International Journal on New Computer Architectures and Their Applications (IJNCAA) 1(3): 599-614
The Society of Digital Information and Wireless Communications, 2011 (ISSN: 2220-9085)

An Empirical Framework Design to Examine the Improvement in Software Requirements through Negotiation

Sabrina Ahmad, Noor Azilah Muda
Faculty of Information and Communication Technology,
Universiti Teknikal Malaysia Melaka
{sabrinaahmad, azilah}@utem.edu.my

ABSTRACT

Negotiation is one promising effort during requirements elicitation process to improve the quality of software requirements. When negotiation is claimed beneficial theoretically, it is important that the deployment of negotiation is examined and the effectiveness of negotiation is evaluated through empirical study. This paper aims at providing an empirical framework design to examine the improvement in software requirements through negotiation. Besides, it elaborates the relevance of negotiation in requirements elicitation process and its effectiveness. An empirical study method is imposed to design the framework. The design is carefully established based the selection of population and participants, the experimental protocol, threats to validity and justification of measures.

KEYWORDS

Empirical framework, design, software requirements, requirements elicitation, negotiation.

1 INTRODUCTION

Many efforts have been done to improve the quality of software requirements. Negotiation is one of the promising efforts during requirements elicitation process especially when it involved various stakeholders. In a process of identifying the right requirements to develop, conflicts are common since stakeholders frequently pursue mismatching goals. Reaching agreements among stakeholders who have different concerns, responsibilities, and priorities is quite challenging. Therefore, negotiation is useful to handle the conflicts and to resolve disagreement between the stakeholders. A part of achieving agreement, the requirements are believed to be improved in quality.

Software requirements quality is usually assessed through verification and validation of intermediate or final product. The requirements are checked against requirements specification, prototypes or the end product. This is known as an analytical approach. It describes an effort to detect the defects within the software development products and fix them.

Meanwhile, a constructive approach is applied while developing the requirements. This approach suggests prevention to ensure that mistakes are minimized during the creation of requirements. In this research, a constructive approach was adopted by enforcing negotiation in the

requirements elicitation process. Negotiation is seen as a preventive action whereby defects are not introduced into the requirements statement. Therefore, the requirements elicitation process which incorporates negotiation is expected to list better quality requirements.

The rationale of adopting the constructive approach and measuring the quality at a very early stage is obvious. History tells us that the greatest number of errors and the errors that are most costly to fix are generated at the beginning stage of software development process. Errors in requirements are the most numerous in the software lifecycle and also the most expensive and time-consuming to correct [1]. The context in which requirements are elicited is usually a human activity, and the problem owners are people. It is seldom technical problems which inhibit productivity and quality [2, 3]. Instead the vast majority of requirements problems are related to human interactions, process and communications. One of the main problems during requirements elicitation is communication and understanding among the stakeholders. This involves conflicts, scope boundary and erroneous interpretation. The argument is supported by Zowghi [3] who believed that requirements elicitation is inherently imprecise as a result of multiple variable factors, a vast array of options and decisions, and communication.

Due to the urgency of quality requirements for quality software, this paper outlines an empirical framework design to allow empirical investigation in order to implement negotiation in requirements engineering and to assess its effectiveness. The design also explains the mechanism of negotiation activities which contribute to the improvement in requirements quality.

2 NEGOTIATIONS IN REQUIREMENTS ENGINEERING

According to Grunbacher [4], negotiation leads to benefits such as understanding project constraints, adapting to change, fostering team learning, revealing tacit knowledge, managing complexity, dealing with uncertainty and finding better solutions. Furthermore, the benefits of negotiation are obvious and many researchers have pointed out its usefulness for requirements engineering [5-11]. None of these studies measured the improvement in requirements after negotiation.

Based on literature, advantages for deploying negotiation are best classified in four categories. They are conflict handling, shared vision, cooperation, and knowledge. Negotiation contributes to **conflict handling** because it facilitates conflict detection and resolution [12]. Before an actual conduct of negotiation, requirements statements are examined to identify conflicts by analysing stakeholders' goals and preferences. The EasyWinWin negotiation approach identifies conflicts manually and relies on the knowledge and expertise of the involved stakeholders and the capabilities of the facilitator [13]. Other researchers have tried to automate or partially automate the task of understanding requirements conflict. For example, Egyed and Grunbacher[14] presented an approach for identifying

conflict and cooperation among requirements based on software attributes and automated traceability. Another example is from Kaiya[15] who introduced a systematic approach to identify conflict through preference metrics in AGORA (attributed goal-oriented analysis). Sequentially, the identified conflicts are then negotiated to seek mutually beneficial solutions that are acceptable by the stakeholders. The negotiation contribution towards conflict handling is also proven empirically by several researchers through experiments [6, 9, 16]. Suppressing or overlooking conflicts is risky and might have serious negative effects on the software development process. Understanding requirements conflicts is thus an important strategy to mitigate software development risks. This is supported by much literature emphasizing the importance identifying and analysing conflicts for the success of system development [17-21].

