# GOSSEC: Goal Oriented Software Sustainability Evaluation Criteria

**Ruzita Ahmad[1]** **Fauziah Baharom[2]** **Azham Hussain[3]**
[1,2,3]*Universiti Utara Malaysia, Sintok Kedah, Malaysia*

rita_azura08@yahoo.com[1]; fauziah@uum.edu.my[2]; azham.h@uum.edu.my[3]

## Abstract

The concepts of sustainability is now aware among the software engineering researchers. It has direct and indirect impacts on three dimensions which are environment, economic and social that results from the development and implementation of the software. Although there are studies on software sustainability evaluation that defines the software sustainability criteria unfortunately, most of the studies are focusing on single criterion rather than come out with holistic criteria of software sustainability. Additionally, the studies also focused on what need to be measured instead of how to perform the evaluation systematically. This limitation was occurred due to lack of defining the measurement goal of each criteria of software sustainability dimensions. Therefore, this study aimed to develop a Goal Oriented Software Sustainability Evaluation Criteria and organize the sustainability criteria using Quality Function Deployment. On top of that, the Goal Oriented Software Sustainability Evaluation Criteria has been constructed using Goal Oriented Measurement approach by adapting the Goal Question Metric method to assist in defining the goal that clearly defined the purposes, perspectives, and point of views of measurement of software sustainability. Hence, the Goal Oriented Software Sustainability Evaluation Criteria provides nine (9) goals and thirty four (34) sub goals for measuring the software sustainability criteria and sub criteria. The findings from the study present a set of criteria and measurement goals which can be used for evaluating software sustainability. The criteria were organized into three dimensions which are environment, economic and social.

**Keywords:** Goal Question Metric, Quality Function Deployment, software sustainability criteria, software sustainability evaluation, sustainability development.

## Introduction

Formerly, software was developed with poor quality [1,2,3]. These problems occurred due to the developer only highlights to maximize the procurement efficiency, increasing organizational profit and financial return [4,5,6]. Besides, the project scope is determined by a few influential stakeholders who focused on minimal design scope in order to maximize the project speed [6,7]. Software architectures were having quality problems which did not support the user action in handling the changes in the environment [2,8,10]. Besides, the complexity of software system had increased the maintenance costs especially when the software is damaged or failed to reflect with the business process and having difficulties to be maintained [1,11,12]. Hence, these scenarios had resulted in failure to achieve the efficiency and reliability of software in order to improve and recover the risks of the system failures and errors in the future [13,14]. Moreover, the software features to protect environmental and individual health might be ignored at all [2,11,15]. Due to these problems, it is vital to practice sustainability design in software development [4,11,16,17,18]. This is because the adaptation of sustainability in software development

can guide developer in producing the right software with the valuable criteria in the right way that can provide the benefits to the environment, economic and social dimensions [4,19].

Sustainability is defined as the development that can fulfill and satisfy the requirements of the current development to the future generations [1,4,20,21]. While, software sustainability refers to the development of software using various resources with aim to achieve the needs of the current and future generations by integrating the sustainability dimensions such as environment, economic and social [2,22,23,24]. Initially, the concepts of software sustainability are inherited from the sustainable development proposed by the Brundtland Commission Report [21] that has been applied in various domains such as in manufacturing, constructing, restoration of natural disasters, soil and erosion, ecosystems, biodiversity, and others including Software Engineering since end of year 2009.

Based on the basic concept, sustainability in Software Engineering was emphasized on three dimension which are environment, economic and social. The environment dimension is concerned with the long term impacts of human activities on natural systems while economic dimension focuses on assets, capital and value added; and finally social dimension covers societal communities that relates to the trust of community in using software system [11,19,22,24,25]. However, the environment dimension is the famous sustainability dimension that had more attention from researchers which had been utilized in their proposed models. For example, Atkinson et al. [26], Sierszeicki et al. [27], Mahaux et al. [20] and Naumann et al. [29] had utilized their models based on energy consumption and energy efficiency. While, Gu et al. [30] proposed a green strategy that reflected to the business area. Additionally, some of them proposed the carbon footprint and carbon gas emission that released by the software equipment and supported the energy efficiency [2,28,30,31,32,33,34,35].

Whereas, some of the literatures focused on combining the dimensions in their proposed models such as worked done by Amri and Saoud [18] which focused on integrating the economic and social dimensions. They had identified sustainability requirements for these two dimensions and applied the requirements in hardware, networks, storage/data management, design and architecture. Furthermore, the earlier study done by Koziolek et al. [24] focused on economic dimension in which the cost efficiency of software system was highlighted in their proposed model. However, in 2013, the researchers had improved their model by focusing on social dimension where the requirements of technology was highlighted in their proposed model [8,11]. The requirement of technology is focused to the security, longevity of information, and data integrity that might be happen when any changing occurred in the environment [11].

In addition, Jansen, Wall and Weis [34] focused on economic and social dimensions, in which the economic dimension is stressed over the entire lifetime of software. While, the social dimension is remained as to support the criteria that specified in economic dimension and indirectly to achieve sustainability [11]. These trends of sustainability design in software development were then highlighted by Durdik et al. [23] in their proposed model. Unfortunately, this model did not specify the requirements of software sustainability in the specified dimensions [11,16]. Therefore, until to date social dimensions have been left behind in software development and needs an in-depth investigation to be emphasized in software development towards software sustainability [36].

In order to ensure the sustainability requirements are fulfilled, the software sustainability evaluation is required to assess the development of software [2, 36]. The purpose of software sustainability evaluation is to provide the decision-makers through an evaluation [11, 23, 37]. This is important to assists stakeholders to determine which actions that need to be followed in an attempt to make software sustainability is exist in present and continuously to the future generation [37, 38, 39]. Furthermore, the software sustainability evaluation can guide the developer in understanding the requirements of software sustainability in each sustainability dimensions and its impacts [12,17,40]. Therefore, the selection of software criteria is the important stage to fulfil the software sustainability requirements via sustainability dimensions [11,40].

The best known software sustainability evaluation model are proposed by Sarkar et al. [37], Koziolek et al. [24], Durdik et al. [23], Kocak et al. [15], Venters et al. [10], Penzenstadler et al. [2] and Software Sustainability Institute [41]. These models are claimed to build with lots of important software sustainability criteria towards long lived software. Based on the investigations, the previous works did not define the criteria completely to support software sustainability in each sustainability dimensions. This limitation arised because they only emphasized the software criteria in sustainability dimensions based on *what* they prefer to measure that relates to the identified dimensions [11,42]. On top of that, the definition of software criteria is limited to the specified sustainability dimensions in their worked [16]. Additionally, they were lacked to apply any tool in organizing the software criteria into sustainability dimensions correspondingly. On top of that, the sustainability requirements in proposing the software criteria does not fulfil by the existing works. Hence, this problem is very closely related to the understanding on the sustainability requirements. Which strongly relates to the purposes of proposing software sustainability criteria and sustainability requirements [43,44]. Bouwers, Deursen, & Visser [45] suggests the successful of software sustainability is the awareness to employ the software criteria into sustainability dimensions before assessing them individually. In a nutshell, the understanding will encourage the awareness to develop software with sustainable software criteria which needs to be aligned with the actions to achieve the objectives [17,44,45].

