Towards Quality Attributes Decision Modeling Approach for a Product Line Architecture

I Made Murwantara

Research and Development Computer Lab., Informatics Dept. Universitas Pelita Harapan, Indonesia

Summary
Hybrid Formal Concept Analysis – Analytical Hierarchy Process (HFA) for decision modeling the product line architecture development give hints to the initiation of the software architecture design. It enables the software architecture to have clear view of variabilities and dependencies in the architecture of a product line. Further, architecture stability which is the issue that arises during architecture configuration, can be coped with this approach. In this approach, the Formal Concept Analysis acts as the cluster manager, that grouping the components to have specific functionality relationship. Then, the Analytical Hierarchy Process calculates the priority of each components. In the case of quality attributes, the HFA groups the components that related to specific quality attributes and its combination. However, some challenges on cross-cutting components and error design may arise in this process. To this, the product line architecture is layered and managed using the Consistency and Variability Manager. However, during product configuration in the Product Line Architecture, software architect have to make decision for components with variability. Furthermore, the software architect must know exactly what will happen to the final product, especially to the quality attributes. To this, this paper proposed the HFA, which support the software architecture of product line to have clear expectation of specific architecture that being configure. The proposed approach demonstrated in a eLearning Product Line.

Key words:
Decision Modeling, Product Line Architecture, Variability, Formal Concept Analysis, Analytical Hierarchy Process

1. Introduction

Software Product Line (SPL) is a new paradigm of software development that heavily reuse the whole artifacts of member products of a product line. It aims to leverage the extensive reuse of software development, and to reduce costs, time of production by maintaining its high quality. In the SPL developments, there are two processes that are Domain Engineering and Application Engineering. Domain Engineering, mainly, identifies the commonalities and variabilities within a given domain. It exploits the reusable artifacts of member product that will be reused for engineering new products in the specific domain. Meanwhile, Application Engineering derives the Domain Engineering result to form the reference architecture.

Product Line Architecture (PLA) is the key success factor of SPL development. PLA compose the artifacts of member products to produce specific architecture. The configuration of specific architecture, mostly, by selecting the components of member products. During selection of components several issues arise, such as relationship and dependency. The dependency may arise as a result of intersection between the software components from different group members, that form the quality attributes. Furthermore, the quality attribute is the prominent issue of dependencies between components. Where, the quality attributes corrrespond to the architecture stability of software architecture design. The change of quality attributes affect the architecture structure. This happens as the product line architecture develops specific architecture by configure the artifacts of member products, that ultimately, affect the whole architecture.

Decision modeling simulates the component composition by taking into account several factors, such as Quality Attributes and Functionalities. Decision modeling also models the components to have the suitable software architecture configuration to answer the requirements. Where, the new product specifications arrive as requirement, that established by querying the architecture elements of member products. In the product line architecture, functionalities derives as the software components. One or more functionalities may exist in one components, and sometimes, it need a group of components to correspond for a functionality. In general, the relationship between components or groups of components resulted on specific or common quality attributes. This, makes the selection of components during product configuration, hard to implement it. In this case, the decision of component selection must taking into account the changes of architecture structures as a result of software architecture configuration.

Decision modeling, in SPL, supports the variability modeling. It presents the analysis of complexity and diversity of products in a specific domain, that efficiently achieve product derivation process. According [1], in the basic model structures, decision model share commonalities. The decision is represented as a set of choices, that comprises of a set of references that forms the decision model. It reveals the decision as unique attributes, and it has dependencies among decisions.

The PLA includes the artifacts of member products of a product line, and develop the architecture for specific product by configuring the components. Primarily, components relates via interaction elements, e.g. connector,

Manuscript received November 5, 2011
Manuscript revised November 20, 2011

ports. The components configuration establish the software architecture. Further, the component relationships are represented by the interaction elements. The SPL derive into PLA, by mapping a feature model into groups of components. Variant components build the constraints for specific product. If more than one variant components, that groups on different component category. Then, software architect needs to decide whether one composition may change the component specification. If the groups of components are the representation of quality attributes, then, they need to make a decision model that able to include quality attributes as the decision value.

