**AN EXPERIMENTAL DESIGN METHOD FOR EVALUATING USABILITY FACTORS OF THE ReChaP PROCESS MODEL**

**Noraini Ibrahim1, Wan Mohd Nasir Wan Kadir2, Safaai Deris3**

Software Engineering Department

Faculty of Computer Science and Information Systems

Universiti Teknologi Malaysia

email: **1**noraini\_ib@utm.my, **2**wnasir@utm.my, **3**safaai@utm.my

**ABSTRACT**

The ultimate aim of this research is to facilitate software users and engineers in implementing the rigorous process of requirement change and propagating the identified impacts to the software design elements.Theproposed approach known as ***ReChaP***, aims to simplify the tedious and error-prone process of change propagation for upstream abstraction and high-level requirement changes to support requirement change management throughout the whole software lifecycle. In particular, the foundations of ReChaP approach are available in twofold: product metamodels (theoretical specification) and process models (implementation specification). Therefore, the main objective of this paper is to design the synthetic experimentation to be conducted; including its procedures and guidelines to be used in performing the planned experimental activities.As a result, the observation findings from this experimentation is hoped to be practical in evaluating the usability quality factor of the two aspects: the proposed process models and the preliminary version of the ReChaP tool.

**Keywords**

Requirement Change Propagation (ReChaP), Product Metamodel, Process Model, Synthetic Experiment.

# 1 INTRODUCTION

A software system normally consists of more than one artefact that are consistently connected and interacted among each other, during the time it is being operating steadily. Therefore, a single change to one of the artefacts may result in several impacts to others. This is a simple example of a change propagation phenomenon in the software context. As pointed out earlier (Rajlich, 2006), change propagation is one of the critical steps of software change management, which is concerned with coordinating changes and managing evolvable software throughout its life-cycle, whereby controlling and implementing change propagation is a tedious process that becomes more complex in large-scale software development. Besides that, producing software systems that are able to adapt themselves to rapid environmental and requirement changes has remained as a topical issue in software engineering research (Lehman, 2002; Pfleeger & M.Atlee, 2006).

Another major problem is the uncertain and incomplete requirements throughout the whole software life cycle that may contribute to the volatility issue of the requirements (Ibrahim, Kadir, & Deris, 2009a; Jönsson & Lindvall, 2005; Nurmuliani, Zowghi, & Powell, 2004; von Knethen & Grund, 2003; Wan Kadir & Loucopoulos, 2004). For that reason, it is important to address the requirements as key sources in managing software changes since changes on a requirement may result to consequent changes to the later phase in software life-cycles i.e. design, testing, maintenance and evolution (Ibrahim, Kadir, & Deris, 2008; Jönsson & Lindvall, 2005; von Knethen, 2002).

The remainder of this paper is organised as follows. Section 2 presents a detailed description on ReChaP approach. Following that, Section 3 elaborates on the contextual background for the experiment. Section 3.1 explains the formulated hypothesis to be tested and the metric to be measured. In addition, the overall steps to be performed in the experimental methods are also presented in Section 3.2. Accordingly, Section 4 presents the experimental design and its procedures in detailed. Section 4.1 describes the criteria of the experimental subjects (participants). In the following Section 4.2, discussion will be made briefly regarding the environment settings in the laboratory. Then, discussion on the implemented case study is presented in Section 4.3, followed by descriptions on the two types of questionnaires that will be distributed to the subjects in Section 4.4. Subsequent to that, Section 5 addresses on the four types of threats to validity issue that should be highlighted in designing this synthetic experiment. Finally, Section 6 concludes the whole findings of this paper and presents the future plans for the next steps in this research.

# 2 DESCRIPTIONS OF ReCHAP APPROACH

In the previous work, we have defined that the requirement change propagation process as being made up from a series of steps to identify the initial affected sets, to analyse the affected artefacts set that need to be changed (in terms of addition, alteration or removal), and to execute the accepted change until there is no more further changes required (Ibrahim et al., 2009a). This research is directed towards addressing the issues on the rigorous requirement change propagation process to support requirement management, especially during software evolution. In particular, the main goals of this work are divided into twofold: i) to develop a product metamodel (theoretical specification), and ii) to develop a process model (implementation specification). Figure 1 illustrates the two pillars of the ReChaP approach. Further descriptions on these two pillars are described in our previous work (Ibrahim et al., 2009a; Ibrahim, Kadir, & Deris, 2009b).



Figure 1: Overview of ReChaP Approach

The first goal is intended to provide a better understanding and solutions for software developers (i.e. software engineers and maintainers) in terms of the theoretical specifications and modelling for working requirement change propagation to the software designs. Meanwhile, the second goal focuses in presenting profound descriptions on a systematic process flow of propagating the requirement changes to the software design elements. The core activities of the proposed requirement change propagation process are described in terms of requirement, analysis, design and evolution phases.