Therefore, this study presented the GOSSEC to solve the limitations in the previous works. GOSSEC defines the description of each software sustainability criteria, and systematically organized the criteria into sustainability dimensions using Quality Function Deployment (QFD) tool. As the main appliances, GOSSEC is developed based on goal oriented measurement approach via adapting GQM method for determining the goal of each software sustainability criteria. The adaptation of QFD tool and GQM method contributes this study in presenting a set of software sustainability criteria with emphasizing the sustainability requirements into environment, economic and social dimensions.

## Literature Review

This section describes the overview of the software sustainability criteria proposed by the best previous studies in the literature. Then, it is followed by elaborating the QFD tools and GOM approach as the main appliance used to develop the GOSSEC.

### Software Sustainability Criteria

Software sustainability criteria is strongly related to the definition and the concept of software sustainability. Most of the researchers defined the software sustainability criteria based on several standard quality models, theories, experiences, views and their understanding towards software sustainability. The sustainability dimensions are the important elements in defining the criteria towards sustainable development. In the context of software engineering, several studies contributed a set of criteria and also known as characteristics or indicators. They are Calero et al. [22], Venters et al. [10], Penzenstadler et al. [2], Koziolek [44] and also an established organization known as Software Sustainability Institute [41].

Research on software sustainability criteria has been introduced as a new field study of sustainability in software engineering, which was initiated by Koziolek [44]. The researcher has proposed a set of criteria in sustainability for long living software with highlighted at maintaining of the cost efficiency and evolved over their entire life cycle which are maintainability, modifiability, portability, ability to evolve, and integrity. According to the researcher, the sustainable development in software products and processes can be achieved by preparing a guideline to conduct the stakeholders through the sustainable requirements. The guideline consists of documenting, prioritizing, analyzing, and tracing the functional and non-functional requirements to an industrial software system, whereby the guideline acts as an important requirement for sustainability [24]. They claimed that the sustainable requirements is helpful in tracing and noticing for long term development systems as it preserves the knowledge about the decision making in every phase of development such as in architecture, design, implementation, testing and maintenance. Each requirement should be analyzed for its potential impact on sustainability in early stage as to improve the upfront design of the systems. On top of that, they claimed that the identified software criteria will much assists to achieve software sustainability. However, the researchers highlighted on environment and economic dimension without directly focuses on social dimension.

According to Venters et al. [10] suggested the concept of sustainability is contributed via the proposed software criteria. They defines the criteria of software sustainability by attaching the elements of a composite requirements and non-functional requirements. The researchers agreed to the Koziolek et al. [44] in stressing on environment and economic dimension in proposing the criteria and sub- criteria for software sustainability. They had declared that their proposed model was achieving the sustainability even though the element of social dimension is united into environment and economic dimensions. In addition, the researchers are closely relates the proposed criteria to software quality characteristic for achieving software sustainability.

Venters et al. [10] has proposed the software sustainability criteria based on McCall model as their benchmark. They claimed that the criteria recommended by McCall model is suitable and relevant to achieve software sustainability. In order, to improve their proposed criteria, the researchers has decomposed several attributes of McCall model into their own. For example, maintainability, reliability, safety and integrity has decomposed as the sub-attributes of dependability. The rest attributes of McCall model is remained. Additionally, the proposed software sustainability criteria were based on the concept of threats and failure as to support the element of the composite requirement towards sustainability. Therefore, the proposed software sustainability criteria consists of efficiency, reusability, scalability, extensibility, interoperability, portability, and dependability.

Calero et al. [22] proposes a set of criteria for software sustainability that involved the elements of energy consumption and resource optimization, which are inherited from ISO/IEC 25010 (2010). The ISO/IEC 25010 proposed the quality characteristics under these two elements such are: functionality, reliability, efficiency, operability, security, compatibility, maintainability and portability that are also broken down into the sub-characteristics. Towards developing software sustainability, the researchers recommended a new characteristic namely perdurability to support software sustainability. They describes perdurability as a software criteria with features of long-lasting software as functionality, modifiability, reusability, changeability and adaptability. On top of that these characteristics finally decomposed as sub-characteristics of perdurability. Besides, the researchers has proposed the software sustainability criteria with highlighted on environment and economic dimensions whereby the social dimension is remained in each stated dimensions indirectly.

Next, Software Sustainability Institute [41] an institute in proposing a model for software sustainability through the development, management and evaluation of the software products. They provides more information about software sustainability with presenting a guideline to the researcher to assess the software products towards sustainability. This private institute has proposed a set of software sustainability criteria which so called as "criterion" that is derived from ISO/IEC 9126-1 standard quality model. There are usability, sustainability and maintainability [11]. The criteria was broken down into sub-criteria and each of them has been composed to a set of questions that are needs to be answered by the stakeholders during the evaluation on their software products. Furthermore, SSI is focused on environment and economic dimension in their proposed criteria for developing software towards software sustainability. The SSI also agreed to the ideas proposed by Koziolek et al. [8,24,44] and Venters et al. [10] to incorporate social dimension with environment and economic dimensions.

The latest of Penzenstadler et al. [2] investigates software sustainability criteria in the three variables namely system, function and time that needs to be defined for setting scope in proposing the sustainability criteria. According to Penzenstadler et al. [12] provides a systematic literature review of sustainability in software engineering. They recommended that several researchers such as Mahaux et al. [20] and Naumann et al. [29] were the best practices in proposing the software sustainability criteria. Besides that, these investigations much supported them in proposing a guideline towards sustainability by focusing on IT changes behaviour that has considerable effect on the society and environment which are introduced by Mahaux et al. [20]. Therefore, Penzenstadler et al. [2] also focuses on direct and indirect impacts that can be affected to economy, society, human beings, and

environment which are aligned by the ideas of Naumann et al. [29]. Although, the researchers has focused on three of sustainability dimensions in their model, unfortunately the researchers were not propose any criteria towards software sustainability. In a nutshell, they proposed a set of indicator for each identified dimension that has been extended to environment, economic, social, individual and technical whereby the generic sustainability model structured is presented in a guideline to achieve sustainability through the values, indicators, regulations and activities.

