Jurnal Teknologi

Full Paper

MULTI ATTRIBUTE ARCHITECTURE DESIGN DECISION FOR CORE ASSET DERIVATION

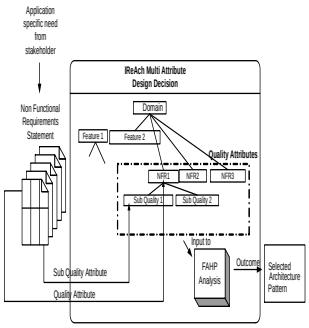
Shahliza A. Halim[a], Dayang N. A. Jawawi[a], Noraini Ibrahim[a], M. Zulkifli M. Zaki[b], Safaai Deris[c]

[a]Software Engineering Department, Faculty of Computing, Universiti Teknologi Malaysia, Johor, Malaysia
[b]RWTH Aachen University, Lehrsthul Informatik 11 - Embedded Software, 52074 Aachen, Germany
[c]Faculty of Creative Technology and Heritage, Universiti Malaysia Kelantan, Kelantan, Malaysia

Article history
Received
2 February 2015
Received in revised form
8 October 2015
Accepted
12 October 2015

*Corresponding author
shahliza@utm.my

GRAPHICAL ABSTRACT



Abstract

Software Product Line (SPL) is an effective approach in software reuse in which core assets can be shared among the members of the product line with an explicit treatment of variability. Core assets, which are developed for reuse in domain engineering, are selected for product specific derivation in application engineering. Decision making support during product derivation is crucial to assist in making multiple decisions during product specific derivation. Multiple decisions are to be resolved at the architectural level as well as the detailed design level, address the need for assisting the decision making process during core asset derivation. Architectural level decision making is based on imprecise, uncertain and subjective nature of stakeholder for making architectural selection based on non-functional requirements (NFR). Furthermore, detail design level involves the selection of suitable features which have the rationale behind each decision. The rationale for the selection, if not documented properly, will also result in loss of tacit knowledge. Therefore, a multi-attribute architecture design decision technique is proposed to overcome the above mentioned problem. The technique combines Fuzzy Analytical Hierarchy Process (*FAHP*) with lightweight architecture design decision documentation to support the decision making during core asset derivation. We demonstrate our approach using the case study of Autonomous Mobile Robot (AMR). The case study implementation shows showed that the proposed technique supports software engineer in the process of decision making at the architecture and detail design levels.

Keywords: Application engineering, software product line, *FAHP*, architecture design decision

Abstrak

Barisan Keluaran Perisian (SPL) adalah pendekatan yang berkesan dalam penggunaan semula perisian di mana aset teras boleh dikongsi dalam kalangan ahli barisan keluaran dengan menekankan aspek kepelbagaian. Aset teras yang dibangunkan untuk digunakan semula dalam bidang kejuruteraan domain dipilih untuk menerbitkan produk khusus dalam bidang kejuruteraan aplikasi. Sokongan pembuatan keputusan semasa proses penghasilan produk adalah penting untuk membantu dalam membuat pelbagai keputusan dalam penghasilan produk secara spesifik. Pelbagai keputusan yang perlu diselesaikan pada peringkat seni bina serta rekabentuk terperinci, menunjukkan bahawa terdapat keperluan bagi membantu dalam membuat keputusan semasa guna semula aset teras. Pembuatan keputusan pada peringkat seni bina adalah berdasarkan kepada pihakberkepentingan yang bersifat tidak menentu, kabur dan subjektif dalam membuat pemilihan seni bina berdasarkan keperluan bukan fungsi (NFR). Tambahan pula, tahap reka bentuk terperinci melibatkan pemilihan ciri yang sesuai dan rasional. Rasional bagi

pemilihan, jika tidak didokumenkan dengan betul, juga akan menyebabkan kehilangan pengetahuan tersirat. Oleh itu, teknik keputusan reka bentuk seni bina pelbagai attribut dicadangkan bertujuan untuk mengatasi masalah yang dinyatakan. Teknik ini menggabungkan *Proses Hierarki Analisis Kabur* (FAHP) dengan dokumentasi keputusan reka bentuk seni bina yang ringan untuk menyokong proses membuat keputusan ketika guna semula aset teras. Kami membuktikan pendekatan ini menggunakan kajian kes robot boleh gerak berautonomi (AMR). Pelaksanaan kajian kes ini menunjukkanteknik yang dicadangkan menyokong jurutera perisian dalam proses membuat keputusan pada peringkat seni bina dan peringkat reka bentuk terperinci.

Kata kunci: Kejuruteraan aplikasi; barisan keluaran perisian, *FAHP*, keputusan reka bentuk seni bina

1.0 INTRODUCTION

The big picture which comprises of process governing the development of core asset and derivation of the core asset for creating product specific application is known as Software Product Line Engineering (SPLE). SPLE has two main development phases: domain engineering and application engineering. Core asset derivation is the process of constructing an individual product from core asset in application engineering. In application engineering, choosing requirements and resolving appropriate architecture structure highlight the needs to understand the variant requirements and specifying them in order to address variants at architectural level.

The derivation is often based on experience, intuition, and domain expert knowledge. As a consequence, the quality of architecture depends on the skills of individual software architect [1]. Furthermore, informal decision during the derivation makes it difficult to trace architectural decisions. This causes difficulties when making changes later. Thus, lack of support in dealing with vagueness and uncertainty in making trade-off between quality attributes has been identified as multi-attribute decision making problem. This problem imposed challenge and complexity during the selection of suitable architecture for product specific derivation [2-5]. Furthermore, lost in tacit knowledge during architecture decision making further complicates the documentation of the rationale behind the design decision made by domain expert or software architect [6-8]. However, the existing approaches are either focusing solely on multi-attributes design decision or documentation of the rationale for the design decision.

Thus, by incorporating decision making support in application engineering, for assisting both the architecture selection and also rationale in selecting suitable components, is seen as reaping the benefit of assisting the domain expert or software architect decision making support at architecture and detail design levels. The main research question for this paper is: "How to support decision making during core asset derivation in application engineering?" Thus, the objective of this paper is to propose a multi-attribute architecture design decision and light weight architecture design decision documentation for core asset derivation in application engineering.

Decision making support is listed as one of the essential requirements for the product derivation process by authors in [1]. Decision making during product specific or core asset derivation is needed for the purpose of assisting multi-attribute design decision for identifying suitable architecture based on the quality attribute of the specific product and also for the purpose of documenting the architecture design decision.

