Jurnal
Teknologi

Full Paper

AN INTEGRATED APPROACH TO FORMULATE A VALUE-BASED SOFTWARE PROCESS TAILORING FRAMEWORK

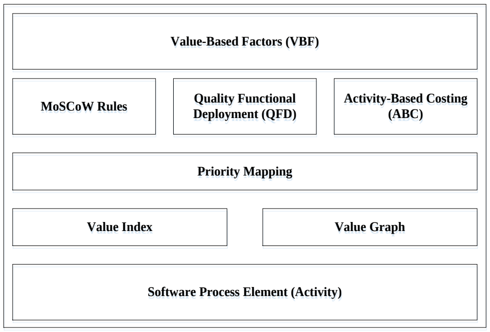
Article history
Received
26 November 2015
Received in revised form
14 January 2016
Accepted
10 October 2016

Noor Azura Zakaria[a], Suhaimi Ibrahim[b], Mohd Naz'ri Mahrin[b]

[a]Department of Computer Science, International Islamic University Malaysia (IIUM), 53100, Kuala Lumpur, Malaysia
[b]Advanced Informatics School (AIS), Universiti Teknologi Malaysia (UTM), 54100 Kuala Lumpur, Malaysia

Corresponding author
noorazura.zakaria@gmail.com

Graphical abstract


Value-Based Factors (VBF)
MoSCoW Rules
Quality Functional Deployment (QFD)
Activity-Based Costing (ABC)
Priority Mapping
Value Index
Value Graph
Software Process Element (Activity)

Abstract

Software process tailoring is an approach to customise the existing software development process or model that able to meet the software project's needs. Software development project is unique and identical from one and another whereby the practices and decision should not be equally treated. Software process tailoring requires knowledge and intuition to make decision such as factors involved in the software project, selection of the suitable software process elements and tailoring operations. Software process tailoring practices focusing more on project characteristics factors and employs ad hoc approach in making the decision. In the absent of value-based factors and systematic method in software process tailoring, subjectivity is embedded in decision making process and the software development project suffers from satisfying the stakeholder. This study presents an integrated approach to formulate a Value-Based Software Process Tailoring Framework (VBSPTF) to overcome this problem. The framework is a combination of value-based factors, MoSCoW rules, Quality Functional Deployment (QFD), Activity-Based Costing (ABC), Priority Map, Value Index and Value Graph. This study perhaps can contribute to the software process tailoring practitioners to be exposed with a systematic method to conduct software process tailoring as well as improving the practices and reducing subjectivity in decision making.

Keywords: Software process tailoring, value-based factors, value-based software engineering

1.0 INTRODUCTION

Software process tailoring is an approach to customise the existing software process or reference model in order to reduce the cost, effort, time and resources of defining the software development process. Even though the purpose is to lighten the development of the software process but the task is not easy and straightforward [1]. It requires extensive task, experience, intuition, right decision and commitment from the software development team members especially the project manager and process designer [2-4]. Another challenge in software process tailoring is that each project is unique which parallel with a claim that *"one size could not fit all"* [5-8]. This means that there is no single framework and guideline which can be used to define the software process in all project environments [5-7]. This is due to the fact that varies factors (e.g. organisation, team and external stakeholder) are involve in the software project environment and affect the software development project [6].

Since there is no one single guideline in performing the software process tailoring, the organisation normally adopt an ad hoc approach because of the time constrain faced by the project team members [9, 10]. The intuition and experience of the experienced project manager or process designer is always involved in this approach. Furthermore, the team members tend to select the old process or utilise the process that familiar to them. The limitation of this practice is that the project manager or process designer are unable to show whether the adapted process is suitable for the new assigned project [11]. With this ad hoc approach also, it decrease the understanding of the software process and knowledge to practice the tailoring which lead to the failure of the software process tailoring [12]. There are standards or reference model that has been used by several researchers to perform tailoring process like Rational Unified Process (RUP) [13-15], ISO/IEC 12207 [16] and V-Modell XT [17]. These standards are comprehensive because it defines the software development projects from the beginning until the end whereby it contains huge of information and one of it is the process elements (actors, roles and activities) of the software project [13]. These standards need to be tailored or downscaled according to the project context because there is no software project that uses all process elements defined in the standards. Guidelines in performing the process tailoring is lacking [13] and due to the comprehensiveness of the standards, the decision is difficult to make in order to select the process elements that are suitable with the project context [12]. The tailoring process is a knowledge intensive activity that requires experienced people to perform the decision making including the suitable factors that should be considered for the activity. It is a tough task for the novice and beginner in the tailoring process domain because they need extra effort and time to do the factors selection and decide the tailoring activity [18].

Research in the software process tailoring domain has taken an effort and initiative to produce supporting automated tool in assisting the practitioners to handle the tailoring activity. They have tried to solve some issues in the tailoring process domain which are correctness [19, 20], consistency [21], similarity checking [3], knowledge support [22, 23], reduce tailoring effort [24] and others [2, 9, 25]. Despite many issues that have been found and solved by several researchers, strategy to perform the software process tailoring remaining unclear [10]. This drawback has been a motivation for Xu and Ramesh [4] and Xu and Ramesh [10] to conduct a research with project environment factors as a centre to perform the tailoring process. They have produced a framework to guide the practitioner in strategising tailoring activity according to the project environment factor. The works have been very much attempting to tailor the software process according to the project environment or characteristics. In addition, most of the research highlighted above has used the project environment factors as input prior to tailoring activity. However, they have treated the project environment factors as equal important and has not distinguished according to the priority [11].

