurately classified based on their experience?
4) Is there a consensus about what constitutes a requirements defect?
5) Can effective pretest and posttests be developed to help quantify experience levels and evaluate the effectiveness of training sessions?

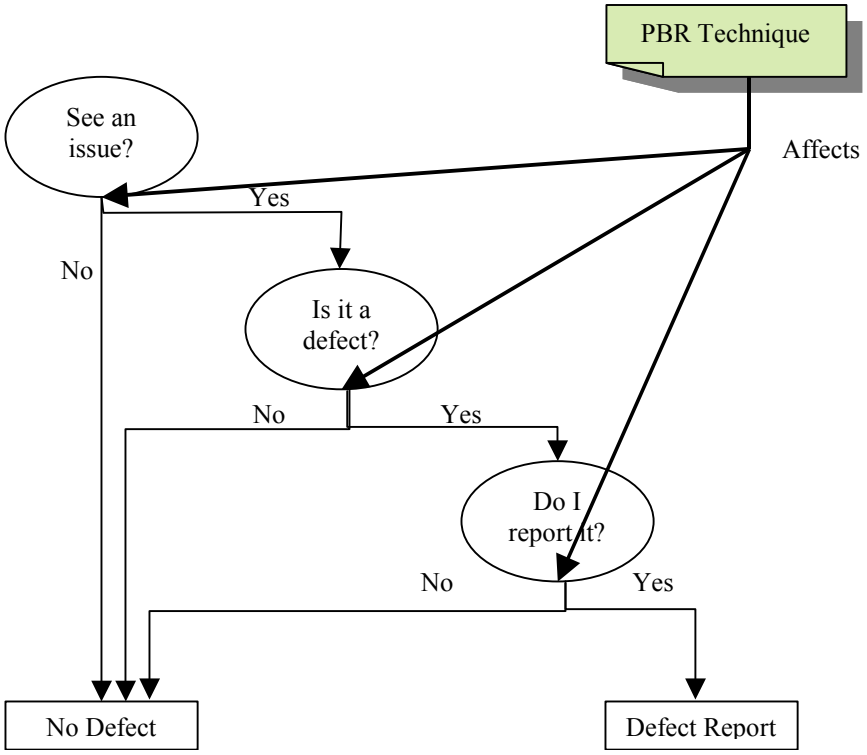

Figure 6 – Decision Tree with PBR

7. ACKNOWLEDGEMENTS

We would like to thank the students of CMSC735 Fall 2002 for participating in the study and providing valuable feedback throughout. We acknowledge support from the NSF Readers' Project (CCR-9900307).

8. REFERENCES

[Basili87] Basili, V., and Selby, R. 1987. Comparing the Effectiveness of Software Testing Strategies. IEEE Transactions on Software Engineering. 13(12): 1278-1296.

[Basili94] Basili, V.R., Caldiera, G., and Rombach, H.D. "Goal Question Metric Paradigm." *Encyclopedia of Software Engineering*. John Wiley & Sons, 1994. 528-532.

[Basili96] Basili, V.R., Green, S., Laitenberger, O., Lanubile, F., Shull, F., Sorumgard, S., Zelkowitz, M.V., "The Empirical Investigation of Perspective-Based Reading," *Empirical Software Engineering – An International Journal*, vol. 1, no. 2, 1996.

[Boehm01] Boehm, B. and Basili, V. "Software Defect Reduction Top 10 List." *IEEE Computer*, 34(1): 135-137, 2001.

[Carver03] Carver, J.C. "The Impact of Background and Experience on Software Inspections." *Ph.D. Dissertation*, University of Maryland, 2003. Also available as University of Maryland, Department of Computer Science Technical Report CS-TR-4476.

[Fagan76] Fagan, M.E., "Design and code inspections to reduce errors in program development." *IBM Systems Journal*. 15(3). 1976. 182-211.

[Kelly92] Kelly, J., Sharif, J, and Hops, J. "An Analysis of Defect Densities Found During Software Inspections." *Journal of Systems and Software*, Feb. 1992, pp. 111-117.

[Laitenberger00] Laitenberger, O., Atkinson, C., Schlich, M. and El Emam, K. "An Experimental Comparison for Reading Techniques for Defect Detection in UML Design Documents." *Journal of Systems and Software*, 53(2), August 2000, 183-204.

[Laitenberger01] Laitenberger, O., El Emam, K., and Harbich, T.. 2001. An Internally Replicated Quasi-Experimental Comparison of Checklist and Perspective-based Reading of Code Documents. IEEE Transactions on Software Engineering. 27(5): 387-421.