Multi-attribute decision making (MADM) is seen as a systematic selection of suitable architecture based on the trade-off between finite numbers of different quality attributes [1] , [9]. Furthermore, authors in [1] have introduced four essential elements for software architecture design decision making techniques where three of the elements are the focus of this paper: quality attribute description, how quality attributes importance is represented, and lastly the fulfillment of the alternative quality attributes in each architecture pattern.

For the first element, quality attribute description is the most important element to be elicited in software development. However not all of the requirements can be seen as impacting the architecture. The notion of architecture significant requirements (ASR) are the quality attributes which have influence to software architecture [10-11]. Therefore, software quality such as performance, efficiency and reliability are among the candidates for ASR. However, using only a single word to describe ASR is not enough as it can be perceived in different context by different stakeholder [1]. Therefore, ASR should be made explicit in order for it to be understood by the stakeholders. One of the plausible way for making ASR explicit is by having a clear taxonomy of its description. There are several approaches such as in [3-4], [9], [12], [14-15] which concentrate on how quality attributes influence the selection of architecture with each approach taking different description for their quality attribute description.

The ambiguities in ASR can be further propagated to the wrong selection of the appropriate architecture, resulting to higher cost to modify or maintain the

software in the future. Therefore, a systematic approach is needed for the purpose of uncovering the appropriate architecture [11]. These difficulties further highlighted the significance of the second and third element listed earlier. Based on Falessi *et al*. [1], precise result is possible if quality attribute importance is represented using elicited weight in Analytical Hierarchical Process (AHP). Furthermore, it is also possible to yield finer results if AHP elicited ratio is used for representing the fulfillment of the alternative quality attributes in architecture pattern. Among the approaches which concentrate on quality attribute description listed in the previous paragraph, only [3], [9], [12], [13] concentrate on AHP to represent quality attribute importance and quality attribute fulfillment among architecture pattern. Thus, among them, only Dhaya and Zayaraz [3] and Zaki *et al*. [15] implement *FAHP* in their approach.

Feature model is the most used model to represent variability in SPL. Feature model is used to decide on which product should be derived from SPL. Feature model represents the variability by showing the variable selections in a form of hierarchical structure. The variable selections can be in the form of optional selection (OR relation), alternative selection, multiple selection and mutual exclusion (XOR relation). Additionally, as described by Capilla and Bosch in [8], there are similarities between decision model and feature model which is also known as variability model in SPL. Due to the similarities, there exist approaches which combine feature model and design decision together such as by Alebrahim and Heisel [17]. There are also approaches which extend the design decision and rationale concept in SPL such as the approach by Thurimella and Bruegge [16].

Architecture design decision (ADD) is the implicit and also tacit knowledge hidden in the mind of an expert software architect. Without a proper documentation of the knowledge, there are no explicit explanations as to why do the architecture is designed in such a way. The reconstruction of software architecture would have to make do with the undocumented knowledge of the software architect. Therefore, ADD is seen as the rationale for recording the decision making process which leads to the design of the architecture in the first place. Perry and Wolf [18] define rationale as "captures the motivation for the choice of architectural style, the choice of elements and the form". Without the documented rationale, the tacit knowledge in designing the architecture will vaporize and it will hinder the future changes to the architecture.

Design rationale approach is one of the earliest approaches to document rationale in architecture design decision and among them are Issue Based Information Systems (IBIS), Questions, Options and Criteria (QOC), and Design Rationale Language (DRL) [19]. In order to represent ADD, the inspiration comes from the design rationale work by Thurimella and Bruegge [20]. However, the design rationale based approach requires a link between questions, options and documentation of rationale which requires much effort from the user [21-22]. Therefore, a more lightweight approach is required and the mostsuitable approach is template based documentation proposed by [22] or based on UML annotation as implemented by Alebrahim and Heisel [17].

The paper is organized as follows: in the next section, background information on the underlying elements comprise of architecture decision making techniques are described. In section 3, the complete architecture decision making process which comprised of two levels of decision making is provided. Section 4 describes the proposed profile representing the proposed technique. Section 5 illustrates the applicability of the proposed technique using AMR as a case study. Related works are discussed in section 6 and finally section 7 concludes the findings and presents future works for the research.

2.0 DECISION MAKING PROCESS FOR CORE ASSET DERIVATION

The proposed multi-attributes architecture design decision documentation technique supports core asset derivation from two perspectives, early design decision making and detail design decision making. Based on literatures, difficulties faced in application engineering is to fulfill the quality requirements of artefact produced during product derivation. This is seen as an early decision making where the most suitable architecture pattern which can fulfill the desired quality requirements. However, there are usually more than one quality requirements suitable for a product and there may be unclear decision in determining the importance of each quality attribute. Therefore multi-attribute decision technique which is able to accommodate fuzziness in decision making is required. Thus, *FAHP*analysis is the technique chosen for assisting the decision making process. Figure 1 shows the first cycle, multi-attribute architecture pattern documentation. From the figure, based on specific application needed from stakeholder, quality attributes binding is done. Quality attributes binding is where the quality attribute and sub quality attribute that are identified during domain analysis are specialized into NFR based on the stakeholder need. The NFR statement for pairwise comparison will be the input for *FAHP* analysis and the analysis will produce the most recommended architecture pattern.

During the core asset derivation, the feature model is used to identify a suitable configuration for developing a product specific application. Suitable variation points in the feature model are selected and binded for the purpose of configuration in detail design decision making For the second cycle, feature based selection according to the design issue is done. Design issue is determined according to the stakeholder need identified during application engineering. Based on the design issue, suitable variation points and variants will be bound or selected. The selection should specify the suitable rationale for the choice taken. The selection of feature is then

updated into the architecture pattern and the outcome will be the product specific architecture based on the stakeholder need. The second cycle is illustrated in Figure 2. The detail description of the elements and mechanisms for each specific process is as elaborated subsequently in section 2.1 and section 2.2.

