Case Study on Requirements Management Tool for Small and Medium Software Projects

Azida Zainol[1†] and Amani Tariq Jamal[2††]

[†]University of Jeddah, Jeddah, Saudi Arabia
[††]King Abdulaziz University, Jeddah, Saudi Arabia
[†]University Utara Malaysia, Malaysia

Summary
In the previous decades, there are many requirements management tools available in the market. However, these tools are expensive, complicated, difficult to learn and too sophisticated for small and medium projects that have resources and budget limitation. Thus, a requirements management tool, known as Requirements Management Tool for Small and Medium Projects (RMT-SMP) is developed on open source platform targeting the small and medium software projects. This paper presents an evaluation of using RMT-SMP during software development projects for small and medium projects in the real industry. This work is steered based on empirical methods in software engineering using case study. In software engineering, case studies are used for validating research, for example, evaluation of new tools, processes, or methods. Thus, the case study research design components are research questions, preposition or hypothesis, unit of analysis, determination of how data are linked to prepositions and criteria interpret the findings. In order to establish quality of case study, we conducted construct validity, internal validity, external validity and reliability. The result of this case study has shown the success of applying the RMT-SMP during software project development for small and medium projects and can be concluded that RMT-SMP is practical and feasible for the small and medium projects. The RMT-SMP encourages the practitioners to have a better approach in managing their requirements during software development projects.

Key words:
Requirements Engineering, Requirements Management, Requirements Management Tool, Case Study on Requirements Management

1. Introduction

These days, developing software projects is becoming difficult and challenging. The majority of software development projects in the USA will take longer, cost more than planned and result in “out of specification‟ products that fail to meet user requirements [1]. Moreover, a report on a survey of over 3800 organizations in 17 European countries concluded that more than 50% of the perceived software problems were in the area of requirements specification and requirements management (RM) [2]. In a study [3] in twelve software companies have revealed that lack of skills and poor staff retention seem to have a significant impact on the capability of the requirements processes to produce good initial sets of requirements

Surveys conducted by [4], [5] that investigated why software projects fail show that projects do not fail for one single reason, but they fail for multiple reasons. Their result has shown that a common problem with failed projects was inadequate requirements when the delivery decision was made (73%) [4]. The customer did not spend enough time with developers to define the requirements properly; this can lead to unrealistic expectations. Then, because the initial requirements are poor, it is not surprising that there are scope changes during the project. Although the software world has changed significantly with several programming and development paradigms, poor requirements is still one of the reasons for software projects failure in this decade, which is also one of the main reasons for software failure in previous decades [1], [4], [5].

Requirements are volatile due to change in needs, processes and technology. This makes manual requirement management a challenging task. To overcome such problems, practitioners developed various tools to collect and manage software requirements [6]. Hence, there are many requirements management tools on the market that claim to support the requirements management activities [6]–[8]. However, not all of these tools on the market are focused solely on requirements management activities.

The use of requirements management tools has become essential when considering the size and complexity of development efforts [9], [10]. There are some commercial off-the shelf- requirements management tools such as IBM Rational DOORS [11] and Rational Requisite Pro [8]. However, these tools use different concepts; have different capabilities and differing degrees of maturity with respect to their applicability in system engineering projects [12] and are more suitable for large sized of projects. In addition, from a website survey by the Incose Group [13] it was revealed that most of the requirements management tool are not focused solely on managing requirements, are difficult to use and expensive, therefore, they are more

Manuscript received Fabruary 5, 2020
Manuscript revised Fabruary 20, 2020

suitable for larger applications. As a result, there is a need for requirements management tool that is available for free, suitable for small and medium projects that have limited resources. Hence, a requirements management tool known as RMT-SMP which is developed on open source platform that is suitable for small and medium projects.

This study aims to explore within the real-life context whether or not the RMT_SMP does provide advantages to the practitioners. Hence, an industrial case study was conducted at Malaysian software industry. It is widely believed in the software engineering domain that real-life case studies are only suitable for an industrial evaluation of software engineering techniques and tools if they are organized and conducted in a sound way [14].

The main objective of this study is to investigate how the requirements management tool could offer a better way of managing requirements for small and medium projects in a real industry setting. The results obtained from this case study cannot necessarily be generalized to represent the Malaysian software industry need and cannot guarantee that similar success would be achieved in other applications because the data collected in the case study is from only two software development projects. However, in this case study, the RMT_SMP tool was found to be capable of promoting a better way of practising requirements management and would lead toward developing quality software within the allocated budget to deliver it at the right time.

2. RMT_SMP

The RMT_SMP is developed targeting for managing software requirements in small and medium software projects that have limited resources in terms of budget, human resources and capital. Hence, the elements of RMT_SMP that composed of general and specific elements are recognized rigorously in order to analyse the tools features [8]. The general elements include the general features that the RMT_SMP tool should have, whereas the specific elements are the requirements that specific for RMT_SMP.

2.1 General Elements

The general elements are important because they describe the features that the tool should accomplish in order to fit the software industry needs. Table 1 below presents the general elements for RM tools, follow by detailed explanations.

Table 1: The general elements

Elements	Description
Usability, simplicity and customization	The tool should be easy to use. Not too much training and administration needed. The tool should not create additional tasks and deployment should not require extensive customization.
Access control	The tool must have tight access control whereby each participant has appropriate access to the data. (Role-based, project-based and task based access control.)
Tailoring and Extensibility	The tool must be adaptable and extensible to the needs of the organization or project.
Description Free licensing and full version availability	The tool should be free licensing that allows the user to use the tool in full version without limitation.
Database centric	The tool should be database centric, but also support document management.

