Quality Attributes Decision Modeling for Software Product Line Architecture

I Made Murwantara
Research and Development Computer Laboratory
Informatics Department. Universitas Pelita Harapan
UPH Tower, Lippo Village, Karawaci, Tangerang, 15811 Indonesia, Phone: +62-21-5460901
e-mail: made.murwantara@uph.edu

Abstrak

Pemodelan keputusan untuk disain arsitektur pada arsitektur piranti lunak product line merupakan salah satu aktifitas yang sedang mengemuka saat ini. Pemodelan keputusan memberikan dukungan pada saat pemilihan komponen piranti lunak untuk memperoleh komposisi yang sesuai dengan kebutuhan. Meskipun demikian, hanya sedikit penelitian yang mengarah pada atribut kualitas. Untuk menjawab permasalahan ini, tulisan ini menghadirkan suatu metoda yang melakukan pencampuran atau hibrida antara Proses Analitik Hirarki dan Analisis Konsep Formal. Permasalahan utama pada pemodelan keputusan untuk arsitektur piranti lunak saat melakukan penilaian suatu atribut kualitas pada suatu konfigurasi arsitektur adalah mengukur dampak dari suatu atribut kualitas yang telah dibuat oleh suatu kelompok komponen piranti lunak. Pada tulisan ini, komposisi komponen piranti lunak yang berhubungan dengan atribut kualitas telah di analisa. Pada akhir tulisan ini, piranti lunak product line untuk e-Learning dihadirkan untuk memperlihatkan bagaimana pendekatan yang di usulkan dapat menyelesaikan permasalahan yang terjadi.

Kata kunci: *Piranti Lunak Product Line, Arsitektur Product Line, Pemodelan Keputusan, Proses Analitik Hirarki, Analisis Konsep Formal, Disain Arsitektur Piranti Lunak*

Abstract

Decision modeling is one of the most prominent activity for architecture design of a software product line architecture. The decision modeling supports the selection of suitable composition of software components from the architecture of member products of a product line. However, only little effort have been devoted to quality attributes. To address this shortcoming, this paper present a method that hybrid the Analytical Hierarchy Process and the Formal Concept Analysis. The key issue of decision modeling that assessing a quality attributes of an architecture configuration is to measure the impact of a quality attributes that made by the set of components. In this paper, we analyze the software components composition that corresponds to the quality attributes. An illustrative example based on the e-Learning software product line is presented to demonstrate of how the proposed approach works.

Keywords*: Software Product Line, Product Line Architecture, Decision Modeling, Analytical Hierarchy Process, Formal Concept Analysis, Software Architecture Design*

1. Introduction

Software Product Line (SPL) develops specific products in a low cost and high quality, that reuses all the member product assets, efficiently. SPLs improves productivity of software development by improving the reusability of all assets of the member products. During 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, 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, we need to decide whether one composition may change the component specification. If the groups of components are the representation of quality attributes, then, we need to make a decision model that able to include quality attributes as a decision value.

Quality Attributes affects system design. It may impact the whole or partial of the software architecture. In the product line architecture, quality attributes may emerge as the dependencies of components that address specific goals, such as security, reliability or usability. 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 we know the best composition that answer the requirement, and How to help the architect engineer to select the suitable alternatives. To address these shortcomings, we need to model the alternatives in a compact form. Further, the decision dependencies may emerge in a complex architecture, and we need to model this dependencies as well.

In this paper we investigate the decision modeling of software architecture design via hybrid Formal Concept Analysis - Analytical Hierarchy Process (HFA) for a product line

architecture. We aim to capture and evolve a SPL's assets so as to gain insight into architecture elements diversity, efficiently.

1.2. 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.

According to Mansell [5], the decision can be differentiated into restricted and unrestricted. 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. 3. 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 approach 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 capture 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

We have analyzed the decision modeling and its related works. After carefully reviewing the existing approaches, we see there are a chance to support decision model by presenting the quality attributes. In our knowledge, the quality attributes involve many components that cross-cut, which is uneasy to solve 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 which. 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.