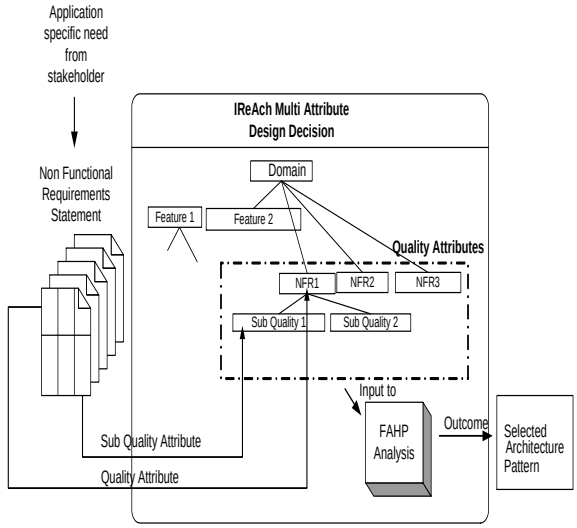


Figure 1 Multi-attribute architecture pattern identification

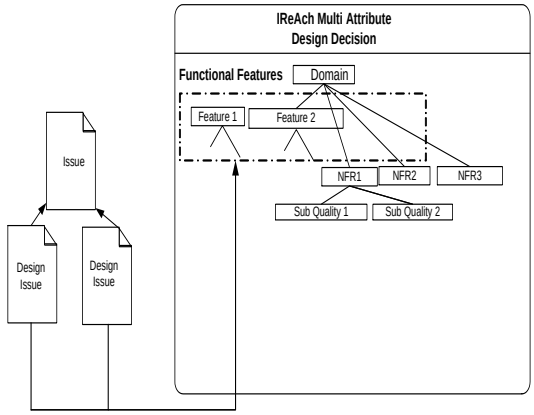


Figure 2 Feature based selection architecture design decision documentation

2.1 Multi-Attribute Architecture Pattern Identification

Enhanced with fuzzy technique, *FAHP* provides the linguistic scale to overcome the deterministic scale originally proposed in AHP. The linguistic scale known as Triangular Fuzzy Number (TFN) represents the fuzziness and uncertainty in human decision making. TFN extends the crisp importance priority in AHPwith fuzzy number whose membership is defined by three numbers, expressed as (l,m,u). The linguistic scale used for the pairwise comparison of quality attribute is adopted from [23] and can be referred in Table 1. Table 1 shows an example of pairwise comparison between efficiency, portability and maintainability. To convert the linguistics scale into its equivalent TFN, the scale and its corresponding TFN value from [24] are adopted and can be referred to in Table 2. Based on the pairwise comparison in Table 1 and its corresponding TFN value referred from Table 2, a fuzzy evaluation matrixA = $(a_{ij})_{n\times m}$ is constructed. The fuzzy evaluation matrixis then used as an input for FAHP analysis. Fuzzy Extent Analysis by Chang [25] has defined the value of fuzzy synthetic extent S_i with respect to the ith criteria as shown in Equation (1). Equation (1) represents fuzzy multiplication and the superscript -1 represents the fuzzy inverse. M_1 and M_2 are convex fuzzy numbers defined by the TFNs (l_1,m_1,u_1) and (l_2,m_2,u_2) respectively.

$$s_i = \sum_{j=1}^{m} M_{C_i}^{j} \left[\sum_{i=1}^{n}\sum_{j=1}^{m} M_{C_i}^{j}\right]^{-1} \quad \text{(Eq.1)}$$

The comparison can be done between M_1 and M_2 as in Equation (2).To compare M_1 and M_2, both values of $V(M_1 \geq M_2)$ and $V(M_2 \geq M_1)$ are needed. In Equation (2), iff represents "if and only if". As shown in Equation(2), for $V(M_2 \geq M_1)$ the highest intersection point, x_d can be determined between the domains of μM_1 and μM_2 with ordinate d as shown in Figure 3 [26]. Equation (3) is used to calculate ordinate d, the possible ordinate for the intersection between M_1 and M_2.

$$V(M_1 \geq M_2) = 1 \text{ iff } m_1 \geq m_2,$$
$$V(M_2 \geq M_1) = \text{ height } (M_1 \cap M_2) = \mu_{M_1}(x_d), \quad \text{(Eq. 2)}$$

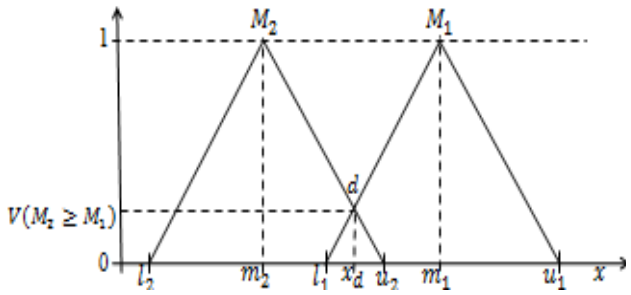


Figure 3 Intersection between M_1 and M_2

$$V(M_2 \geq M_1) = \text{ height } (M_1 \cap M_2) =$$
$$\frac{l_1 - u_2}{(m_2 - u_2) - (m_1 - l_1)} = d \quad \text{(Eq. 3)}$$

The degree of possibility for a convex fuzzy number M to be greater than the number of k convex fuzzy numbers M_i(i=1, 2,…,k) can be given by the use of the operations min and can be defined by Equation (4)

$$V(M \geq M_1, M_2 \ldots, M_k = V[(M \geq M_1) \text{and} (M \geq M_2)$$
$$\text{and } \ldots\ldots \text{ and} (M \geq M_k)]$$
$$= \min V(M \geq M_i), i = 1, 2, \ldots, k. \quad \text{(Eq. 4)}$$

Assume that $d'(A_n) = \min V(S_i = S_k)$, where k = 1, 2, …, n, $k \neq i$, whereA is a fuzzy evaluation matrix and n is the number of criteria. Then a weight vector (W') is given by Equation (5) with the normalized weight vectors (W)can be referred in Equation (6).

$$W' = (d'(A_1), d'(A_2), \ldots, d'(A_m))^{\mathrm{T}} \quad \text{(Eq. 5)}$$

$$W = (d(A_1), d(A_2), \ldots, d(A_m))^{\mathrm{T}} \quad \text{(Eq. 6)}$$

Table 1 Pairwise Comparison based on Linguistic Scale

Pairwise Comparison	Pairwise comparison: Importance of one criterion to another										
	Software Quality	Absolutely Important	Strongly Important	Fairly Important	Weakly Important	Equally Important	Weakly Important	Fairly Important	Strongly Important	Absolutely Important	Software Quality
Is Efficiency more important than Portability?	Efficiency		X								Portability
Is Efficiency more important than Maintainability?	Efficiency								X		Maintainability
Is Portability more important than Maintainability?	Portability							X			Maintainabil

Table 2 Linguistic Scale and TFN Value[a]

Priority	Linguistic Scale	Triangular Fuzzy Number (TFN)	Description
1	Equally Important (Eq. Imp)	(1,1,1)	Two quality attributes (*i* and *j*) contribute equally to the objective
2	Intermediate values between adjacent scale	(1,2,3)	Used to represent a compromise between the preferences 1 and 3
3	Weakly Important (W. Imp)	(2,3,4)	One quality attributes is weakly more important than the other
4	Intermediate values between adjacent scale	(3,4,5)	Used to represent a compromise between the preferences 3 and 5
5	Fairly Important (F. Imp)	(4,5,6)	One quality attributes is fairly more important than the other
6	Intermediate values between adjacent scale	(5,6,7)	Used to represent a compromise between the preferences 5 and 7
7	Strongly Important (S. Imp)	(6,7,8)	One variable is strongly more important than the other
8	Intermediate values between adjacent scale	(7,8,9)	Used to represent a compromise between the preferences 7 and 9
9	Absolutely Important (A. Imp)	(9,9,9)	One variable is absolutely more important than the other

Architectural pattern is used in this research as it has predictable non-functional properties where each pattern consists of one or more quality attributes. To enable the architecture pattern to be used in *FAHP*, the identified quality attributes must be given a numeric value or score as to how high or low the quality attributes are in the specified architecture pattern. This is referred in Section 2 as fulfillment of

quality attributes in software architecture pattern. The normalized weight vector identified earlier using fuzzy AHP calculation is then multiplied with the score value.