The first goal of this research has been achieved beforehand. The product metamodels have been evaluated using two industrial strength case studies of Integrated Facilities and Assets Maintenance Management System (IFAMMS) and healthcare application of MediNET system. Our initial findings concluded that the two case studies are worthwhile to be considered for observation in evaluating the applicability aspects of the metamodels for the ReChaP approach. The case studies helped to theoretically specify the definition of volatile requirements as main sources of changes in software evolution. In our case, the proposed metamodels act as a key role and essential enabler in providing specific traceability; with fairly accurate links from requirement level to the software design elements. Additionally, the metamodel promotes suitable requirement types to exhibit the volatility features. Apart from that, the proposed metamodels also helps in reducing the difficulties to express the requirement specification using natural languages. The element of pre-defined templates relatively helps to define new requirement specification. As a result, it offers simpler way that helps to reduce human efforts and fewer errors. Also, the suggested templates are reusable, which will result to higher flexibility of software systems specifications, and at the same time helps to express the requirement specifications in a consistent manner and more standardised way. Nevertheless, any revision of the defined product metamodels is subjected to the future evaluation on further case studies.

At present, the proposed process models however, have not been evaluated so far. The process models are adopted using a well-accepted Software Process Engineering Meta-Model (SPEM) v2.0 by OMG ((OMG), April 2008). The processes could provide guidelines especially for software engineers to assist them in simplifying the change propagation process for evolutionary software development. In addition to that, the defined processes will capture the three main components namely activities, roles (actors) and work products (software artefacts) that are generally related in the development and evolution of a software system. The proposed process models characterise responsible people such as the system users, system designers and analysts. Their specific roles are also illustrated in terms of used cases as well as the detailed activities that those people perform. Furthermore, the process models also highlight all the associated software artefacts that are being manipulated, in particular during the implementation of requirement changes.

Throughout the research, a preliminary version of ReChaP tool has been developed to realise the direction of this research. The main concern of this prototype is to perform a proof-of-concept of the two main elements in the proposed ReChaP approach. From the first perspective of the product metamodel element, the tool realises the traceability provided by the defined metamodels, which act as a key player in enabling the change propagation process. While from the second perspective of the process models element, the prototype realises the approach by offering the systematic guidelines to support and simplify the phenomenally time consuming and expensive efforts of the change propagation process.

Besides that, the main target of the ReChaP tool is to demonstrate the automation aspect in managing the changes for requirement levels, whereby the major aim is to simplify the tedious change propagation process from requirement levels to the software design levels. It is hoped that the automated strategies would help to exploit the tool’s efficiency not only by reducing the human efforts to intervene with the manual propagating tasks, but at the same time would lessen the human errors in predicting the impacted elements in software artefacts. However, in the context of software engineering research paradigm, the prototype tool development itself is not sufficient to prove the efficiency of the concepts in the proposed ReChaP approach. Therefore, the synthetic experiment aims at an empirical observation of the usability quality factor, in order to evaluate the proposed process models of the ReChaP approach and to perform a proof-of-concept of the preliminary version of ReChaP tool.

# 3 EXPERIMENTAL CONTEXT

A synthetic experimentation is a classical scientific method that can be used to evaluate empirical studies in the software engineering research and practices (Sjoberg et al., 2005; Wohlin et al., 2000). The basic purpose of synthetic experiment is to analyse any influences towards observation of the result by replicating or reproducing the simpler version in the laboratory (Zelkowitz, 2009). According to (Zelkowitz & Wallace, 1998), synthetic experiment is defined as:

*“A replicated experiment that is conducted in a smaller artificial environment, but in a realistic setting compared to the real projects.”*

In the context of this research, synthetic experiment is another method in explanatory research, which aims to look intensively at the generic scientific foundation for the proposed elements in the ReChaP approach. Thus, this method is also expected to better understand the real implementation and causal relationships (dependent vs. independent variables) on the phenomenon of requirement change propagation process. Due to that, the purpose of the designed synthetic experiment is to empirically and scientifically evaluate the third and fourth objective of this research as follows (Ibrahim et al., 2009b):

* *To propose a software process model that simplifies the requirement change propagation into software designs*
* *To develop a custom-made tool as a proof-of-concept of the research that applies both metamodel and process elements in the proposed ReChaP approach*

In particular, the rest of this section describes on the hypothesis and the software quality metrics to be obtained during the experiment. Also, the experimental method is further explained based on the planned steps to be conducted during the synthetic experiment.