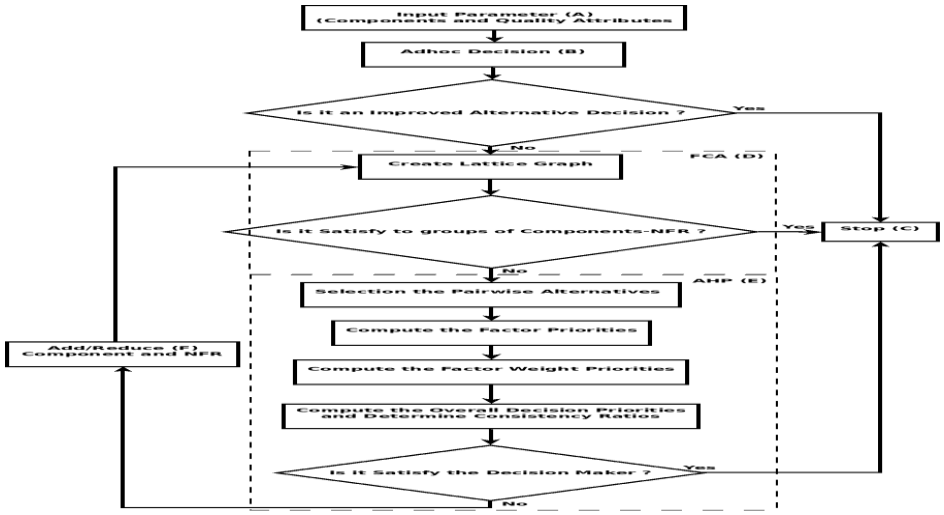


Figure 1. Hybrid Formal Concept Analysis – Analytical Hierarchy Process

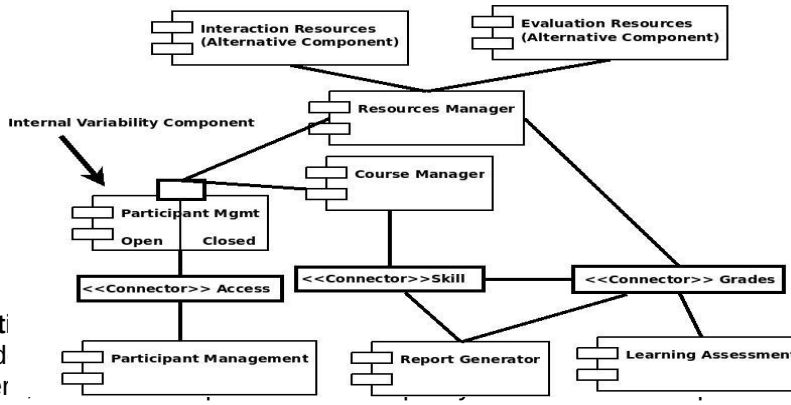
In our decision model approach, as depicted in Figure 1, the FCA analyzes the possibility of components cluster to answer the requirement question, and the AHP form the range of critical decision. The AHP Quality attributes (or Non-Functional Requirements (NFR)) affect the architecture design of a PLA, 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:

- Identify the relevant component variant to specific quality attributes.
- Predict the significance of each variable components that identified in quality attributes.
- Define the architecture design for quality attributes in component model.

We introduce 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 our study is that an initial parameter of components and quality attributes are created at the beginning, and used as an initial parameter input into FCA. (b) FCA will terminate if the alternative is satisfied (d), in terms of grouping the components against the quality attributes after the generated lattice graph (c), otherwise, (e) 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, (f) 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

We demonstrate HFA approach using a e-Learning Product Line Architecture (el-PLA) based on MOODLE [9], 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 our case, Security is handled by several components, which are component “Authorization Controller”, “User Account Manager”, “Key Generator”, and “Roles Access Manager”. In general, if we implement High Security into a system, then we will experience of low network speed, 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”. We found that many cross-cut components to the quality attributes which is difficult to configure during architecture design.



3.1. Identi

Id ature model [10]. Wher ure 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. 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 we have reliability, which is a quality attributes. From this perspective, functionalities relationship of components may be grouped as a quality attributes. The problems emerge when we try to separate the group of functionality within components. It may change the quality attributes, logically, and also change the functional structure of software architecture. However, we can measure the value of functionalities that impact into quality attributes, by qualifying groups of components.