Negotiation also promotes **shared vision** among multiple stakeholders. The negotiation process addresses the stakeholders' concerns and thus establishes shared vision to achieve mutual understanding. This is supported by other researchers as they also claimed that one of the negotiation benefits is to establish shared vision [9, 22, 23]. Throughout a negotiation process, stakeholders share their interests of the requirements they need and thus provide understanding to other stakeholders. This process allows various stakeholders to acknowledge others' concerns for the benefits of the system to be developed. Usually, stakeholders contribute incomplete, vague, and often inconsistent statements and ideas about their objectives, assumptions, and expectations. As they work together to negotiate their requirements, they give the project shape, and their merged visions emerge into a system that other stakeholders can accept. If, on the other hand, the stakeholders do not negotiate together, there is little chance the resulting system will accommodate their needs and the project will often fail. Negotiation is, therefore, essential to achieve mutually satisfactory agreements.

The shared vision and the satisfactory agreement increase the level of **cooperation** and trust among the stakeholders. As negotiation processes explore the stakeholders concerns, needs and visions and their ideas towards developing a reliable and workable system are acknowledged. The acknowledgement leads to cooperation as the agreement is a group decision which recognize the various stakeholders viewpoints [24]. This is also proven by empirical study and reported experience in literature [6, 16, 25]. The cooperation among the stakeholders is important to support the development process along the way and to ensure the success of the system being developed. At the end of the day the developed system provides functions the stakeholders need to assist their business process.

Further, the negotiation process improves the shared **knowledge** gained by the stakeholders. Usually, stakeholders state their needs towards the intended system with an implicit knowledge of their own work. A statement can be easily misinterpreted or misunderstand by the others. Through the negotiation process, stakeholders

need to explain and elaborate their requirements in order to provide understanding to others. In addition negotiation invokes the exploration of solutions before reaching agreement. Also, through negotiation, stakeholders are forced to justify the need of the requirements they request and the rationale of having the said requirement. The negotiation process therefore narrows the knowledge gap and reveals the tacit knowledge of the multiple stakeholders [5, 9, 26, 27].

3 THE MECHANISM OF NEGOTIATION PROCESS

This section explains the mechanism of negotiation process implemented in this research. Explains here are the negotiation activities, concepts and terms used throughout the research and related works which motivates the framework design.

3.1 Negotiation Activities

Explained here is an overview of the interactions that happen to produce agreed requirements through negotiation. Details of the components of negotiation are as follows:

- *Define and share the glossary* – This process allows the stakeholders to define and to share the meaning of important keywords. A clear and explicit definition yields the same interpretation used in the requirements statements. The same interpretation is useful to assist multiple stakeholders to understand definitions without ambiguity. There are at least two literature which support the fact that sharing the glossary is important to prevent inconsistency in interpretation [13, 15]. EasyWinWin methodology comprises activities of gathering, elaborating, prioritizing and negotiating requirements. Additionally, in order to avoid the occurrence of misinterpretation, EasyWinWin includes the 'capture a glossary of term' sub-activity wherein stakeholders can define and share the meaning of important terms and words appearing in the requirements statements. AGORA adopts a scoring technique that initially focuses on vertical conflicts in preference matrices in order to systematically find discordances in interpretations.
- *Identify conflicts* – Negotiation process focuses on conflicts identification to gather the attention of the stakeholders on problematic requirements. This effort motivates them to work together in order to find a resolution. Conflicts do not necessarily contain defects but may contain possible defects which are worth unfolding, justifying and assessing thoroughly. There are at least three literature which agree and prove that conflict identification is useful to identify possible defects, which in turn leads to resolution. Boehm [23] who introduced EasyWinWin as a tool based on negotiation methodology incorporates an activity called '*Identify Issues, Options, and Agreements'* to register conflicts as a foundation from which to propose resolution options and therefore provides the foundation to negotiate agreements. Kaiya[28], who introduced AGORA provides

systematic conflict identification through preference matrices value, also proved that conflict identification is useful to identify which requirement should be improved and refined. Robinson et al [29] introduced conflict-oriented approach to identify and to remove conflicts in order to have a better structure requirements.

