Evolving and Packaging Reading Techniques Through Experimentation

Victor R. Basili
Experimental Software Engineering Group

Institute for Advanced Computer Studies
and
Department of Computer Science
University of Maryland
and
Fraunhofer Center For Experimental Software Engineering Maryland

Evolving Knowledge
Model Building, Experimenting, and Learning

Understanding a discipline involves building models,
e.g., application domain, problem solving processes

Checking our understanding is correct involves
- testing our models
- experimentation

Analyzing the results of the experiment involves learning, the encapsulation of knowledge and the ability to change and refine our models over time

The understanding of a discipline evolves over time

Knowledge encapsulation allows us to deal with higher levels of abstraction

This is the paradigm that has been used in many fields,
e.g., physics, medicine, manufacturing.
What do these fields have in common?
They evolved as disciplines when they began applying the cycle of model building, experimenting, and learning.
Began with observation and the recording of what was observed.
Evolved to manipulating the variables and studying the effects of change in the variables.

What are the differences of these fields?
The objects they study, the properties of those object, the properties of the system that contain them, the relationship of the object to the system, and the culture of the discipline.
This effects how models are built and analyzed, how experimentation gets done.

Like other disciplines, software engineering requires the cycle of model building, experimentation, and learning.
The study of software engineering is a laboratory science.
We need to understand the nature of the processes, products and the relationship between the two in the context of the system.
Research and Development have a symbiotic relationship.
Research needs laboratories to observe & manipulate the variables - they only exist where developers build software systems.
Development needs to understand how to build systems better - research can provide models to help.
The Experimental Discipline

Classes of Experimental Studies

Experiment Classes

<table>
<thead>
<tr>
<th>#Projects</th>
<th>One</th>
<th>More than one</th>
</tr>
</thead>
<tbody>
<tr>
<td># of Teams</td>
<td>One</td>
<td>Single Project</td>
</tr>
<tr>
<td>per Project</td>
<td></td>
<td>Variation</td>
</tr>
<tr>
<td>More than one Project</td>
<td></td>
<td>Replicated Project</td>
</tr>
<tr>
<td>one</td>
<td></td>
<td>Subject-Project</td>
</tr>
</tbody>
</table>

Sign of maturity in a field:
level of sophistication of the goals of an experiment understanding interesting things about the discipline

For software engineering that might mean:

Can we build models that allow use to measure and differentiate processes and products?

Can we measure the effect of a change in a particular process variable on the product variable?

Can we predict the characteristics of a product (values of product variable) based upon the model of the process (values of the process variables), within a particular context?

Can we control for product effects, based upon goals, given a particular set of context variables?
The Experimental Discipline

Sign of maturity in a field:
- a pattern of knowledge built from a series of experiments

Does the discipline build on prior (knowledge, models, experiments).

Was the study an isolated event?

Did it lead to other studies that made use of the information obtained from it?

Did the building of knowledge exist in one research group or environment, or has it spread to others - researchers building on each other’s experimental work?

For example, inspections, in general, are well studied experimentally.

However, there has been very little combining of results, replication, analysis of the differentiating variables.

Reading Techniques

Reading is a key technical activity for analyzing and constructing software artifacts.

Reading is a model for writing.

Reading is critical for reviews, maintenance, reuse, ...

What is a reading technique?
- a concrete set of instructions given to the reader saying how to read and what to look for in a software product.

More specifically, software reading is
- the individual analysis of a software artifact
  - e.g., requirements, design, code, test plans
- to achieve the understanding needed for a particular task
  - e.g., defect detection, reuse, maintenance
Reading Techniques

Early experiments (Hetzel, Meyers) showed very little difference between reading and testing.

But reading was simply reading, without a technological base.

We discuss a series of experiments at the University of Maryland and at NASA used to learn about, evaluate, and evolve reading techniques.

This example:
- shows multiple experimental designs
- provides a combination of evaluation approaches
- offers insight into the effects of different variables on reading.

The experiments start with
- the early reading vs. testing experiments
- to various Cleanroom experiments
- to the scenario based reading techniques currently under study.