## 3.1 Hypothesis and Variables

The design for the experiment is based on the formulated hypotheses such as the following:

**Null hypothesis, H0 usability:**

There is no (significant) difference in simplifying requirement change propagation process to software designs by a proposed ReChaP approach that consists of two components: i) product metamodel and ii) process model.

**Alternative hypothesis, H1 usability:**

There is (significant) difference in simplifying requirement change propagation process to software designs by a proposed ReChaP approach that consists of two components: i) product metamodel and ii) process model.

## 3.2 Usability Quality Factor and Criteria

The vital concern in defining the software quality factors is to define the attribute to be obtained and the metrics that measure the attribute. In this experiment, the attribute is a ***“simplicity”*** of the proposed software process for ReChaP approach; whether it could support the requirement management and software evolution by simplifying the change propagation process from requirement level to software designs artefacts.

In particular, the ***“usability”*** metrics that measure the simplicity attribute is planned to be obtained for static analysis. Figure 2 shows the relations between the usability quality factor to be measured with the simplicity attribute of the proposed process model to be obtained. Consequently, ***“usability”*** quality factor is defined as:

*“The capability of the software to be understood, learned, used and liked by the user, when the used is under specified conditions”* ("ISO 9126-1," 2001)



Figure 2: Usability quality factor

Five main criteria are chosen to analyse the usability factor as illustrated in Figure 3. These criteria are chosen based on Quality in Use Integrated Measurement (QUIM) enhancement model corresponding from several different standards (e.g., ISO 9241, ISO/IEC 9126, IEEE Std.610.12) for usability measurement (Seffah, Donyaee, Kline, & Padda, 2006). Each of the criteria is briefly explained as follows:

1. ***Effectiveness***, which concerns on whether the software product is capable in helping users to perform the required task with accuracy and completeness.
2. ***Efficiency***, which concerns on whether the software product is capable in helping users to perform the required tasks within an appropriate amount of time wih regards to the aimed effectiveness.
3. ***Satisfaction,*** which concerns on users’ feelings and real experience when performing experimental task using the tool.
4. ***Learnability,*** which concerns on whether the tool is easy to use and can be mastered.
5. ***Usefulness,*** which concerns on the handy features and functionalities provided by the tool, whether it helps to solve users’ tasks in an acceptable way.



Figure 3: The usability criteria to be obtained.

In this targeted experiment, usability factor is the dependent variable to be obtained (i.e. effectiveness, efficiency, satisfaction, usefulness and learnability). As for the independent variable, we have determined two variables that can be manipulated and controlled, namely;

1. Type of the system under investigation (the domain application system which is library system case study)
2. Type of requirement change scenarios ; simple and complex scenarios ( in terms of addition, deletion and modification)

## 3.3 Experimental Method

Figure 4 illustrates the steps to be taken in performing the synthetic experimentation. The experimental tasks are divided into three main phases, namely; i) introduction session to research experimentation, ii) comprehensive briefing on the approach, and iii) the research experimental implementation phases.

First, the introduction session starts with the explanation on the procedures and guidelines to be performed during the experiment. Then, the subjects are required to fill-in the debriefing questionnaires that consist of the information on demographic data such as personal details, working background and subject’s experience in software engineering practices.

The session continues with a briefing on the presentation of the proposed ReChaP approach, description on the library system as a chosen case study, and finally the tutorial to demonstrate how to use ReChaP tool. The tutorial will expose and give detailed instructions to the participants on the main functionalities of the tool.

During the third phase, which is the experimental tasks implementation phase, the subjects are instructed to complete all the following tasks based on the guidance in the proposed software process models (Refer to Appendix B).

1. Task 1 (Requirement Phase): Understanding the system context for library system case study, as well as capturing all functional and non-functional requirement statements for the library system.
2. Task 2 (Analysis Phase): Analysing the library system architectures.
3. Task 3 (Design Phase): Defining the requirement specifications and architectural design for library system application.
4. Task 4 (Evolution Phase): Managing requirement change request and propagating requirement change to the software design elements.



Figure 4: The design of experimental methods.

In Task 4, during the evolution phase, two sets of scenarios for requirement change will be given. Finally, once all of the required tasks are completed, participants will be asked to fill-in the post-experiment questionnaire. In this questionnaire, participants will be asked on their perception and judgement on the usability aspects of the prototype tool, whether the prototype tool provides the support to simplify the change propagation process from the requirement to the software design.

# 4 EXPERIMENTAL DESIGN

In this section, specific planning, procedures and guidelines are presented in details for the synthetic experiment to be conducted in a near future. The planning includes information gathered regarding the subjects or participants that will involve the environment settings of the venue (laboratory), as well as the detailed treatment on the implementation of the chosen case study.

