MODELLING QUALITY ATTRIBUTES IN FEATURE MODELS IN SOFTWARE PRODUCT LINE ENGINEERING

Guoheng Zhang, Huilin Ye and Yuqing Lin
School of Electrical Engineering and Computer Science, University of Newcastle, Callaghan 2308, NSW, Australia

Keywords: Quality attributes assessment, Non-functional requirement framework, Software product lines and analytic hierarchical process.

Abstract: In software product line engineering, product configuration is the process of selecting the desired features from a feature model based on customers' functional requirements. The quality attribute assessments for a configured product are neglected in most existing product configuration approaches. As we know, the key issue of assessing quality attributes for a configured product is to measure the interdependencies between functional features and quality attributes. To address this issue, we have proposed a quantitative-based approach to establish the interdependencies based on analytic hierarchical process (AHP) in our previous work. In this paper, we supplement our previous work from two aspects: first, we adapt non-functional requirement (NFR) framework to identify quality attributes for a software product line and extend current feature models to represent the identified quality attributes; second, we develop an evaluation method to check the consistency of domain experts' judgments to ensure the effectiveness of our approach. A simplified tourist guide software product line is used as an example to illustrate our approach.

1 INTRODUCTION

A software product line (SPL) enables to create a number of similar products by selecting and composing the reusable software artefacts (Pohl, 2005). The commonalities and variabilities of SPL members are identified during domain analysis and modeled as features in a feature model (White, 2007). A feature model is mostly represented as a feature tree where nodes represent features and edges represent the selection relationships among features. From a feature model, a specific member product can be derived by selecting the desired features based on customers' requirements and feature relationships specified in the feature model. However, quality attributes, such as performance, security and development cost, are usually handled until the target product is produced and tested in the system testing phase (Montagud, 2009). It is costly to fix the problems if we find that the produced product cannot meet the customers' requirements on quality attributes. Therefore, the quality attributes of a target product should be assessed in the earliest stage of product development in software product lines—product configuration in a feature model.

The key issue of assessing quality attributes for a configured product derived from a feature model is to measure the different impacts on the quality attribute imposed by different functional features. To address this issue, we have developed a quantitative-based approach which uses analytic hierarchical process to measure the interdependencies between a quality attribute and its correlated contributors, the functional features which have impact on the quality attribute (Zhang, 2010). Although the quality attributes of a configured product can be easily assessed based on the measured interdependencies, the completeness and effectiveness of our approach have not been fully achieved in our previous work. In this paper, we aim to supplement our previous work from two aspects: first, we will develop a systematic method of identifying and representing quality attributes that are critical for a software product line using an adapted non-functional requirements framework (Chung, 2000) to improve the completeness of our approach; second, we will develop an evaluation method of checking the correctness of domain experts' judgments to ensure the effectiveness of our approach.

The rest of this paper is organized as follows: Section 2 will introduce how to identify and

Zhang G., Ye H. and Lin Y..
MODELLING QUALITY ATTRIBUTES IN FEATURE MODELS IN SOFTWARE PRODUCT LINE ENGINEERING.
DOI: 10.5220/0003646602490254
In *Proceedings of the 6th International Conference on Software and Database Technologies* (ICSOFT-2011), pages 249-254
ISBN: 978-989-8425-77-5

represent quality attributes for a software product line; Section 3 will briefly review our previous work about how to measure the interdependencies between functional features and quality attributes. Section 4 will introduce an evaluation method to check the correctness of domain experts' judgments. Section 5 illustrates how to assess quality attributes for a configured product based on the measured interdependencies. We conclude this paper and identify the future work in Section 6.

2 IDENTIFYING AND REPRESENTING QUALITY ATTRIBUTES

In this section, we will propose a systematic approach of identifying the most critical quality attributes for the software systems in a software product line (SPL) and represent the identified quality attributes in feature models. An adapted non-functional requirement (NFR) framework (Chung, 2000) is used to identify quality attributes and current feature models are extended to represent the identified quality attributes. A tourist guide SPL is used as an example to illustrate our approach. Fig. 1 shows the feature model of the simplified tourist guide SPL. The semantics of feature relationships have been described in detail in our previous work (Zhang, 2010).