[Messick95] Messick, S. (1995). Validity of psychological assessment: Validation of inferences from persons' responses and performances as scientific inquiry into score meaning. American Psychologist, 50, 741-749.

[Ostrand88] Ostrand, T.J. and Balcer, M.J. "The Category Partition Method for Specifying and Generating Functional Tests." CACM 31(6): 676-686. June 1988.

[Parnas85] Parnas, D.L., and Weiss, D.M. "Active Design Reviews: Principles and Practice." *Proc of 8th International Conference on Software Engineering*, 1985, p132-136.

[Porter95] Porter, A., Votta, L., and Basili, V. 1995. Comparing Detection Methods for Software Requirements Inspections: A Replicated Experiment. IEEE Transactions on Software Engineering 21(6): 563-575.

[Porter97a] Porter, A.A., and Johnson, P.M. "Assessing Software Review Meetings: Results of a Comparative Analysis of Two Experimental Studies." *IEEE Transactions on Software Engineering*. Vol. 23, no. 3. March 1997. pp 129-145.

[Porter97b] Porter, A.A., Siy, H.P. and Votta, L.G. "Understanding the Effects of Developer Activities on Inspection Interval." In *Proceedings of the 19th International Conference on Software Engineering*. 1997. Boston, MA.

[Powers02] Powers, D. E. "Self-assessment of Reasoning Skills." Educational Testing Service. Research Report RR-02-22. 2002.

[Sauer00] Sauer, C., Jeffery, D.R., Land, L., and Yetton, P. "The Effectiveness of Software Development Technical Reviews: A Behaviorally Motivated Program of Research." *IEEE Transactions on Software Engineering*. SE-26(1) : 1-14.

[Schneider92] Schneider, G.M., Martin, J., and Tsai, W.T. "An Experimental Study of Fault Detection in User Requirements Documents." *ACM Transactions on Software Engineering and Methodology* 1(2): 188-204.

[Selby87] Selby, R., Basili, V., and Baker, T. "Cleanroom Software Development: An Empirical Evaluation." *IEEE Transactions on Software Engineering*, 13(9): 1027-1037

[Shull98] Shull, F. Developing Techniques for Using Software Documents: A Series of Empirical Studies. PhD Thesis, Computer Science Dept., University of Maryland. 1998.

[Shull00] Shull, F., Rus, I., and Basili, V. "How Perspective-Based Reading Can Improve Requirements Inspections." *IEEE Computer*, 33(7), July 2000, p. 73-79.

[Shull01] Shull F., Carver, J. and Travassos, G. H. 2001. An Empirical Methodology for Introducing Software Processes. Proc. European Software Engineering Conference, Vienna, Austria, 288-296.

[Shull02] Shull, F., Basili, V., Carver, J., Maldonado, J., Travassos, G., Mendoca, M., and Fabbri, S. "Replicating Software Engineering Experiments: Addressing the Tacit Knowledge Problem." In *Proceedings of the 2002 International Symposium on Empirical Software Engineering*, p. 7-16, Nara, Japan. 2002.

[Siy96] Siy, H.P. "Identifying the Mechanisms Driving Code Inspection Cost and Benefits." *Ph.D. Dissertation, University of Maryland*, 1996.

[Trochim00] Trochim, W. *The Research Methods Knowledge Base, 2nd Edition*. Internet WWW page, at URL: trochim.human.cornell.edu/kb/index.html (version current as of August 02, 2000).

[Wood97] Wood, M., Roper, M., Brooks, A., and Miller, J. 1997. Comparing and Combining Software Defect Detection Techniques: A Replicated Empirical Study. Proc. 6th European Software Engineering Conference/5th ACM SIESOFT Symposium on the Foundations of Software Engineering, Zurich, Switzerland, 262-277.

[Votta93] Votta, L.G., "Does Every Inspection Need a Meeting?" *Proceedings of ACM SIGSOFT'93 Symp. Foundations of Software Engineering*, Assoc. for Computing Machinery, Dec. 1993.

[Zhang99] Zhang, Z.; Basili, V.; and Shneiderman, B. 1999. Perspective-based Usability Inspection: An Empirical Validation of Efficacy. Empirical Software Engineering. An International Journal 4(1): 43-70.