A debate on the software economics has proposed a new perspective which integrating the values in the software engineering [26, 27]. It does not rely on the success project that has been produced solely but the values that it can contribute either to the software process or software product. The value-neutral approach that usually been adopted has shortage in the value perspective in the software development which contributes to the failure of the software project [26]. The project values can influence the business value perspective [28]. The value perspective varies according to the project context or the organisation need. The values can be seen in terms of economic values (e.g.: cost) [29-32] or non-monetary values (e.g.: performance, quality, satisfaction) [33-36].

With the important of values in the software project and software engineering practices, a study can be performed by incorporating the value with the software process tailoring. Moreover, many studies have been performed in the software process tailoring domain but explanation on the factors selection and association with the values is still lacking. In addition, software process tailoring practices focusing more on project characteristics factors and employs ad hoc approach in making the decision. This is because, software process tailoring involves decision making to select appropriate software process element and tailoring operation. In the absent of value-based factors and systematic method in software process tailoring, subjectivity is embedded in decision making process and the software development project suffers from satisfying the stakeholder. Therefore, a research to develop a framework that incorporates the value-based concept and systematic method to tailor the software process tailoring is needed. This is the primary aim of this study.

2.0 METHODOLOGY

This section explains on the formulation of the Value-Based Software Process Tailoring Framework (VBSPTF). The VBSPTF consists of two phases which is pre-tailoring phase and post-tailoring phase.

2.1 Pre-tailoring Phase

The purpose of the pre-tailoring phase is to suggest to the practitioner on the suitable software process element (activity) and tailoring operation that should be consider for tailoring purpose.

2.1.1 Value-Based Factors

The VBSPTF begins with a list of value-based factors. The value-based factors are vital input in order to conduct the software process tailoring. The value-based factors are list of factors that are important and valuable to be considered in software development to produce a valuable product. Therefore, the value-based factors were used in software process tailoring domain to deliver a value-based tailored process which indirectly able to produce a valuable product. The list of value-based factors was obtained after conducting the Systematic Literature Review (SLR) study [37] and exploratory survey [38]. There are 28 value-based factors that was classified into four categories which is Success-Critical Stakeholder (SCS), Business Strategy (BS), Project Characteristics (PROJC) and Product Characteristics (PRODC).

2.1.2 MoSCoW Rules

In this study, the selected prioritisation technique was MoSCoW rules in order to assess the value-based factors according to priority. The selection was based on the two criteria which are flexibility to implement and well-defined definition of the prioritisation scale. MoSCoW rules is one of the numeral assignment techniques to prioritise the requirements based on four priority group [39]. MoSCoW is an acronym based on the four priority groups which is 'Must have', 'Should have', 'Could have' and 'Won't have' [40, 41]. The definition of them is listed below and derived from Hatton [41].

- **Must have** – requirements are not negotiable; the failure to deliver this requirements would result in the failure of the entire project.
- **Should have** – Features that would be nice to have if at all possible.
- **Could have** – Features that would be nice to have if at all possible but slightly less advantageous than the 'should have'
- **Won't have** – These requirements are not unimportant but they will definitely not be implemented in the current software project. They may, at a later stage be created.

Requirements that have similar priority will be grouped accordingly. This technique gives flexibility to the implementer to prioritise the requirements and less cumbersome to be implemented since the definition is clear for each of the priority group.

A weightage has been given to the MoSCoW priority group for the purpose of relationship calculation between value-based factors and software process element. The weightage was adapted from Vinay, *et al.* [42]. The study has used four level of weightage which is 9, 6, 3 and 0 depending on the strength of support between hard goal requirements with the soft goal. Weightage '0' means that there is no support between the hard goal requirements with the soft goal. In the context of this study, the first three weightage was adopted while weightage '0' was replaced by weightage '1'. This is because, the meaning of the last group in MoSCoW rules ('Won't have') is not null since the requirements will be considered in later stage. The definition of the MoSCoW rules was adapted that reflect to the domain of this study together with the weightage is tabulated in Table 1.

Table 1 Definition and Value of MoSCoW Rules

Term	Definition	Value
'Must have'	Must have this value-based factors in the software project to meet the value-based needs in terms of success-critical stakeholder, business strategy, project characteristics and product characteristics. The value-based factors are fundamentals and without them the final output is less valuable.	9
'Should have'	Should have this value-based factors if possible because of its important, but it is not vital for software project success.	6
'Could have'	Could have this value-based factors, but it can be left out if it has less impact.	3
'Want to have but will not have this time around'	Would like to have this value-based factors, but it can be considered later.	1

2.1.3 Quality Functional Deployment (QFD)

Software process tailoring requires an input about software process elements (artefact, activity and roles) before tailoring process can be done. However, the selection process can be vague whereby the software process elements that should be include can be questionable in terms of its suitability or appropriateness. Most of the studies in software process tailoring domain used project characteristics but the process to choose the appropriate software process elements according to the project characteristics context is not clear. In the context of this study, QFD was selected to choose the software process element based on the identified value-based factors.