The fulfillment of the alternative quality attribute in each architecture pattern is adopted from [9] where the authors used discrete ordinal integer values of x ∈ [-2,2] for the score.

$$X = \begin{Bmatrix} -2 \text{ if symbol} = -- \\ -1 \text{ if symbol} = - \\ 0 \text{ if symbol} = 0 \\ 1 \text{ if symbol} = + \\ 2 \text{ if symbol} = ++ \end{Bmatrix} \quad \text{(Eq. 7)}$$

Using the score, eight architecture patterns from real time system domain [27] are analyzed, and the result of the analysis of the fulfillment of the quality attribute taken from ISO9126 in each of the eight architecture patterns areshown in Table 3. Using the normalized weight from *FAHP* and the fulfillment or score for the quality attributes for each architecture pattern, weight scoring method will be used to identify which architecture pattern has the highest results of the best matched architecture.

Table 3 Quality attributes fulfillment for architecture pattern[a]

Quality Attribute / Architecture Pattern	*Functionality*	*Reliability*	*Usability*	*Efficiency*	*Maintainability*	*Portability*
Layered	+	-	-	++	++	++
Five-Layer	+	-	-	++	-	++
Microkernal	++					
Channel	-	+	-	++	+-	+
Recursive Containment	++	-	-	-	-	-
Hierarchical	-	-	-	-	++	++
Virtual Machine	-	-	-	+	-	++
Component Based	++	-	-	--	++	-

2.2 Feature Based Selection Architecture Design Decision Documentation

Although representing design decision making provides a number of benefits, the means to capture design decision and its rationale is still an open issue [6]. This paper focuses on a lightweight approach to annotate the feature model with the design issue and rationale for choosing the variation points and variants. The annotation is based on the SysML profile extension which will be discussed in the following section.

In this second cycle of detail design decision making, the first step is to define the *Issue*based on the need given by the stakeholder for developing an application specific product. From the *Issue*, the second step is identifying the *Design Issues* and a suitable variation point and variants are selected and stored in *Configuration* tag value. Rationale is also included as to why the specified variants are selected and stored in *Rationale* tag value.

3.0 SYSML PROFILE FOR CORE ASSET DERIVATION

Profile extension is used as it helps to model the important elements in SPL which are variability and commonality. SysML profile is extended in order to represent Domain Analysis, Domain Requirements, Domain Architecture and Architectural Design Decision. The complete profile extension can be referred to in Figure 4. The focus of this paper is on the profile extension in the Domain Analysis, Domain Requirement and Architecture Design Decision section.

The profile starts with Domain Analysis where quality attributes such as reliability or maintainability can be clarified using the NFR stereotype which can be refined using ISO9126 quality attributes model. The ISO 9126 standard is chosen because the standard has a clear taxonomy, which comprises of sub quality attributes and suitable metrics to enable a measurable quality attributes. The quality attributes will then be used in the *FAHP* analysis. The analysis is not included in the profile and is shown as a dashed line in Figure 4. The second part of the SysML profile extension is to document architecture design decision. To realize this goal, the extension includes stereotypes for representing the issue and design issue in selecting the variation point in feature model. The design issue is further described using configuration and rationale tag. *Configuration* tag stores the value of the configuration of variation points and variants selected, while the *Rationale* tag stores the reason why the specific variation point is chosen. Due to both feature and decision models are modeled together, the variation point in the feature model has *trace* relationship with the design issue stereotype.