## 4.1 Experimental Subjects

Twenty subjects are targeted to be participated in this synthetic experiment. All post-graduate students from Software Engineering (SE) background are cordially invited. All experimenters are expected to have minimal understanding on SE theories and principles, and must be exposed to the object oriented development techniques (i.e. Unified Modelling Language-UML). This pre-condition is very important to keep abreast of their motivation in participating the experiment until its completion.

## 4.2 Laboratory Environment Settings

The synthetic experiment is planned to be conducted in the laboratory at Software Engineering Department, Faculty of Computer Science and Information System in Universiti Teknologi Malaysia. It is expected that there will be twenty people volunteering as experimental subjects. All participants are required to sit in groups. Each group contains two members. Each computer is supported by Windows XP operating systems, Generic Modelling Environment (GME) case tool version 7.0, and the preliminary version of ReChaP tool.

## 4.3 Experimental Materials

In addition, each group is supplied with a set of documentation to support the experimentation process:

1. Experimentation guidelines and procedures
2. The presentation slides on the background theory of requirement change propagation
3. A library system case study with change scenarios to be solved
4. Preliminary version of ReChaP tool user manual, and the process models.
5. The questionnaires.

### *a) Questionnaires*

There will be two types of questionnaires to be answered by the experimental subjects during the Phase 1 and Phase 3 of the experiment (as mentioned in Section 3.2, Experimental Method).

1. **Debriefing Questions:** refer to pre-experiment questions to be answered before the experimental tasks begin. The purpose of this questionnaire is to collect the related demographic data on participant such as personal details, their background and experiences on software engineering theoretical and concepts, their motivation estimation in participating the experiment, as well as their level of knowledge for library system as a chosen domain application. (Refer to Appendix A for details)
2. **Post-Experiment Questions:** are carefully designed in order to obtain five criteria of the usability factor in evaluating the proposed software process and the preliminary tool of ReChaP approach. The questions are composed into of two parts, namely, i) the user’s perception on the feelings and experience in using the tool and ii) the impacts of the given guidelines that derived from the proposed software process in simplifying the requirement change propagation to software design scenarios in order to solve the required experimental tasks.

Sample questions sets are referred and adopted from Software Usability Measurement Inventory (SUMI) (Kirakowski, 1994). SUMI is an industry standard evaluation questionnaire for assessing quality of use of software by end users.

In addition to that, the quality ratings for each question are based on ordinal Likert scale of 5 options to be chosen (1: Strongly disagree, 2: Disagree, 3: Undecided, 4: Agree, 5: Strongly agree). Appendix B shows the drafted questions of post-experiment questionnaire. The questionnaire is designed to map all the five criteria of the usability factor towards obtaining the evaluation analysis of the proposed product metamodel, process model as well as preliminary tool of ReChaP in reducing the human errors and intervention (effort) when simplifying the requirement change propagation process. Two criteria namely effectiveness and efficiency are related with the participant’s ability to correctly complete the tasks within certain periods. Therefore, these two criteria are evaluated for each assigned task for more simple and complex change scenarios. Examples for experimental tasks that are planned for both simple and complex change scenarios can be found in Appendix C.

## b) Case Study Evaluation

The application and evaluation on the Library System case study is intended to objectively investigate the actual implementation for all defined product metamodels of ReChaP approach in a realistic environment of software development life cycles. The suitable requirements for library application domain will be specified based on the requirement template definition facility as provided by the prototype. The related software design artefact such class diagram and state diagrams will then be created for all the defined requirement specifications. The following Figure 5 depicts the six packages of library system domain that will be implemented during the experiment.



Figure 5: Library System Packages

### 4.4 Experimental Procedures

There will be two rounds in the experiment. During the first round, participants are instructed to perform the simple change scenario (i.e. modification). In Round 2, they need to perform the complex change scenario (i.e. add & delete tasks). The experiment procedure is illustrated in Figure 6.



Figure 6: Experimental Procedures

### 4.5 Experimental Tasks

The examples of experimental tasks that will be performed can been seen in Appendix C. Basically, the subjects will be guided to perform the following tasks for the library system domain case study:

1. Create the aforementioned six packages of the whole library system class diagram (i.e. Registration, Loan/Borrow, Return/Checkout, Renewal, Reserve and Search). The packages models are created based on the previous designed class metamodel.
2. Create two state diagrams of book and patron that indicate the behavioural parts of the library system. These two state diagrams are created based on the previous designed state metamodel.
3. Managing requirement
   1. Adding new requirement
   2. Deleting the current requirement
   3. Modification of the current requirement
4. Simple and complex change scenarios.