- *Share perspectives, views, and expectations of the requirements* – This process allows the stakeholders to clarify and to further elaborate the requirements statements. The clarification and further elaboration of the requirements revealed the stakeholders perspectives, views and expectations towards the requirements statements. Grunbacher [9] stated that people know more than they can tell. Also, implicit stakeholders' goals, hidden assumptions, unshared expectations often result in severe problems in the later stages of software development. There was at least one literature on negotiation method which supported the fact that sharing perspectives, views, and expectations of the requirements was important to reveal tacit knowledge. EasyWinWin includes the '*Brainstorm stakeholder interests*' to allow the stakeholders to share their goals, perspectives, views, and expectations by gathering statements about their win conditions. This activity had proven beneficial during implementation using real-world negotiation as reported in [9].
- *Assess the system feasibility* – This process allows the stakeholders to assess the system feasibility from the perspective of resource feasibility and dependency feasibility. Resource feasibility means that the requirements are assessed if the subset of requirements can be built within time and cost constraints. While dependency feasibility means that all requirements in the subset are assessed if all the dependencies are included in the subset. This effort assists the stakeholders to make informed decision on the practicality of the agreed set of requirements. There were at least two literatures on project management which supported the fact that assessing system feasibility was important to ensure the success of the software project [30, 31]. The literature discussed the scenario of the possibility of an infeasible system if the project resources were not considered during requirements engineering process.
- *Justify the requirements needs* – This process allows the stakeholders to justify the needs and the importance of the requested requirements. The stakeholders need to think through the requirements and consider why one requirement is more important than the other in order to justify them to other stakeholders. There was at least one empirical evidence which supported the fact that requirements justification forced the stakeholders to think thoroughly on the requirements need and importance. In a small team, negotiation is exercised and based on observation; it is reported [32] that the negotiation process forced the participants to justify the need on every request demanded in order to gain other participants' understanding.

- *Prioritize the requirements* – This process allows the stakeholders to prioritize the requirements statements, to define and narrow down the scope of work and to gain focus. Prioritization makes it possible to gauge the importance a client feels regarding each requirement in respect of a software solution being able to fulfil their needs. There was at least one literature on negotiation method which supported the fact that requirements prioritization managed to narrow down the focus which assisted in group agreement [13]. Through prioritization [33], if it is not feasible to complete all of projects requirements, it is still possible to see which requirements are most important to the customer and implement those before the less important ones. This means that a project which has not had all its requirements fulfilled can still be of high value to a customer when it has fulfilled the customers' most important requirements. In an example, Karlsson et al [34] showed that 94% of the project value can be delivered for about 78% of the possible maximum cost.

3.2 The Underlying Concept

Mohammed [35] stated that having agreement between parties is paramount. Negotiation is deployed in this research to achieve agreement between the system's stakeholders in order to identify a set of requirements to be developed. Negotiation is usually understood to be a bargaining process between two or more parties to identify or to resolve people's needs of a system. A common bargaining process is between customer and developer to agree on the requirements to be developed and the project cost and time. The objective is to achieve an agreement on a business deal and then to proceed with the agreed software development.

Four key concepts need to be emphasized here as these are the concepts applied in the research:
Consensus-based negotiation is applied in this research in which the system's stakeholders, working together, reach group objectives rather than compete against each other. The group objective is mainly the development of a system which benefits the organization and at the same time represents the key stakeholders' perspectives and perceptions [27]. The main concern with regards consensus is not to reach unanimity but rather that all the stakeholders are committed to accept the consensual decision and feel that their perspectives and ideas are acknowledged in a cooperative manner. The consensus decision making is adopted because it is based on the belief that each stakeholder has some part of the truth while no one person contributes all. It is also based on a respect that all persons involved in the decision making be considered. Consensus enables a group to take advantage of all group members' ideas. It is a reasonable expectation that a decision based on a combination of thoughts would be of a higher quality than any individual decision. Choudhury et al [36] stated that working in a group provides a wide range of advantages by sharing information, generating ideas, making decisions and reviewing the effects of the decisions. Ideally, the group will reach a better decision than

an individual because collective knowledge and expertise of the group is greater than that of any individual. Further, people are more likely to implement and accept decisions they have accepted by consensus [36, 37]. **Consensus-based negotiation** may be summarized as:

- Agreement on the decisions by *all* the key stakeholders;
- Acceptance of consensual decision and acknowledgement of the stakeholders' perspectives and ideas;
- Respect for all persons involved in the decision making process;
- Use of the collective knowledge and skills of the group and
- Creation of collaborative environment among the stakeholders.