As to conclude, several models in previous work has investigated on environment and economic dimensions without highlighting the social dimension individually. They were recommended that social dimension is specified into environment and economic dimensions in which the impacts of criteria that proposed in environment and economic dimensions will reflect to the social dimension indirectly. Table 1 summaries the software sustainability criteria that gathered from reviewing by the related works.

**Table 1.**
**Software Sustainability Criteria Contributed by Previous Works.**

| Researcher | Features Types | Criteria | References | Focused Dimension |
|---|---|---|---|---|
| Koziolek [44] | Characteristic | Maintainability<br>Modifiability<br>Portability<br>Evolvability<br>Integrity | Software Quality | Environment and economic |
| Software Sustainability Institute [41] | Criterion | Usability<br>Sustainability<br>Maintainability | ISO/IEC9126 | Environment and economic |
| Calero et al. [22] | Characteristic | Reliability<br>Maintainability<br>Portability<br>Perdurability | ISO/IEC25010 | Environment and economic |
| Venters et al. [10] | Characteristic | Extensibility<br>Interoperability<br>Dependability<br>Portability<br>Reusability<br>Scalability<br>Efficiency | McCall Model | Environment and economic |
| Penzenstadler et al. [2] | Indicator | Economic<br>Environment<br>Social<br>Technical<br>Individual | Mahaux et al., [20] and Naumann et al., [29] | Environment, economic and social |

**Quality Function Deployment (QFD) Tool**

The Quality Function Deployment (QFD) was first developed in Japan by Yoji Akao in the late 1960s. QFD is a process and set of tools used to effectively define customer requirements and convert them into detailed engineering specifications and plans to produce the products that fulfil those requirements [46]. It is a useful tool in translating the Voice of the Customer (VoC) into new products systematically [46, 47]. The VoC concept as the role to highlight the top-down approach, in which VoC (the WHATS) is customer's requirements are matched with the appropriate technical response along the top (the HOWS) [48]. The QFD answered the question of What and How in this study according to an appropriate technical response for each customer's requirement can be organized systematically.

The concept of customer's requirement pointed by QFD is suitable and relevant to the weight value assigned by the experts in this study through the proposed assessment mechanism. This is because QFD determines the importance of the weights for the customer requirements [48,49]. Additionally, QFD is widely used and combined with AHP technique for deriving weight values and frequently applied by various studies [48,49,50].

This study adopted QFD tool to organize the criteria obtained from theoretical and exploratory studies into sustainability dimension in systematic way. The gathered criteria is structured into sustainability dimension based on the theory suggested by literatures review and the opinion and suggestion from the best practices of software organization through the exploratory study performed in this research. Consequently, this tool is combined with AHP method through the proposed assessment mechanism of *i*-SSEM with aimed to perform the accurate results effectively. The assessment mechanism of an *i*-SSEM is presented in another paper.

**Goal Oriented Measurement (GOM)**

The Goal oriented measurement is an approach used for defining measurement goals in any development to achieve the determined objective [51]. Basically, the measurement goal is related to the criteria and sub-criteria that related to a specific object [52]. Thus, in software development the object can be related to any entity such as software process, software product or people. The entity contains a set of attributes either external or internal attributes that need to be clearly identified. The measurement of these attributes must be specified entirely which consisting of what, who, when, where, why, and how [53]. In order to identify the specified measurement goal, the GQM is used in this study.

The GQM method allowed user to determine the goal, question, and metric in a hierarchical acts as a guideline of measurement. This method is developed by Basili and Weiss [53]. According to this approach, the specific goals are formulated and then questions of each goal that need to be answered is derived to help to attain the goals. Finally, the metrics are defined in the third step as a platform of measurement. The development of metrics are based on the questions that have been developed to evaluate the identified criteria.

In this paper, the measurement goal of each software sustainability criteria has been defined by adapting a goal definition template proposed by Basili et al. [51]. The templates consisting of three elements which are purpose, perspective and environment as shown in Table 2. Table 2 illustrates the adapted template to define goals in the specified measurement. The element of *Purposes* and *Perspectives* are remained to the original template,

while element of *Environment* is modified to the context of sustainability dimensions such as environment, economic, and social.

**Table 2.**
**Adapted Templates for Goal Definition.**

| Element | Description |
|---|---|
| *Purposes* | To (characterize, evaluate, predict, motivate) the (process, product, model, metric) in order to (understand, assess, manage, engineer, learn, improve) it. |
| *Perspective* | Examine the (cost, effectiveness, correctness, defects, changes, product metrics, reliability, and etc) from the point of view of the (developer, manager, customer, corporate perspective and etc). |
| *Environment* | In the following context of (environment, economic, and social dimension). |

## The Development of Goal Oriented Software Sustainability Evaluation Criteria (GOSSEC)

In general, our study has proposed a model for software sustainability evaluation namely *integrated*-Software Sustainability Evaluation Model (*i*-SSEM). The *i*-SSEM has been constructed by referring to Evaluation Theory which consists of six main components. The GOSSEC is one of the components of the *i*-SSEM. GOSSEC is constructed based on the findings from theoretical and exploratory studies. Findings from the theoretical study comprises of the principal domain of sustainability development as highlighted in the Brundtland Commission Reports [21], several standards and models of software quality, and the identified criteria of software sustainability. While findings from the exploratory study were obtained from the survey amongst software practitioners in Malaysia. The identified software sustainability criteria were defined and classified into three dimensions which are environment, economic and social. The proposed software sustainability criteria were then organized by using QFD tool. Then, the GQM method was adapted to construct the goal of each criteria. Next, the questions and metrics were derived from each goals. Finally, in order to ensure the correctness, completeness and understandability, the GOSSEC was verified by using expert review approach which involves academician and software practitioners who had knowledge and experience in software evaluation and software sustainability domain. Next sub sections elaborates the methodology in developing GOSSEC.

## Methodology

The methodology used for developing GOSSEC consists of three main phases which are organization of software sustainability criteria, definition of each software sustainability criteria, and Formulation of goal for each criteria.

### Organization of Software Sustainability Criteria

As mention earlier the software sustainability consists of three dimensions which are environment, economic and social. The criteria for software sustainability were identified through theoretical and exploratory studies. Table 3 presents the identified criteria obtained from the and exploratory studies. Then by referring to ISO 25010, findings from the studies had finalized that there are nine criteria of software sustainability need to be included in formulating the GOSSEC. The criteria are functional suitability, performance efficiency, compatibility, usability, reliability, security, maintainability, portability and impactibility (as shown in Table 4) . Table 4 shows the finalized criteria of software sustainability where

every criteria will be contributed to evaluate the environment, economic and social dimensions.