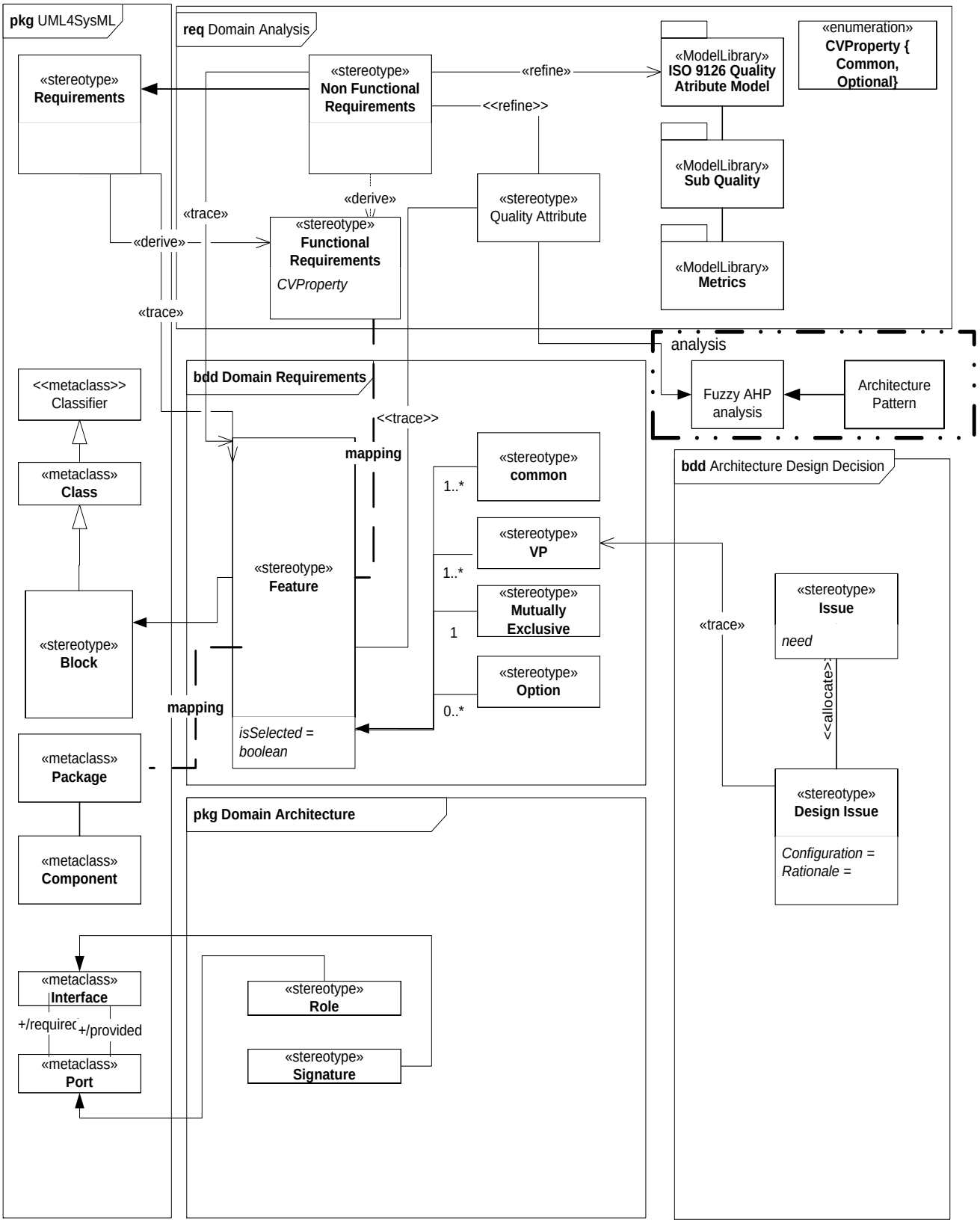


Figure 4 Extension of SysML profile

4.0 CASE STUDY APPLICABILITY

4.1 Case Study Design and Planning

Case study design and planning starts with the goal for case study implementation. The goal of this case study is to derive core asset using the proposed multi-attribute design decision technique as described in the previous section.

4.2 Data Collection

Data used for this study can be classified as third degree data collection [28]. This is because the data comes from independent analysis done on requirements specification, manual documentation and research publication from similar applications of Autonomous Mobile Robot (AMR). There are four AMR identified based on the research collaboration done at Embedded Real Time and Software Engineering Research Lab (ERetSEL, Universiti Teknologi Malaysia). The four AMR are AMR for research, AMR for teaching, i-wheelchair and intelligent scooter Universiti Teknologi Malaysia. The fifth AMR is the parking assistant based on the work of [29]. The requirements from similar applications are gathered for the purpose of common and variability analysis. Another third degree data collection comes from the analysis of quality attributes fulfillment in each architectural pattern. There are eight architecture pattern accumulated from domain specific real time patterns documentation by [27] as tabulated in Table 3.

4.3 Architecture Significant Requirements (ASR)

The first step is to understand the application specific need given by the stakeholder. Therefore the need specified in this case study is as follows:

To build an indoor wheelchair AMR for severely handicapped people

For the quality attribute binding, there are three quality attributes identified for the specified need, namely efficiency, maintainability and portability. For efficiency, there are two sub qualities involved, that are resource based and time based qualities. NFR statement was created to further refine the quality attributes and sub quality attributes identified for the specified AMR, where NFR statement for efficiency is as shown in Figure 5(a). NFR statement for quality attributes portability and maintainability are as shown in Figure 5(b) and 5(c) respectively.

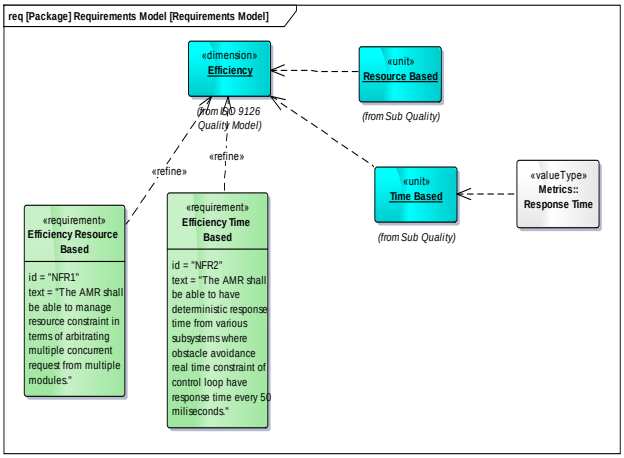


Figure 5(a) NFR statement for efficiency

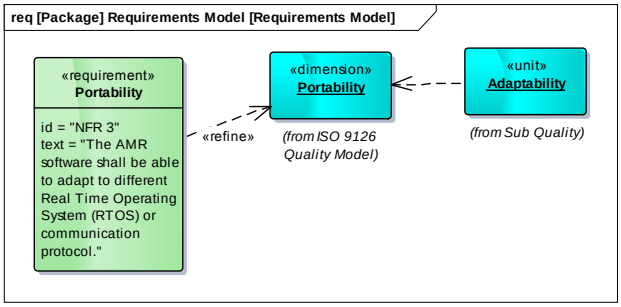


Figure 5(b) NFR statement for portability

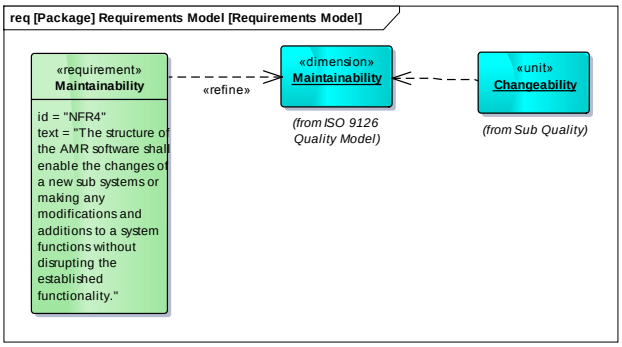


Figure 5(c) NFR statement for maintainability

4.4 Multi-Attribute Architecture Decision

The pairwise comparison for NFR as revealed earlier in Table 1, shows that efficiency is strongly important than portability and represented as S.Imp in Table 4. Table 1 also shows Maintainability is strongly important than Efficiency and the pairwise comparison is also represented as S. Imp in Table 4. Pairwise comparison between portability and maintainability in Table 1 shows Maintainability is Fairly Important than Portability and denoted as F. Imp in Table 4.

Table 4 Pairwise comparison of quality attributes importance[a]

	Efficiency	Portability	Maintainability
Efficiency	-	S. Imp	
Portability		-	
Maintainability	S. Imp	F. Imp	-

Pairwise comparison of the quality attributes importance in Table 4 has to be converted into its equivalent TFN value. For the conversion, the linguistic scale and its corresponding TFN value from Table 2 is referred. Based on the conversion, a fuzzy evaluation matrixA = $(a_{ij})_{n\times m}$ is constructed and the value can be referred in Table 5. The fuzzy evaluation matrixis then used as an input for *FAHP* weight calculation as described in subsequent paragraphs.