Quality Attributes affects system design. When, the quality attribute considered in the architecture configuration, it give significant impact to the architecture structure. In the product line architecture, quality attributes may emerge as the dependencies of components that address specific goals, such as security, reliability or usability. In addition, the Product line architecture differs to the traditional architecture in terms of quality attributes. Where, the traditional software architecture do not deal with variability. Further, Complex dependencies of components that emerge in the product line architecture need to be clarify. It pin points the presence of variability in the specific software architecture.

Groups of components establish the quality attributes. Where, the modification of components composition change the quality attribute. The decision of which alternatives should be included into the composition affect the whole architecture. For example, if there are two optional components, which are “Interaction Resources” and “Evaluation Resources”. When one of those optional component is composed to the component “Participant Management”, then it will form two quality attributes possibilities, which are “Reliability” and “Usability”. The composition of “Interaction Resources” and “Participant Management” form “Reliability”. Similarly, the composition of “Evaluation Resources” and “Participant Management” establish “Usability”. However, how do the software architect knows the best composition that answer the requirement, and How to help the architect engineer to select the suitable alternatives. To address these shortcomings, modeling the alternatives in a compact form is essential. Further, the decision dependencies may emerge in a complex architecture, and to model this dependencies as well.

This paper investigated the decision modeling of software architecture design via hybrid Formal Concept Analysis - Analytical Hierarchy Process (HFA) for a product line architecture. It aim to capture and evolve a SPL's assets so as to gain insight into architecture elements diversity, efficiently.

1.1 Decision Modeling in Product Line Architecture

Decision modeling in PLA manages and supports the choices of development path by composing or decomposing software architecture elements. Development path reveals the software architecture composition in a concise manner. In general, the software architecture comprises of components and its relationships. Regardless the interaction elements that build the relationship, the software component is the main elements that compose the software architecture.

Decision modeling in PLA consists of two parts [5], firstly, defines the structures and elements to build the decision model. And, specifies the decision characteristics. In the first part, the decision groups into sets of decision model. Then, it is organized into tree form to determine the type of decision that found in the decision model.

In the Component-based Software Engineering (CBSE), the reuse of components is the key success factor [3]. Thus, the reusable components must be developed from the application domain point of view, not from a specific application does. In this case, decision should pinpoint the functionalities as the orchestration of components. Decision that coming from an instantiation process, therefore, generate the component instantiation which is different to the original one. Above all, the decision model only in the build level components, and the logical composition [4] can be achieved via the functionality within component.

The decision can be differentiated into restricted and unrestricted [5]. The restricted decision contains restrictions specification. While, the unrestricted decision have the constraints specification that do not support other constraints, which differs to its data constraints. Both, the restricted and the unrestricted decision may exist in the same specification of decision model. The key success factor of decision process is a stable architecture, that won't change the structure and logic of software architecture, in the PLA, during architecture design activity.

1. 2 Related Work

In the component-based development, Kobra [2] uses a tabular notation as their decision model. Kobra have separate decision level, from simple decisions to advanced decisions. The resolution of those level specify the selection for each product instances. The Kobra approaches using UML as their modeling tool and the model is configured to have specific architecture structure. A decision can be mapped to more than one architecture, and the decision type may vary, based on the variability type, e.g. Optional and Variant. Dependencies is captured as the resolution of decision.

DOPPLER [1] decision model comprises of a set of decision and their dependencies. The decision process

begins by answering the question that asked to the customer, where each decision have a unique name. And, the answer depend on the type of the decision (Boolean, string). The range of allowed answer is restricted by validity condition. Further, the decision hierarchically depend to the other decision and it must be resolve before other decision logically accepted.

2. Research Method

In this research, several decision modeling approaches and its related works have been analyzed. After carefully reviewing the existing approaches, there are chances to support the decision model by presenting the quality attributes. In a product line architecture, the quality attributes involve many components that cross-cuts, which is uneasy to resolve for a product line architecture design. It is frequently the case however that the design error sometimes emerge when the alternatives do not have, both, its value and rank of importance. In this case, if the alternatives can be pre-computed, the decision will be concise. Both, Analytical Hierarchy Process (AHP) and Formal Concept Analysis (FCA) can be used to support a decision model in a product line architecture, by composing or decomposing groups of components and their relationship in an architecture design. AHP [7] aims to have best decision that suits the goals, that using pair method. The AHP compose the decision into hierarchy, that each sub-composition can be examined independently. The elements of hierarchy corresponds to any quality attributes elements, that are groups of software components. Subsequently, each quality attributes are valued based on the evaluation of critical, effective, and impact. After that, the AHP convert those value into numerical value, and compared to entire range of component compositions. The result is the representation of alternatives that is offered to the architecture designer.