QFD is an adaptation of Total Quality Management (TQM) which able to relate ideas to ideas, ideas to data and data to data. It was used in Japan in late sixties to support them with product design [43]. The QFD is able to link between customer needs with important components in order to produce a quality product. This is the reason of selecting QFD technique to be incorporated in the software process tailoring domain. Figure 1 illustrated the QFD matrices and data required in the context of this study.

There are five matrices used in this study which is explained below:

A – Value-Based Factors: This is a section to place all identified value-based factors as explained in 2.1.1.

B – Prioritisation: This section contains prioritization value that has been determined by the practitioner. The MoSCoW rules was used as prioritization technique which is explained in 2.1.2.

C – Software Process Element (Activity): This section requires the practitioner to place the software process element that is utilized in the organization. In the context of this study, the software process element is activities involved during software development in each phase.

D – Relationship: This section requires the judgement from the practitioner on the strength of relationship between each software process element (activity) with each of value-based factors. The rating value to be assign in this section followed suggestion given by Cohen [43] whereby 9-strong relationship, 3-moderate relationship, 1-slight or possible relationship and 0/blank-no relationship. This is a relationship rating criteria which later on will be given to the practitioner to conduct the relationship assessment (explain further in Section 3).

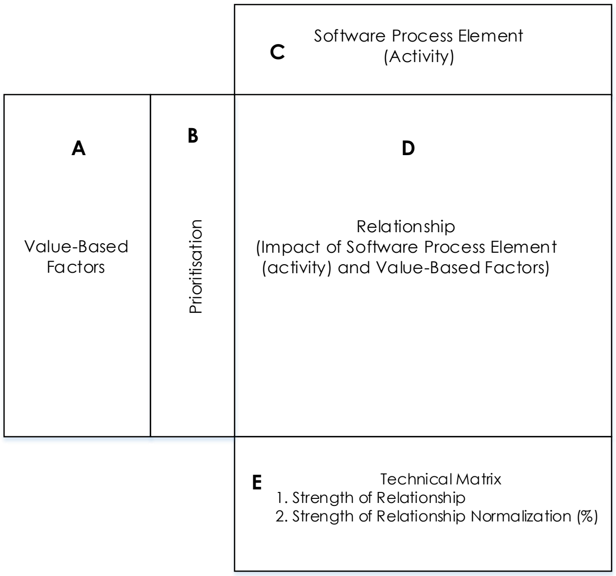

C
Software Process Element
(Activity)
A
Value-Based
Factors
B
Prioritisation
D
Relationship
(Impact of Software Process Element
(activity) and Value-Based Factors)
E
Technical Matrix
1. Strength of Relationship
2. Strength of Relationship Normalization (%)

Figure 1 QFD Matrices

E – Technical Matrix: In this section produces two outputs after completing section A-D in the QFD diagram.

- Strength of Relationship - *total of multiplication between value-based factors and software process element (activity)*. The formula is outlined below:

$$SR_{activity\,j} = \sum_{i=1}^{n} (VBF_i) * (Relationship_{VBFi,activityj}) \quad (1)$$

where $SR_{activity\,j}$ is total score for strength of relationship for activity $_j$ while I and j is a value of 1,2,.......,n.

- Strength of relationship normalization (%) – *percentage of total of multiplication between value-based factors and software process element (activity)*. The formula is outlined below:

$$SRN_{activity\,j} = \frac{SR_{activity\,j}}{\sum_{j=1}^{n} SR_{activity\,j}} * 100 \quad (2)$$

where, $SRN_{activity\,j}$ is total score for strength of relationship normalization for activity$_j$ while and j is a value of 1,2,.......,n.

2.1.4 Activity-Based Costing (ABC)

The ABC technique used in this study purposely to estimate the cost incurs in software development activities. The ABC methodology defines list of activities that bring value and assign cost for each of the activity [44]. This technique suit with the context of this study because it involves cost estimation in each activity in the software development process. This study was inspired by a work from Gunasekaran *et al.* [45] to implement the ABC for cost estimation purpose. In order to use this technique, the practitioner must determine the cost driver for the activity. This study used man per hour as the cost driver. This means that the average output produced by one worker in one hour. The cost driver is calculated based on rate (MYR) and volume (Hour). This means that each activity determined in the software development is calculated based on hourly basis in Malaysian Ringgit (MYR). The cost estimation terms and definition used in this study is shown in Table 2.

Table 2 Cost Estimation Terms & Definition

Terms	Definition
Cost Driver	Unit of an activity that causes the change in activity's cost.
Cost Driver Rate (MYR)	Amount of indirect or variable cost that is assigned to each unit of cost driver activity.
Cost Driver Volume (Hour)	Amount of time (in hour basis) required to complete the activity.

There are two outputs after conducting the cost estimation process which is outlined below:

- Total Cost Estimation (MYR) – This is a total cost estimation in Malaysian Ringgit for each software process element (activity). Total cost estimation is a summation of (Cost Driver Rate) X (Cost Driver Volume) for each task.

