Information and Software Technology 65 (2015) 14–38

Contents lists available at ScienceDirect

Information and Software Technology

journal homepage: www.elsevier.com/locate/infsof

How have we evaluated software pattern application? A systematic mapping study of research design practices

Maria Riaz [a,*], Travis Breaux [b], Laurie Williams [a]

[a] North Carolina State University, Dept. of Computer Science, Raleigh, NC, USA
[b] Carnegie Mellon University, Institute for Software Research, Pittsburgh, PA, USA

ARTICLE INFO

Article history:
Received 7 October 2014
Received in revised form 2 April 2015
Accepted 2 April 2015
Available online 9 April 2015

Keywords:
Software pattern
Mapping study
Systematic review
Empirical evaluation
Empirical design

ABSTRACT

Context: Software patterns encapsulate expert knowledge for constructing successful solutions to recurring problems. Although a large collection of software patterns is available in literature, empirical evidence on how well various patterns help in problem solving is limited and inconclusive. The context of these empirical findings is also not well understood, limiting applicability and generalizability of the findings.
Objective: To characterize the research design of empirical studies exploring software pattern application involving human participants.
Method: We conducted a systematic mapping study to identify and analyze 30 primary empirical studies on software pattern application, including 24 original studies and 6 replications. We characterize the research design in terms of the questions researchers have explored and the context of empirical research efforts. We also classify the studies in terms of measures used for evaluation, and threats to validity considered during study design and execution.
Results: Use of software patterns in maintenance is the most commonly investigated theme, explored in 16 studies. Object-oriented design patterns are evaluated in 14 studies while 4 studies evaluate architectural patterns. We identified 10 different constructs with 31 associated measures used to evaluate software patterns. Measures for 'efficiency' and 'usability' are commonly used to evaluate the problem solving process. While measures for 'completeness', 'correctness' and 'quality' are commonly used to evaluate the final artifact. Overall, 'time to complete a task' is the most frequently used measure, employed in 15 studies to measure 'efficiency'. For qualitative measures, studies do not report approaches for minimizing biases 27% of the time. Nine studies do not discuss any threats to validity.
Conclusion: Subtle differences in study design and execution can limit comparison of findings. Establishing baselines for participants' experience level, providing appropriate training, standardizing problem sets, and employing commonly used measures to evaluate performance can support replication and comparison of results across studies.

Contents

* Corresponding author.
E-mail addresses: mriaz@ncsu.edu (M. Riaz), breaux@cs.cmu.edu (T. Breaux), williams@csc.ncsu.edu (L. Williams).

http://dx.doi.org/10.1016/j.infsof.2015.04.002
0950-5849/

1. Introduction

While the availability of frameworks and tools enable developers to build more complex systems by reusing software components, software development remains a knowledge-intensive problem solving activity, largely relying on available skills and expertise [1]. Experts in any field are rare; capturing and sharing expert knowledge helps to raise the baseline performance of a novice (or non-expert) [2]. Software patterns are a popular approach for capturing and sharing expert knowledge for constructing successful solutions to recurring problems [3]. Despite the popular adoption of patterns in software engineering, how do we know that software patterns meet their goal of successfully capturing and sharing expert knowledge across the software development community to support problem solving? Applying software patterns during problem solving, much like most of software engineering, is a human-centered activity. The merit of a pattern can be assessed in terms of how successfully a practitioner employs a pattern to solve a particular commonly occurring problem. Empirical research evaluating the use of software patterns by software engineers (i.e., studies that involve human participants) can help in establishing whether a given pattern helps in problem solving.

The empirical evidence on how well various patterns help in problem solving is limited, often inconclusive, and merits further investigation. The context of empirical findings is also not well understood which limits applicability and generalizability of the findings. Furthermore, to lend validity to study findings and isolate confounding factors, empirical studies need to be replicated and synthesized. A clear description of the empirical protocol and a clear description of the context of the study is pre-requisite in replicating and synthesizing studies [4]. Systematic reviews, such as a mapping study, can help gather and categorize existing empirical research design practices, identify common research protocols, context factors and measures used to evaluate software patterns. Such information can be of use to researchers designing new studies or replicating existing studies.

To this end, the objective of this paper is to characterize the research design of empirical studies exploring software pattern application involving human participants.

We conducted a systematic mapping study [5] of existing literature on software pattern application in an empirical setting to aggregate existing research design practices. We characterize the research design in terms of the questions researchers have explored and the context of empirical research efforts including participants, patterns, training and tasks. We also classify the studies by what investigators' measured to evaluate pattern applications, and by threats to validity considered during study design and execution. We have established the following research questions to focus our analysis. We address each of these questions in the section referenced in parentheses.

RQ1. What research questions and hypothesis have been empirically investigated in reference to the application of software patterns? (Section 4.1).
RQ2. What is the context of empirical research efforts, such as participant demographics, preparatory material, patterns and task details, which investigate the application of software patterns? (Section 4.2).