In the FCA [8], it aims to have natural clusters of attributes and object input data. Where, the set of all the share common attributes are clustered as object cluster, and the set of all attributes that shared to object cluster as property cluster. The property cluster correspond one-to-one with object cluster, and a pair comprising of object cluster and property cluster forms a concept. This concept build from the mathematical axiom that is called lattice, and well known as concept lattice.

In this decision model approach, as depicted in Figure 1, the FCA analyzes the possibility of components cluster to answer the requirement question, and the AHP forms the range of critical decision..

The AHP Quality attributes affect the architecture design of a PLA, it reveals the importance of components that forms the architecture as a unique collaboration of functionality. Each component may have more than one functionalities, depend on its relationship to other components. The proposed approach includes the following elements:

1] Identifying the relevant component variant to specific quality attributes.
2] Predicting the significance of each variable components that identified in quality attributes.
3] Defines the architecture design for quality attributes in component model.

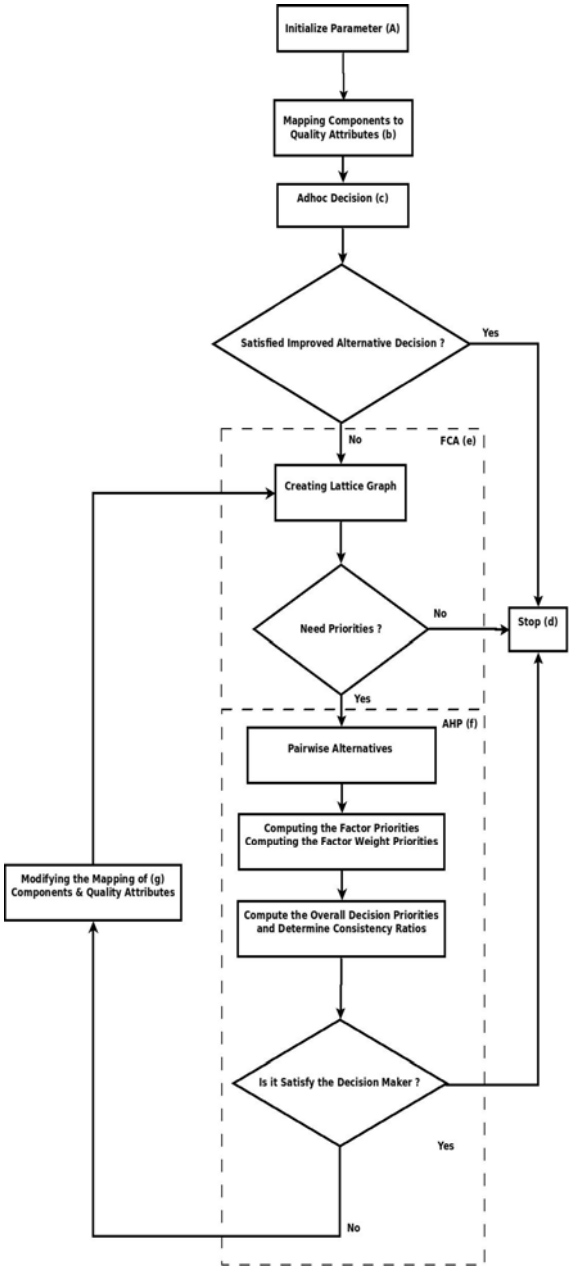


Figure 1. Hybrid Formal Concept Analysis – Analytical Hierarchy Process Approach (HFA)