Next, each criteria has been structured into a set of measurable sub criteria (as shown Table 6). These criteria and sub criteria were then aligned to each sustainability dimensions using QFD. The QFD is a quality tool that help to translate the Voice of the Customer (VoC) into new products systematically. The VoC (WHATS) refers to the sustainability dimensions were matched with the appropriate technical response along to the top (HOWS) which refers to the software sustainability criteria. These relationships were presented by adaptation of House of Quality (HoQ) structure that consists of the degree of importance weights or rating scales assigned by the stakeholders. The rating scales are represented as the relationship matrix, in the middle of HoQ.

**Table 3.**
**The Identified Criteria through Theoretical and Exploratory Study.**

| **Sustainability dimension** | **Identified criteria from theoretical study** | **Identified criteria from exploratory study** |
|---|---|---|
| Environment | Functional Suitability<br>Performance Efficiency<br>Compatibility<br>Maintainability<br>Portability<br>Impact | Functional Suitability<br>Performance Efficiency<br>Reliability<br>Maintainability<br>Portability |
| Economic | Functional Suitability<br>Performance Efficiency<br>Compatibility<br>Usability<br>Reliability<br>Security<br>Maintainability<br>Portability<br>Impact | Functional Suitability<br>Performance Efficiency<br>Compatibility<br>Reliability<br>Security<br>Maintainability |
| Social | Functional Suitability<br>Performance Efficiency<br>Compatibility<br>Usability<br>Security<br>Portability<br>Impact | Functional Suitability<br>Compatibility<br>Usability<br>Reliability<br>Security<br>Maintainability<br>Portability |

**Table 4.**
**Software Sustainability Criteria.**

| Sustainability dimension | Identified criteria |
|---|---|
| Environment | Functional Suitability<br>Performance Efficiency<br>Compatibility<br>Usability<br>Reliability<br>Security<br>Maintainability<br>Portability<br>Impactibility |
| Economic | |
| Social | |

Figure 1 presents the adaptation of HoQ to organize software sustainability criteria in systematic way.

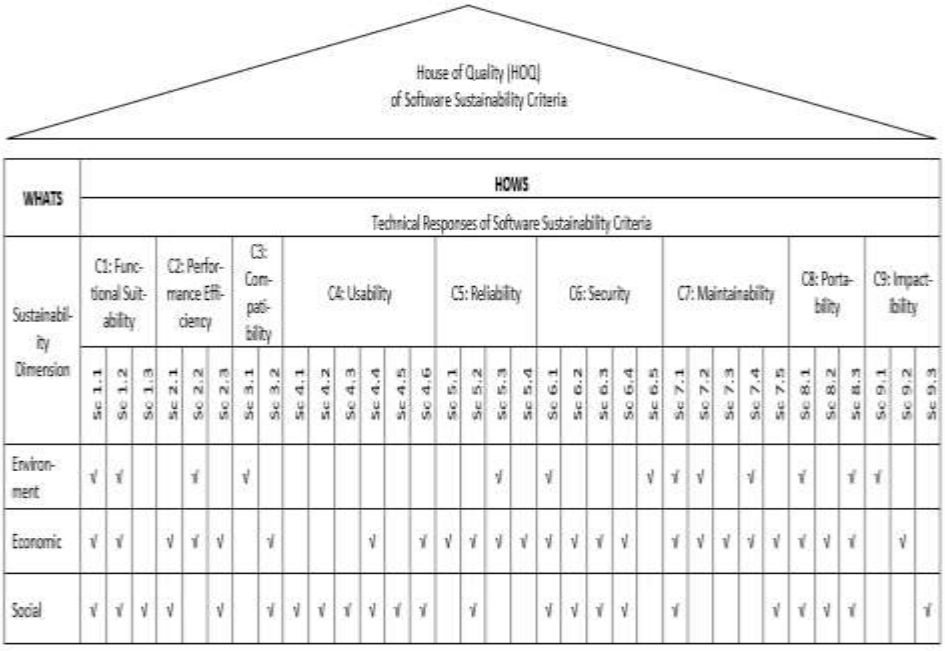


**Figure 1. HOQ Structure of Software Sustainability Criteria.**

## Definition of Software Sustainability Criteria

This phase focused on determining the definition of each software sustainability criteria. Figure 1 provides the definitions of each criteria which were formulated for each dimensions accordingly. The definition was formulated by referring to several standard quality models and the best practices of software sustainability. According to the theory from the literatures stated that, each criteria of software sustainability can be considered to be defined in each sustainability dimensions in which it was rely on the aim of the specified goal for the identified criteria and its dimension [7,40]. Therefore, each identified criteria for software sustainability in this study was remained in environment, economic and social dimensions correspondingly. Table 5 presents the definition of software sustainability criteria in each dimension.

**Table 5.**
**Definition of Software Sustainability Criteria**

| Dim. | Criteria | Definition |
|---|---|---|
| Environment | C1:Functional Suitability | To assess the degree to which a product or system provides the minimal impacts, in which the function performed the accurate result to avoid waste due to the un-functional of computing resources |
| | C2:Performance Efficiency | To assess the degree to which a product or system provides features with establish the time and energy behaviour to support the green software development |
| | C3: Compatibility | To assess the degree to which a product or system provides the features of software that can share the environment without adverse impact on their functionality |
| | C4: Usability | To assess the degree to which a product or system provides the features that required to be usable in which the product or system can be reused to protect the environment |
| | C5:Reliability | To assess the degree to which a product or system provides the features to perform the specified functions for a specified period of time to avoid waste of un-reliable tasks/functions |
| | C6:Security | To assess the degree to which a product or system provides the features to ensure the information and data have the legal accessibility and authorization |
| | C7:Maintainability | To assess the degree to which a product or system provides the features of effectiveness and efficiency in which the tasks/functions support energy efficiency |
| | C8:Portability | To assess the degree to which a product or system can effectively and efficiently be adapted and transferred from one hardware, software or other operational from one environment to another |
| | C9:Impactibility | To assess the user acceptance towards environment impacts with focused on the way of software is created, used, maintained and disposed with minimal impacts on environment |
| Economic | C1:Functional Suitability | To assess the degree to which a product or system provides the accurate function or tasks to minimize the cost of development |
| | C2:Performance Efficiency | To assess the degree to which a product or system provides the features with energy saving to monitor and control the cost of investment |
| | C3: Compatibility | To assess the degree to which a product or system provides the features of data formats and protocol can exchangeable with two or more systems, products and components which can control the cost of investment |
| | C4: Usability | To assess the degree to which a product or system provides the effectiveness and efficiency in a specified context of use to increase productivity and reduces costs |
| | C5:Reliability | To assess the degree to which a product or system provides the features to predict and control the faults that caused of higher investment |
| | C6:Security | To assess the degree to which a product or system provides the features in protecting information and data to reduce risk of capital value in long term profit |
| | C7:Maintainability | To assess the degree to which a product or system can be reused, modified, changed and tested with the lower cost of maintenance |
| | C8:Portability | To assess the degree to which a product or system provides the features of adaptable and transferrable with effectively and efficiently to decrease the software investments |
| | C9:Impactibility | To assess the user acceptance towards economic impacts in developing software with lower cost of development and maintenance to survive |