The prominent problem in the decision model for a PLA, is to find the most suitable or matching components. Furthermore, it should answer the quality attributes by grouping the components, dynamically. As we have the FCA result, the quality attributes still have problems on how the group of components answer the quality attributes. To address this shortcomings, we hybrid the FCA and AHP. As shown in Figure 2, we hybridize 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.

There are still ambiguous on specific quality attributes decision as seen on Figure 2. To address, 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 we correspond to the component functionalities. We 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.

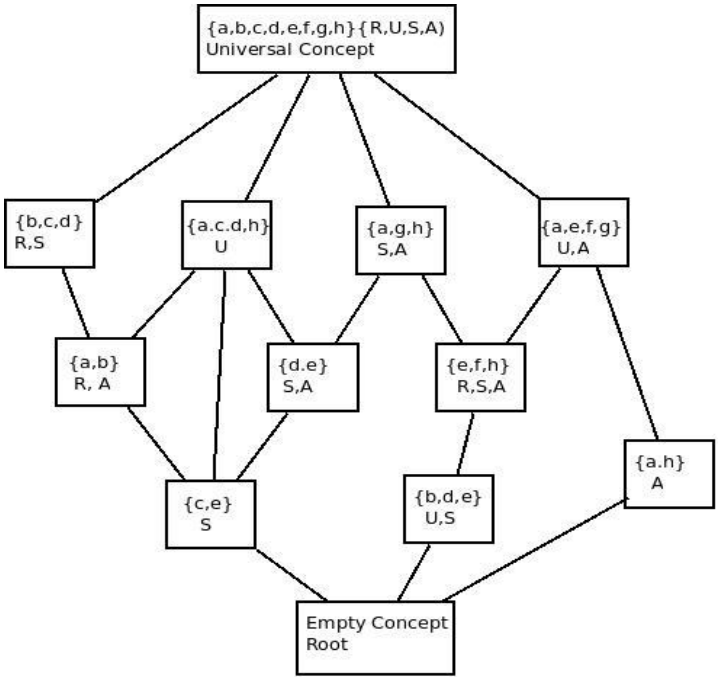


Table Legend:

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 2. el-PLA Formal Concept Analysis Graph

Table 1. Pairwise comparison matrix for the alternatives with respect to Security and Availability of e-Learning Participant

	Participant Management	**Learning Assessment**	**Participant Mgmt**	**Priorities**
Participant Management	1	7	9	0,794
Learning Assessment	1/7	1	3	0,124
Participant Mgmt	1/9	1/3	1	0,082

Table 2. Pairwise comparison matrix for the alternatives with respect to Usability

	Participant Management	**Resources Manager**	**Interaction Resources**	**Participant Mgmt**	**Priorities**
Participant Management	1	5	9	3	0,588
Resources Manager	1/5	1	7	5	0,204
Interaction Resources	1/9	1/7	1	1/9	0,048
Participant Mgmt	1/3	1/5	9	1	0,016

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 “Resource Manager” reach second priorities (0,204), and the “Interaction Resources” on the next priorities (0,048). It indicates that the usability should include both “Resources Manager” and “Interaction Resources” because of its relations to external variability. As the “Interaction Resources” is the alternative component, then this comparison have sharpening the configuration process.

3.3. The Architecture Design

The architecture design is a design that resolve the quality attributes of requirements. It builds the architecture that have to taking into account the composition of quality attributes into specific architecture which is configured from the artifacts of member products. In this approach, the software architecture should have explicit information of components as the alternative of choices, regardless of the component complex dependency. In addition, the architecture that is resulted from this process, will provides several advantages as follows; Firstly, the alternatives of variability can be predicted before it is choose. Secondly, the impact of alternatives guides the software architect of how one of the choices may leverage its architecture. Thirdly, Variabilities in the components of architecture can be clarify, in terms of component selection. And the last, the architecture can be designed to address the quality attributes efficiently, in regard to the internal and external variabilities of the software component.

4. Conclusion

In this paper, we already explained the hybrid AHP-FCA to automate the decision of component composition in a PLA. The software architecture can be organized to have quality attributes without changing the functionalities reside in the components. Although, the PLA dynamically configure, it won't change the architecture structure. 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, we would like to investigate the component complex dependencies that corresponds to Quality Attributes. And formalize the complex dependencies in the Product Line Architecture.

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 of Consumer Electronic Software Product Lines. 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; 1(1). 83-98
[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.