2.2 Specific Elements

The specific element is defined in the Table 2 below, followed by detail explanation.

Table 2: The specific elements

Elements	Description
Requirements identification	The tool should support the identification of requirements. The requirements ID, which is a number for each individual requirement is mandatory.
Requirements classifying and viewing	The tool must be able to classify requirements into logical user defined groups.
Requirements base-lining	The tool should be able to manage functional and non-functional requirements that the development team has committed to implement in a specific release.
Change control	The tool must : offer the possibility of handling formal change requests. Track all changes and kept in the database. The tool should be able to update the requirements document.
Version control	The tool should be able to identify: Requirements document versions Individual requirements versions
Status tracking	The tool has to : Define possible requirement statuses Record the status of each requirement Reporting the status distribution of all requirements.
Requirements tracing	The tool ought to : Define links to other requirements Define links to other system elements
Use Case specification generation	The tool must be able to generate Use Case specifications documents. The tool uses predefined document definitions to generate documents with current data from the database
List of requirements generation	The tool should be able to generate a list of requirements as a support documents.
Requirements linking to system elements	The tool should be able to keep functional requirements, the design components and code modules that address each requirement, and the test cases that verify its correct implementation.
Authentication procedure	The tool should allow individuals with different roles to log in to the tool. The tool should restrict its functions to the different users.
Project definition	The tool should allow a project to be defined in order to keep requirements separately from other projects.

Create user	The tool should be able to create user id and password with different roles. This is important for the user to log in and use the tool efficiently.

3. Methodology

3.1 Case Study Research Design

Case study has been chosen as a method to carry out our research since it is a scientific or empirical method used when we want to test whether RMT-SMP giving positive impact on software development in the real world in a specific context instead of having ad–hoc RM. This is because case study research has grown in reputation as an effective methodology to investigate and understand complex issues in real world settings [15]–[17]. In software engineering, case studies are used for validating research, for example, evaluation of new tools, processes, or methods. Thus, case study is designed based on 5 components [18] :

3.1.1 Research Questions

This research will address the following research question: RQ: How the best practices of requirements management that is applied into RMT-SMP tool has a positive impact on encouraging the RE practitioners to have a better approach for managing requirements in developing the small and medium software projects?

3.1.2 Propositions or Hypothesis

Hypothesis is an educated guess that keeps the research in the right direction [18]. The hypothesis of the case study is defined as:

1. The RE practitioners themselves indicate that the RMT-SMP has a positive influence on the requirements management practices when compared to ad-hoc requirements management practices.
2. The RMT-SMP is considered suitable for the given Malaysian software projects when the tool's features can meet the general and specific elements.

3.1.3 Unit of Analysis

There are two software development project size that are compared, that is, small and medium size software projects. A project is considered as small when overall atomic requirements are less than 500 and medium project when the number of atomic requirements is between 500-1000 [19], [20]. Based on these conditions and considerations, a project called E-Filing is identified at company Z (the name of the company is withheld for reason of private and confidential). Company Z is a semi-government agency and the case study is conducted at their Information Technology Department. As the development of E-filing involved 250 requirements, this is considered as small projects. Another project is identified in Company Y and it is known as Human Resource Management System (HRMS). The number of requirements is 650 and it is consider as medium project.
It is important to identify a small project similar to E-filling project and a medium project similar to HRMS. In addition, the small and medium projects should have the similar characteristics as E-filling and HRMS; and were previously carried out in company Z and Y. Thus, in company Z, a previously small project is identified as a Project Management and Monitoring System (PMMS). While in company Y, a prior medium project is known as Office Documents Management System (ODMS). The comparison is conducted for small projects; between E-filling and PMMS and medium projects; between HRMS and ODMS. In PMMS and ODMS, the requirements management practices were ad-hoc and there were no requirements management tool getting involved. On the other hand, in E-filling and HRMS, the requirements management practices are defined and using the RMT-SMP to manage their requirements. The results are significant to show the comparison.

3.1.4 Determination of How Data are Linked to Prepositions

Data collected during case study should be a reflection of the proposition and mapped to it [18]. Table 3 shows the metric for comparison used in this research, which reflects our proposition and research questions. There are a lot of other factors that interplay with each other to contribute to the success of requirements management, such as the knowledge of RE practitioners and management commitment. Moreover, there are also many variables that are required to be measured and controlled during the case study. However, in this case study, the focus is on the positive and negative effects that will bring the success of requirements management. This is important in order to identify the merits or problems based on the empirical evaluation under the context of the case study, since it is beyond the scope of the research to investigate the causal relationship of all factors interacting in the case study as well as in the research.

Table 3: Variables to be measured

No	Variables to be measured	Notes
1	Total number of (atomic) requirements in the final requirements specification	Atomic requirements are defined as lower level requirements with one specific function and cannot be further broken down into a lower function (Salzer, H. 1999).