EXPERIMENTAL LEARNING MECHANISMS

Series of Studies

<table>
<thead>
<tr>
<th># of Teams per Project</th>
<th># Projects</th>
</tr>
</thead>
<tbody>
<tr>
<td>One</td>
<td>One</td>
</tr>
<tr>
<td>More than one</td>
<td>More than one</td>
</tr>
<tr>
<td>3. Cleanroom</td>
<td>4. Cleanroom</td>
</tr>
<tr>
<td>(SEL Project 1)</td>
<td>(SEL Projects, 2,3,4,...)</td>
</tr>
<tr>
<td>2. Cleanroom at Maryland</td>
<td>1. Reading vs. Testing</td>
</tr>
<tr>
<td></td>
<td>5. Scenario reading vs. ...</td>
</tr>
</tbody>
</table>
Blocked Subject Project Study

Testing/Reading Strategies Comparison

Code Reading vs Functional Testing vs Structural Testing

Study: fault detection effectiveness, cost, classes of faults detected
Experimental design: Fractional factorial design at NASA/CSC

Some Results

Code reading (by stepwise abstraction) more effective than functional testing (equivalence partitioning)
efficient than functional or structural testing (100% stmt coverage)
Different techniques more effective for different defect classes

Developers don’t believe reading is better, not motivated to read

---

Blocked Subject Project Study

Testing/Reading Strategies Comparison

Fractional Factorial Design

<table>
<thead>
<tr>
<th>Advanced Subjects</th>
<th>Code Reading</th>
<th>Functional Testing</th>
<th>Structural Testing</th>
</tr>
</thead>
<tbody>
<tr>
<td>S1 S2</td>
<td>P1 P2 P3</td>
<td>P1 P2 P3</td>
<td>P1 P2 P3</td>
</tr>
<tr>
<td>S3</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>S4</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Intermediate Subjects</th>
<th>Code Reading</th>
<th>Functional Testing</th>
<th>Structural Testing</th>
</tr>
</thead>
<tbody>
<tr>
<td>S5 S6 S7</td>
<td>P1 P2 P3</td>
<td>P1 P2 P3</td>
<td>P1 P2 P3</td>
</tr>
<tr>
<td>S9 S10</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>S11</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>S12</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Junior Subjects</th>
<th>Code Reading</th>
<th>Functional Testing</th>
<th>Structural Testing</th>
</tr>
</thead>
<tbody>
<tr>
<td>S18 S19 S20</td>
<td>P1 P2 P3</td>
<td>P1 P2 P3</td>
<td>P1 P2 P3</td>
</tr>
<tr>
<td>S20 S21 S22</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
</tbody>
</table>

Blocking by experience level and program tested
Replicated Project Study

Cleanroom Study

Cleanroom process vs. non-Cleanroom process
Study: effects on the process, product, developers
Experimental design: 15 three-person teams at UMD

Some Results
Cleanroom developers were motivated to read better
Reading by step-wise abstraction more effective and efficient
Does Cleanroom scale up? Will it work on a real project?

Single Project Study

Cleanroom in the SEL

Cleanroom process vs. Standard SEL Approach
Study: effects on the effort distribution, cost, and reliability
Experimental design: Flight Dynamics project in the SEL

Some Results
Reading by step-wise abstraction effective and efficient
Reading appears to reduce the cost of change
Better training needed for reading by stepwise abstraction
Will it work again? Can we scale up more?
Multi-Project Analysis Study

Cleanroom in the SEL

Revised Cleanroom process vs. Standard SEL Approach

Study: effects on the effort distribution, cost, and reliability

Experimental design: Three Flight Dynamics projects in the SEL

Some Results

Reading by step-wise abstraction
- effective and efficient in the SEL
- appears to reduce the cost of change

Better training needed for reading by stepwise abstraction

Better reading techniques needed for other documents, e.g., requirements, design, test plan

Can we improve the reading techniques for requirements and design documents?