| | | |
|---|---|---|
| Social | C1:Functional Suitability | To assess the degree to which a product or system provides the accurate functions or tasks to support the reasonable and acceptable outcomes to achieve the specified intended objective |
| | C2:Performance Efficiency | To assess the degree to which a product or system provides the features to quick response on the user or system tasks to meet the specified target |
| | C3: Compatibility | To assess the degree to which a product or system provides the required functions efficiently while sharing common environment and resources with other product without detrimental impact on any other products |
| | C4: Usability | To assess the degree to which a product or system provides the features of software that enable user participation, accessibility and satisfaction in a specified context of use |
| | C5:Reliability | To assess the degree to which a product or system provides the features in which a system, product or component is operational and accessible when required for use |
| | C6:Security | To assess the degree to which a product or system provides the features to secure accessibility, participation and trustworthiness when using software |
| | C7:Maintainability | To assess the degree to which a product or system provides the tasks/functions that supported to achieve user objective and expectation |
| | C8:Portability | To assess the degree to which a product or system provides the satisfaction to the users with the features of adaptable and transferable actions |
| | C9:Impactibility | To assess the user acceptance towards social impacts of user and software functions are connectedness to each other to satisfy on using a product or system |

The proposed definitions of software sustainability criteria have been verified through expert review approach. The experts were identified from among academician and software practitioners who had knowledge and experiences in software sustainability. Table 6 presents the results of the identified software sustainability criteria consist of nine (9) criteria and thirty four (34) sub criteria. This result is used as the set of software sustainability criteria proposed by GOSSEC.

**Table 6.**
**The Identified Criteria for Software Sustainability**.

| Characteristic | Sub-Characteristic |
|---|---|
| 1. Functional Suitability | Functional Correctness<br>Functional Completeness<br>Functional Appropriateness |
| 2. Reliability | Maturity |
| | Fault Tolerance |
| | Recoverability |
| | Availability |
| 3.Performance Efficiency | Time Behavior |
| | Resource Utilization<br>Capacity |
| 4. Usability | Appropriateness Recognizabi |
| | Learnability |
| | Operability |
| | User error protection |
| | User interface aesthetics |
| | Technical Accessibility |
| 5. Security | Confidential |
| | Integrity |
| | Non-repudiation |
| | Accountability |
| | Authenticity |
| 1. 6. Compatibility | Co-existence |
| | Interoperability |
| 2. 7. Maintainability | 3. Modularity |
| | Reusability |
| | Analysability |
| | Modification Stability |
| | Testability |
| 8. Portability | Adaptability |
| | Installability |
| | Replaceability |
| 9. Impactibility | Environment Acceptance |
| | Economic Acceptance |
| | Social Acceptance |

## Formulation of Measurement Goal

As mentioned earlier, the measurement goal of each identified criteria was formulated by adapting the GQM templates. The adaptation of GQM potentially solve the limitation of the previous works, in which the goal is determined in precisely by presenting the purposes, perspectives and utilizing the context of environment towards environment, economic and social dimensions. Additionally, GQM guides to achieve the aimed of measurement whereby the focusses on what, who, when, why, where and how to measure. This can solve the limitation of the existing works that only highlighted on the what need to measure.

The GOSSEC contains nine (9) goals and thirty four (34) sub goals based on the identified software sustainability criteria and sub criteria. Earlier, the goal of software sustainability is defined as to evaluate software sustainability from software products and processes with focused on environment, economic and social dimensions. Then, the goal of each criteria and sub criteria are defined. Figure 2 presents the goal

structure of software sustainability criteria in GOSSEC. The next sub section discussing an example of GQM adaptation.

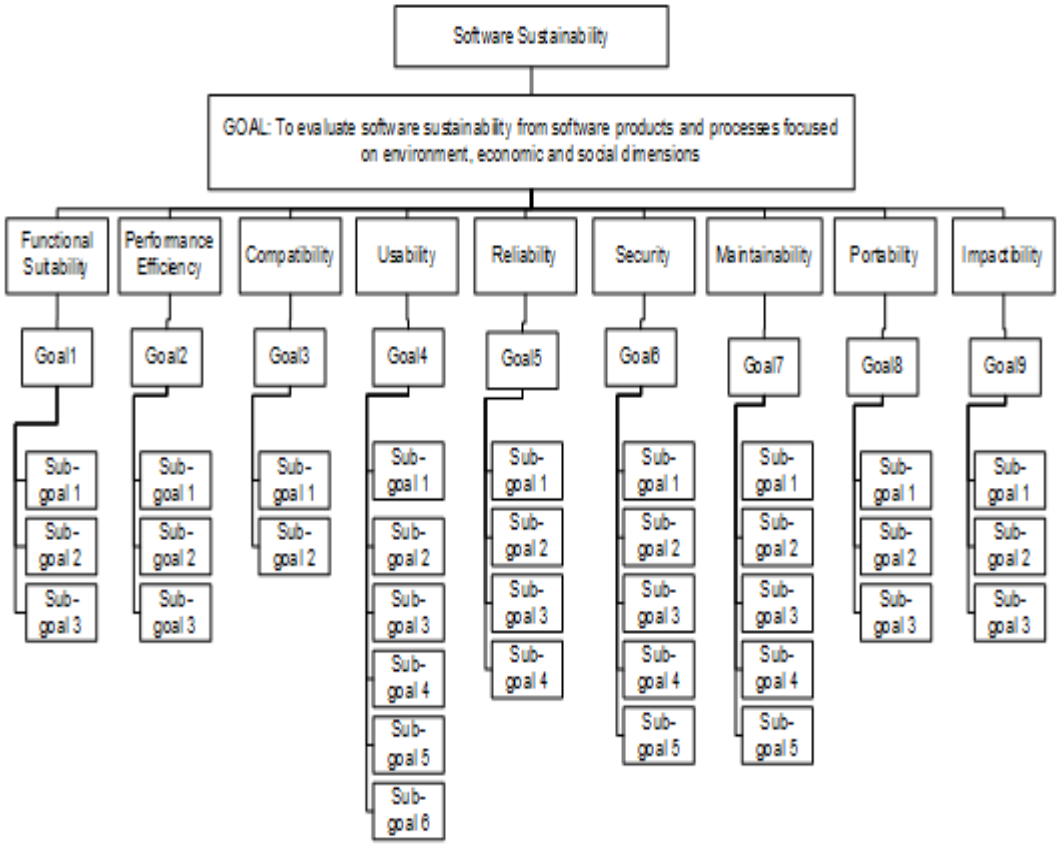


**Figure 2. Goal Structure of Software Sustainability Criteria.**

## Adaptation of GQM for Functional Suitability Criteria

Functional suitability is organized into environment, economic and social dimensions through its sub criteria such as functional completeness, functional correctness, and functional appropriateness. All sub criteria have an impact towards achieving software sustainability. Table 7 elaborates the goal definition templates for the criteria.