2	Number of analysts involved	Analyst plays the role of requirements engineers as well
3	Number of developers involved	Analyst can also be a developer or tester when they are required by the project.
4	Number of original requirements	
5	Number of requirements deleted	
6	Number of requirements rejected	
7	Number of change request	
8	Number of requirements change approved	
9	Number of requirements change rejected	
10	Number of requirements change evaluated	
11	Number of requirements change verified	
12	Number of requirements change modified	
13	Number of requirements change completed	
14	Number of completed change request	
15	Project duration	Include the planned duration and actual duration
16	Effort in person-month	Can be calculated from variable 3 and 15
17	Cost overrun in terms of the Effort in person-month	Can be calculated from variable 3, 15 and 16
18	Software project budget	
19	The number of software product quality expectation	

3.1.5 Criteria to Interpret Finding

Any findings and conclusions will be made on the basis of data collected during case studies keeping in view the research questions and propositions along with the statistical analysis.

3.2 Criteria for Judging Quality of Research Design

There are four tests as described in [18] to establish quality of case study which are :

3.2.1 Construct Validity

Construct validity ensures correct operational measures chosen for the concepts being studied. Table 3 shows 19 variables that were measured in this research which reflects our research questions as well as proposition.

3.2.2 Internal Validity

Internal validity is inapplicable to case studies that are not concerned with causal situation. In our research, each inference is given its due consideration and rationale during the research design.

3.2.3 External Validity

Within case studies, it means that the results can be generalized to similar cases to those that were studied. In our research, two size of software development projects were compared to ensure our results can be used to generalizable.

3.2.4 Reliability

Reliability means that if the same procedures were employed on the same case study again (perhaps by another researcher), the researcher should arrive at the same results/findings earlier recorded. In our research, these steps were documented and executed.

3.3 Implementing and Monitoring the Case Study

This case study involves the development of two different software projects at different companies. Thus, it is important to monitor the projects' progress and compared the results with the plan. In addition, the authors involved directly with the projects in order to ensure the tool is conducted accurately. Although the tool is being introduced and trained with the team members, the authors keep on monitoring the team members when they used it. The data collected during the project development are summarized in the Table 4 for small projects and Table 5 for medium project.

Table 4 : Result from E-filling and PMMS

Variables measured		**E-Filing**	**PMMS**
Total number of (atomic) requirements in the final requirements specification		250	225
Number of analysts involved		4	4
Number of developers involved		4	4
Number of original requirements		130	120
Number of requirements deleted		25	0*
Number of requirements rejected		20	0**
Number of change request		10	0***
Number of requirements change approved		9	0***
Number of requirements change rejected		1	0***
Number of requirements change evaluated		9	0***
Number of requirements change Verified		9	0***
Number of requirements change Modified		9	0***
Number of requirements change completed (installed as work product)		9	0***
Number of completed change request		9	0***
Project duration	Planned	6 months	6 months
	Actual	6 months	9 months
Effort in person-month	Planned	24	24
	Actual	24	36

Cost overrun in terms of the Effort in a person – month	Number	0	12
	% over the total effort of the project	0	50%
Software project budget (RM)	Planned	10,000	15,000
	Actual	9,500	18,000
The number of software product quality expectation	Planned	6	6
	Actual	6	4
Notes: 0* indicates that the no requirement was deleted 0** indicates that requirement rejected was never recorded 0*** indicates that no requirements change management was conducted			

Table 5 : Result from HRMS and ODMS

Variables measured		**HRMS**	**ODMS**
Total number of (atomic) requirements in the final requirements specification		650	680
Number of analysts involved		6	6
Number of developers involved		6	6
Number of original requirements		500	490
Number of requirements deleted		50	0*
Number of requirements rejected		20	0**
Number of change request		17	0***
Number of requirements change approved		15	0***
Number of requirements change rejected		2	0***
Number of requirements change evaluated		15	0***
Number of requirements change verified		15	0***
Number of requirements change modified		15	0***
Number of requirements change completed (installed as work product)		15	0***
Number of completed change request		15	0***
Project duration	Planned	9 months	10 months
	Actual	9 months	18 months
Effort in person-month	Planned	54	60
	Actual	54	108
Cost overrun in terms of the Effort in person-month	Number	0	48
	% over the total effort of the project	0	80%
Software project budget (RM)	Planned	50,000	60,000
	Actual	48,000	75,000
The number of software product quality expectation	Planned	10	10
	Actual	10	7
Notes: 0* indicates that the no requirement was deleted 0** indicates that requirement rejected was never recorded 0*** indicates that no requirements change management was conducted			

4. Result Analysis

This section describes the result analysis from the quantitative analysis of using RMT-SMP for small and medium software projects. Furthermore, the qualitative analysis of RMT-SMP also been discussed in this section.

4.1 Quantitative Analysis

The data collected during the development of E-filling was compared with the previous project, PMMS, which did not have proper RM practices. Both of these projects have similar project attributes. Table 4 and Fig. 1 present the result of comparison between these projects. It can be seen that, both of the projects are similar in number of analysts and developers as well as having the same project duration. Even though the E-filling has 25% more requirements than PMMS, the E-Filling project was able to complete its development as planned, within the budget and met the software quality expectations. Moreover, the E-Filling did not have cost overrun, as it developed the project on time, while the PMMS project had a 50% overrun. From the budget allocated, it can be seen that the E-Filling project managed to be developed within the budget. On the other hand, the PMMS project failed to control the budget, as it exceeded the budget by approximately 20%. When comparing the software product quality, it also showed that the E-Filling project managed to meet the quality expectations, while the PMMS project failed to do that.

From Table 4, the PMMS project involved no designated repository for documenting the rejected documents. Although these were only rejected requirements, they might have been useful in the future. Additionally, the PMMS project did not have the changes management activity. So, when there were any changes in the requirements, it was difficult to handle. This became more difficult, if the changes occurred after the analysis phase.