Table 5 Fuzzy evaluation matrix for the quality attribute[a]

	Efficiency	Portability	Maintainability
Efficiency	(1,1,1)	(6,7,8)	(1/6,1/7,1/8)
Portability	(1/6,1/7,1/8)	(1,1,1)	(1/4,1/5,1/6)
Maintainability	(6,7,8)	(4,5,6)	(1,1,1)

FAHP weight calculation follows the equations as described in Section 3.1. The following calculations show sequence of steps for the calculation.

Fuzzy Synthetic Extent calculation by applying Equation (2).

$S_{Efficiency}$ = (0.36, 0.33, 0.37)
$S_{Portability}$ = (0.26, 0.05, 0.28)
$S_{Maintainability}$ = (0.55, 0.52, 0.60)

M extent analysis objects comparison using Equation (3) and Equation (4).

$V(S_{Efficiency} = S_{Portability}) = 1$
$V(S_{Efficiency} = S_{Maintainability}) = 1$
$V(S_{Portability} = S_{Efficiency}) = 0.4387$
$V(S_{Portability} = S_{Maintainability}) = 1$
$V(S_{Maintainability} = S_{Efficiency}) = 1.8$
$V(S_{Maintainability} = S_{Portability}) = 1$

Weight vector object calculation with Equation (5)

d'(efficiency) =V (Sc1 = Sc2, Sc3)
= min(1,1)
=1

d'(portability) =V (Sc2 = Sc1, Sc3)
= min(0.4387,1)
= 0.4387

d'(maintainability) =V (Sc3 = Sc1, Sc2)
= min(1.8,1)
= 1

Weight vector is normalized to obtain a non fuzzy number using Equation (6).

The weight vector before normalization is:
W' = (1,0.4387, 1).

The value of weight vector after normalization with respect to Efficiency, Portability and Maintainability is:
$W = (0.410, 0.180, 0.410)^T$

Using the weight from *FAHP* and the score value obtained from the fulfillment of quality attributes for each architecture pattern as shown in Table 3, the multiplication of both values determines which architecture pattern has the highest ranking. From the result as shown in Table 6, the highest value comes from Layered pattern.

Table 6 Ranking of architecture pattern based on weight and score calculation

	Efficiency	Maintainability	Portability
Layered	0.82	0.36	0.82
Five Layered	0.82	-0.18	0.82
Channel	0.82	0.18	0.41
Hierarchical	-0.41	0.36	0.82

4.5 Architecture Design Decision Documentation

The first step is to define the issue. The issue is based on the need identified earlier in section 4.3. Related design issues are specified as shown in Figure 6, where there are three allocated design issues for the identified issue Selection of feature suitable for the design issue is done afterwards. For each selection of features, it will be recorded in the configuration tag value and the rationale for the selection is recorded in the rationale tag value. The design issues and the recorded tag values with the selected features represent the ADD documentation concept for this research. The selected features together with its associated design issues are as shown in Figure 7. For the purpose of clarity, the design issue together with the configuration of selected feature variants and its rationale aretabulated in Table 7.