- Cost Estimation Normalization (%) – This is a value of normalization after gain result of total cost estimation.

$$CEN_{activityj} = \frac{TCE_{activityj}}{\sum_{j=1}^{n} TCE_{activityj}} * 100 \qquad (3)$$

where, $CEN_{activityj}$ is total score for cost estimation normalization for activity$_j$, while $TCE_{activityj}$ is total cost estimation for activity$_j$ and j is a value of 1,2,…….,n.

2.1.5 Priority Map

In order to suggest the suitable tailoring operation to the practitioner, a priority mapping graph was used as visualisation aid. The priority mapping graph is a plot of strength of relationship against cost. The value was obtained after activity in 3.1.3 and 3.1.4 was completed. The x-axis represents percentage of cost while *y*-axis represents percentage of strength of relationship. The graph was scaled in a range of 1 to 10 on both axes. Therefore, data used for x-axis is strength of relationship plot normalization while data used for *y*-axis is cost estimation plot normalization. The priority map graph was divided into four distinct regions as shown in Figure 2.

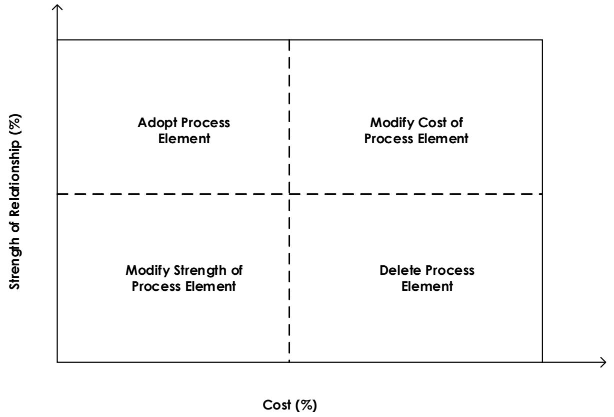

Strength of Relationship (%)
Adopt Process Element
Modify Cost of Process Element
Modify Strength of Process Element
Delete Process Element
Cost (%)

Figure 2 Priority Mapping Graph

The four regions in a priority map graph recommend the tailoring operation that should be considered by the practitioner for tailoring purpose. The regions and suggested tailoring operation is shown in Table 3.

Table 3 Priority Map Regions & Tailoring Operation

Region	Suggested Tailoring Operation
Upper left (Low Cost & High Strength)	Adopt process element
Upper right (High Cost & High Strength)	Modify the cost of the process element
Below left (Low Cost & Low Strength)	Modify the strength of the process element
Below right (High Cost & Low Strength)	Delete process element

2.2 Post-tailoring Phase

The purpose of the pre-tailoring phase is to examine the value index for process element that falls in the modify region. This is to ensure that the process element that needs to be tailored is valuable and cost accepted. The practitioner is able to decide either to include the process element (activity) which falls under modification regions for tailoring purpose or it can be eliminated.

2.2.1 Cost Re-estimation

Cost re-estimation is required if the software process element (activity) was placed in the modify region during pre-tailoring activity. The purpose is to measure the value of each process element to identify either the process element should be considered for modification or deletion. The practitioner needs to re-assess the cost only for each of the activity. The value for strength of relationship for the activity remains the same as value determined in pre-tailoring activity.

The cost re-estimation process is similar to cost estimation which used ABC technique as explained in 3.1.4. In this process, the practitioners must re-assess the cost of the process element (in order to reduce the cost) and re-assess the strength of relationship of process element (in order to increase the strength). In the cost re-estimation process, the cost may reduce or increase based on the task required to complete the process element (activity).

2.2.2 Value Index

The value index can be calculated once the cost re-estimation process is completed. The value index is used to rank a system of components by their perceived value [46]. The value index is an indicator to measure the value of the process element (activity) based on two data (strength of relationship and cost estimation). In this context, the purpose of the value index is to indicate either the process element (activity) which falls under modification regions should be included for tailoring or it can be deleted. The value index is a ratio of strength of relationship and cost estimation. The formula to calculate the value index is described as follows:

$$VI_{actj} = \frac{Strength\ of\ Relationship\ Normalization_{actj}}{Cost\ Estimation\ Normalization_{actj}} \quad (4)$$

where, VI_{actj} is Value index for activity $_j$ and j is a value of 1,2,……..,n.

The value index 1 and above is considered good and below 1 indicates that the software process element (activity) needs improvement or can be omitted. A value graph is provided to plot the software process element (activity) by using Cost Estimation Normalization (%) for axis-x and Strength of Relationship Normalization (%) for axis-y. A 45° line is a divider in the graph to indicate the good value or vice versa as illustrated in Figure 3. The software process element (activity) placed above the line is considered good and below line needs further improvement.