The possible reasons behind these circumstances are that the E-Filling project incorporated proper practice when managing requirements, as well as having a tool to help the team members to conduct requirements management activity. Thus, this can be seen as formal evidence that by incorporating the best practices and having a tool when conducting the requirements management activity leads to delivering software within the budget, on time and without compromising the software quality expectations.

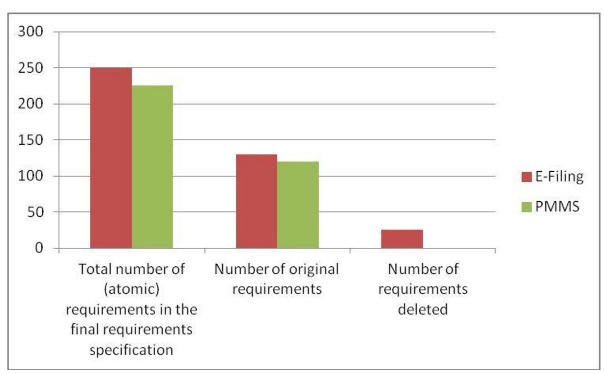

Fig. 1 Comparison of variables between E-filing and PMMS

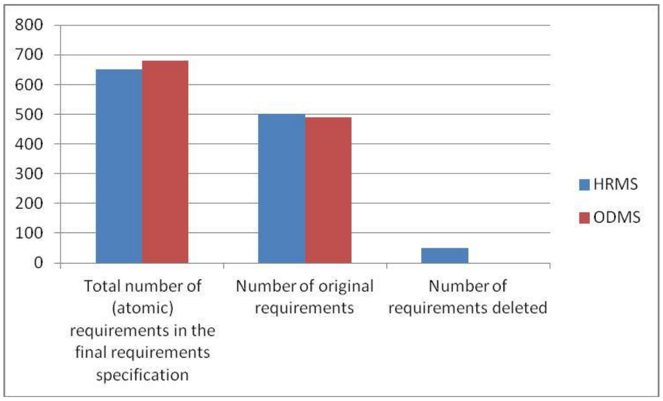

Fig. 2 Comparison of variables between HRMS and ODMS

The results of comparison for medium projects are presented in Table 5 and Fig. 2. The projects have almost similar project attributes but they are different in terms of managing their requirements management during software project development. In the HRMS project the requirements are carefully managed by using RMT-SMP while the ODMS project involves ad-hoc requirements management practices. Although both of the projects had the same number of analysts and developers, HRMS managed to deliver software on time and within budget. The ODHM has 30 atomic requirements more than HRMS and the duration is a month more than HRMS, however, ODHM did not successfully complete the projects on time. ODMS took about another 8 months to complete and was able to satisfy 7 software product quality expectations out of 10. The budget for developing the ODHM also overran. Figure 8-2 shows the number of requirements for HRMS and ODMS. It is clearly seen that ODMS did not compile the deleted requirements. The HRMS was able to manage all requirements change requests and that all the changes went through a well-managed procedure. However, in the ODMS project, the change requirements requests were not cautiously handled or documented. Thus, it can be concluded that the reason ODMS failed to deliver software on time and within budget was because ODMS did not have RM practices and the tool to support it. From the comparisons, it has clearly shown that RMT-SMP plays a vital role in managing requirements in small and medium projects.

4.2 Qualitative Analysis

In addition to quantitative analysis, a survey was conducted among the requirement engineers, project managers and developers who were involved in the software projects development. The objective of this survey is to evaluate the RMT-SMP using the ISO 9126 quality model. In order to construct questions that are related to the ISO9126 quality model, a Goal Questions Metric is used. The following sections review the ISO 9126 and Goal Questions Metric, which is then followed by the results.

4.2.1 Qualitative analysis

4.2.1.1 ISO 9126

The ISO 9126 standard was developed in 1991 by the International Organization for Standardization (ISO) in order to provide a framework for evaluating software quality, which was then refined over a further ten year period [21]. The standard is used as a tool to identify the quality considered in each application.
ISO 9126 standard is a constructive model for evaluation of the quality of basic information providing and rational decision making to avoid costly mistakes [22], [23]. ISO 9126 defines a quality model with six characteristics namely functionality, reliability, usability, efficiency, maintainability, and portability which are further subdivided into 22 characteristics [24], [25] as depicted in the Fig. 3 and the explanation in Table 7.

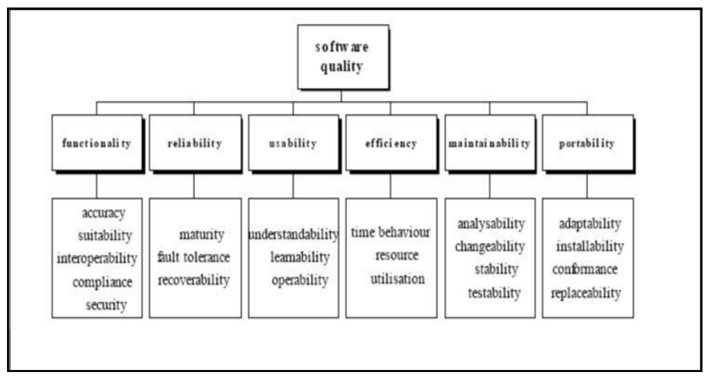

Fig 3 Software Quality ISO 9126 [26]

Table 6: ISO 9126 Characteristics and sub characteristics [27]

Characteristics	Sub Characteristics	Explanation
Functionality	Suitability	Can software perform the tasks required?
	Accurateness	Is the result as expected?
	Interoperability	Can the system interact with another system?
	Compliance	Is the system compliant with standard?
	Security	Does the system prevent unauthorized access?
Reliability	Maturity	Have most of the faults in the software been eliminated over time?
	Fault tolerance	Is the software capable of handling errors?

