Introduction to Empirical Experiments in Software Engineering

Anleitung zur wissenschaftlichen Arbeit

vorgelegt von
Michal Musial
Dezember 2010

Abstract

Empirical experiments play an important role in the evaluation of a software system. To verify or reject whether a software system has a desired effect, empirical experiments have to be performed. This document helps us to understand the steps we need to take to test out our hypothesis about a software system and contains advice for people who want to conduct an empirical experiment.

1. Introduction

This document describes methods we used to test our hypothesis "*TaskManager* reduces the time being spent on completing a programming task". *TaskManager is* a tool, which supports developers during the completion of programming tasks. Our ideas described in this document, can be used for the evaluation of other software systems.

The first step is to research our topic, because the experiment we are trying to perform might be built upon ones we have done previously or other people have already performed similar experiments. If we have gained knowledge about our topic, we can identify the main goal of a software system. The main goal should be defined, in a way to demonstrate its

practical usefulness for future users of the system we want to evaluate with the experiment. In the context of *TaskManager* the goal was "to reduce the time being spent on completing a programming task". Having defined the main goal we can create a hypothesis which states that the software system satisfies our goal. As a next step, we define variables having influence on our goal and criteria to measure those variables. According to the principle of *divide and conquer, v*ariables help to divide the hypothesis and its main goal into subgoals. Having identified these variables, we can conduct a controlled experiment and measure the variables. In the last step, we can empirically confirm the effect of improving navigation of software artifacts with our tool.

2. Evaluation by Experiment

To conduct a controlled experiment, we need to define the experimental context and the experimental design. The main goal of the experimental context is to ensure that objectives of the experiment are well-defined. The experimental context consists of the following parts:

Background Information: information about the topic of the experiment, related work in the area of our experiment and a description of used technologies and motivation of the experiment. Background information should provide necessary introduction to the experiment for other researchers.

Hypothesis: proposed explanation for an observable effect of the software that we would like to validate with our experiment. Hypotheses define the goal of the experiment.

For each of the hypotheses we have to define the experimental design that describes our proceeding during the experiment. The experimental design contains the following parts:

Variables: should correspond to the main activities, which have to be performed by a *subject* in order to achieve the goal. We distinguish between two types of variables: independent and dependent. The independent variables can be measured directly during the observation of the subjects. The dependent variables cannot be measured directly since they are derived from independent variables. This means, that when defining variables we have to find relationships between independent variables to know how they influence dependent variables. In general there is one dependent variable related to the hypothesis. This variable describes the main goal of the hypothesis. In the case of *TaskManager* the *subject* is a developer and the goal is to complete a programming task. An example of a variable can be the time spent on navigation through the lines of code, measured in time units such as seconds.

Subjects: by performing *tasks* subjects enable us to measure certain variables. To get reliable results and to see significant differences between using a new solution for a specific problem and the base line, it is required to test our hypothesis with two groups of subjects: an experimental group and a control group. The experimental group is asked to solve tasks using the new solution which we want to evaluate and the control group solves its tasks using the base line solution – in case of *TaskManager,* the standard version of the IDE. To rule out the differences in subject's experience, it is important that both groups consist of subjects with near equal expertise. Chosen subjects should be relatively close to the population of interest that will use the software we want to evaluate. All subjects should also have comparable

experience in using software technology related to our experiment. The group size also has a meaningful impact on the reliability of the results. For experiments such as the one conducted when developing *Senseo* [2][3], it is strongly advised to conduct the experiment with larger groups of subjects. To find the correct size of the group, we suggest to define:

- Margin of error (ME). This is our measure of precision
- Alpha (A). This is the significance level.
- Critical standard score (Z). This is the value where the cumulative probability that our hypothesis is true is equal to 1- alpha

Having these input defined we can predict the number of observations and so the group sizes (N) as following:

$$N= (Z^2 + A^2) / ME^2$$

Tasks: activities are defined within tasks. Those activities correspond to the hypothesis we want to test. Tasks should not be too complex and define clearly what the subject is asked to do. This allows us to measure more precisely. Tasks should cover all kinds of problem regarding our hypothesis. When designing the tasks we also describe how subjects are assigned to the tasks. For instance assigning can be done randomly. Besides observing subjects by solving the tasks and deriving results from these observations, we can also ask subjects directly about their considerations and experience about our software system. A questionnaire is a means to do so.