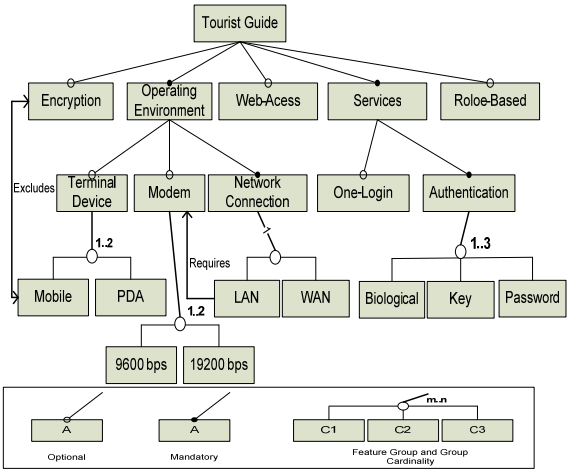

Figure 1: A feature model of tourist guide system SPL.

First, we identify the most critical quality attributes for software product line members by adapting NFR framework (Chung, 2000). It is difficult and time-consuming for domain experts to identify the quality attributes that are critical for software systems in software product line because of the elusive nature of quality attributes. The NFR template from NFR framework provides detailed classification and description for each specific quality attribute, such as usability, availability, reliability, performance, security, scalability, modifiability, and reusability. In our approach, the NFR template can be used as a checklist to identify the most critical quality attributes for a software product line.

The quality attributes identified from NFR template are always abstract. The degree of specificity of the identified abstract quality attributes, such as security and performance, would not permit non-functional requirements analysis and we need to refine the abstract quality attributes into more detailed quality attributes which have more semantics. NFR framework provides a sort catalogue for each abstract quality attribute based on the development knowledge taken from the literature and industrial experiences. A NFR sort catalogue summarizes the potentially set of concepts (sorts) of the quality attribute in a hierarchy and can serve as a rich set of alternatives to choose from as well as check-points to guard against omitting any important concerns in quality attribute decomposition. If a sub-quality attribute in the NFR sort catalogue is critical for one or more product line members, in will be included into the decomposition, otherwise it will be excluded from the decomposition.

Finally, we represent the abstract quality attributes and the refined sub-quality attributes in a feature model. Kang et al. defined a feature as "the prominent or distinctive user-visible aspect, quality or characteristic of a software system or systems" (Kang, 1998). Based on this definition, each quality attribute can be modeled as a feature. Therefore, we extend current feature models with a sub-feature tree to represent quality attributes. This sub-feature tree is named as quality attribute (*QA*) feature tree and its included features are named as quality attribute (*QA)* features. The root of the *QA* feature tree is the overall quality attribute which represents the overall "goodness" of the system. The second level of the *QA* feature tree is formed by the abstract quality attributes identified from NFR template. Typically, performance, security, usability and availability are the children of the root. Under each of these quality attributes are specific quality attribute refinements. The leaves of the *QA* feature tree are the detailed sub-quality attributes that are concrete enough for prioritization and analysis.

Following the above three steps, we identify three abstract quality attributes performance, low

cost and security for the tourist guide software product line, refine them into sub-quality attributes and extend the original feature model with a sub-quality attribute feature tree as illustrated in Fig. 2.

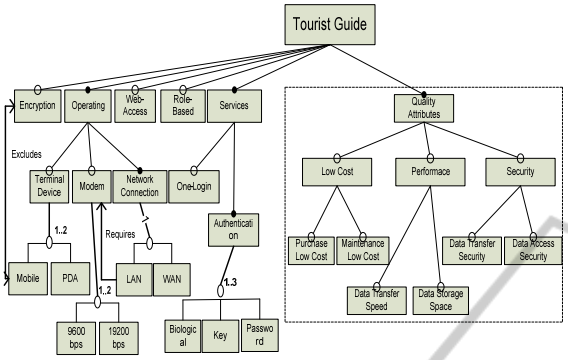

Figure 2: The extended feature model of tourist guide SPL.

3 MEASURING THE INTERDEPENDENCIES

In this section, we briefly review our previous approach of measuring the interdependencies between functional features and quality attributes based on analytic hierarchical process. The following steps need to be followed:

- identify the contributors, the functional features that contribute to a quality attribute (*QA*) feature
- prioritize the contributors of a *QA* feature based on their relative importance to the *QA* feature
- identify the relationships among functional features with respect to contributing to the *QA* feature
- calculate the overall impact of a configured product with respect to the *QA* feature

The identification of the contributors of a quality attribute is based on NFR framework (Chung, 2000) and domain experts' knowledge and experience. We have identified the contributors of "data transfer speed" (DTS) in tourist guide SPL as "Encryption", "PDA", "Mobile", "Modem19200", "Modem9600", "LAN", "WAN". Among these contributors, "Encryption" has negative impact on DTS and we transform the negative impact of "Encryption" on DTS when it is included into the product (selected status) to a positive impact on DTS when it is excluded from the product (deselected status). We use $\neg f$ to represent feature f in deselected status and define the set of contributors of *QA* as *RF (QA)*. Thus we have: *RF (DTS)* = {LAN, WAN, PDA, Mobile, Modem9600, Modem19200, ¬Encryption}.