	Recoverability	Can the software resume working & restore lost data after failure?
Usability	Understandability	Does the user comprehend how to use the system easily?
	Learnability	Can the user learn to use the system easily?
	Operability	Can the user use the system without much effort?
	Attractiveness	Does the interface look good?
Efficiency	Time behaviour	How quickly does the system respond?
	Resource	Does the system utilize resources efficiently?
Maintainability	Analyzability	Can faults be easily diagnosed?
	Changeability	Can the software be easily modified?
	Stability	Can the software continue functioning if changes are made?
	Testability	Can the software be tested easily?
Portability	Adaptability	Can the software be moved to other environments?
	Installability	Can the software be installed easily?
	Conformance	Does the software comply with portability standards?
	Replaceability	Can the software easily replace the software?

4.2.1.2The Goal Question Metrics Approach

The Goal Question Metrics (GQM) approach was developed in the early 1980s by Victor R. Basili and his colleagues during their work at NASA Goddard Space Flight Centre for evaluating defects for a set of project [28]. It views the measurement process holistically by identifying the measures on the basis of measurement goals and interpreting them in order to access the level of achievement of the identified goals [29].

GQM has been defined in terms of three main levels, namely conceptual, operational and quantitative. In the conceptual level, the GQM defines the goals for an object so that further action is taken in the development of questions and related metrics. The operational level serves as a link between the operational level and the quantitative level. In this level, questions are developed to clarify and elaborate the goals in order to provide a base for identification of metrics. The quantitative level is the metric base which helps in the identification of those metrics. It is a set of data that is associated with every question in order to answer it in a quantitative way.

4.2.1.3 The Result

This section describes the result of the questionnaire; arrange by the ISO 9126 quality characteristics. Then, the result of user satisfaction is presented and the number in the box indicates the number of respondents who selected the corresponding answer.

Quality factor: Functionality

1. Suitability

In the suitability factor, there are five questions which are shown below as in Table 8. It can be concluded that most of the respondents claimed that the tool is able to perform the RM activity as required because the number of respondents who select strongly agree is relatively high.

Table 7: The data for suitability factor

Suitability		gly A	gre e	e ut	sa gr	y D.
1	Did the RM tool manage your requirements during software development?	10	0	0	0	0
2	Did the RM tool document all your requirements?	10	0	0	0	0
3	Did the RM tool manage change control?	8	1	1	0	0
4	Did the RM tool manage version control?	8	2	0	0	0
5	Was the RM tool able to trace the requirements?	8	2	0	0	0

2. Accuracy

There are six questions which represent the accuracy factor. It can be seen from Table 9 that most of the respondents claimed that the accuracy factor was good. Results from using the RMT-SMP tool then, are as expected because the numbers of respondents that select strongly agree and agree is high.

Table 8: The data for accuracy factor

Accuracy		gly A	gre e	e ut	sa gr	y D.
6	Did the RM tool generate list of requirements?	9	1	0	0	0
7	Did the RM tool generate use case specification?	10	0	0	0	0
8	Did the RM tool show the traceability between requirements?	10	0	0	0	0
9	Did the RM tool illustrate the traceability up to their sources and down to corresponding design, source code and test cases?	10	0	0	0	0
10	Did the RM tool define a set of status values for a requirement, and monitoring status throughout the project?	9	1	0	0	0
11	Did the RM tool manage the document versions and requirements revisions?	7	3	0	0	0

3. Interoperability

The result concluded that the RMT-SMP is able to interact with another system as seven respondents select strongly agree and two respondents agree as in Table 10.

Table 9: The data for interoperability factor

Interoperability		Strongly Agree	Agree	Neutral	Disagree	Strongly Disagree
12	Can the RM tool interact with another tool?	7	2	1	0	0

4. Security

From the result below in Table 11 it can be concluded that the RM tool does prevent unauthorized access. It can be seen that 100% of the respondents claimed that the tool allows different actors to access the tool and prevent unauthorized access.

Table 10: The data for security factor

Security		gl y A	gr e	e ut	ra a gr	y D.
13	Did the RM tool allow different actors access?	10	0	0	0	0
14	Did the RM tool prevent unauthorized access?	10	0	0	0	0

Quality factor: Reliability

Table 12 below shows the result for the reliability quality factor that includes maturity, fault tolerance and recoverability as sub-qualities. It can be seen that the number of respondents who selected agree and strongly agree for maturity, fault tolerance and recoverability are higher than neutral, while none of them selected disagree or disagree. Thus, it can be concluded that the majority of the respondents claimed that most of the faults in the tool had been eliminated over time, that the tool is capable of handling errors, and that it can resume working and restore data after failure.