**Table 7.**
**Goal Definition Templates.**

| | |
|---|---|
| Purposes | To evaluate the functional suitability in order to assess it. |
| Perspectives | Examine the functional completeness, functional correctness and functional appropriateness. |
| Environment | In the context of environment, economic and social dimensions |

In order to achieve the goal stated to assess the functional suitability of the software system, the sub-goal, question and metric for each sub criteria are developed as presented in the Table 7, 8 and 9 respectively.

*i. Functional Completeness Sub-Criteria*

Functional completeness was measured through the metric of "functional coverage". According to the environment dimension, this metric contributed to avoid un-functional of computing equipment/hardware to deliver the tasks in effectively that can reflect to avoid waste. While in economic dimension, the functional coverage metric will detect the missing function (if any) when the system or software product does not have the ability to perform a function that is specified. This ability perform the accurate function and result to minimize the cost of software development. Dealing to the social dimension, this metric have the ability to support in achieving the user objective when using the software. Table 8 represents the adaptation of GQM for functional completeness.

**Table 8.**
**Goal Definition of Functional Completeness.**

| **Functional Completeness Sub Goal** | |
|---|---|
| *Purposes* | To evaluate the functional completeness in order to assess it. |
| *Perspective* | Examine the proportion of the specified functions implemented from developers and user's point of view. |
| *Environment Context* | In the following context:<br>1) To avoid un-functional of computing equipment/hardware to deliver the tasks effectiveness that can reflect to avoid waste (environment dimension),<br>2) To maintain and control the cost involved that related to un-functional equipment/hardware and software system (economic dimension)<br>3) To support in achieving the user objective (social dimension) |
| **Question** | |
| Q1 | What proportion of the specified functions has been implemented? |
| **Metric Name: M1 Functional Coverage** | |
| X = 1 – A/B<br>A = Number of functions missing<br>B = Number of functions specified | |

*ii. Functional Correctness Sub-Criteria*

Functional correctness is measured through the metric "functional correctness". This metric evaluate the proportion of functions provides the correct results. An incorrect function allowed the un-functional hardware or software that reflected to the un-effectiveness of the software functions to deliver the results. This issue was contributed to the environment dimension. Besides that, an incorrect function reflects to the developer and maintainer that are possibly need to review, test and determine whether the function successfully provides the suitable outcomes to the specific objectives as already defined in the requirement specifications. This issue contributes to the economic dimension and in the same time involved the user satisfaction towards using the software function (social dimension). Table 9 describes the adaptation of GQM for functional correctness.

**Table 9.**
**Goal Definition of Functional Correctness**.

| **Functional Correctness Sub Goal** | |
|---|---|
| *Purposes* | To evaluate the functional correctness in order to assess it. |
| *Perspective* | Examine the proportion of functions to provide the correct results from developer's, maintainer's and user's point of view. |
| *Environment Context* | In the following context:<br>1) To preserve the environment health and protection via controlling the waste development.<br>2) To control the cost involved in the requirements and specifications when occur incorrect functions.<br>3) To achieve the user intended objectives and satisfaction. |
| **Question** | |
| **Q1** | What proportion of functions provides the correct results? |
| **Metric Name: M2 Functional Correctness** | |
| X = 1 – A/B<br>A = Number of functions that are incorrect<br>B = Number of functions considered | |

*iii. Functional Appropriateness Sub-Criteria*

The functional appropriateness are measured by the metrics of "functional appropriateness of usage objective and "functional appropriateness of usage system". These metrics function typically be considered for the most important to identify each of the usage objective that can be pursued in the system and also can be calculated collectively across all the usage objectives to provide a system measure. These measurements are to support the function performed and provided the suitable or reasonable results in order to achieve user intended objectives. Thus, it is contributed to provide satisfaction to the user pertaining to the results performed in the specified usage and also for the whole usage objective in the system. In a nutshell, the metrics are supported the social dimension. Table 10 describes the adaptation of GQM for functional appropriateness.

**Table 10.**
**Goal Definition of Functional Appropriateness.**

| **Functional Appropriateness Sub Goal** | |
|---|---|
| *Purposes* | To evaluate the functional appropriateness in order to improve it. |
| *Perspective* | Examine the proportion of the functions required by the user provides appropriateness outcome to achieve a specific usage objective from user's point of view. |
| *Environment Context* | In the following context:<br>1) To achieve the specified usage objective (social dimension) |
| **Question** | |
| **Q1** | What proportion of functions required by the user provides appropriate outcome to achieve a specific usage objective? |
| **Metric Name: M3 Functional Appropriateness of Usage Objective** | |
| X = 1-A/B<br>A = Number of function missing or incorrect among those that are required for achieving a specific usage objective<br>B = Number of functions required for achieving a specific usage objective | |
| **Metric Name: M4 Functional Appropriateness of Usage System** | |
| X = $\sum_{i=1ton} Ai\ /n$<br><br>A*i* = Appropriateness score for usage objective *i*, that is, the measured value by the metric functional appropriateness of usage objective for *i*-th specific usage objective<br>n = Number of usage objective | |

## Conclusion

This paper provides discussion on how the GOSSEC was developed based on the findings from theoretical and exploratory studies. The exploratory study was investigated amongst software practitioners in Malaysia. The investigation much helps to identify the software sustainability criteria towards achieving sustainability dimensions. The proposed GOSSEC consists of nine (9) criteria and thirty four (34) sub criteria of software sustainability. For each criteria and sub criteria, the measurement goals were generated by adapting Goal Definition template. The goal of each criteria was formulated by highlighting three main elements which are purposes, perspectives, point of views in the following context of environment with respect to achieve software sustainability. The software sustainability criteria was systematically organized and mapped into sustainability dimensions using the QFD tool. The organization of each criteria into sustainability dimensions were based on its sub criteria and its capability to achieve the stated goals. The results of organization supports in defining the description of each criteria towards software sustainability. Additionally, the adaptation of GQM in GOSSEC will assist the software practitioners to ensure that the software sustainability criteria can be systematically measured. Therefore, GOSSEC is beneficial to the researchers, software sustainability practitioners, and software development community in proposing the important criteria of software sustainability and the measurement goals.