Scenario-Based Reading Definition

An approach to generating a family of reading techniques, called operational scenarios, has been defined to be
- procedurally defined
- document and notation specific
- goal driven
- tailorable to the project and environment
- focused to provide a particular coverage of the document
- empirically verified to be effective for its use
- usable in existing methods, such as inspections

So far, four techniques have been defined and studied:

- perspective based reading: for requirements documents
- defect based reading: for requirements documents
- scope based reading: for design reuse
- use based reading: for user interface design

All techniques have been applied in a series of experiments
\textbf{Scenario-Based Reading Definition}

- Need to characterize the “model of use”: how the information in a document is used for a particular task in a particular environment.

\begin{itemize}
  \item Abstractions of Information: A model of what information is important, and how it is best organized
  \item Uses of Information: A model of the process by which the task is accomplished
\end{itemize}

\begin{itemize}
  \item Initial procedures for identifying information in this document that is relevant
  \item Initial procedures for using the information to accomplish the task
\end{itemize}

\textbf{Reading Technique:}
Tested practices for accomplishing a particular task

\textbf{Reading for Analysis: Scenario-Based Reading Definition}

So far, two different scenario-based reading techniques have been defined for requirements documents:
- defect based reading
- perspective based reading

\textbf{Defect based reading} focuses on different defect classes
  e.g., missing functionality and data type inconsistencies
  Existing definition for reading SCR style documents

\textbf{Perspective-based reading} focuses on different customer perspectives, e.g., reading from the perspective of the designer, tester, or end-user
  Existing definition for reading natural language requirements documents.
**Perspective-Based Reading**

**Abstractions of Information:**
- Design plan
- Test plan
- User manual

**Uses of Information:**
- Check consistency
- Check completeness
- Check correctness . . .

**Reading Technique:**
- For detecting defects in requirements
- Allow reviewers to use their usual procedures
- Create questions aimed at checking each attribute
- Ask reviewers to create the appropriate abstraction, then answer a series of questions tailored to the abstraction

---

**Reading for Analysis: Blocked Subject Project Study**

**Perspective-Based Reading**

**Study Goal:**
Analyze perspective-based reading, NASA’s current reading technique to evaluate and compare them with respect to their effect on fault detection effectiveness in the context of an inspection team from the viewpoint of quality assurance.

**Environment:**
- NASA/CSC SEL Environment
- Requirements documents:
  - Two NASA Functional Specifications: ground support subsystems
  - Two Structured Text Documents: ATM machine, Parking Garage
- All documents seeded with known defects

**Experimental design:**
- Metric: Percent of defects detected
- Blocked subject-project: Partial factorial design
- Replicated twice: November 1994 and June 1995
- Subjects: 25 subjects in total
Reading for Analysis: Perspective-Based Reading Experiment

Major Results

PBR most successful in the generic domain
both at the individual and team level
caught more defects
captured different defects depending on the perspective

PBR not sufficiently tailored to the NASA environment in terms of
document contents, notation and perspectives
PBR successful at the team level for only one of the NASA documents
caught more defects
did not catch significantly different defects

The relative benefit of PBR is higher for teams

Potential Improvements

Be more specific to avoid subjects falling back to familiar technique
Tailor PBR to the domain to achieve full potential

---

Goal of Perspective-Based Reading (PBR):
detect defects in a requirements document
focus on product consumers

Controlled experiment run twice with NASA professionals:

- **Pilot Study**
- **Main Study**

*Team Detection Rate*

- local tech.
- PBR

- **generic docs.**
- **flight dyn. docs.**
Defect-Based Reading

Study Goal:
Analyze defect-based reading, ad-hoc reading and check-list based reading to evaluate and compare them with respect to their effect on fault detection effectiveness in the context of an inspection team from the viewpoint of quality assurance.

Environment:
University of Maryland graduate courses
Requirements documents written in SCR notation
Water Level Monitoring System, Cruise Control System

Experimental design:
Blocked subject-project: Partial factorial design
Replicated twice
Subjects: 48 subjects in total

Major Results

- Scenario readers performed better than Ad Hoc and Checklist Readers improvement of about 35%

- Scenarios helped reviewers focus on specific fault classes but were no less effective at detecting other faults

- Checklist reading was no more effective than Ad Hoc reading
Reading for Analysis: Defect-Based Reading Experiment

Goal of Defect-Based Reading (DBR):
detect defects in a requirements document
focus on defect classes

Controlled experiment run twice with UMD graduate students:

<table>
<thead>
<tr>
<th>Team Detection Rate</th>
<th>Ad Hoc</th>
<th>Checklist</th>
<th>DBR</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.4</td>
<td>0.5</td>
<td>0.6</td>
<td></td>
</tr>
</tbody>
</table>

Reading for Analysis: Use-Based Reading

Definition

Perspective-based reading for user interface analysis

Used three different perspectives:
expert-based reading
novice-based reading
error-based reading

Based upon perspectives, provide a set of questions that needed to be tailored to the particular interface being inspected

This provides each reader with an inspection procedure

The reader should then find anomalies and assess the document from their particular point of view.
Use-Based Reading

Study Goal:
Analyze use-based reading and heuristic evaluation to evaluate and compare them with respect to their effect on anomaly detection effectiveness in the context of an inspection team from the viewpoint of quality assurance.

Environments:
- Bureau of Standards (24), 2 web interfaces for census data collection
- HCI undergraduate/graduate class (44), studied a web site
- HCI professional SE class (42), studied same web site

Experimental designs:
- Dependent variable: number of detected anomalies of different types (expert, novice, error)
- Independent variables: procedure (UBR, HE) procedure application (alone, pair)
- Replicated experiment, random selection wrt dependent variables

Major Results
Use-based (Perspective-based) reading, compared to heuristic evaluation did not require more inspection time, got a better or equivalent preference rating by performers, was more effective in detecting related anomalies, overall found more anomalies at individual level, for paired application of the procedures, and for simulated teams.

The effectiveness was consistently shown with different subjects, interfaces, and time constraints.

Paired teams performed better than individuals.
High Level Reading Goals

We differentiate two goals for reading techniques:

**Reading for analysis:**
Given a document, how do I assess various qualities and characteristics?

- **Assess for**
  - product quality
  - defect detection

- **Useful for**
  - quality control, insights into development

**Reading for construction:**
Given a system, how do I understand how to use it as part of my new system?

- **Understand**
  - what a system does
  - what capabilities do and do not exist

- **Useful for**
  - maintenance
  - building systems from reuse

Reading for Construction

White-Box Frameworks

We proposed two reading techniques for frameworks:
Given the object model of your application and the OO framework

**System-wide technique:**
- Find the class in the framework hierarchy that best matches the functionality you are seeking
- Determine how to parameterize that class and how to implement it as part of your application

**Task-oriented technique:**
- Find the example in the example set that best matches the functionality you are seeking
- Determine which piece of the example is relevant and how to implement it as part of your application

Controlled Experiment with UMD students
**Reading for Construction**

**Some Results: White-Box Framework Experiment**

The effectiveness of an example-based technique is heavily dependent on the quality and breadth of the example set provided.

Example-based techniques are well-suited to use by beginning learners.

A hierarchy-focused technique is not well-suited to use by beginners.

Teams who began their implementation using an existing example for guidance seemed more effective than those who began implementing from scratch.

Teams who were able to stay close to their original object model of the system during implementation seemed more effective.

---

**Other work in Developing Techniques**

We are studying
other perspective-based techniques, e.g., use-case driven perspective
Does this perspective find defects not caught by other perspectives?
Do better defined PBR procedures provide better results?
object oriented design reading techniques
scenarios based upon defect classes (UMD)
scenarios based upon perspectives (Fraunhofer IESE)

Can use-case driven reading technique be used in the context of a product line to help generate generic use cases for the product line?
What support processes and tools are necessary?

What other families of techniques can we develop based on empirical evaluation parameterized for use in different contexts
Reading Techniques for OO Design

- Target Artifacts:
  - Requirements Specification
  - Use-Cases
  - High Level Design
    - Class Diagrams
    - Class Descriptions
    - State Machine Diagrams
    - Interaction Diagrams

Vertical reading
Horizontal reading

---

Reading Techniques for OO Design

Technology
- General Goal
- Specific Goal
- Incremental (Iterate)

Solution
Family
Technique

PROBLEM SPACE
- Reading
  - Analysis
  - Defect Detection
  - Usability

Design
- Requirements Code
- User Interface
- SCR
- English
- Screen Shot

OO Diagrams
- Traceability
  - Horizontal
  - Vertical
- Defect-based
  - Omission
  - Inconsistency
- Incorrect
- Developer-Tester
  - User-based