This paper introduces a decision modeling method: the Hybrid Formal Concept Analysis - Analytical Hierarchy Process (HFA). It aims to offer solutions for complex problem and automate the suitable alternatives selection. The basic process of HFA is shown in Figure 1. (a) The general idea of HFA in this research is that an initial parameter of components and quality attributes are created at the beginning, and (b)used as an initial parameter input into FCA. (c) FCA will terminate if the alternative is satisfied (d), in terms of grouping the components against the quality attributes after the generated lattice graph (e), otherwise, (f) the selection of pairwise alternatives will be obtained. In addition, it computes the pairwise alternatives to get factor priorities. After that, the factor weight priorities are computed. The result of weight priorities are computed, to have the overall decision priorities, and then, consistency ratios are determined. Subsequently, if the alternatives that is provided by this approach satisfied, the process is finished. Otherwise, (g) the components can be added some more or reduced, and the Quality Attributes may have modification. Then, the process start from the lattice creation.

3. Results and Discussion

A e-Learning Product Line Architecture (el-PLA) based on MOODLE [9] was used to show the feasibility of this approach, that is depicted in Figure 3. In the el-PLA, several Quality Attributes exists, such as Security, Reliability, and Usability. The deployment that involved the Internet and Intranet create the high risk of the system in terms of security. In this case, Security is handled by several components, which are component “Authorization Controller”, “User Account Manager”, “Key Generator”, and “Roles Access Manager”. In general, if High Security is implemented into a system, then the network speed will be low, low process speed or too many credential must be provided. In the Reliability, the readiness of the system to serve users is the biggest challenge. The system have to be able to cope all operational in routine circumstances. The components that correspond to this quality attributes are “Course Manager”, “Participant Manager”, “User Account Manager”, “Display Controller” and “Roles Access Manager”. Meanwhile, the usability that correspond to the easiness to use and to learn is handled, primarily, by the following components; “Display Controller”, “Course Manager”, “Event Trigger”, “User Account Manager” and “Content Manager”. It found that many cross-cut components corresponds to the quality attributes which is difficult to configure during architecture design.

3.1. Identifying the component variable for quality attributes.

Identification of pertinent components, primarily, derive from the feature model [10]. Where, the domain expert defines the quality attributes of correspond feature models, already. Each components that represent groups of features will be influenced from the functional features that correspond to quality attributes. In this case, the domain expert's knowledge and experience play an important role in the identification process. In particular, quality attributes may be cross cut the architecture. In this case, one component may be included for more than one quality attributes [11]. For example, the component “Content Manager” and “Participant Manager” are included for the quality attributes of secure learning. The component “Content Manager” have two optional alternatives. The first, in term of content that the alternatives are Single or Share. The second, in term of learning content resources location that it may be stored internal in the same server or external on different server. The component “Participant Manager” also have alternatives, that are closed participant which means only specific users are allowed to join, or open participant that everyone may join to the learning system. And, if the decision is high security, then the alternatives of closed participant in the “Participant Manager”, and single content and the internal content resources of “Content Manager” should be appeared as alternatives. On the other hand, when quality attributes of Usability Learning is decided. Then, the relevant alternatives are open participant of “Participant Manager”, and single content and the internal content resources of “Content Manager” will be represented as alternatives elements. Both usability learning and high security, showed us that the quality attributes may cross-cut the architecture decision. In another word, the quality attributes should be simulated and pre-computed to have a good decision.

3.2. Predicting the Significance of Components for Quality Attributes

In the product line architecture, components cross-cut among the architecture [12]. In order to have quality attributes in the configuration, the components must be represented explicitly. For example, as illustrated in Figure 2, relationship between component “Participation Management” and “Evaluation Resources” will create a managed learning functionality. Meanwhile, the relationship between component “Participant Management” and “Course Manager” represent the restricted user functionality. Further, if both functionalities are collaborated, then the reliability will be cultivated, which is a quality attributes. From this perspective, functionalities relationship of components may be grouped

as a quality attributes. The problems emerge when there is a separation of the group of functionality within components. It may change the quality attributes, logically, and also change the functional structure of software architecture. However, the value of functionalities that impact into quality attributes can be measured, by qualifying groups of components [13]. The prominent problem in the decision model for a PLA, is to find the most suitable or matching components [14]. Furthermore, it should answer the quality attributes by grouping the components, dynamically [15]. As shown by the FCA result, the quality attributes still have problems on how the group of components answer the quality attributes. To address this shortcomings, a hybrid approach of the FCA and AHP is formed. As shown in Figure 2, the hybrid of the FCA and elements are mapped to each location of cluster that match to the present of quality attributes and components. Then, all highest values in the FCA graph are structured, hierarchically, from the highest to the lowest, for example {a,b}R,A means the component "Participant Management" and "Course Manager" cluster into "Reliability" and "Availability" which values are {aR, aA, bR, bA} . After that, all cluster's value are ranked. The highest value of the result shows the best decision that match to the quality attributes need.