# 5 THREATS TO VALIDITY

Validity on the conducted research is very subjective to be assured. Nevertheless, it is a must to deal with the validity issue in order to ensure the evaluation result and analysis made is convincing and not prejudice. More importantly, the analysis result should be reliable enough to be trusted as well (Runeson & Höst, 2009). This section explains on the four kinds of threats to validity that should be considered in designing the planned experiment. These four types of validity are commonly used to justify synthetic experiment in software engineering research. However, to this extent, the validities as described here have not been finalised since the synthetic experiment has not yet been undertaken. It is therefore crucial that we thoroughly ~~to~~ consider the pre-caution steps in minimising the threats to be handled and identify the effect from the causal relationships.

## 5.1 Internal validity

The internal validity is referring to the conclusion made on the cause-effect or causal relationships of independent variables with the dependent variables (Easterbrook, Singer, Storey, & Damian, 2007; Runeson & Höst, 2009). This validity may also concern on the side effects of any parameters or undetected source of bias, that arise from the experimental design and analysis result (Kitchenham et al., 2008). Three kinds of possible threats are identified, namely; maturation effect, selection effect and instrumentation effect.

### *a) Maturation effect*

1. Blind experiment is an important consideration in scientific method to minimise the subject’s expectation bias from occurring (Kitchenham et al., 2008). Thus, subjects will not be informed on the specific hypothesis to be validated in this synthetic experimentation.
2. Unfamiliarity towards the prototype tool and less experience in handling GME case tool may affect the result. To prevent this, a detailed tutorial on the step by step procedure in using the prototype tool is planned to be an agenda in the experimentation session as shown in Table 1.
3. There will be two rounds of experimentation for requirement change scenarios; simple and complex. Thus, subject’s learning effects in performing the experimental tasks are moderately controlled.

### *b) Selection effect*

1. Experimental subjects are selected based on their software engineering background and priority is given to those with experience in object oriented development technique (i.e. UML techniques). Moreover, as previously mentioned by (Kamsties, Von Knethen, & Reussner, 2002), selecting students as the experimental subjects are useful as a pilot study for later experiment in the real industry environment.
2. Subjects are also required to sit in a group of two persons, so that they can have a discussion during the experiment and this situation can reduce the effort per subject, at the same time to prevent from any plagiarist activities among the groups.

### *c) Instrumentation effect.*

1. Library domain application system is chosen as a case study, because every subject is familiar with the functionalities and the system environment. Therefore, the unfamiliarity effect of domain application’s knowledge is avoided.
2. The experiment session is well-planned so that all subjects are given sufficient time to answer questions in both pre and post questionnaires, while performing the experimental tasks. The following Table 1 shows the planning for each activity and the time that is expected to be taken.

Table 1: Experimental Agenda

|  |  |
| --- | --- |
| **Activity** | **Session Duration** |
| **Phase I : Introduction to Research Experimentation** | |
| * Description on Research Procedures * Filling out Debriefing Questionnaire | * 5 minutes * 10 minutes |
| **Phase II : Briefing on ReChaP Approach** | |
| * Presentation on “Requirement Change Propagation” * Tutorial on how to use the ReChaP prototype tool * Briefing on the case study that will be implemented | * 15 minutes * 30 minutes * 30 minutes |
| **Phase III : Experimental Implementation** | |
| * Applying Library System case study using the ReChaP prototype tool * Filling out Post-Experiment Questionnaire | * 1 hour 30 minutes * 30 minutes |
| Break | 15 minutes |
| Summary, Question and Answer (non formal) | 30 minutes |

## 5.2 External validity

This type of validity significantly concerns on whether the same experimentation and the results' analysis shall be extended and generalised on to other persons, place and time (Jedlitschka, Ciolkowski, & Pfahl, 2008). First of all, it is not a guarantee that the future experimental results can be generalised to other current systems. It is because the evaluated library case study is not representing the completed and running library system. Therefore, further empirical investigation is compulsory, if the results are planned to be generalised to other systems; which depending on the nature of the organisation, its complexities and the volatility features in existing requirements in their business environment.

Secondly, even though the synthetic experiment to be conducted is a laboratory testing in academic settings, the expected subjects are post-graduate students that have attended the SE course previously, and they also have minimal experiences in industry. Besides that, they are motivated to gain practical experience involving the user testing in the SE research project that is applicable to their course theories.

## 5.3 Construct validity

This type of validity is concerning on whether *“the theoretical constructs are interpreted and measured correctly”* (Easterbrook et al., 2007). In simpler words, it means; are we measuring what we actually believe (Bratthall, Johansson, & Regnell, 2000)? In the context of the designed experiment, the usability metrics are translated and examined carefully based on the criteria suggested by QUIM model (Seffah et al., 2006). The design for post-experimental questionnaires is greatly focussing on the five criteria in the usability criteria, namely: efficient, effectiveness, satisfaction, learnability and usefulness. Therefore, the questionnaires are really devoted to interpret the measurement aspect of usability quality factor.