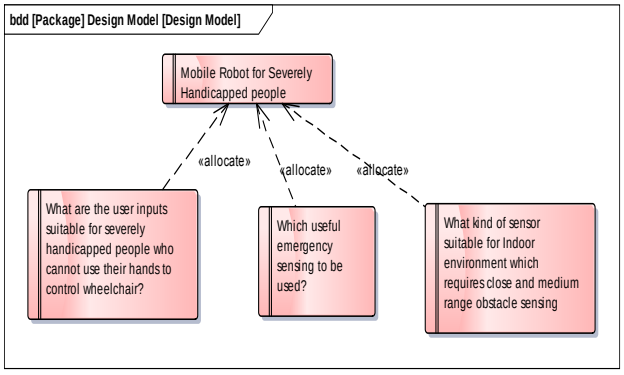


Figure 6 Issue and design issues for AMR case study

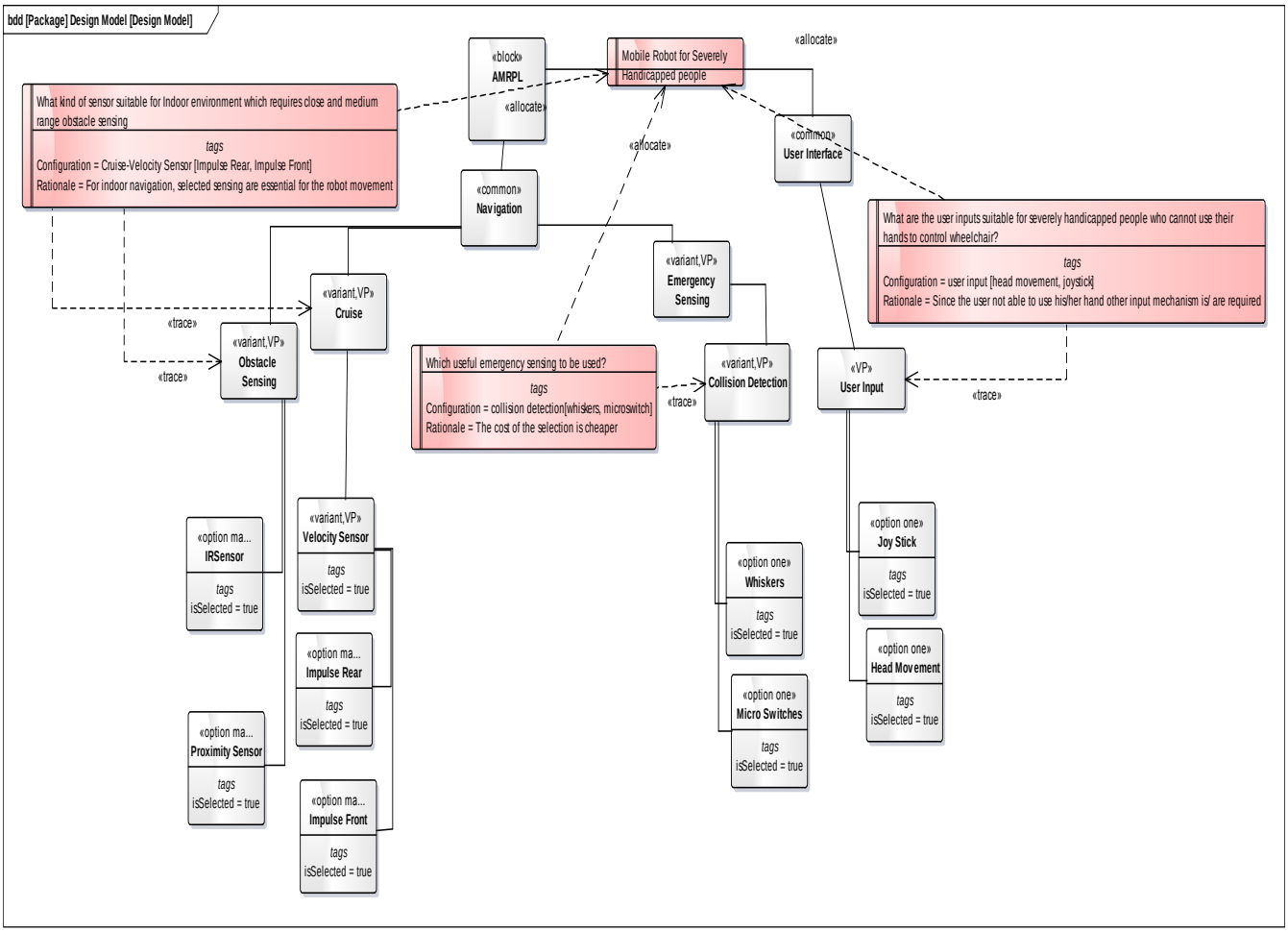


Figure 7 Selected features and the corresponding design issue

Table 7 Design issue and its corresponding configuration and rationale

Design Issue	Configuration Tag Value	Rationale Tag Value
What kind of sensor suitable for indoor environment which requires close and medium range obstacle sensing?	Velocity Sensor = [Impulse Rear, Impulse Front] Obstacle Sensing = [IR Sensor, Proximity Sensor]	For indoor navigation, selected sensing are essential for the robot movement
Which useful emergency sensing to be used?	Collision Detection = [Whiskers, Micro Switch]	The cost of the selection is cheaper
What are the input suitable for severely handicapped people who cannot use their hands to control the wheel chair?	User Input = [Head Movement, Joystick]	Since the user not able to use his/her hand other input mechanism is/are required

4.6 Case Study Discussion

The core assets are derived using the application engineering process. In this process, the quality and sub quality attributes are further being bind with NFR statements where it can give a concrete meaning to the ASR. The taxonomy identified from ISO 9126 is compatible enough to represent quality attributes such as Efficiency, Portability and Maintainability. However, there is a possibility that the taxonomy is not sufficient enough to represent every possible quality attributes such as scalability and performance. The quality attributes further assist in the pairwise comparison for the purpose of ranking the most suitable architecture. The linguistic scale and its corresponding TFN value further helps in overcoming the uncertainty and fuzziness in making trade-off for the quality attribute. The pairwise comparison using *FAHP* also shows that the technique is practical in quantitatively identifying the most suitable architecture pattern.

Furthermore, the design issues also helps in the identification of possible selection of variation points and variants in the feature model which can be recorded at the configuration tag value. Aside from that, the rationale for the selection of the variation points and variants can also be recorded in the rationale tag value. Strategies that might enhance the ADD documentation might involve a generated document for the ADD based on the value input at the modeling level. This gives an ample room for describing the configuration and rationale due to the little amount of display space in the model view. The tabulated display of design issues and its corresponding configuration and rationale as shown in Table 7 helps in relating the design decisions and features selected as suggested by Capilla and Bosch in [8].

5.0 RELATED WORK

The importance of ASR in influencing the architecture is undeniable as shown by the incorporation of the ASR in the form of quality attributes description in several approaches such as in [3-4], [9], [12–15], [20]. However, majority of the approaches do not have a complete representation of the quality attribute using the taxonomy identified from ISO 9126 except the approach used in Thurimella and Bruegge [20].

There are several approaches such as [3], [9], [12,-13] that concentrate on AHP to represent quality attribute importance and quality attribute fulfillment among architecture pattern. However, among them only Dhaya and Zayaraz [3] and Zaki *et al*. [15] which have the same notion as this paper in implementing *FAHP* for multi-attribute decision making. The incorporation of linguistic scale and TFN to accompany AHP help to overcomeuncertainty and fuzziness in making trade-off for the quality attribute. However, both approaches [3] and [15] do not have a clear taxonomy in representing quality attributes.

Thurimella and Bruegge [20] present an approach which combines variability and design rationale. However, the design rationale based approach requires a link between questions, options and documentation of rationale which requires much effort for the user. Therefore, a more lightweight approach is required and the most suitable approach is template based documentation proposed by [21] or based on UML annotation as implemented by Alebrahim and Heisel [17]. In this paper, the latter approach of annotating the feature model for the purpose of documenting design decision is implemented. However, this paper differs in terms of implementing design decision for feature selection instead of implementing it during component configuration as in [17].

This research has implemented the three most prominent elements for MADM in software architecture design decision making [1], namely quality attributes description, the importance of quality attributes and the description of fulfillment of quality attribute. The implementation of the three elements is seen as a systematic guidance in making decision in deriving architectural design during application engineering in SPL. The first element uses ISO 9126 taxonomy and implements it in the form of model based annotation as in Figure 5(a-c) which contributes towards a clearer taxonomy for quality attributes. Furthermore, to represent the importance of quality attributes for the stakeholder, the second element uses elicited weight in the form of stakeholder pairwise comparison of quality attributes using *FAHP*. The use of *FAHP* further assists in overcoming the vagueness and uncertainty experienced by the stakeholder during decision making.In addition, for the third element, ordinal scale of +, -, ++ and – are used as a representation for the purpose of describing the fulfillment of quality attributes for each alternative architecture as shown in Table 3. The ordinal scale is seen as a suitable measure for the quality attributes since there is no precise quantification for quality attributes fulfillment. Lastly, as there will be numerous rationale in architecture decision making during application engineering, design decision documentation annotationis perceived as suitable to record the tacit knowledge during design decision. The combination of these elements that have not yet been implemented by other researchers can be seen to support the architecture design decision during core asset derivation in SPL.

6.0 CONCLUSIONS AND FUTURE WORK

The imprecise, uncertain and subjective nature of human or in this case stakeholders in making decision on the suitable quality attributes for SPL and also rationale for selecting suitable components for the PLA leads to a less objective decision. Therefore, a multi-attribute decision making with the ability to handle the uncertain and subjective nature of

human decision is proposed by using *FAHP*. Moreover, selected architecture pattern based on the quality attribute ensures that the architecture has the quality desired by the stakeholder.