Appendix A

Experience Questionnaire

Name _____

General Background

Please estimate your English-language background:
__ I am a native speaker.
__ English is my second language. [Please complete both of the following.]
My reading comprehension skills are:
__ low
__ medium
__ high
My listening and speaking skills are:
__ low
__ medium
__ high

What is your previous experience with software development in practice? (Check the bottom-most item that applies.)
__ I have never developed software.
__ I have developed software on my own.
__ I have developed software as a part of a team, as part of a course.
__ I have developed software as a part of a team, in industry one time.
__ I have worked on multiple projects in industry.

Please explain your answer. Include the number of semesters or number of years of relevant experience. (E.g. "I worked for 10 years as a programmer in industry"; "I worked on one large project in industry"; "I developed software as part of a class project"; etc…)

_____
_____

Software Development Experience

Please rate your experience in this section with respect to the following 5-point scale:
(Please include any relevant comments below each section)
1 = No experience; 2 = learned in class or from book; 3 = used on a class project;
4 = used on one project in industry; 5 = used on multiple projects in industry

Experience with Lifecycle Models

- Experience with the *Waterfall Model* 1 2 3 4 5
- Experience with the *Spiral Model* 1 2 3 4 5
- Experience with the *Incremental Development Model* 1 2 3 4 5

Comments:

Experience with Requirements

- Experience writing requirements 1 2 3 4 5
- Experience interacting with users to write requirements 1 2 3 4 5
- Experience writing use cases 1 2 3 4 5
- Experience reviewing requirements 1 2 3 4 5
- Experience reviewing use cases 1 2 3 4 5
- Experience changing requirements for maintenance 1 2 3 4 5

Comments:

Experience in Design

- Experience in design of systems 1 2 3 4 5
- Experience in design of systems from requirements/use cases 1 2 3 4 5
- Experience with creating Object-Oriented (OO) designs 1 2 3 4 5
- Experience with creating Structured Designs 1 2 3 4 5
- Experience with reading OO designs 1 2 3 4 5
- Experience with the Unified Modeling Language (UML) 1 2 3 4 5
- Experience changing designs for maintenance 1 2 3 4 5

Comments:

Experience in Coding

- Experience in coding, based on requirements/use cases 1 2 3 4 5
- Experience in coding, based on design 1 2 3 4 5
- Experience in coding, based on OO design 1 2 3 4 5
- Experience in maintenance of code 1 2 3 4 5

Comments:

Experience in Testing

- Experience in testing software 1 2 3 4 5
- Experience in testing, based on requirements/use cases 1 2 3 4 5
- Experience with Unit Testing 1 2 3 4 5
- Experience with Integration Testing 1 2 3 4 5
- Experience with System Testing 1 2 3 4 5
- Experience with Acceptance Testing 1 2 3 4 5
- Experience with _____ (fill in name of a technique that you are familiar with) 1 2 3 4 5
- Experience with _____ (fill in name of a technique that you are familiar with) 1 2 3 4 5

Comments:

Other Experience

- Experience with software project management? 1 2 3 4 5
- Experience with software inspections? 1 2 3 4 5

Comments:

Experience in Problem Domains

We will use answers in this section to understand how familiar you are with various systems we may use as examples or for assignments during the class.

Please rate your experience in this section with respect to the following 3-point scale:
1 = I am really unfamiliar with the concept. I have never done it.
3 = I have done this a few times, but I am not an expert.
5 = I am very familiar with this area. I would be very comfortable doing this.

How much do you know about:

- Applying for a loan or mortgage? 1 3 5
- Using a parking garage? 1 3 5
- Using an ATM? 1 3 5
- Renting movies from a video rental store (e.g. Blockbusters)? 1 3 5

Appendix B – PBR Techniques

Perspective Based Reading –Version 1 (High level)

Perspective based reading is the concept that the various stakeholders of a document should read it to find out if their needs are satisfied by the document. In doing so, it is hoped that the reader will find defects and be able to assess the document from their particular point of view.

Test-Based Reading Technique

For each requirement or functional specification, generate a test case or set of test cases that allow you to ensure that an implementation of the system satisfies the requirement or functional specification.

Inputs: **A set of requirements or functional specifications**

Outputs: **An initial list of test cases**
A list of defects in the requirements

Use any standard testing approach that you are familiar with, to generate a set of test cases incorporating test criteria into the test suite. In doing so, ask yourself the following questions about each test case:

1. Do you have all the information necessary to identify the item being tested and the test criteria? Can you generate a reasonable test case for each item based upon the criteria?

2. Can you be sure that the tests generated will yield the correct values in the correct units?

3. Are there other interpretations of this requirement that the implementer might make based upon the way the requirement or functional specification is defined? Will this affect the tests you generate?