Table 11: The data for reliability factor

Reliability		ngl y A	Agr ee	Neu tral	Dis agre	ngr y Dis
15	Were most of the faults in the RM tool eliminated over time?	4	5	1	0	0
16	Were most of the faults in the RM tool eliminated over time?	4	5	1	0	0
17	Can the software resume working and restore lost data after failure?	3	6	1	0	0

Quality factor: Usability

The usability factor is important to evaluate the tool in order to ensure the tool is easy to use. The sub-quality factors are comperhensibility, learnability, operability, and attractiveness. Based on Table 13, the numbers of respondents who selected agree and strongly agree is relatively high. While, none of the respondents selected neutral, disagree and strongly disagree. Hence, it can be concluded that most of the respondents claimed that the tool is easy to use as well as agreed that:

- The user comprehends how to use the tool
- The user can learn to use the tool easily
- The user can use the tool without much effort

Table 12: The data for usability factor

Usability		gl y A	gr e	e ut	ra a gr	y D.
18	Did the user comprehend how to use the RM tool easily?	5	5	0	0	0
19	Can the user learn to use the RM tool easily?	1	9	0	0	0
20	Can the user use the RM tool without much effort?	4	6	0	0	0
21	Did the interface look good?	10	0	0	0	0
22	Did the user get lost while using the RM tool	10	0	0	0	0

Quality factor: Efficiency

The intention of evaluating the efficiency factor is to know how efficient the tool is. The efficiency factor includes time behaviour, resource and utilization as sub-quality factors. Most of the respondents selected agree and strongly agree while none of them selected neutral, disagree or strongly disagree as presented in Table 14. This is a good indicator to show that the tool is efficient to use as well as showing that the tool does respond quickly and utilizes resources efficiently.

Table 13: The data for efficiency factor

Efficiency		gl y A	gr e	e ut	ra a gr	y D.
23	Does the RM tool respond quickly?	6	7	0	0	0
24	Does the RM tool utilize resources efficiently?	7	3	0	0	0

Quality factor: Maintainability

The maintainability factor concentrates on evaluating how easy is to modify the software. There are four sub-qualities for maintainability factors, which are analyzability, changeability, stability and testability. The respondents were aware of the following statements when questioned.

- Faults are easy to diagnose
- The tool is easy to modify
- The tool continue functioning if changes are made
- The tool can be tested easily

Table 15 below shows that majority of the respondents selected strongly agree and agree to indicate that the tool is easy to modify. Thus, it can be concluded that the tool is easy to modify.

Table 14: The data for maintainability factor

	Maintainability	gl y A	gr e e	e ut	ts a	gr y D
25	Can faults be easily diagnosed?	3	7	0	0	0
26	Can the RM tool be easily modified?	1	8	1	0	0
27	Can the RM continue functioning if changes are made?	3	7	0	0	0
28	Can the RM tool be tested easily?	5	5	0	0	0

Quality factor: Portability

The portability factor focuses on evaluating how easy is to transfer to another environment. In this factor, the sub-qualities factors are adaptability, installability, conformance and replaceability. In addition, the respondents also claimed that:

- The tool can be tested easily
- The tool can be moved to other environments
- The tool can be installed easily
- The tool can easily replace other software

Figure 16 below shows that the majority of the respondents selected strongly agree and agree to indicate that the tool is easy to transfer to another environment.

Table 15: The data for portability factor

	Portability	ngl y A	Agree	Neutral	Disagre	gr y Dis
29	Can the RM tool be tested easily?	5	5	0	0	0
30	Can the RM tool be moved to other environments easily?	9	1	0	0	0
31	Can the RM tool be installed easily?	8	2	0	0	0
32	Can the RM tool easily replace other software?	6	4	0	0	0

User satisfaction

In this user satisfaction, five questions ask the user to select their satisfaction concerning their various needs. The level of satisfaction ranges from very satisfied, satisfied, neither satisfied nor dissatisfied, dissatisfied to very dissatisfied. The result is presented in Table 17 below. It can be seen that most of the respondents are very satisfied or satisfied in using this tool, as the number of respondents who select them are relatively high. None of the respondents select neither satisfied nor dissatisfied, dissatisfied or very dissatisfied. This is a good indicator that the tool is competent in satisfying the user's needs during the software project development.

Table 16: The data for user satisfaction

	User Satisfaction	gl y A	gr e e	e ut	ts a	gr y D
1	Does the RM tool help you to manage your requirements?	10	0	0	0	0
2	Does the RM tool encourage you to conduct best RM practice?	10	0	0	0	0
3	Does the RM tool help you to identify the errors at the earlier stage?	3	7	0	0	0
4	Does the RM tool provide help and guidance in managing requirements?	10	0	0	0	0
5	Overall, is the RM tool capable of handling RM activity?	10	0	0	0	0

5. Discussion

From the result of the quantitative and qualitative data it could be concluded that the practitioners in small and medium projects have improved their RM practices compared to the development during the previous project. In addition, the requirement engineer, project manager and developer emphasized that using the RMT-SMP tool during the software project development had a positive influence on RM activity. From the qualitative data, it can be seen that the respondents claimed that RMT-SMP is a tool that encourages them to have a better approach in practicing the RM activity. The following observations were made based on the qualitative and quantitative data collected throughout this case study:

- The requirements changes were handled carefully, analyzing, evaluating and modifying systematically. This indicates that the RMT-SMP is capable of handling the RM especially if there are any requirements changes required after the analysis phase.
- Every requirement identified in the projects was documented and it could be tracked at any time during software development. Even the rejected requirements are documented and able to be previewed at any time. This shows that there are requirements abandoned and the tool is able to perform as a repository for the references
- The RMT-SMP is evaluated using the ISO 9126 quality model framework and it shows that the features of the tool are capable of handling RM in small and medium projects.
- In terms of user satisfaction, it shows that the users are satisfied with what the tool has offered them in order to complete the RM activity.
- The project was developed in a team with four different actors. Although they played different roles, they were able to use the tool based on their needs. Thus, it can be stated that the tool is able to be used in every phase of software development.
- RM is a part of RE that is not the sole duty of requirement engineers. The involvement of developers and senior management in the process of managing requirements under the leadership of requirement engineers has a positive impact on the project.