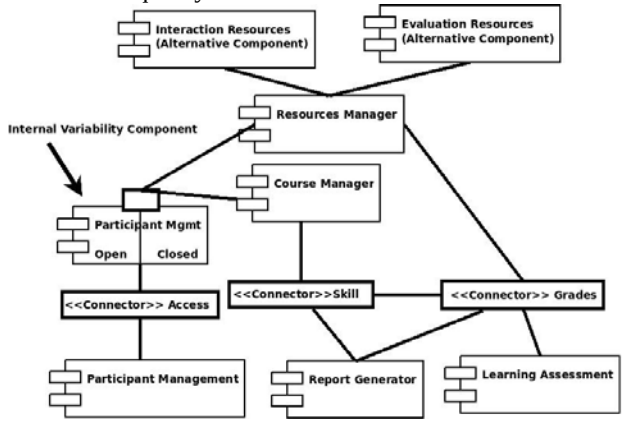
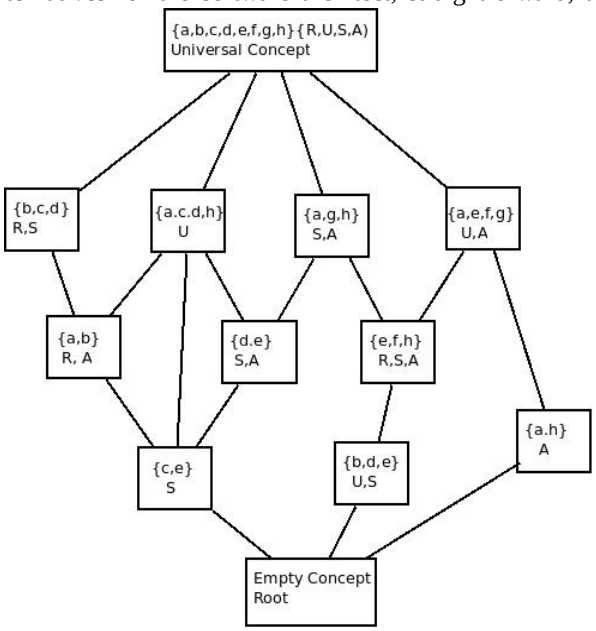


Figure 2. The e-Learning Product Line Architecture Component Model

There are still ambiguous on specific quality attributes decision as seen on Figure 2. To address this shortcoming, we need to analyze the cross product of clusters in the FCA. Indeed, the perfect match of component composition to quality attributes may differ significantly, if it correspond to the component functionalities. This approach propose to use challenge and conquer method. For example, each clusters in the FCA graph have functional and quality attribute map. If functional composition affect the quality attributes, then, component should have its best match to the quality attributes information. The AHP sharpening the alternatives of Quality Attributes by decomposing the FCA result into specific comparison matrix. The comparison matrix have their values from the domain expert. Each matrix may forms a simple or complex matrix. If it is a complex matrix, the priorities result must be evaluated, as seen on table 1. In prominent, the complex matrix comprises of more than one quality attributes. The initial idea to address this problem is by creating similar value to the quality attributes element. After that, qualifying the other matrix that correspond to the same quality attributes, or straightforward gives the comparison value. If it is a simple matrix, the result of priorities may be used as alternatives for the software architect, straightforward, as



shown on table 2. The priorities signs the significant of component to address the quality attributes.