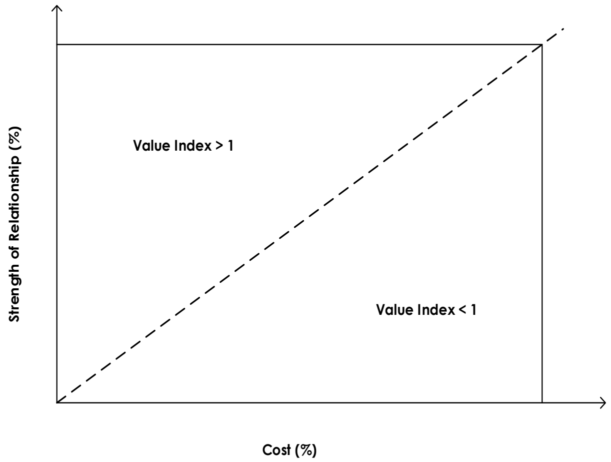


Figure 3 Value Graph

3.0 RESULTS AND DISCUSSION

3.1 Implementing VBSPTF

The approach selected to formulate the VBSPTF which was discussed in the previous section is visualised in Figure 4.

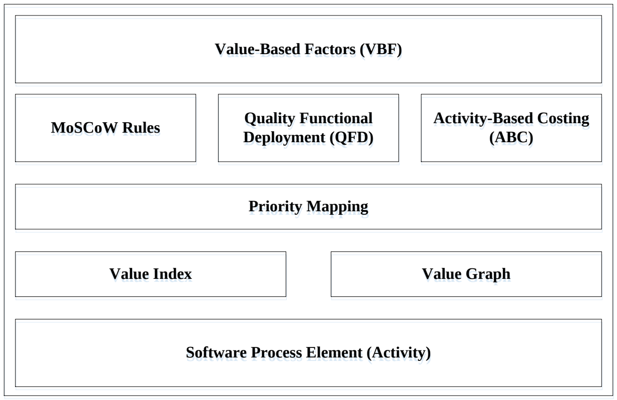

Value-Based Factors (VBF)
MoSCoW Rules
Quality Functional Deployment (QFD)
Activity-Based Costing (ABC)
Priority Mapping
Value Index
Value Graph
Software Process Element (Activity)

Figure 4 Integrated Approach in VBSPTF

3.2 Implementing VBSPTF

This section briefly explains the implementation step to use the VBSPTF. Figure 5 depicted the flow chart to implement VBSPTF.

Step 1: *Value-Based Factors Identification.* This step requires the practitioner to identify the suitable value-based factors which fulfill the project's needs. The practitioner must know the project's environment to identify the value-based factors before proceed with other tailoring activities.

Step 2: *Value-Based Factors Assessment.* In this step, the practitioner needs to assess the identified value-based factors for prioritization purposes. Value-based factors assessment criteria are given to the practitioner by using MoSCoW rules (explained in 2.1.2) to guide them in the assessment process.

Step 3: *Relationship Assessment between Value-Based Factors with Software Process Element.* The practitioner is required to assess the relationship between the identified value-based factors with the software process element. This can be done in the developed QFD environment (explained in 3.1.3). Besides value-based factors, other inputs required to conduct the relationship assessment is the software process element in each phase of software development. In this context, the software process element is the activity required in the software development. The practitioner may select the process element (activity) based on existing model such as ISO 12207 or RUP or best practices used in their organization. An example of software process element (activity) in testing phase based on RUP is plan test, design test, implementation test, perform integration test, perform system test and evaluation test. A relationship rating criteria is provided to the practitioner (explained in 2.1.3) to help them in assessment process.

Step 4: *Cost Estimation.* In this step, cost estimation is conducted based on ABC technique (explained in 2.1.4). The practitioner is required to estimate the cost for each software process element (activity) by listing possible tasks needed to complete the software process element. The total estimation cost is the multiplication of cost driver rate (MYR) with cost driver volume (Hour). Once completed the cost estimation process, a tailoring operation suggestion is given to the practitioner based on four options which is highlighted below (explained in 2.1.5).

- Adopt software process element
- Modify the cost of process element
- Modify the strength of process element
- Delete process element

Step 5: *Cost Re-estimation.* Cost re-estimation is required to be conducted for software process elements (activity) that falls under modify regions in the priority mapping graph (explained in 2.2.1). The practitioner needs to re-estimate the cost only while the strength of relationship is maintained. In order to reduce the cost of the software process element (process element that falls under high cost and high strength), the practitioner is suggested to revisit the list of determined tasks to complete the process element. The practitioner may delete some of the tasks or merge some of them. When deletion or merging some of the tasks, the role and artefact associated to the activity is also reduced. This may help in reducing the cost of the process element. On the other hand, to increase the strength of relationship (process elements that fall under low cost and low strength), the practitioner may add some tasks for the process element (activity). However, adding task to the process element (activity) may increase the cost of the project. This requires experience from the practitioners to conduct the cost re-estimation. Once re-estimation process is completed, a value graph (explained in 2.2.2) is provided to help the practitioner to make a decision either the process element (activity) is really required to be considered for modification or deletion.

4.0 CONCLUSION

Software process tailoring is an extensive work that requires knowledge and experience from knowledgeable practitioners. Current practices in software process tailoring are very much focusing on project characteristics and relying on ad hoc approach to conduct software process tailoring. Besides project characteristics, there are several of factors in software development project that are valuable to be considered to be included to conduct the software process tailoring. Ad hoc approach and replicating the process that familiar to the organization to execute the tailoring process may include subjectivity in decision making which indirectly reduce the product quality and stakeholder satisfaction.