In the second step, we adapt analytic hierarchical process (AHP) (Hallowell, 2007), to prioritize the identified contributors of *QA* based on their relative importance for satisfying *QA*. A comparison matrix is generated and the relative impact of individual contributors on *QA* is calculated using AHP tools. We use *Relative Importance Value* (*RIV*) to represent the calculated relative impact of each individual contributor on *QA*. The expression *RIV (QA, F)* is used to represent the *RIV* of feature *F* on *QA* if *F* has positive impact on *QA* in selected status and *RIV (QA, ¬F)* is used to represent the *RIV* of feature *F* on *QA* when *F* has positive impact on *QA* in deselected status. We have calculated the *RIVs* of the contributors in *RF (DTS)*, such as *RIV (DTS, ¬Encryption) =15.45* and *RIV (DTS, LAN) = 33.37*.

Once we have calculated the relative impact of individual contributors in *RF (QA*), we can calculate the overall impact on *QA* made by a set of contributors from *RF (QA)*. We define the overall impact as *Overall Importance Value* (*OIV*) and use *OIV (QA, fg)* to represent the *OIV* of a set of contributors *(fg)* which is a subset of *RF (QA)*. As the contributors often affect *QA* interdependently, we defined four types of feature groups in our approach: *SumGp*, *AvgGp*, *MaxGp* and *MinGp* to indicate the relationships among the contributors in *RF (QA)* with respect to affecting *QA*. The detailed definitions of the above feature groups can be found in our previous work (Zhang, 2010). In the *RF (DTS)* we have identified four feature groups: {¬Encryption} is a *SumGp*; {Modem19200, Modem9600} and {PDA, Mobile} are two *AvgGp*; and {LAN, WAN} is a *MinGp*.

A configured product may include a subset of *RF (QA)* for satisficing *QA* and we define the set of features which contribute to *QA* in a configured product as a *valid selection* (*VS*) with respect to *QA* and represented as *VS (QA)*. Then the overall impact of a configured product on *QA* can be considered as the overall importance value of its included *VS (QA)*. The calculation of *OIV (QA, VS (QA))* is based on definitions of different feature groups and the detailed calculation process can be found in (Zhang, 2010). The *OIV* is not intuitive for stakeholders to understand. It must be compared with *OIVs* of all other valid selections with respect to *QA* to represent its relative *QA* level. We define the normalized overall importance value (*NOIV*) which is in [0…1] scale to represent the relative *QA* level of a valid selection compared with other valid selections. The

detailed normalization method can be found in (Zhang, 2010). Following our approach, for a valid selection of DTS in the tourist guide SPL: {LAN, Modem 9600, ¬Encryption}, its *OIV* is 55.70 and its *NOIV* is 0.89 which represents a relative high level of DTS.

4 CHECKING DOMAIN EXPERTS' JUDGMENTS

Domain experts' judgments are needed in our AHP-based approach. If there are errors in domain experts' judgments, the relative importance values of individual contributors of a quality attribute will be incorrect. Based on the incorrect *RIVs*, the predicted quality attributes for a configured product will be wrong. Therefore, one critical issue of our approach is to ensure the correctness of domain experts' judgments.

In our AHP-based approach, domain experts need to make judgments on $n\ (n-1)/2$ pair-wise comparisons for n contributors of quality attribute QA, X_1 , X_2 ,...X_n based on their contribution to QA. The results of pair-wise comparisons are written into a comparison matrix as shown in table 1 where C_{ij} represents the importance intensity value of comparing X_i with X_j. A priority vector (PV) which consists of the relative importance values of all the contributors of QA involved in the comparison can be calculated from the comparison matrix. Then we use the relative importance values in PV to measure the interdependencies between QA and its contributors. Based on the interdependencies, we can assess the QA level for any configured product.

Table 1: A comparison matrix made by domain experts.