Table Legend:

a	Participant Management	e	Evaluation Resources
b	Course Manager	f	Report Generator
c	Resource Manager	g	Learning Assessment
d	Interaction Resources	h	Participant Mgmt
R	Reliability	U	Usability
S	Security	A	Availability

Figure 3. e-Learning Product Line Architecture Formal Concept Analysis Graph

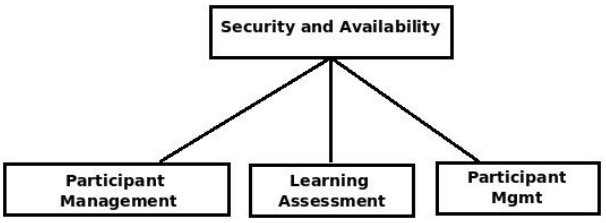


Figure 4. Security and Availability Hierarchy Structure

Based on the Formal Concept Analysis result, the groups of component are build into hierarchy structure. In the e-Learning Product Line case, the quality attribute may have specific or mixed specification, i.e. Security mixed with Availability. In this case, the mixed hierarchy structure can be seen in Figure 4.

In the mixed quality attributes, the software component do not have its specific quality attributes. As depicted in Figure 4, although one component may arrive as mixed quality attributes, on the other configuration it also act as other specific quality attributes. As seen in Figure 5. The quality attribute Usability also have component "Participant Mgmt" , that previously mention as the member of mixed quality attributes in the Security and Availability. Therefore, this component may have internal variability. To make it crystal clear, the component model is the reference. As seen in Figure 1, the component model have the component "Participant Mgmt", which have optional and closed as its internal variability. And, the clustering process in the Formal Concept Analysis have identified it as a component. However, when the Analytic Hierarchy Process did its job, the internal variability emerge as a decision. On the other hand the component "Participant Management" did not explicitly define its variability in the component model. Then, it must have something missing during component derivation from the feature model, and this problem resolve during architecture design process. Ultimately, this approach explicit the internal variability by reviewing the groups of component to correspond the quality attributes.

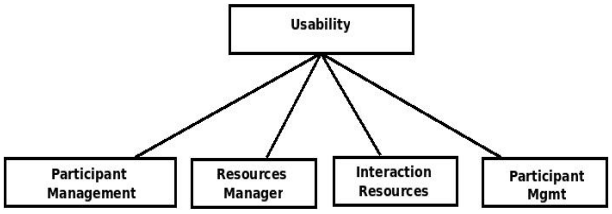


Figure 5. Usability Hierarchy Structure

Table 1. Pairwise comparison matrix for the alternatives of Security and Availability

	Participant Management	Learning Assessment	Participant Mgmt	Priorities
Participant Management	**1**	**5**	**3**	**0.658**
Learning Assessment	**1/5**	**1**	**1**	**0.156**
Participant Mgmt	**1/3**	**1**	**1**	**0.185**

In Table 1, the priorities of the component "Participant Management" is the highest to establish the security and availability, therefore, this component must be included in the architecture configuration. However, the internal variability of component "Participant Mgmt" need further investigation, whether to include "Open" or "Close" options. In this case, an experts' knowledge must be considered as a decision. Meanwhile, in the usability, the component "Interaction Resources" reach second priorities (0,246), and the "Resources Manager" on the next priorities (0,123), as illustrated in Table 2. It indicates that the usability should include both "Resources Manager" and "Interaction Resources" because of its relations to the external variability. As the "Interaction Resources" is the alternative component, then this comparison have sharpening the configuration process.

Table 2. Pairwise comparison matrix for the alternatives of Usability

	Participant Management	Resources Manager	Interaction Resources	Participant Mgmt	Priorities
Participant Management	**1**	**5**	**3**	**7**	**0.571**
Resources Manager	**1/5**	**1**	**1/3**	**3**	**0.123**
Interaction Resources	**1/3**	**3**	**1**	**3**	**0.246**
Participant Mgmt	**1/7**	**1/3**	**1/3**	**1**	**0.064**

3.3. The Architecture Design

The composition of components forms the quality attributes. Representing the component with internal variability should also taking it into account. In this approach, the internal variability explicit by grouping one or more components to single or mixed quality attributes. By clustering the software components, the functionality of the component with internal variability that forms the quality attributes can be selected clearly. To this, the quality attributes in the architecture design may have a stable architecture. This condition supports the software architecture to configure the specific architecture of member products of a product line without changing the architecture structure.