An example of a task used in the experiment concerning *TaskManager* was an implementation of a new software feature in a specially prepared experimental project.

Having variables, subjects and tasks defined, we are ready to conduct the controlled experiment. During the experiment we observe the subjects solving the tasks given to them. This observation is needed to measure our variables, to note any problems that do arise during the experiment and to discover possible usability issues with the tool being evaluated. While the observation is being taken, we have an opportunity to record values of our variables, for instance time spent on each activity. It is significant that the experiment is performed with both groups of subjects under the same conditions since only in that situation measurements and comparisons make sense. The goal is to compare and to find differences between the results of both the control and the experimental group. Statistics coming from the experiment are used to confirm or reject our hypothesis.

3. Learning from the Experiment

The goal of the experiment is to validate the hypothesis and answer questions that arose during the evolution of the software system. In this part of the experiment, we assess how the results of the experiment confirm or reject our hypothesis. The validation of the hypotheses is based on an analysis of the result differences between the control and the experimental group. The fundamental criterion when learning from the experiment is to interpret the results in the experimental context. Thanks to the results we can better understand and explain observed differences.

Data collected during the experiment is not standardized. For instance one variable can be measured in time units and another one in the number of mouse clicks. This means that we have to be careful when defining and comparing measurements. Having data and feedback

from the subjects, we can validate or reject our hypothesis. If we are not sure that the defined variables, subjects and tasks will be useful in the context of our hypothesis, we can perform a pretest before the effective experiment takes place. Feedback from the subjects gives us valuable information about possible improvements and extensions to our tool and report about its usability and user acceptance.

4. Best Practices

Performing the experiment with two groups of subjects (control & experimental) rather than one, allows us to measure time differences when solving the tasks. The reason of every significant observed difference between the control and experimental group should be presented in our work.

Using statistical methods such as statistical tests [5] (i.e. t-test, Wilcoxon-Test, Mann-Whitney test) enable us to compute the significance level. The significance level gives us the probability of observing data at least as extreme as that observed given that our hypothesis is true. It is to remember that there is a difference between important statistical significance and practical importance of the results.

It is desired that the results be readable. A convenient way to present the results of our calculation is to use tables and diagrams. Most people find it much easier to study data by looking at diagrams than huge blocks of text. When presenting results, it is not enough to show only the final result. We should explain the methods and how they were used to get the final results. This is relevant because any conclusion about the experiment should be related to the results. Furthermore, this ensures that our results can be verified by other researchers.

It is recommended to measure not only the variables focused on our experimental context. Recording other variables allows us to detect adverse and positive effects of the software being evaluated. When collecting data it can be useful to define criteria describing the quality of gathered data. This helps to ensure that data of lower quality has less or even no influence on the final result.

One of the suggested ways to get feedback from the subjects is to use a questionnaire. Questions given in the questionnaire should concern different aspects of our hypotheses and provide quantified data regarding our experiment. Formulating a question that gives the possibility to express the answer in form of Likert scale ensures to get quantified data. Open questions cannot be measured directly, but they are useful to get feedback about possibilities for the improvement and extension of evaluated software.

Bibliography

[1] Michal Musial. TaskManager: Integrating a Task Manager into IDE. Bachelor Arbeit, Universität Bern, Bern, Switzerland , 2010.

[2] Marcel Härry. Senso: Augmenting Eclips with Dynamic Information. Master Arbeit, Universität Bern, Bern, Switzerland , 2010.

[3] David Röthlisberger, Marcel Härry, Alex Villazón, Danilo Ansaloni, Walter Binder, Oscar Nierstrasz, and Philippe Moret. Exploiting Dynamic Information in IDEs Improves Speed and Correctness of Software Maintenance Tasks. In Transactions on Software Engineering, 2010

[4] Bas Cornelissen, Andy Zaidman, Arie Deursen, and Bart Rompaey. Trace Visualization for Program Comprehension: A Controlled Experiment. In Proceedings 17th International Conference on Program Comprehension (ICPC), p. 100—109, IEEE Computer Society, 2009

[5] Harvey Motulsky. Intuitive Biostatistics (ISBN 0-19-508607-4), Oxford University Press Inc, 2010