## 5.4 Conclusion validity

This type of threat is concerning on to which extent the conclusion drawn is correct, on the result analysis of the statistical tests of data (Runeson & Höst, 2009). To this point, any validity issue on this conclusion threat has not been determined yet since we have not conducted the planned experiment. For that reason, there is insufficient information that has been collected to perform the statistical tests to report on the overall observation on the evaluated usability factor of the process models and the preliminary version of ReChaP tool.

# 6 SUMMARY & CONCLUSION

This research performs an exploratory study to investigate whether the proposed ReChaP approach can fairly resilient (support) the software evolution by simplifying the requirement change propagation into software design. The foundations of ReChaP approach are twofold: product metamodels (theoretical specification) and process models (implementation specification). In particular, the preliminary version of ReChaP tool is also developed based on these two pillars. It is an automated tool that realised the two pillars of ReChaP approach, which aims to minimise human error and user intervention (effort) in simplifying the tedious requirement change propagation tasks.

In this paper, design and procedures of the synthetic experiment to be conducted for the proposed requirement change propagation, that is the ReChaP approach is presented. The objective of the synthetic experiment is to evaluate empirically the usability quality factor in terms of the second elements in ReChaP approach; the proposed process model and to proof-of-concept the preliminary version of ReChaP tool. The usability factor is empirically measured in terms of five criteria, namely efficient, effectiveness, learnability, satisfaction and usefulness. In particular, the paper elaborates in detailed on the specific guidelines and related procedures in performing the synthetic experiment. In addition to that, the identified threats to validity are also looked at carefully in designing this experiment. Therefore, it is hoped that any potential threats could be restricted and reduced.

# 7 FUTURE WORKS

# It is hoped that this targeted experimentation shall be able to be conducted in the near future. Once conducted, the qualitative result of the synthetic experimentation will be analysed and the static analysis should be presented for the improvement of the overall ReChaP approach. The qualitative result is hoped to complement the explanatory research method in the study, in order to retain valuable insight into evaluating the proposed process model for simplifying the tedious requirement change propagation tasks. More importantly, the conclusion made is an initial step for empirical studies in this research, that would enable for a reasonable answer to be found in accumulating a body of knowledge to facilitate the usability issue of the proposed processes (Shull, Carver, & Travassos, 2001).

# ACKNOWLEDGMENTS

This research is partially funded by the ***Science Fund*** Grant by *Ministry of Science and Technology (MOSTI)* under ***Vote No.79265.*** The authors would like to thank *MOSTI* and *Universiti Teknologi Malaysia (UTM)* for their financial support. In particular, the authors would also like to acknowledge the respective individuals and members of Software Engineering department for the informative discussions, supportive suggestions and invaluable feedbacks during the time that this research was being conducted.

**APPENDIX A**

**DEBRIFING QUESTIONNAIRE**

**Personal Details and Working Background**

**Student ID (e.g. PC/MC- Optional) : \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Qualifications (e.g. Master in SE/ Bachelor in Comp.Sc) :\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

1. What was your previous job? (Tick ***√*** in the box/fill in the blank)

|  |  |  |
| --- | --- | --- |
|  | a. | Programmer/Software Developer |
|  | b. | Software Engineer /System Analyst |
|  | c. | Project Leader / Project manager |
|  | d. | Configuration engineer |
|  | e. | Quality engineer |
|  | f. | Other (please specify) \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ |

**Please circle your responses to the following statements based on the scales below.   
(Choose ONE answer only)**

1. How do you rate your experience in software engineering practice?

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **None** | **Little** | **Average** | **Substantial** | **Profesional** |
| **1** | **2** | **3** | **4** | **5** |

1. How do you rate your experience in software development?

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **1** | **2** | **3** | **4** | **5** |

1. How do you rate your experience in object-oriented design using UML modelling?

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **1** | **2** | **3** | **4** | **5** |

1. How do you rate your experience in software change management/maintenance/evolution?

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **1** | **2** | **3** | **4** | **5** |

1. How do you rate your experience in performing requirement change propagation?

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **1** | **2** | **3** | **4** | **5** |

1. What kind of software artefacts that you have been involved with in performing change request? (Tick ***√*** all that applies OR fill in the blank if specified otherwise)

|  |  |  |
| --- | --- | --- |
|  | a. | Source code |
|  | b. | Design |
|  | c. | Requirements |
|  | d. | Test cases |
|  | e. | Documentation |
|  | f. | Others (please specify) \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ |

**Motivation and Domain Knowledge Background**

1. In overall, kindly estimate how motivated you are to participate in this experimentation.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Not** | **Poorly** | **Fairly** | **Well** | **Highly** |
| **1** | **2** | **3** | **4** | **5** |