The **stakeholder** is a term that refers to any person or group who will be affected by the system directly or indirectly. Stakeholders include end users who interact with the system and everyone else in an organization that may be affected by its installation. Other system stakeholders may be engineers who are developing or maintaining related systems, business managers, domain experts and trade union representatives [38]. However, it is inappropriate and impossible to have all of the system stakeholders in the requirements elicitation process. It is impractical to involve a huge number of people in a face-to-face negotiation process. Negotiations practice [6] usually involves the key stakeholders (also known as success-critical stakeholders) to determine success. These stakeholders are the key people to represents their group interests and may include the end users, the system owner and managers who collaborate and are actively involved in decision making to achieve mutually satisfactory agreements. Therefore during the empirical study, only the key stakeholders involve to represent the key people.

The **'silent objective'** is enforced to the empirical study. It means the researcher's purpose for the experiments will be not revealed to the participants performing the negotiation. The 'silent objective' is not revealed in the experiments' instruction to the participants. This 'silent objective' is employed to allow the participants to merely exercise negotiation without knowing the underlying objective of the researcher. If the objective is revealed, the participants will tend to prioritise wrongly by striving to achieve the objective without having negotiation. This is to ensure that this research is purely assessing the negotiation process and the results obtained from that process.

A **defect** is defined by the researcher as summarized here. The literature is rife with inconsistent usage of this term. For example, McConnell [39] makes no distinction between errors and defects in the examples he cites in his book. On the other hand, Humphrey [40] elaborately states a bug is a defect but not all defects are bugs, and all defects result from errors but not all errors produce defects. Even the software measurements collected by authoritative organizations reflect a lack of consensus; Christensen et al [41] stated that NASA and DoD used the term "defects" while the Software Engineering Laboratory refers to "errors" and the Army refers to

"faults" and "anomalies". Pressman [42] define defect as a deviation between the specification and the implementation, detected after release to the customer (or the next activity in the software process). This is supported by a definition in IEEE [43] and SWEBOK [44] in which the standard define defect as product anomaly and a quality problem discovered after the software has been released to end-users respectively. These definitions fit the big picture of software development in which the specification can be checked against the end product to recognize the existence of defect or not.

This research is using the term defect to represent requirements defect which may occur during requirements elicitation process. However, defect in this research refers to the nonconformance of requirements at a very high level of requirements elicitation phase. As this research has a limitation within RE phase only, the general definition of defect as stated above is not appropriate. The requirement defect at this stage is checked within the written requirements statement and against the conformance of the stakeholders needs. Aligned with that, at this stage, only a number of defect attributes which associate with several quality attributes is relevant. Lauesen et al [45] looked into the effort to prevent defects early in the process life-cycle, defined requirement defect as "although the product works as intended by the developers, the users and customers are not satisfied with it. They may find it too difficult to use or unable to support certain user tasks. Unstated user expectations (tacit requirements) and misunderstood requirements are typical examples". Similar research [46-48] which looks into requirements defects in this early stage line up more or less the same defect attributes in their research. Therefore, by definition, a defect is a nonconformance of requirements in requirements statements and customers' needs based on the requirements comprehensibility, completeness, consistency, feasibility and correctness. Customers' needs are represented by the high level requirements statements listed as agreed requirements following negotiation.

3.3 Related Works

This sub-section elaborates the motivation which influenced the negotiation process introduced in this research. The process was designed to provide negotiation facility during the requirements elicitation process among multiple stakeholders. The basic features were conflict detection and resolution, requirements exploration and requirements prioritization to assist in achieving group decision. Discussed below are current methods and techniques which motivate the negotiation process introduced in this research.

In terms of conflicts and misinterpretation detection, EasyWinWin [23] is identified as a useful negotiation methodology with collaborative tools which provides electronic brainstorming, categorizing and polling. It includes the "capture a glossary of terms" sub-activity wherein stakeholders can define and share the meaning of important terms and words appearing in the requirements statements. This effort requires the

stakeholders to create a record of the glossary. Once the glossary is recorded, it can be viewed by all the stakeholders involved in the negotiation. Also, EasyWinWin has a tool called quality attribute risk and conflict consultant (QARCC) which systematically provides suggestions to the stakeholders regarding the possibilities of potential conflicts by using a knowledge base. In the knowledge base, pairs of conflicting quality attributes are stored. However, the success of this approach largely depends on the quality of the knowledge base and in general it is a huge effort to build such a knowledge base. The development of the knowledge base needs project backgrounds, documentations and histories of previous projects to allow archiving and mapping on the potential conflicts. This effort is useful if only the organization has the history of previous projects. What if a knowledge base is not available? The approach introduced to detect conflicts in this research does not require such a knowledge base in advance. This research adopts a scoring technique to systematically detect the conflicts. In this activity, individual stakeholder need to assign a score value for every requirement based on individual preference. Whenever the scores differ, there are conflicts. Even though the approach used in this research does not require knowledge base as in EasyWinWin, the benefit of knowledge sharing among the stakeholders emphasized in EasyWinWin is noted.