The result from the quantitative and qualitative data is used to compare the RMT-SMP features against the elements of RM tool described in Section 2. Table 18 and 19 below summarize the result. The result below is considered valid in this case study only, and it cannot be generalized to represent the Malaysian software industry as a whole because this case study only involves two software development projects.. However, the results obtained shown that the RMT-SMP features met the general and specific elements. Thus, it can be concluded that, in this case: for the E-filing project, the RMT-SMP is suitable for the given Malaysian software project.

Table 17: The comparison of RMT-SMP features with general elements

General Elements	RMT-SMP
Usability, simplicity and Customization	√
Access control	√
Tailoring and Extensibility	√
Free licensing and full version availability	√
Database centric	√

Table legend: √-FULLY SUPPORTED, X–NOT SUPPORTED, P-PARTIALY SUPPORTED, ?-NOT KNOWN

Table 18: The comparison of RMT-SMP features with specific elements

Specific Elements	RMT-SMP
Requirements identification	√
Requirements classifying and viewing	√
Requirements base-lining	√
Change Control	√
Version Control	√
Status Tracking	√
Requirement Tracing	√
Use Case Specification generation	√
List of requirements generation	√
Requirements linking to system elements	√
Authentication procedure	√
Project definition	√
Create user	√

Table legend: √-FULLY SUPPORTED, X–NOT SUPPORTED, P-PARTIALY SUPPORTED, ?-NOT KNOWN

The case study presented in this chapter is an example that indicates the benefits and help that the RMT-SMP can provide for the small project: E-filling, and the medium project: HRMS. However, the result from this case study cannot be used as formal evidence that the RMT-SMP will always provide the best solution for managing requirements and/or the success of software project development. Thus, the following factors that reduce the validity of the case study have been recognized:

- Management commitment
  The management of the two projects had different levels of commitment to the RM process. Management of the E-filling and HRMS projects gave support for using RM practices during software development. However, this is not considered as a major effect on the good result in this case study.
- Learning effects and training
  Learning effect plays a role because both of the projects are in the same domain application. However, since the PMMS and ODMS projects were previous projects and not conducted at the same time as the E-filling and HRMS projects, so there could be any difference in learning effects in implementing PMMS and ODMS. Thus, learning effect should not be considered as major reason that lead to the success in this case study.
- Other factors
  The factors related to the personal attitudes and experiences of the software development team have influenced the answers to the questionnaire. However, this is only a minor effect toward the success of this case study.

On the other hand, based on this case study, it is likely to state that:

- The RMT-SMP is suitable for the given Malaysian small and medium software projects when the software project could be delivered on time, within the budget and met the quality expectations.
- The RE practitioners themselves indicate that the RMT-SMP had a positive influence on the RM practices.
- The RE practitioners themselves acknowledge that the RMT-SMP had a positive influence on encouraging them to practice the best RM activity.
- The RMT-SMP is capable of handling RM from the perspective of the ISO9126 quality model.

Moreover, the result from this case study supports the fundamental assumption made by the RE community that getting high-quality requirements, as well as documenting early on will reduce rework and overall cost development.

6. Conclusion

This paper describes a case study of implementing RMT-SMP in the Malaysian software companies from the quantitative analysis perspective. In addition, the qualitative analysis is also conducted in order to show the feasibility of RMT-SMP for the small and medium projects in the Malaysian software industry.

The result of this case study has shown the success of applying the RMT-SMP during software project development for small and medium projects. Therefore, it can be concluded that the hypothesis defined in the section 3 is true and confirms that:

Using the best practices of requirements management that is applied into RMT-SMP tool rather than using ad-hoc requirements management practices has a positive impact on encouraging the RE practitioners to have a better approach for managing requirements in developing the small and medium software projects.

As a conclusion, the objective of this case study which is to investigate how the requirements management tool could offer a better way of managing requirements for small and medium projects is achieved and relevant to the Malaysian software industry.