1. Please choose the specific reasons for the above answer. (Tick ***√*** all that applies OR fill in the blank if specified otherwise)

|  |  |
| --- | --- |
|  | a. To gain practical experience involving the user testing in the SE research project |
|  | b. To apply the related SE theories and knowledge from my coursework |
|  | c. Others (please specify) \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ |

1. In overall, kindly estimate how well you understood the Library system as the case study application; in terms of its environment and functionality.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Not** | **Poorly** | **Fairly** | **Well** | **Highly** |
| **1** | **2** | **3** | **4** | **5** |

**APPENDIX B**

**POST-EXPERIMENT QUESTIONNAIRE**

For every task that will be assigned, participant will be asked to evaluate the efficiency and effectiveness criteria as below:

1. **Efficiency :**

**Task Start Time** (e.g.: 9.30am)

**Task Finish Time**

1. **Effectiveness** 
   * Can you complete this task? **(Tick ONE only)**

Answer: **Yes**

**No**

After all tasks are given, participant will be asked to evaluate the rest of the three criteria involving learnability, satisfaction and usefulness for the whole tasks such as the followings:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Learnability** | **Strongly disagree** | **Disagree** | **Undecided** | **Agree** | **Strongly agree** |
| 1. It is easy to use the tool | **1** | **2** | **3** | **4** | **5** |
| 1. It is easy to learn to use and the tool it can be mastered | **1** | **2** | **3** | **4** | **5** |
| **Satisfaction** | **Strongly disagree** | **Disagree** | **Undecided** | **Agree** | **Strongly agree** |
| 1. Allows an easier way to perform specific changes on requirements (i.e. addition, deletion and modification). | **1** | **2** | **3** | **4** | **5** |
| 1. Allows an easier way to propagate changes from requirement to class and state diagrams. | **1** | **2** | **3** | **4** | **5** |
| 1. ReChaP prototype tool can fairly simplify the requirement change propagation process in supporting the software evolution. | **1** | **2** | **3** | **4** | **5** |
| **Usefulness** | **Strongly disagree** | **Disagree** | **Undecided** | **Agree** | **Strongly agree** |
| 1. It allows me for an easier selection to classify the types of requirement. | **1** | **2** | **3** | **4** | **5** |
| 1. It allows me for an easier way to define the requirement specifications. | **1** | **2** | **3** | **4** | **5** |
| 1. It allows me for an easier way to choose an input statement for requirement specifications (i.e. by fill-in the **“< >”** placeholders) | **1** | **2** | **3** | **4** | **5** |
| 1. It allows me for an easier way to express the requirement in a consistent manner | **1** | **2** | **3** | **4** | **5** |
| 1. It allows me for an easier way to create software design model (class and state diagrams) based on the provided metamodels | **1** | **2** | **3** | **4** | **5** |