Attributed Goal-oriented Analysis (AGORA)[15] introduced a scoring technique that focused on vertical conflicts and diagonal conflicts in preference matrices. Vertical (off diagonal) conflicts systematically find conflicts in interpretations and diagonal (the main diagonal of the matrices) conflicts systematically find conflicts in stakeholders' interest. However, AGORA requires a well trained facilitator to facilitate the requirements elicitation process who understands how AGORA works and who is capable of handling the entire process. Also, during the process with AGORA, the stakeholders need to guess what other stakeholders think of every requirement and assign a score to it. If the variance of the score is high, it is believed that there might be conflicts in interpretation with the requirement and further elaboration is required. On the other hand, the approach in this research does not require a trained facilitator to assist the elicitation process because neither tools nor complicated graphs nor matrices are used.

This research adapts and simplifies the scoring technique in AGORA [28] to detect the conflicts in preference among multiple stakeholders. The vertical conflicts which identify interpretation issues are not included as misinterpretation and inconsistent conflicts are managed in the face-to-face negotiation process which reveal tacit information and shared common understanding.

The scale of scoring technique used in this research is adapted from the MoSCoW technique [49]. MoSCoW is a prioritisation technique used in business analysis and software development to reach a common understanding with stakeholders on the importance they place on the delivery of each

requirement. The capital letters in MoSCoW stand for:
M - MUST have this.
S - SHOULD have this if at all possible.
C - COULD have this if it does not affect anything else.
W - WON'T have at this time but WOULD like in the future.

Table 1: The Scale for Requirements Prioritization

Scale	Meaning
4	Must have this
3	Should have this if at all possible
2	Could have this if it does not affect anything else
1	Will not have this time but would like in the future
0	Must never have this

In this research, this method was converted into a numbered scale from 0 to 4 in which an item was added to scale 0 meaning 'Must never have this.' This item was introduced to provide an option if the stakeholders do not want the particular requirement to be included. This is possible in a circumstance of requirements which are requested by a stakeholder but is not wanted by the other. For example, lecturers would like to have their students' photos to be tagged along the electronic report card for prompt recognition but on the other hand the students are not comfortable to have their photos online. In this example, the lecturers' representative suggests a requirement to have the students' photos online but the students' representative choose to exclude the requirements. Hence, the 'Must never have this' is the best option to represent the students' preference. Table 1 state the scale used in this research.

A cycle of explanation and elaboration in the negotiation phase in this research is designed to promote understanding, to allow the stakeholders to make informed decisions and therefore achieve consensus. This approach is influenced by Delphi technique which is usually used to survey and to collect the opinions of experts. The Delphi technique is widely used and accepted method for gathering data from respondents within their domain of expertise. The technique is designed as a group communication process which aims to achieve a convergence of opinion on a specific real-world issue [50, 51]. The strength of Delphi, in contrast to other data gathering and analysis techniques, employs multiple iterations designed to develop a consensus of opinion concerning a specific topic via questionnaires. It is noted that Delphi usually keeps the stakeholders isolated. However, this research adapted the iterative process of Delphi to converge the stakeholders' opinions in face-to-face iteration format. This activity allows information sharing emphasizing the justification of the "need" or "not need" of the software requirements.

4 THE EMPIRICAL FRAMEWORK DESIGN

This section describes the framework design deployed in the research based on guidelines by Kitchenham et al [52]. It provides guideline and control on the population being studied, the rationale for sampling from that population, the process for allocating and administering the empirical study, and threats to validity to the study. Throughout this

section, empirical study is mentioned several times but is not reported in this paper as it focuses on the framework design and the theory behind it.

4.1 The Subjects

This sub-section defines the population from which the participants for the study were drawn, the process by which the participants were selected and the process by which the participants were assigned to the study

The study was done in a series of tutorial sessions at The University of Western Australia. Two course units with at least 20 people each were involved in two semesters to allow several trial runs and the actual study to take place. The units were Software Requirements and Project Management (CITS3220) and Software Engineering Industry Project Leadership (CITS4222). The units shaped the students to become effective team members, undertake problem identification, formulation and solution and apply their knowledge of basic science and engineering skills

These two units were identified as the most suitable units to provide the right group of students with the right level of knowledge to deploy the study for this research. These were students with a software engineering knowledge background. Particularly they were equipped with the theory and concept of negotiation through formal lecture before the study. Some had working experience in software development.

In order to avoid the presence of bias, the participants' assignment to the groups and to the role they were playing was random. The participants who had special ability, such as people with working experience or a high achiever, were identified by the unit coordinator and divided evenly among groups. This was done to avoid the possibility of having a distinguish group which consist of brilliant participants who would produce very good negotiation results. Good results may not represent the effectiveness of negotiation but simply the participants' intelligent guesses. Hence, in this research, on top of random group assignment, extra effort to avoid the presence of bias is necessary.