SOLUTION SPACE
- Defect-based
  - Expert Novice Error

---
Reading Techniques for OO Design
First Experiment at UMCP

• Some results from the experiment:
  – developers agreed that using some kind of OO reading technique is worthwhile
  – some developers said that they would like to use the same techniques again but, the mechanisms used to instrument them should be improved. The study allowed us to identify weaknesses in the first version of the techniques that have led to a second version
  – It was possible to demonstrate that reading techniques can be used as part of design inspections, and do help reviewers detect defects

Refining a High Level Focus

Object of Study

Focus

Process

Effect on Product

Analysis

Requirements

Reading

Defect Detection

English
Families of Reading Techniques

**Reading Process:**
- Analyze reading techniques to evaluate their effectiveness on products from the point of view of the knowledge builder in the context of variable set

**Effect:**
- Class
- Goal

**Test Plan:**
- Code
- Design
- Requirements
- User Interface

**Product:**
- Type
- Notation

**Solution Space:**
- Scope Based
- Defect Based
- Perspective Based
- Usability Based

**Family:**
- Expert
- Novice
- Tester
- User
- Developer

**Technique:**
- Wide
- Oriented
- Ambiguity

**PROBLEM SPACE:**
- Construction
- Analysis
- Reuse
- Maintenance
- Defect Detection
- Usability

**SCR:**
- English
- Screen Shot
- Framework
- Code

**Project:**
- Code
- White Box
- Black Box

**Source:**
- Library

**System:**
- Wide
- Task

**Task:**
- Oriented
- Incorrect
- Omission
- Ambiguity
**Focused Families of Analysis Techniques**

G3 Analyze a set of processes focused to provide a particular coverage of an artifact to evaluate their ability to detect anomalies from the point of view of the knowledge builder in the context of (variable set).

**PROBLEM SPACE**

- Process/Analysis/Reading
- Anomaly Detection
- Requirements
- User Interface

**Object of Study**

- Focus
- Artifact
- Notation

**SOLUTION SPACE**

- Defect Based
- Perspective Based
- Usability Based

**Family**

- Expert
- Novice
- Error
- Tester
- User
- Developer

- SCR
- English
- Screen Shot
- Ambiguity
- Incorrect
- Omission
- Inconsistent

**Building Laboratory Manuals**

Laboratory manuals can be used to:
- Document processes
- Provide artifacts
- Offer a mix of experimental designs and analysis techniques
- Provide a basis for balancing threats to validity
- Support meta-analysis

Several laboratory manuals already exist:
- Reading vs. Testing
- Defect Based Reading
- Perspective Based Reading
- Framework Reading

Several experiments have been replicated under the same and differing contexts using these manuals.

Some progress has been made in doing meta-analysis.
Building Laboratory Manuals

ISERN

- organized explicitly to share knowledge and experiments
- has membership in the U.S., Europe, Asia, and Australia
- represents both industry and academia
- supports the publication of artifacts and laboratory manuals

It can be used to

- help define and replicate studies and techniques
- support the development of evaluation approaches for software engineering
- contribute to the laboratory manuals.

Conclusions from Experiments

- Able to combine the results of several experiments and build up our knowledge about software processes
  - We can effectively design and study techniques that are procedurally defined, document and notation specific, goal driven, and empirically validated for use
  - We can demonstrate that a procedural approach to a software engineering task could be more effective than a less procedural one under certain conditions (e.g., depends on experience)
  - A procedural approach to reading based upon specific goals will find defects related to those goals, so reading can tailored to the environment
  - et. al.
Conclusions about Knowledge Building

- Benefit to Researchers:
  - ability to increase the effectiveness of individual experiments
  - offers a framework for building relevant practical SE knowledge
  - provides a way to develop and integrate laboratory manuals
  - generate a community of experimenters

- Benefits to Practitioners:
  - offers some relevant practical SE knowledge
  - provides a better basis for making judgements about selecting process
  - shows importance of and ability to tailor “best practices”
  - provides support for defining and documenting processes
  - allows organizations to integrate their experiences with processes