## Acknowledgments

We would like to thank the Ministry of Higher Education, Malaysian for providing the funding for this research through RAGS entitled "A New Yardstick For Software Survivabilitty Evaluation Of e-Government Applications Using Quality Functional Deployment (QFD) And Goal Question Metric (GQM) Approaches" (SO Code: 12736)

**CONFLICT OF INTERESTS.**
**There are non-conflicts of interest .**

## References


[1] F. Albertao, J. Xiao, C. Tian, Y. Lu, K. Zhang, K. C. Liu, Measuring the Sustainability Performance of Software Projects. 7th International Conference on E-Business Engineering, 2010. Pp. 369–373. doi:10.1109/ICEBE.2010.26

[2] B. Penzenstadler, and H. Femmer, "A generic model for sustainability with process- and product-specific instances," In Proceedings of the 2013 workshop on Green in/by Software Engineering, 2013, pp. 3-8. ACM.

[3] J. Cabot, S. Easterbrook, J. Horkoff, L. Lessard, S. Liaskos, and J. Mazón, "Integrating sustainability in decision-making processes: A modelling strategy," In *ICSE: 31st International Conference on Software Engineering*, 2009, pp. 207-210. IEEE.

[4] C. Becker, S. Betz, R. Chitchyan, L. Duboc, S. M. Easterbrook, B. Penzenstadler, N. Seyff, and C. C. Venters, "Requirements: The key to sustainability," *IEEE Software*, vol. 33 no. 1, pp. 56-65, 2016. Springer, Cham.

[5] B. Penzenstadler, B. Tomlinson, and D. Richardson, "Re4s: Support Environmental Sustainability by Requirements engineering, In *First International Workshop on Requirements Engineering for Sustainable Systems*, 2012, pp: 34-39.

[6] L. M. Hilty, and B. Aebischer "ICT for sustainability: An emerging research field," *ICT Innovations for Sustainability*. 2015, pp. 3-36. Springer

[7] R. U. Khan, S. U. Khan, R. A. Khan & S. Ali, "Motivators in Green IT-outsourcing from Vendor's Perspective: A Systematic Literature Review", *Proceedings of the Pakistan Academy of Sciences*. vol. 4 no. 52, pp. 345-360, 2015.

[8] H. Koziolek, D. Domis, T. Goldschmidt, and P. Vorst, "Measuring Architecture Sustainability," *IEEE Software*, vol. 30, no. 6, pp. 54–62, 2013. doi:10.1109/MS.2013.101

[9] C.C. Venters, L. Lau, M. K. Griffiths, V. Holmes, R. R. Ward, C. Jay, C. E. Dibsdale, and J. Xu, J, "The blind men and the elephant: Towards an empirical evaluation framework for software sustainability", *Journal of Open Research Software*, 2014, vol. 1 no 2.

[10] C. C. Venters, N. Seyff, C. Becker, S. Betz, R. Chitchyan, L. Duboc, D. McIntyre, and B. Penzenstadler, "Characterizing sustainability requirements: A new species, red herring, or just an odd fish?" In *39th International Conference on Software Engineering ICSEBuenos Aires, Argentina*, 2017.

[11] C. C. Venters, R. Capilla, S. Betz, B. Penzenstadler, T. Crick, S. Crouch, C. Carrillo, "Software Sustainability: Research and Practice from a Software Architecture Viewpoint", *Journal of Systems and Software*. 2017. Doi: 10.1016/j.jss.2017.12.026

[12] B. Penzenstadler, "Sustainability in software engineering: A systematic literature review for building up a knowledge base", Technical report, Technische Universität München, 2012. Available: http: //www4.in.tum.de/~penzenst/sources/SysLitRev_Protocol_TUM-I1201.pdf.

[13] J. Schelp, and S. Aier, "SOA and EA — Sustainable Contributions for Increasing Corporate Agility". *Proceedings of the 42nd Hawaii International Conference on System Sciences*, 2009. pp. 1–8.

[14] J. Governor, "SOA : An On Ramp To Sustainability", *Redmonk Greenpaper*, (March), 2009. pp. 1–9.

[15] S. A. Kocak, "Green Software Development and Design for Environmental Sustainability," *Journal of Green Engineering*, 2012.
[16] T. Kehrer, and B. Penzenstadler, "An Exploration of Sustainbility Thinking in Research Software Engineering, *7th Intl. Workshop on Requirements Engineering for Sustainable Systems (RE4SuSy)*, 2018, pp. 1-8.
[17] R. U. Khan, S. U. Khan, R. A. Khan, and S. Ali, "Motivators in Green IT-outsourcing from Vendor's Perspective: A Systematic Literature Review," *Proceedings of the Pakistan Academy of Sciences*, vol. 52, no. 4, pp.345-360, 2015.
[18] R. Amri, and N. B. B. Saoud, "Towards a Generic Sustainable Software Model," *Fourth International Conference on Advances in Computing and Communications, Cochin*, 2014, pp. 231-234. IEEE.
[19] R. Chitchyan, C. Becker, S. Betz, L. Duboc, B. Penzenstadler, N. Seyff, and C. C. Venters, "Sustainability design in requirements engineering: State of practice," In *Proceedings of the 38th International Conference on Software Engineering Companion, ser. ICSE-SEIS '16*, pp. 533–542, 2016. ACM.
[20] M. Mahaux, P. Heymans, and G. Saval, "Discovering sustainability requirements: an experience report," In *International Working Conference REFSQ*, pp. 19–33. Springer.
[21] G. H. Brundtland, and UN World Commission on Environment and Development, *Our Common Future*. Oxford University Press, 1987.
[22] Z. Durdik, B. Klatt, H. Koziolek, K. Krogmann, J. Stammel, and R. Weiss, "Sustainability guidelines fo long-living software systems," In 2012 28th IEEE International Conference on Software Maintenance (ICSM), 2012, pp. 517–526. IEEE.
[23] H. Koziolek, "Sustainability Evaluation of Software Architectures : A Systematic Review," *Journal of Environmental Assessment Policy and Management*, 2011.
[24] B. Penzenstadler, A. Raturi, and D. Richardson, "Safety, Security, Now Sustainability: The Nonfunctional Requirement for the 21st Century," *IEEE Software*, vol. 31, pp. 40–47, 2014.
[25] C. Atkinson, T. Schulze, and S. Klingert, "Facilitating Greener IT through Green Specifications", *IEEE Software*, 2014, vol 3, no 31, pp: 56–63.
[26] K. Sierszecki, T. Mikkonen, M. Steffens, T. Fogdal, and J. Savolainen, "Green Software: Greening What and How Much?", *IEEE Software*, 2014, vol 3, no 31, pp: 64–68.
[27] D. Mendez and B. Penzenstadler, "Artefact-based Requirements Engineering: The AMDiRE Approach", *Requirement Engineering Journal*, 2014.
[28] S. Naumann, M. Dick, E. Kern, and T. Johann, "The GREENSOFT Model: A reference model for green and sustainable software and its engineering", *Sustainable Computing: Informatics and Systems*, 2011, vol 4, no 1, pp: 294 – 304.
[29] Q. Gu, P. Lago, and S. Potenza, "Aligning Economic Impact with Environmental Benefits: A Green Strategy Model", In *First International Workshop on Green and Sustainable Software*, 2012.
[30] M. Mahaux, P. Heymans, and G. Saval, "Discovering sustainability requirements: an experience report", In *International Working Conference REFSQ*, 2012, pp. 19–33.
[31] D. Stefan, E. Letier, M. Barrett, and M. Sawicki, "Goal-oriented system modelling for managing environmental sustainability". In *The 3rd Workshop on Software Sustainability*, pp: 1–4. Available: http://www0.cs.ucl.ac.uk/staff/e.letier/publications/2011 [Accessed: July 18, 2013].