In addition, a role play empirical study always comes with the dilemma of whether the participants are really playing a role or simply incorporating their personal judgment. Expecting that each participant would be more committed to a specific priority when given a clear role and in order to minimize that possibility, the participants were given instruction and guideline on how to play the role of the system's stakeholder. In addition, to assist the participants to feel the responsibility of being the system's stakeholders, the description scenario and the candidate requirements were given to them in advance. These reading materials helped because the description scenario described the need of the system and the concern of different stakeholders and the candidate requirements were carefully tailored to the specific stakeholder's needs. In addition, observation done by the researcher, her supervisor and unit coordinator throughout the experiment discovered that most of the participants were playing the role given to them; this

is due to the peer assessment for the unit of the tutorial session where the study was done.

4.2 Empirical Study Procedures

This sub-section defines the empirical study unit, describes the study design and explains the procedures.

Each unit was a group of four or five participants exercising negotiation. The number of groups available for each study was treated as a replication of the treatment. Every study involved four to six groups exercising negotiation. Hence, negotiation was essential and exercised by all the groups. The results from the study produced a list of software requirements which had been negotiated among the participants within a group and measured respectively.

Initially in the empirical study procedures, all handouts such as the instruction sheet, the description scenario, the candidate requirements and the decision forms, were given to the participants. Next, a briefing on the background knowledge of the experiment took place. This was followed by instructions which were supported by a sample overview of step-by-step activities. The participants' assignment to the groups with the role to play during the experiment was then given and followed. Ample time was given to the participants to understand the roles and the candidate requirements prepared for them. The participants were then asked to make an individual decision based on resource constraints on which requirements should be implemented. This activity acted as a control situation in which decisions were made individually and obviously no negotiation was involved. It also provides a basis for systematic conflicts detection. This was then followed by a negotiation to achieve a group decision. When the consensus was achieved or the time limit ended, the decision forms were collected and the post mortem was deployed. In the post mortem session, feedback from the participants was gathered to learn if the study was successful and to note weaknesses, if any, for future references.

4.3 Threats to Validity

First the 'silent objective' was defined as in Section 3.2. The participants should not have been aware of the aims and measurement being employed. The purpose was to hide the desired outcome of the experiments which might have influence the participants' decisions. This is usually known as "blind experiments" to prevent participants' expectations from influencing the results [52]. On top of this, the variables which were identified to be measured in the study such as the agreement level and the quality values were unknown to the participants. The silent objective was enforced to let the participants focus only on exercising negotiation in order to achieve group decision without considering the variables to be measured from the output.

Second was the double measurement for the requirements quality. In a series of studies to measure the quality of requirements, the requirements produced by the negotiation effort were discussed, tested, analysed and proven twice. Two types of methods and measurements were deployed separately with different

groups of participants; and yet produced similar result that is improvement in quality. The double measurement was seen to give a redundant check and to support one method with another.

Third was the blind marking. Kitchenham et al [52] stated that a researcher's enthusiasm for their own work may bias the trial. Therefore, a third party was involved to assist the researcher to collect and to mark the results purely based on the data collected. It was then analysed and measured by Cohen's kappa. Cohen's Kappa [53] is an index of inter-rater reliability that is commonly used to measure the level of agreement between two sets of dichotomous ratings or scores. The measurement involved an independent statistician, who ensured that the results were represented and reported correctly.

5 CONCLUSION

It is crucial to ensure that the process of empirical study is carefully designed. This is to guarantee that the data being collected is reliable to support the underlying theory. Therefore, the framework elements which, consist of the identification of population and participants, the flow of empirical study procedures and threats to validity are crucial items to ensure the reliability of the study output. Besides that, since negotiation effort is seldom applied during the requirements elicitation process, the relevance and the advantages of negotiation are explained and argued. In conclusion, this paper provides a dynamic fundamental framework on how to go about deploying empirical study in requirements elicitation with negotiation.