# REFERENCES

1. (OMG), O. M. G. (April 2008). *Software & Systems Process Engineering Meta-Model Specification (SPEM) Version 2.0*.
2. Bratthall, L., Johansson, E., & Regnell, B. (2000). Is a design rationale vital when predicting change impact?-A controlled experiment on software architecture evolution. *Product Focused Software Process Improvement*, 77-100.
3. Easterbrook, S., Singer, J., Storey, M. A., & Damian, D. (2007). Selecting empirical methods for software engineering research. *Guide to advanced empirical software engineering*, 285-311.
4. Ibrahim, N., Kadir, W. M. N. W., & Deris, S. (2008, 26-31 October 2008). *Comparative Evaluation of Change Propagation Approaches towards Resilient Software Evolution.* Paper presented at the The Third IEEE International Conference on Software Engineering Advances (ICSEA'08), Sliema, Malta.
5. Ibrahim, N., Kadir, W. M. N. W., & Deris, S. (2009a). *Propagating Requirement Change into Software Designs to Resilient Software Evolution.* Paper presented at the The 16th IEEE Asia Pacific Software Engineering Conference (APSEC'09), Penang, Malaysia.
6. Ibrahim, N., Kadir, W. M. N. W., & Deris, S. (2009b). *ReChaP: An Approach for Simplifying Requirement Change Propagation to Resilient Software Evolution*. Paper presented at the The Fifth Postgraduate Annual Research Seminar 2008 (PARS’ 09).
7. ISO 9126-1. (2001).
8. Jedlitschka, A., Ciolkowski, M., & Pfahl, D. (2008). Reporting experiments in software engineering. *Guide to advanced empirical software engineering*, 201-228.
9. Jönsson, P., & Lindvall, M. (2005). Impact Analysis. In A. Aurum & C. Wohlin (Eds.), *Engineering and Managing Software Requirements* (pp. 26): Springer, Berlin, Heidelberg, New York.
10. Kamsties, E., Von Knethen, A., & Reussner, R. (2002, 9-10 September 2002). *A Controlled Experiment on the Understandability of Different Requirements Specifications Styles.* Paper presented at the International Workshop on Requirements Engineering: Foundation for Software Quality (REFSQ’02), Essen, Germany.
11. Kirakowski, J. (1994). SUMI Questionnaire Homepage. Retrieved 1 February, 2010, from <http://sumi.ucc.ie/>
12. Kitchenham, B., Al-Khilidar, H., Babar, M. A., Berry, M., Cox, K., Keung, J., et al. (2008). Evaluating guidelines for reporting empirical software engineering studies. *Empirical Software Engineering, 13*(1), 97-121.
13. Lehman, M. M. (2002). Software Evolution. *Encyclopedia of Software Engineering, 2*, 1507-1513.
14. Nurmuliani, N., Zowghi, D., & Powell, S. (2004). *Analysis of requirements volatility during software development life cycle.* Paper presented at the The IEEE Australian Software Engineering Conference (ASWEC'04).
15. Pfleeger, S. L., & M.Atlee, J. (2006). *Software Engineering Theory and Practice*. New Jersey, USA: Pearson Prentice Hall.
16. Rajlich, V. (2006). Changing the Paradigm of Software Engineering. *Communications of the ACM, 49*(8), 67-70.
17. Runeson, P., & Höst, M. (2009). Guidelines for conducting and reporting case study research in software engineering. *Empirical Software Engineering, 14*(2), 131-164.
18. Seffah, A., Donyaee, M., Kline, R. B., & Padda, H. K. (2006). Usability measurement and metrics: A consolidated model. *Software Quality Journal, 14*(2), 159-178.
19. Shull, F., Carver, J., & Travassos, G. H. (2001). An empirical methodology for introducing software processes. *Software Engineering Notes, 26*(5), 288-296.
20. Sjoberg, D. I. K., Hannay, J. E., Hansen, O., Kampenes, V. B., Karahasanovic, A., Liborg, N., et al. (2005). A Survey of Controlled Experiments in Software Engineering. *Software Engineering, IEEE Transactions on, 31*(9), 733-753.
21. von Knethen, A. (2002). *Change-oriented requirements traceability. Support for evolution of embedded systems.* Paper presented at the International Conference on Software Maintenance.
22. von Knethen, A., & Grund, M. (2003). *QuaTrace: a tool environment for (semi-) automatic impact analysis based on traces.* Paper presented at the 19th IEEE International Conference on Software Maintenance (ICSM).
23. Wan Kadir, W. M. N., & Loucopoulos, P. (2004). Relating evolving business rules to software design. *Journal of Systems architecture, 50*(7), 367-382.
24. Wohlin, C., Runeson, P., Höst, M., C.Ohlsson, M., Regnell, B., & Wesslén, A. (2000). *Experimentation in Software Engineering – An Introduction*. Boston, MA: Kluwer Academic Publishers.
25. Zelkowitz, M. V. (2009). An update to experimental models for validating computer technology. *Journal of Systems and Software, 82*(3), 373-376.
26. Zelkowitz, M. V., & Wallace, D. R. (1998). Experimental Models for Validating Technology. *Computer*, 23-31.

Ms. Noraini Ibrahim obtained her B.Sc. degree in Computer Science from Universiti Teknologi Malaysia (UTM). Subsequently, she completed her Master Engineering degree in Software Engineering from University of Queensland, Australia. She is currently attached with UTM as a lecturer and pursues her Ph.D. in Faculty of Computer Science and Information System (FSKSM), UTM. Her research interests include software maintenance and evolution, as well as requirement engineering area.



Dr. Wan M.N. Wan Kadir is an Associate Professor in Software Engineering Department, FSKSM, UTM and his current position is Deputy Dean (Development) of FSKSM. He received his B.Sc. degree in Computer Science from UTM, Master Science degree from UMIST, and Ph.D. in the field of Software Engineering from The University of Manchester. His research interests range from the various Software Engineering knowledge areas based on the motivation to reduce the cost of development and maintenance, as well as to improve the quality of large and complex software systems.

Dr. Safaai Deris is a Professor in Software Engineering Department, FSKSM, UTM. He is currently seconded to the Centre of ICT in UTM as a Director. He is also a Director of Artificial Intelligence and Bioinformatics Group (AIBIG), UTM. He received his Master Engineering degree in Industrial Engineering and Doctor of Engineering in System and Computer Science from Osaka Pref University, Japan. His research interests include various Software Engineering, Artificial Intelligence and Bioinformatics areas.