4. Is there another requirement or functional specification for which you would generate a similar test case but would get a contradictory result?

5. Does the requirement or functional specification make sense from what you know about the application or from what is specified in the general description?

Perspective Based Reading –Version 2 (Step-by-step)

Perspective based reading is the concept that the various stakeholders of a document should read it to find out if their needs are satisfied by the document. In doing so, it is hoped that the reader will find defects and be able to assess the document from their particular point of view.

Reading Technique for Category Partition Testing

Generate a set of test cases that allows you to ensure that an implementation of the system satisfies the requirements. Follow the procedure below to generate the test cases, using the questions provided to identify defects in the requirements.

Inputs: **A set of new requirements**
Output: **An initial list of test cases**
A list of defects in the requirements

1) Read through the requirements and identify *functional units*, which are either
   - Top-level user commands; **OR**
   - Functions described that may be called by other functions

   And record them on the Form.

   Q1.1 Does the functional unit make sense from what you know about the application or from what is specified in the general description?

   Q1.2 If you are using a top-level user command, are all the necessary subpieces adequately described?

   a) For each functional unit identify the *parameters* (inputs and outputs) and *environmental conditions* (system states at time of operation), and record them on the form.

      Q2.1 Do you have all the information necessary to identify the inputs to the functional unit? Based on the general requirements and your domain knowledge, are these inputs correct for this functional unit?

      Q2.2 Has the description of any of the necessary inputs been omitted?

      Q2.3 Are any inputs specified which are not needed?

      Q2.4 Do you have all the information necessary to identify the outputs of the functional unit?

      Q2.5 Do the inputs and outputs make sense from what you know about the application domain and from the general description of the system?

   b) Then for each of the parameters and environmental conditions either:
      i) Identify the *choices* (possible values) and group them into *categories* (major properties or characteristics) for each of these parameters or environmental conditions; **OR**
      ii) Identify the *categories* and then enumerate the specific *choices* within each one.

      And record them on the form.

      Q3.1 Can you identify specific choices for each parameter and environmental condition?
      Q3.2 Do you have enough information to identify or group the choices into categories?
      Q3.3 Do the requirements indicate that a choice belongs in more than one (mutually-exclusive) category?
      Q3.4 Are there other categories of inputs that are not described by the requirements but should be?

2) Based on the choices defined above, create a set of test cases, using the form, for each functional unit such that all combinations of choices are covered. Eliminating combinations that represent impossible situations may reduce the number of test cases.

Q4.1 Do you have enough information to create the necessary test cases?
Q4.2 Are there other interpretations of the functional unit than an implementer might make based on the description?
Q4.3 Is there another functional unit for which you would generate a similar test case, but would get a contradictory result?
Q4.4 Can you be sure that the tests generated will yield the correct output?

Acknowledgements:

These guidelines are based on information found in:

Ostrand, T.J. and Balcer, M.J. "The Category Partition Method for Specifying and Generating Functional Tests." CACM 31(6): 676-686. June 1988.

Appendix C – Post Experiment Questionnaire 1

Post-Experiment Questionnaire Version 1 (High level procedure)

Name: _____

Please note that your answers on this questionnaire will *not* affect your grade in any way. These are questions we need to know in order to make effective use of the data from the study.

1. Training

How effective did you think the training was? Did it help you understand the procedures? Was there anything that was missing or could have been done better?

Describe in detail the method that you used to develop your test cases. If you used a predefined method, give the name of that method.

2. The Techniques

What did you think of the level of detail provided to you in the reading technique? Was there enough information for you to do your job? Was there too much information? Explain.

How would you change the technique to make it more effective?

Did you have any problems using the technique?

Would you use the technique again? Explain.

3. Executing the techniques in inspections

Was there background or training other than what you received in class that you needed to apply PBR?

Were the defect classes well defined? Useful? Complete? Why/Why not?

Did you find defects that you didn't report? Why/Why not?

Post-Experiment Questionnaire Version 2 (Step-by-step procedure)

Name: _____

Please note that your answers on this questionnaire will *not* affect your grade in any way. These are questions we need to know in order to make effective use of the data from the study.

1. Training

Had you learned about or used Category Partition Testing prior to this class? Explain?

How effective did you think the training was? Did it help you understand the procedures? Was there anything that was missing or could have been done better?

Did you get a clear understanding of the Category Partition Testing technique from the training? What could have been done better? Would you use Category Partition Testing again? Explain.