6 REFERENCES

1. Ahmad, S., M. Reynolds, and T. Woodings. *Understanding Requirements Engineering*. in *International Conference on Engineering and ICT*. 2007. Melaka, Malaysia.
2. Damian, D.E.H. and D. Zowghi. *The impact of stakeholders' geographical distribution on managing requirements in a multi-site organization*. in *IEEE Joint International Conference on Requirements Engineering*. 2002. Germany: IEEE Computer Society.
3. Zowghi, D. and C. Coulin, *Requirements Elicitation: A Survey of Techniques, Approaches and Tools*, in *Engineering and Managing Software Requirements*, A. Aurum and C. Wohlin, Editors. 2005, Spriner-Verlag: Berlin. p. 19-41.
4. Grunbacher, P. and N. Syeff, *Requirements Negotiation*, in *Engineering and Managing Software Requirements*, A. Aurum and C. Wohlin, Editors. 2005, Springer-Verlag: Berlin. p. 143-158.
5. Al-Karaghouli, W., S. AlShawi, and G. Fitzgerald, *Negotiating and Understanding Information Systems Requirements: The Use of Set Diagrams.* Requirements Engineering, 2000. **5**(2): p. 93-102.
6. Boehm, B. and A. Egyed. *Software Requirements Negotiation: Some*

Lessons Learned. in *20th International Conference on Software Engineering*. 1998. Kyoto, Japan: IEEE Computer Society.
7. Damian, D.E.H. and D. Zowghi. *An insight into the interplay between culture, conflict and distance in globally distributed requirements negotiations*. in *36th Annual Hawaii on International Conference*. 2003. Big Island, Hawaii.
8. Davis, A.M., *Just Enough Requirements Management*. 2005, New York: Dorset House.
9. Grunbacher, P. and R.O. Briggs. *Surfacing Tacit Knowledge in Requirements Negotiation: Experiences using EasyWinWin*. in *34th Hawaii International Conference on System Science*. 2001. Hawaii: IEEE Computer Society.
10. Mohan, K. and B. Ramesh, *Traceability-based knowledge integration in group decision and negotiation activities.* Decision Support Systems, 2007. **43**(3): p. 968-989.
11. Nuseibeh, B. and S. Easterbrook, *Requirements engineering: A Roadmap*, in *Conference on The Future of Software Engineering* 2000, ACM Press: Limerick, Ireland p. 35 - 46
12. Damian, D.E.H., *Challenges in Requirements Engineering*, in *Computer Science Technical Report*. 2000, The University of Calgary: Calgary.
13. Grunbacher, P. and B. Boehm. *EasyWinWin: A Groupware-Supported Methodology for Requirements Negotiation*. in *8th European Software Engineering Conference held jointly with 9th ACM SIGSOFT International Symposium on Foundations of Software Engineering*. 2001. Toronto, Canada.
14. Egyed, A. and P. Grunbacher, *Identifying Requirements Conflicts and Cooperation: How Quality Attributes and Automated Traceability Can Help.* IEEE Softw., 2004. **21**(6): p. 50-58.
15. Kaiya, H., et al., *Improving the detection of requirements discordances among stakeholders.* Requirements Engineering, 2005. **10**: p. 289-303.
16. Boehm, B. and I. Hoh, *Conflict Analysis and Negotiation Aids for Cost-Quality Requirements.* Software Quality Profesional, 1999. **1**(2): p. 38-50.
17. Boehm, B. and I. Hoh, *Identifying Quality-Requirement Conflicts.* IEEE Softw., 1996. **13**(2): p. 25-35.
18. Hans, W.N., et al., *Managing Multiple Requirements Perspectives with Metamodels.* IEEE Softw., 1996. **13**(2): p. 37-48.
19. Nuseibeh, B., *Conflicting Requirements: When the customer is not always right.* . Requirements Engineering, 1996. **1**(1): p. 70–71.
20. Curtis, B., H. Krasner, and N. Iscoe, *A field study of the software design process for large systems.* Communications of the ACM, 1988. **31**: p. 1268-1287.
21. Nuseibeh, B., J. Kramer, and A. Finkelstein. *ViewPoints: meaningful relationships are difficult!* in *Software Engineering, 2003. Proceedings. 25th International Conference on*. 2003.
22. Lee, M. and B. Boehm, *The WinWin Requirements Negotiation System: A*