This is the objective of this study whereby to present the formulation of framework for software process tailoring by incorporating value-based factors and systematic method (it is called as value-based software process tailoring framework). The purpose is to provide systematic method by using integrated approach which able to reduce subjectivity in decision making and satisfied the stakeholder. The integrated approach embedded in the VBSPTF is value-based factors, MoSCoW rules, Quality Functional Deployment (QFD), Activity-Based Costing (ABC), Priority Map, Value Index and Value Graph. This framework may contribute to the software process tailoring domain and improve the current practices used by the practitioner to tailor the software process in the organization. The VBSPTF is needed to undergo evaluation process in order to ensure the feasibility and usefulness to conduct software process tailoring.

Acknowledgement

This research is funded by the Research University grant of Universiti Teknologi Malaysia (UTM) under the Tier1-Vot no. 12H66. The authors would like to thank the Research Management Centre of UTM and the Malaysian ministry of education for their support and cooperation including students and other individuals who are either directly or indirectly involved in this project.

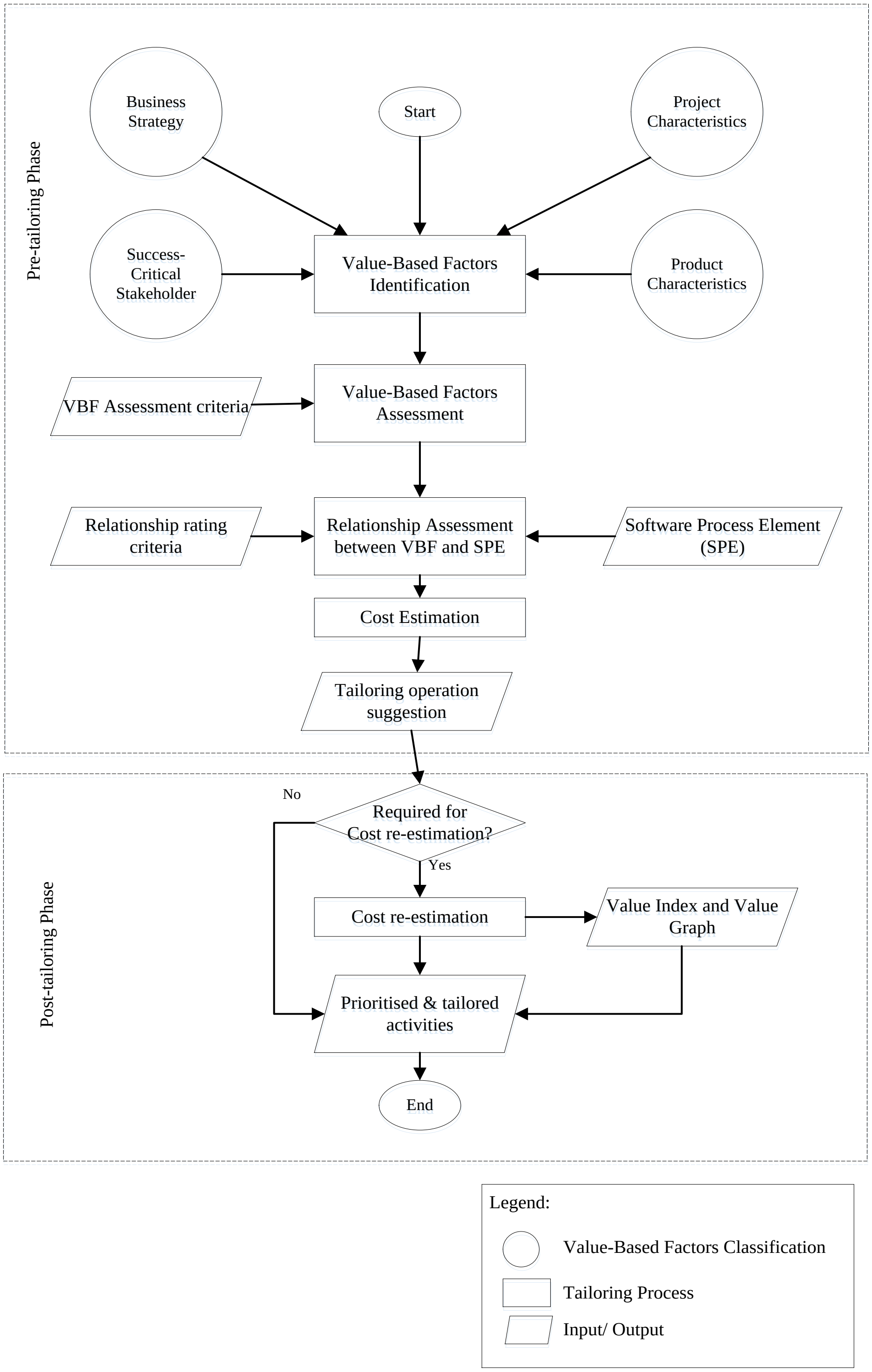

Pre-tailoring Phase
Business Strategy
Start
Project Characteristics
Success-Critical Stakeholder
Value-Based Factors Identification
Product Characteristics
VBF Assessment criteria
Value-Based Factors Assessment
Relationship rating criteria
Relationship Assessment between VBF and SPE
Software Process Element (SPE)
Cost Estimation
Tailoring operation suggestion
Post-tailoring Phase
No
Required for Cost re-estimation?
Yes
Cost re-estimation
Value Index and Value Graph
Prioritised & tailored activities
End
Legend:
Value-Based Factors Classification
Tailoring Process
Input/ Output

Figure 5 VBSPTF Implementation Flow Chart

References

[1] Magdaleno, A. M., 2010. An Optimization-Based Approach To Software Development Process Tailoring. *Second International Symposium on Search Based Software Engineering (SSBSE), 2010*. Benevento. 40-43.