2. The Techniques

What did you think of the level of detail provided to you in the reading technique? Was there enough information for you to do your job? Was there too much information? Explain.

How would you change the technique to make it more effective?

Did you have any problems using the technique?

Would you use the technique again? Explain.

3. Executing the techniques in inspections

Was there background or training other than what you received in class that you needed to apply PBR?

Were the defect classes well defined? Useful? Complete? Why/Why not?

Did you find defects that you didn't report? Why/Why not?

Appendix D – Questionnaire results

This questionnaire dealt with issues surrounding the training, the techniques themselves, and the execution of the techniques. The table below presents a summary of the answers provided by the subjects. To analyze the subjects' answers to the questionnaire, the idea of constant comparison used in grounded theory was applied. To do the analysis, some specific answer categories logically related to each question were created *a priori*. Then the answers from the subjects were analyzed and placed into the predetermined categories. If necessary, the categories were refined by adding, removing, or creating sub-categories based on the particular answers provided by the subjects. Table 6 presents a summary of how many subjects fell into each category. If a particular question did not apply to a group of subjects, then the table entry in that column is shaded. Particularly interesting observations from the table are highlighted below.

		Group 1	**Group 3**	**Group 4**
Perspective Experience		High	Low	Low
Process Detail		No Model	Model included	No Model
Model		Own	Category Partition Testing	Own
Number of Subjects		6	10	6
Training Questions	**Answer Categories**			
How effective was the training?	Helpful	3	5	3
	Adequate	1	4	1
	Insufficient	1	1	1
Did the training help you understand the procedures?	Yes	3	2	2
	No	0	1	0
Was anything missing?	Details on CPT		5	
	Test Case creation details	2	0	2
	Examples	4	2	2
Did you get a clear understanding of CPT?	Yes		3	
	No		5	
What could have been done better on CPT?	More examples		5	
Would you use CPT again?	Yes		7	
	No		2	
Technique Questions	**Answer Categories**			
How was the level of detail in the technique?	Enough	2	8	4
	Not Enough	4	2	2
How would you make the technique more effective?	None	0	2	2
	Add details about CPT		7	
	Change details of Technique	2	0	2
	Add details about test cases	3	0	0
	Other	1	1	0
Did you have problems using the technique?	No	3	2	1
	Yes – with CPT		6	
	Yes – with the questions	1	1	0
	Yes – with the domain	0	1	0
	Yes – with creating test cases	1	0	3
	Yes - other	0	0	0
Would you use the technique again?	No	0	1	2
	Yes	5	7	3
	Yes, in some cases	1	2	0

Execution Questions	Answer Categories			
Was other background or training needed?	No	0	5	2
	Yes – on CPT		1	
	Yes – on Testing	2	0	1
	Yes – on Test cases	3	0	2
	Yes – on the Domain	1	1	0
	Yes – on the Technique	1	2	1
Were the defect classes well defined?	Yes	5	8	3
	Yes, except for 1	0	2	1
Were the defect classes useful?	Yes	5	7	6
	No	1	0	0
Were the defect classes complete?	Yes	6	5	3
	No	0	1	2
Did you find defects you did not report?	No	4	7	3
	Yes – because of assumptions	1	1	1
	Yes – because of domain	0	1	2
	Yes – no specific reason	1	1	0

Appendix E – Post Experiment Questionnaire 2
Understanding Requirements

The following questions are related to, but separate from, the reading technique experiment you have just finished. Some of the questions ask for answers that are unrelated to testing. Please answer them to the best of your abilities, but take no more than 30 minutes to answer all the questions. There are no right or wrong answers.

1) Requirements can be ordered and formatted in different ways. Did the functional specification style of this document help or hurt your understanding of the static structure of the desired system? Did it help or hurt your understanding of the desired behavior of the system? Would another style of presentation such as use cases, system diagram, or narrative have been better?

2) A requirements document must define the boundary of a system and provide enough details to design the system. Many defects are related to problems of scope and level of detail. Did these requirements provide you with enough information to define the scope of the system? Were there enough details to start designing the system? Or were there so many that they overly constrain the system? Would addressing the concerns you raised in your defect list fix the problems you mentioned here?

3) By reviewing this requirements document did you learn anything new about parking garage systems in general? Did anything about this parking garage conflict with your prior understanding of parking garage systems? Did you gain any deeper insights into the needs of such a system? Please give an example for each question or answer 'none.'

4) Please add any other comments you might have on how your own experience, the reading technique and this particular requirements document might make this a hard, or easy, project to work on.