However, the selection of components to be included in an architecture of product line need to have clear information, of how the internal and external variability within component can be configured explicitly. The Formal Concept Analysis already grouped the components which correspond to specific or mixed quality attributes. In the specific quality attribute, a group of component can be pairwise compared to have its priorities using Analytical Hierarchy Approach. If in a group of component have mixed quality attributes. Then, the components have more than one quality attributes membership. To this, the component may have internal variability, or it may have different point of view when it combined to different components.

In this research goals, software architect defines its decision based on the information of the importance of components to the quality attributes. So, this approach will

not proposed the best component for the architecture design. Where, the final decision is on the software architect judgment.

4. Conclusion

In this paper, the hybrid Analytical Hierarchy Process – Formal Concept Analysis (HFA) to automate the decision of component composition in a Product Line Architecture already explained. The software architecture can be organized to have quality attributes without changing the functionalities of the components. The FCA has grouped the components to the specific or compound quality attributes, and the AHP measure the priorities of each group of components that correspond to one or more quality attributes. This approach have sharpen the decision of components by taking into account the quality attributes. In addition, the impact of a decision may be predicted and error design can be reduced.

In the future, formalizing the decision model is the next research objective.

Acknowledgement

This research was funded by Lembaga Penelitian dan Pengabdian Masyarakat Universitas Pelita Harapan (LPPM UPH) research grant under the project number P-002-FIK/IX/2011.

References

[1] K. Schmid, R.Rabiser, P. Grunbacher. Comparison of Decision Modeling Approaches in Product Lines. Proceeding Variability Modeling of Software Intensive Systems, ACM, 2011.

[2] C. Atkinson. Component-Based Product Line Development: The KobrA Approach. LNCS 1234, Springer, 2000.

[3] K. Kang. Issues in Component-Based Software Engineering. International Workshop on Component-Base Software Engineering, 21st ICSE, Los Angeles, 1999.

[4] C. Park, S. Hong, K. Son, J. Kwon. *A Component Model Supporting Decomposition and Composition*. Proceeding of SPLC, Kyoto, Japan, 2007.

[5] J. X. Mansell, D. Sellier. *Decision Model and Flexible Component Definition Based on XML Technology*. Proceeding of PFE, LNCS 3014, Springer, 2003.

[6] R. Mazo. *Using Constraint Programming to Verify DOPPLER Variability Model.* Proceeding of Variability Modeling of Software Intensive Systems, ACM, 2011.

[7] T. L. Saaty. Decision Making with AHP. *International Journal Services Sciences*. 2008.

[8] B.A. Davey. *Formal Concept Analysis: Introduction to Lattices and Orders*. Cambridge University Press, 2002.

[9] W. Rice. Moodle. Packt Publishing. 2006.

[10] K. Pohl, G. Bockle, F. van der Linden. Software Product Line Engineering: Foundation, Principles, and Techniques. Springer-Verlag, Berlin, 2005.

[11] I. M Murwantara. Initiating Layers Architecture Design for Product Line Architecture. Proceeding of URKE, Bali, 2011.

[12] P. J. Clemente, J. Hernández, J. M. Conejero, and G. Ortiz, "Managing crosscutting concerns in component based systems using a model driven development approach," *Journal of Systems and Software*, vol. 84, no. 6, pp. 1032-1053, Jun. 2011.

[13] T. M. Dao, H. Lee, and K. C. Kang, "Problem Frames-Based Approach to Achieving Quality Attributes in Software Product Line Engineering," *2011 15th International Software Product Line Conference*, pp. 175-180, Aug. 2011.

[14] F. Gilson and V. Englebert, "Towards Handling Architecture Design , Variability and Evolution with Model Transformations," *Architecture*, pp. 39-48, 2011.

[15] T. C. Harrison and a P. Campbell, "Attempting to Understand the Progress of Software Architecture Decision-Making on Large Australian Defence Projects," *2011 Ninth Working IEEE/IFIP Conference on Software Architecture*, pp. 42-45, Jun. 2011.

I Made Murwantara received the M.S. degrees in Computer Science from the University of Indonesia in 2002. During 2004-now, he stayed in Informatics Department, Faculty of Computer Science, Universitas Pelita Harapan. He now with Research and Development Computer Lab., Informatics Dept, Universitas Pelita Harapan, Indonesia.