[2] Hurtado, J. A. and Bastarrica, C., 2009. Process Model Tailoring As A Mean For Process Knowledge Reuse. *2nd Workshop on Knowledge Reuse, KREUSE, Falls Church, Virginia, USA*.

[3] Kang, D., Song, I.-G., Park, S., Bae, D.-H., Kim, H.-K., and Lee, N., 2008. A Case Retrieval Method For Knowledge-Based Software Process Tailoring Using Structural Similarity. *15th Asia-Pacific Software Engineering Conference, 2008. APSEC'08*. 51-58.

[4] Xu, P. and Ramesh, B. 2007. Software Process Tailoring: An Empirical Investigation. *Journal of Management Information Systems.* 24: 293-328

[5] Cameron, J. 2002.Configurable Development Processes. *Communications of the ACM.* 45: 72-77

[6] Clarke, P. and O'Connor, R. V. 2012. The Situational Factors That Affect The Software Development Process: Towards A Comprehensive Reference Framework. *Information and Software Technology.* 54: 433-447

[7] Jeners, S., Clarke, P., O'Connor, R. V., Buglione, L., and Lepmets, M. 2013. Harmonizing Software Development Processes with Software Development Settings–A Systematic Approach. *Systems, Software and Services Process Improvement*. Springer. 167-178.

[8] Keenan, F. 2004. Agile Process Tailoring And Problem Analysis (APTLY). *Proceedings of 26th International Conference on Software Engineering, 2004. ICSE 2004* Scotland, United Kingdom. 45-47.

[9] Rui, H., Hao, W., and Zhiqing, L. 2009. A Software Process Tailoring Approach Using a Unified Lifecycle Template. *International Conference on Computational Intelligence and Software Engineering, 2009. CiSE 2009.* Wuhan, China, 1-7.

[10] Xu, P. and Ramesh, B. 2008. Using Process Tailoring To Manage Software Development Challenges. *IT Professional.* 10: 39-45

[11] Heindl, M. and Biffl, S. 2005. A Case Study On Value-Based Requirements Tracing. *Proceedings Of The 10th European Software Engineering Conference Held Jointly With 13th ACM SIGSOFT International Symposium On Foundations Of Software Engineering*. Lisbon, Portugal. 60-69.

[12] Xu, P. 2005. Knowledge Support In Software Process Tailoring. *Proceedings of the 38th Annual Hawaii International Conference on System Sciences, 2005. HICSS'05.* Waikoloa, Hawaii. 87c-87c.

[13] Hanssen, G. K., Westerheim, H., and Bjørnson, F. O. 2005. Tailoring RUP to a defined Project Type: A Case Study. *Product Focused Software Process Improvement*. Springer. 314-327.

[14] Ibarguengoitia, G., Salazar, J., Ramirez, A., and Sánchez, M. 2003. A Procedure For Customizing A Software Process.

[15] Westerheim, H. and Hanssen, G. K. 2005. The Introduction And Use Of A Tailored Unified Process-A Case Study. *31st EUROMICRO Conference on Software Engineering and Advanced Applications, 2005.* Porto, Portugal. 196-203.

[16] Demirörs, O., Yildiz, A., Demirörs, E., and Tarhan, A. 2000. Tailoring ISO/IEC 12207 For Instructional Software Development. *EUROMICRO Conference*. 2300-2300.

[17] Armbrust, O., Ebell, J., Hammerschall, U., Münch, J., and Thoma, D. 2008. Experiences And Results From Tailoring And Deploying A Large Process Standard In A Company. *Software Process: Improvement and Practice.* 13: 301-309

[18] Hikichi, K., Fushida, K., Iida, H., and Matsumoto, K. i. 2006. A Software Process Tailoring System Focusing To Quantitative Management Plans. *Product-Focused Software Process Improvement*. Springer. 441-446.

[19] Yoon, I.-C., Min, S.-Y., and Bae, D.-H. 2001. Tailoring And Verifying Software Process. *Software Engineering Conference, 2001. APSEC 2001. Eighth Asia-Pacific*. 202-209.

[20] Park, S.-H. and Bae, D.-H. 2013. Tailoring A Large-Sized Software Process Using Process Slicing And Case-Based Reasoning Technique. *IET Software.* 7: 47-55

[21] Pereira, E. B., Bastos, R. M., and Oliveira, T. C. 2007. A Systematic Approach To Process Tailoring. *International Conference on Systems Engineering and Modeling, 2007. ICSEM'07*. 71-78.

[22] Henninger, S. and Baumgarten, K. 2001. A Case-Based Approach To Tailoring Software Processes. *Case-Based Reasoning Research and Development*. Springer. 249-262.