[32] J. Taina, "Good, Bad, and Beautiful Software - In Search of Green Software Quality Factor," *CEPIS UPGRADE XII* (4), pp. 22–27, 2011. Available: www.cepis.org [Accessed: December 7, 2013].

[33] Jansen, A. Wall, and R. Weiss, "TechSuRe: A method for assessing technology sustainability in long lived software intensive systems," *SEAA 2011*: *Proceedings of the 37th EUROMICRO Conference on software engineering and advanced applications*, Oulu, Finland, 2011. IEEE.

[34] M. Hinai, and R. Chitchyan, "Social sustainability indicators for software: Initial review," In Proceedings of the Third International Workshop on Requirements Engineering for Sustainable Systems (RE4SuSy), 2014.

[35] S. S. Mahmoud, and I. Ahmad, "A green model for sustainable software engineering," *International Journal of Software Engineering and Its Applications*, vol. 7, no. 4, pp. 55–74, 2013.

[36] S. Sarkar, A. C. Kak, and G. M. Rama, "Metrics for measuring the quality of modularization of large-scale object-oriented software" *IEEE Transactions on Software Engineering*, 2008, vol 5, no 34, pp: 700–720. doi:10.1109/TSE.2008.43

[37] E. Pavlovskaian, " Sustainablity Criteria: Their indicators, control, and monitoring with examples from the biofuel sector," Environmental Sciences Europe, 2014.

[38] K. Roher, and D. Richardson, "A proposed recommender system for eliciting software sustainability requirements," In 2013 2nd International Workshop on User Evaluations for Software Engineering Researchers (USER), 2013, pp. 16-19. IEEE.

[39] E. Bouwers, V. Deursen, and A. J. Visser, "Evaluating usefulness of software metrics: an industrial experience report," In Proceedings of the 2013 International Conference on Software Engineering, 2013, pp. 921–930. IEEE.

[40] Software Sustainability Institute, "*Software Sustainability Evaluation"*, Available: http://www.software.ac.uk/ [Accessed: December 12, 2018].

[41] R. K. Singh, H. R. Murty, S. K. Gupta, A. K. Dikshit, "An overview of sustainability assessment methodologies" *Ecological Indicators*, 2009. vol 1, no 15, pp: 189-212.

[42] Calero, M. A. Moraga, and M. F. Bertoa, "Towards a Software Product Sustainability Model," *Journal of Sustainability*, 2013. Available: http://arxiv.org/abs/1309.1640 [Accessed: September 28, 2014].

[43] H. Koziolek, D. Domis, T. Goldschmidt, P. Vorst, and R. J. Weiss, "MORPHOSIS: A lightweight method facilitating sustainable software architectures", *Proceedings of the 2012 Joint Working Conference on Software Architecture and 6th European Conference on Software Architecture*, 2012, pp: 253–257. Doi:10.1109/WICSA-ECSA.212.40.

[44] Penzenstadler, M. Khurum, and K. Petersen, "State of the Practice for Sustainability as an Explicit Objective". *In Intl. Conf. on Requirements Engineering: Foundations for Software Quality*, 2014. Available: http://www.ics.uci.edu/~bpenzens/2014refsqEmpTrack/ [Accessed: December 26, 2015].

[45] L. Cohen, *Quality Function Deployment: how to make QFD work for you.* Reading, MA: Addison-Wesley. 1995.

[46] F. P. M. Shafinah, B. Fauziah, and D. Aziz, " ESPAC Model: Extended Software Process Assessment And Certification Model For Agile And Secure Software Processes," *ARPN Journal of Engineering and Applied Sciences*, vol. 10, no. 3, pp. 1364 – 1373, 2015.

[47] Alinezad, A. Seif, and N. Esfandiari, "Supplier evaluation and selection with QFD and FAHP in a pharmaceutical company", *The International Journal of Advanced Manufacturing Technology*, 2013, vol 1-4 no 68, pp: 355-364. doi: 10.1007/s00170-013-4733-3

[48] Garibay, H. Gutierrez, and A. Figueroa, "Evaluation of a digital library by means of quality function deployment (QFD) and the Kano model", *The Journal of Academic Librarianship*, 2010. Vol 2 no 36, pp: 125-132. doi: 10.1016/j.acalib.2010.01.002

[49] Ishizaka, and A. Labib, "Analytic Hierarchy Process and Expert Choice: Benefits and limitations" *OR Insight*, 2009 vol 4 no 22, pp: 201–220. doi:10.1057/ori.2009.10

[50] V. R. Basili, G. Caldiera, and H. D. Rombach, "The goal question metric approach," *Encyclopedia of Software Engineering*, *2*, 528–532, 1994.

[51] V. R. Basili, J. Heidrich, and M. Lindvail, "Bridging the Gap between Business Strategy and Software Development Why Measurement?", *International Conference on Information Systems*, 2007, pp*:* 1–16.

[52] S. L. Pfleeger, *Software Engineering: Theory and Practice, 2nd ed*. Upper Saddle River, N.J: Prentice Hall, 2001.

[53] V. R. Basili, and D. M. Weis, "A methodology for collecting valid software engineering data", IEEE Transaction on Software Engineering, Vol. SE-10, No. 6, 1984.