References

[1] L. J. May, “Major Causes of Software Project Failures,” CrossTalk J. Def. Softw. Eng., pp. 9–12, 1998.
[2] T. Javed, M. e Maqsood, and Q. S. Durrani, “A study to investigate the impact of requirements instability on software defects,” ACM SIGSOFT Softw. Eng. Notes, vol. 29, no. 3, pp. 1–7, 2004.
[3] T. Hall, S. Beecham, and A. Rainer, “Requirements problems in twelve software companies : An empirical analysis,” Software, IEEE Proc., vol. 149, no. November, pp. 153–160, 2002.
[4] N. Cerpa and J. M. Verner, “Why did your project fail?,” Commun. ACM - Find. Fun Comput. Sci. Educ., vol. 52, no. 12, pp. 130–134, 2009.
[5] A. Mandal and S. C. Pal, “Identifying the Reasons for Software Project Failure and Some of their Proposed Remedial through BRIDGE Process Models,” Int. J. Comput. Sci. Eng., vol. 3, no. 1, pp. 118–126, 2016.
[6] A. Shah, M. A. Alasow, F. Sajjad, and J. Javed Akbar, “An evaluation of software requirements tools,” in Eighth International Conference on Intelligent Computing and Information Systems (ICICIS), 2017, pp. 278–283.
[7] M. Lang and J. Duggan, “A Tool to Support Collaborative Software Requirements Management,” Requir. Eng., vol. 6, no. 3, pp. 161–172, 2001.
[8] Y. Sharma and A. K. Sharma, “Evaluation of the Software Requirement Tools,” Int. J. Eng. Res. Technol., vol. 3, no. 3, pp. 950–954, 2014.
[9] T. Hammer and L. Huffman, “Automated RM – Beware HOW You Use Tools: An Experience Report,” in Proceedings of IEEE International Symposium on Requirements Engineering: RE ’98, 1998, pp. 34–40.
[10] A. Altalbe, “Software Requirements Management,” Int. J. Adv. Res. Artif. Intell., vol. 4, no. 4, pp. 64–68, 2015.
[11] IBM, “IBM Rational DOORS,” 2018. [Online]. Available: https://www.ibm.com/us-en/marketplace/requirements-management.
[12] M. Hoffmann, N. Kühn, and M. Bittner, “Requirements for Requirements Management Tools,” in Proceedings of the 12th IEEE International Requirements Engineering Conference (RE’04), 2004, pp. 301–308.
[13] S. Innovations, “INCOSE Requirements Management Tool Survey Response,” 2019. [Online]. Available: https://www.innoslate.com/incose-requirements-management-tool-survey-response/.
[14] C. Wohlin, P. Runeson, M. Höst, M. C.Ohlsson, B. Regnell, and A. Wesslén, Experimentation in Software Engineering. 2012.
[15] H. Harrison, M. Birks, R. Franklin, and J. Mills, “Case study research: Foundations and methodological orientations,” Helena Harrison, Melanie Birks, Richard Franklin, Jane Mills, vol. 18, no. 1, 2017.
[16] J. Tetnowski, “Qualitative case study research design,” Perspect. Fluen. Fluen. Disord., vol. 25, no. 1, pp. 39–45, 2015.
[17] Kitchenham, B. and L. Pickard, “Case Study for Method and Tool Evaluation,” IEEE Softw., pp. 52-62., 1995.
[18] R. K. Yin, Case Study Research and Applications Design and Methods, 5th ed. Thousand Oaks, CA: Sage, 2014.
[19] A. Jorge, E. Steve, and W. Greg, “Requirements in the wild: How small companies do it,” in 15th IEEE International Requirements Engineering Conference, 2007, pp. 39–48.
[20] S. Khankaew and S. Riddle, “A review of practice and problems in requirements engineering in small and medium software enterprises in Thailand,” in IEEE 4th International Workshop on Empirical Requirements Engineering (EmpiRE), 2014.
[21] A. Abran and J. W. Moore, Guide to the Software Engineering Body of Knowledge. Los Alamitos,CA: IEEE Computer Society Press., 2001.
[22] G. Quirchmayr, S. Funilkul, and W. Chutimaskul, “A Quality Model of e-Government Services Based on the ISO/IEC 9126 Standard.,” in The Proceedings of International Legal Informatics Symposium (IRIS), 2007, pp. 45–53.
[23] Karabasevic, D., M. Maksimovic, D. Stanujkic, P. Brzakovic, and M. Brzakovic, “The evaluation of websites in the textile industry by applying ISO/IEC 9126-4 standard and the EDAS method.,” Ind. Textila, vol. 69, no. 6, p. 489, 2018.
[24] S. L. Pfleeger and J. M.Atlee, Software Engineering: Theory and Practice (Third Edit.). Prentice Hall, 2006.
[25] A. Paz, F., Diaz, E., Paz, F. A., Moquillaza, “Application of the Usability Metrics of the ISO 9126 Standard in the E-Commerce Domain: A Case Study.,” in International Conference on Intelligent Human Systems Integration, 2019, pp. 352–356.
[26] A. Losavio, F., Chirinos, L., Matteo, A., Levy, N., RamdaneCherif, “ISO Quality Standards for Measuring Architectures.,” J. Syst. Softw., vol. 72, no. 2, pp. 210–223, 2004.
[27] B. B. Chua and L. E. Dyson, “Applying the ISO 9126 Model to the Evaluation of an e-Learning System.,” in Proceeding of ASCILITE 2004, 2004, pp. 184–190.
[28] V. R. Basili, G. Caldiera, and H. D. Rombach, “The Goal Question Metric Approach.,” Encycl. Softw. Eng., pp. 528-532., 1994.
[29] R. an Solingen, V. Basili, G. Caldiera, and H. D. Rombach, “Goal question metric (gqm) approach.,” Encycl. Softw. Eng., 2002.

Azida Zainol An Assistant Professor at Faculty of Computer Science and Engineering, University of Jeddah. Her reseach areas of interest are software engineering, software enginering in machine learning and green computing.

Amani Jamal Assistant Professor, Faculty of Computing and Information Technology, King Abdulaziz University . She completed an M. Sc. degree of Computer Science from Concordia University, Canada, in Data Warehousing and On-line Analytical Processing in 2009. She got her PhD degree of Computer Science from Concordia University, Canada, on 2015, in Pattern Recognition and Artificial Intelligence. Her PhD research was conducted in Centre for Pattern Recognition and Machine Intelligence(CENPARMI). She works as an Assistant Professor in Faculty of Computing and Information Technology, King Abdulaziz University, Computer Science department. Her

research areas of interest are software engineering, AI,machine learning, pattern recognition, document analysis, big data and IoT.