[23] Xu, P. and Ramesh, B. 2003. A Tool For The Capture And Use Of Process Knowledge In Process Tailoring. *Proceedings of the 36th Annual Hawaii International Conference on System Sciences, 2003.* Waikoloa, Hawaii, 7.

[24] Hurtado Alegría, J. A., Bastarrica, M. C., Quispe, A., and Ochoa, S. F. 2014. MDE-based Process Tailoring Strategy. *Journal of Software: Evolution and Process.* 26: 386-403

[25] Li, J. and Mao, M. 2009. A Case Study on Tailoring Software Process for Characteristics Based on RUP, in *International Conference on Computational Intelligence and Software Engineering, 2009. CiSE 2009.* 1-5.

[26] Boehm, B. 2003. Value-based Software Engineering: Reinventing. *ACM SIGSOFT Software Engineering Notes.* 28: 3.

[27] Boehm, B. and Sullivan, K. 2000. Software Economics: A Roadmap. *Proceedings of the conference on The future of Software Engineering*. Limerick, Ireland. 319-343.

[28] Suma, V., Shubhamangala, B., and Manjunatha Rao, L. 2013. Customization Of Quality Models In Software Projects To Enhance The Business Value. *Advance Computing Conference (IACC), 2013 IEEE 3rd International*. 1479-1485.

[29] Warburton, R. D. 2011. A Time-Dependent Earned Value Model For Software Projects. *International Journal of Project Management.* 29: 1082-1090

[30] Müller, M. M. and Padberg, F. 2003. On The Economic Evaluation Of XP Projects. *ACM SIGSOFT Software Engineering Notes*. 168-177.

[31] Li, J., Ma, Z., and Dong, H. 2008. Monitoring Software Projects With Earned Value Analysis And Use Case Point. *Computer and Information Science, 2008. ICIS 08. Seventh IEEE/ACIS International Conference on*. 475-480.

[32] Hanna, R. A. 2009. Earned Value Management Software Projects. *Third IEEE International Conference on Space Mission Challenges for Information Technology, 2009. SMC-IT 2009*. 297-304.

[33] Wagner, K. W. and Durr, W. 2006. A Five-Step Method For Value-Based Planning And Monitoring Of Systems Engineering Projects. *32nd EUROMICRO Conference on Software Engineering and Advanced Applications, 2006. SEAA'06.* Cavtat/Dubrovnik (Croatia). 282-290.

[34] Kim, C.-K., Lee, D. H., Ko, I.-Y., and Baik, J. 2007. A Lightweight Value-Based Software Architecture Evaluation. *Eighth ACIS International Conference on Software Engineering, Artificial Intelligence, Networking, and Parallel/Distributed Computing, 2007. SNPD 2007*. 646-649.

[35] Cabrero, D., Garzas, J., and Piattini, M. 2007. Choosing the Best Design Strategy from Requirements. A Value-Based Approach. *IEEE International Conference on Exploring Quantifiable IT Yields, 2007. EQUITY'07.* Washington DC, USA. 87-94.

[36] Azar, J., Smith, R. K., and Cordes, D. 2007.Value-oriented Requirements Prioritization In A Small Development Organization. *Software, IEEE.* 24: 32-37.

[37] Zakaria, N. A., Ibrahim, S., and Mahrin, M. N. 2015. A Survey of Value-Based Factors in Software Development. *Advanced Science Letters.* 21: 3351-3354.

[38] Zakaria, N. A., Ibrahim, S., and Mahrin, M. N. 2015. Examining Value-Based Factors in Software Development: A Survey Study in Malaysian Public Sector. Presented at the 24th Australasian Software Engineering Conference (ASWEC 2015), Adelaide, Australia.

[39] Vestola, M. 2010.A Comparison of Nine Basic Techniques for Requirements Prioritization. Helsinki University of Technology.

[40] Stapleton, J. 1997. *DSDM, Dynamic Systems Development Method: The Method In Practice*. Cambridge University Press

[41] Hatton, S. 2008. Choosing The Right Prioritisation Method. *19th Australian Conference on Software Engineering, 2008. ASWEC 2008.* Australia. 517-526.

[42] Vinay, S., Aithal, S., and Sudhakara, G. 2013. A Quantitative Approach Using Goal-Oriented Requirements Engineering Methodology And Analytic Hierarchy Process In Selecting The Best Alternative.

Proceedings of International Conference on Advances in Computing. 441-454.

[43] Cohen, L. and Cohen, L. 1995. *Quality Function Deployment: How To Make QFD Work For You*. Addison-Wesley Reading, MA.

[44] Ríos-Manríquez, M., Colomina, C. I. M., and Pastor, M. L. R.-V. 2014. Is The Activity Based Costing System A Viable Instrument For Small And Medium Enterprises? The Case Of Mexico. *Estudios Gerenciales.* 30: 220-232.

[45] Gunasekaran, A., McNeil, R., and Singh, D. 2000. Activity-based Management In A Small Company: A Case Study. *Production Planning & Control.* 11: 391-399.

[46] De Marle, D. J. 1992. Value Measurment, In *Value: Its Measurement, Design, And Management*. M. L. Shillito and D. J. De Marle, Eds. ed New York: John Wiley & Sons.