Furthermore, the selection of different alternatives in designing the product specific application is not being recorded explicitly. This leads to lost or forgotten design rationale. However, not many practitioners are interested in documenting their rationale. Thus a lightweight decision representation is required in order to help in choosing the suitable variation points for product derivation process. In conclusion, the multi-attribute design decision technique helps in deriving application specific product in providing a fuzzy based linguistic to overcome the uncertainty and subjectivity in making decision regarding to the quality attribute. The design decision document is also seen as a lightweight approach by annotating the feature model with rationale of choosing the specific feature for the product derivation.

As a future work, tool support should be implemented based on the proposed technique. As for now,the difficulty is to establish an automated environment for supporting the proposed approach with the extension of *FAHP* multi-criteria design decision. Furthermore, a complete rule is needed for the mapping between the architecture pattern and the feature selection in enabling a product specific architecture derivation. Therefore, further investigation needs to be done in order to integrate the proposed extension models and the multi-criteria design decision concept and to define the rule in a more formal way for the tool. Moreover, the incorporation of a new series of standard known as *Software Product Quality Requirements and Evaluation* (SQuaRE) should be investigated to replace the use of ISO9126 in defining the quality attributes terms.

Acknowledgement

We are grateful for the UTM scholarship to Author 1.Furthermore, we express our gratitude profoundly to Graduate Supervision Grant, Universiti Teknologi Malaysia (UTM) and Ministry of Science and Technology (MOSTI) under Vote No. 4S064 for their financial support. Our profound appreciation also goes to ERetSEL lab members for their continuous support in the working of this paper.

References

[1] Falessi D., Cantone G., Kazman R., and Kruchten P. 2011. Decision-making Techniques for Software Architecture Design: A Comparative Study. *ACM Computing Survey*. 43: 1-28.

[2] Jansen, A. and J. Bosch. 2005. Software Architecture as a Set of Architectural Design Decisions. *5th Working IEEE/IFIP Conference on Software Architecture (WICSA'05)*. 109-120.

[3] Dhaya, C. and Zayaraz, G. 2012. Fuzzy based Quantitative Evaluation of Architectures using Architectural Knowledge. *International Journal of Advanced Science and Technology*. 49: 137-154.

[4] Babu, D., Rajulu, G., Reddy R., Kumari A. 2010. Selection of Architecture Styles using Analytic Network Process for the Optimization of Software Architecture. *International Journal of Computer Science and Information Security*. 8(1).

[5] Moaven, S., J. Habibi, H. Ahmadi, and A. Kamandi, A Decision Support System for Software Architecture-Style Selection. *2008. Sixth International Conference on Software Engineering Research, Management and Applications, 2008*. 213-220.

[6] Weinreich, R. and G. Buchgeher. 2010. Integrating Requirements and Design Decisions in Architecture Representation. Lecture Notes in Software Engineering (LNCS), Springer Berlin Heidelberg. 86-101.

[7] Lytra, I., H. Eichelberger, H. Tran *et al.* 2014. On the Interdependence and Integration of Variability and Architectural Decisions. *Proceedings of the Eighth International Workshop on Variability Modelling of Software-Intensive Systems - VaMoS '14*. 1-8.

[8] Capilla, R. and Bosch J. 2013. Software Variability and Design Decision. In Systems and Software Variability Management, R. Capilla, J. Bosch, and K.-C. Kang, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg. 287-292.

[9] Galster, M. Eberlein, A. and Moussavi, M. 2010. Systematic Selection of Software Architecture Styles. *Journal of IET Software*. 4(5): 349.

[10] Chen, L., Babar M. A., and Nuseibeh B. 2013. Characterizing Architecturally Significant Requirements. *Journal of IEEE Software*. 30: 38-45.

[11] Niu, N., Xu, L., Cheng, J. and Niu Z. 2013. Analysis of Architecturally Significant Requirements for Enterprise Systems. *IEEE System Journal.* 8(3): 850-857.

[12] Kim, J., Park, S. and Sugumaran V. 2008. DRAMA: A Framework for Domain Requirements Analysis and Modeling Architectures in Software Product Lines. *Journal of System Software*. 81: 37-55,.

[13] Moaven, S., J. Habibi, H. Ahmadi, and A. Kamandi. 2008. A Fuzzy Model for Solving Architecture Styles Selection Multi-Criteria Problem. Second UKSIM European Symposium on Computer Modeling and Simulation. 388-393.

[14] Zhang, Y. Y.Sun, Peng, X. Cui, and H. Mei. 2011. Towards Quality Based Solution Recommendation in Decision-Centric Architecture Design. *Proceedings of the 23rd International Conference on Software Engineering and Knowledge Engineering (SEKE)*. 776-781.

[15] Zaki, M. Z. M. Jawawi, D. N. A, Halim, A. S. Hamdan N. M. 2013. Multi-Criteria Architecture Style Selection for Precision Farming Software Product Lines Using Fuzzy AHP. *International Journal of Advanced Software Computing Application*. 5: 3.

[16] Thurimella, A. K. and B. Bruegge. 2012. Issue-based Variability Management. *International Software Technolology*. 54(9): 933-950.

[17] Alebrahim, A. and M. Heisel. 2012. Supporting Quality-Driven Design Decisions by Modeling Variability. *Proc. 8th Int. ACM SIGSOFT Conference on Quality Software Architecture. - QoSA '12*. 43.

[18] Perry, D. E. and Wolf, A. L. 1992. Foundations for the study of Software Architecture, ACM SIGSOFT Softw. Engieering Notes. 17(4).

[19] A. Dutoit, R. McCall, I. Mistrik, and B. Paech. 2006. *Rationale Management in Software Engineering*. Springer Verlag, Heidelberg, Germany. 1-449.

[20] Thurimella A. and S. Ramaswamy. 2012. On Adopting Multi-Criteria Decision-Making Approaches For Variability Management In Software Product Lines. *Proceeding of 16th Software Product* Line. 32-35.

[21] Salvador, van der Ven, J. Jansen J., Nijhuis, G. and Bosch, J. 2006. *Design Decisions: The Bridge Between Rationale and Architecture. Rationale Management in Software Engineering.* Springer Berlin Heidelberg. 329-348.
[22] Tyree J. and Akerman, A. 2005 Architecture Decisions?: Demystifying Architecture. *Journal of IEEE Software*. 22: 2.
[23] Özdagoglu A. and G. Özdagoglu. 2007. Comparison of AHP and FUZZY AHP for the Multi-criteria Decision Making Processes with Linguistic Evaluations. Istanbul Ticaret Üniversitesi Fen Bilim. Derg. 6: 65-85.
[24] Ayhan, M. B. 2013. A Fuzzy AHP Approach for Supplier Selection Problem: A Case Study in a Gear Motor Company. *International Journal of Managing Value and Supply Chains (IJMVSC)*. 4(3): 11-23.