Model-Driven Approach. 1996, CiteSeer.
23. Boehm, B., et al. *Software requirements as negotiated win conditions*. in *Requirements Engineering, 1994., Proceedings of the First International Conference on*. 1994.
24. Darke, P. and G. Shanks, *Stakeholder viewpoints in requirements definition: A framework for understanding viewpoint development approaches.* Requirements Engineering, 1996. **1**(2): p. 88-105.
25. Hoh, P.I. and D. Olson, *Requirements Negotiation Using Multi-Criteria Preference Analysis.* Universal Computer Science, 2004. **10**(4): p. 306-325.
26. Robinson, W.N. and V. Volkov, *Supporting the Negotiation Life Cycle.* Communication of ACM, 1998. **41**(5): p. 95-102.
27. Price, J. and J. Cybulski, L. *The Importance of IS Stakeholder Perspectives and Perceptions to Requirements Negotiation*. in *Australian Workshop on Requirements Engineering*. 2006. Adelaide.
28. Kaiya, H., H. Horai, and M. Saeki. *AGORA: Attributed Goal-Oriented Requiremnts Analysis Method*. in *IEEE Joint International Conference on Requirements Engineering*. 2002. Essen, Germany.
29. Robinson, W.N. and V. Volkov, *Conflict-Oriented Requirements Restructuring*, in *GSU CIS working paper*. 1996: Georgia State University, Atlanta.
30. Atkinson, R., *Project management: cost, time and quality, two best guesses and a phenomenon, its time to accept other success criteria.* International Journal of Project Management, 1999. **17**(6): p. 337-342.
31. Boehm, B., *Software Engineering Economics*. 1981, New Jersey: Prentice Hall.
32. Smith, M. (2005) *A Small Experiment in International Negotiations: Chuo Law School, Japan and Chulalongkorn Law Faculty, Thailand*. **19**, 216-220.
33. Hatton, S. *Software Requirements Prioritisation: The Client's Perspectives*. in *Fifteenth University of Western Australia, School of Computer Science & Software Engineering Research Conference*. 2006. Yanchep, Western Australia: CSSE, University of Western Australia.
34. Karlsson, J. and K. Ryan, *Supporting the Selection of Software Requirements*, in *Proceedings of the 8th International Workshop on Software Specification and Design*. 1996, IEEE Computer Society.
35. Mohammed, S. and E. Ringseis, *Cognitive Diversity and Consensus in Group Decision Making: The Role of Inputs, Processes, and Outcomes.* Organizational Behavior and Human Decision Processes, 2001. **85**(2): p. 310-335.
36. Choudhury, A.K., R. Shankar, and M.K. Tiwari, *Consensus-based intelligent group decision-making model for the selection of advanced technology*. Decision Support Systems, 2006. **42**(3): p. 1776-1799.
37. Price, J. and J. Cybulski, *Consensus Making in Requirements Negotiation:The Communication Perspective.* Australasian Journal of

Information Systems, 2005. **13**(1): p. 209-224.
38. Sommerville, I., *Software Engineering 7th Edition*. 2004, U.S: Addison-Wesley.
39. McConnell, S., *Rapid Development: Taming Wild Software Schedules*, ed. W. Redmond. 1996: Microsoft Press.
40. Humphrey, W.S., *Managing the software process*. 1989: Addison-Wesley Longman Publishing Co., Inc. 494.
41. Christensen, M.J. and R.H. Thayer, *The Project Manager's Guide to Software Engineering's Best Practices*. 2002, Los Alamitos, CA.: IEEE Computer Society Press.
42. Pressman, R.S., *Software Engineering A Practitioner's Approach 6th Edition*. 2005, New York: McGraw Hill.
43. *IEEE Standard Glossary for Software Engineering Terminology. IEEE Standard 610.12-1990. .* 1990, IEEE Computer Society.
44. *Software Engineering—Guide to the Software Engineering Body of Knowledge (SWEBOK)*. 2007, Standards Australia.
45. Lauesen, S. and O. Vinter. *Preventing Requirement Defects*. in *Sixth International Workshop on Requirements (REFSQ'2000)*. 2000. Stockholm.
46. Biffl, S., B. Freimut, and O. Laitenberger, *Investigating the cost-effectiveness of reinspections in software development*, in *Proceedings of the 23rd International Conference on Software Engineering*. 2001, IEEE Computer Society: Toronto, Ontario, Canada.
47. Biffl, S. and M. Halling, *Investigating the defect detection effectiveness and cost benefit of nominal inspection teams.* Software Engineering, IEEE Transactions on, 2003. **29**(5): p. 385-397.
48. Halling, M., S. Biffl, and P. Grünbacher, *An economic approach for improving requirements negotiation models with inspection* Requirements Engineering, 2003. **8**(Number 4): p. 236-247.
49. *MoSCoW Prioritisation*, in *Reducing Your Acceptance Testing Risk*. 2007, Coley Consulting.
50. Linstone, H.A. and M. Turoff, *The Delphi Method: Techniques and Applications* 1975: Addison-Wesley Pub. Co., Advanced Book Program
51. Hsu, C.C. and B.A. Sandford, *The Delphi Technique: Making Sense Of Consensus.* Practical Assessment, Research & Evaluation, 2007. **12**(10).
52. Kitchenham, B., et al., *Preliminary guidelines for empirical research in software engineering.* IEEE Trans. Softw. Eng., 2002. **28**(8): p. 721-734.
53. Cohen, J., *A coefficient of agreement for nominal scales.* Educational and Psychological Measurement, 1960. **20**(1): p. 37-46.