	X_1	X_2	···	X_{n-1}	X_n
X_1	1	C_{12}	⋯	$C_{1(n-1)}$	C_{1n}
X_2		1	⋯	$C_{2(n-1)}$	C_{2n}
…			1	…	…
X_{n-1}				1	$C_{(n-1)n}$
X_n					1

When making pair-wise comparisons in AHP method, domain experts may make two kinds of errors. Firstly, a domain expert may make inconsistent pair-wise comparisons for the contributors of QA. For example, in the comparison matrix of table 1, if $(C_{ij}>0) \wedge (C_{jk}>0) \wedge (C_{ik}<0)$, there will be conflicts among these three comparisons, because we can deduce that C_i is more important than C_k as $C_{ij}>0$ illustrates that C_i is more important than C_j and $C_{jk}>0$ illustrates that C_j is more important than C_k. However, this deduction conflicts with $C_{ik}<0$ which means C_i is less important than C_k. In this case, we say that the domain expert makes inconsistent pair-wise comparisons.

Inconsistencies in one domain expert's judgments can be identified by checking the consistency ratio (CR) of the comparison matrix. AHP allows small inconsistency in judgements because human is not always consistent. A CR below *0.1* is acceptable (Hallowell, 2007). We also adapt *0.1* as a borderline to check whether all the pair-wise comparisons made by one domain expert are consistent. The calculation of CR is supported by most AHP tools. In our approach, if CR of a comparison matrix is above *0.1*, the domain expert needs to identify the inconsistent pair-wise comparisons and modify the comparison matrix until the CR is below *0.1*.

The second kind of errors that a domain expert may make is the biased judgments. For example, a feature X has significant contribution to a quality attribute QA in one domain expert's opinion. However, X is not that important to QA in reality. The *RIV (QA, X)* calculated based on the domain expert's judgment must be higher than its real value. To avoid the biased judgments made by one domain expert, we adapt two domain experts in our approach and measure the consistency between two domain experts' judgments. Assume that two domain experts generate two comparison matrixes: matrix □ and matrix □ for the feature set $X_1, X_2...X_n$ and calculate two priority vectors PV_1 and PV_2 respectively. The $NOIV_1\ (QA, VS_i)$ illustrates the *NOIV* of valid selection VS_i based on PV_1 while the $NOIV_2\ (QA, VS_i)$ illustrates the *NOIV* of valid selection VS_i based on PV_2. Then we use the following formula (1) to measure the consistency between two domain experts' judgments.

$$AD = \sqrt{\frac{1}{n}\sum^{VS_S}(NOIV_1(QA,VS_i) - NOIV_2(QA,VS_i))^2} \quad (1)$$

The above formula represents the *average difference* (AD) between the assessed QA levels based on PV_1 and assessed QA levels based on PV_2 for all valid selections VS_s from *RF (QA)*. The minimum value of AD is 0.0 when PV_1 and PV_2 are completely same while the maximum value of AD is *1.0* when PV_1 and PV_2 are completely contrary. A smaller AD can represent higher consistency

between two priority vectors and further illustrate higher consistency between two domain experts' judgments. Using our approach to assess quality attributes, if the expression $|NOIV\ (QA, VS_i)\text{-}NOIV\ (QA, VS_j)|<0.1$, we consider that VS_i and VS_j have the same QA level. Then we can deduct that if AD is less than *0.1*, we consider that the assessed quality attribute level based on PV_1 and the assessed quality attribute level based on PV_2 are the same. Therefore, the borderline of AD is 0.1. If the AD is higher than *0.1*, the domain experts need to find the biased judgments and modify the comparison matrix until AD is less than *0.1*.

5 ASSESS QUALITY ATTRIBUTES

The interdependencies between qualities attribute features and functional features are very complex. We have design an xml-based representation schema in (Zhang, 2010) to represent the interdependencies between a *QA* feature and its contributors. Fig. 3 shows the representation schema of "data transfer speed" in the tourist guide SPL. The semantics of the elements and attributes in the representation schema are described as follows:

- Element <QA> represents a quality attribute feature *QA*.
- Element <Relevant Features> represent all the contributors of *QA* in *RF (QA)*.
- Element <Feature> under element <Relevant Features> represents a contributor of *QA*. Its attribute "name" illustrates the contributor name and attribute "RIV" illustrates the relative importance value of the contributor on *QA*.
- Element <Feature Groups> represents all the feature groups in *RF (QA)*.
- Element <FG> under element <Feature Groups> represents a feature group. Its attribute "type" illustrates the type of feature group and attribute "included features" illustrates all the features in the feature group.
- Element <VS_OIV> represents the overall importance value of valid selections from *RF (QA)*. Its attribute "max_oiv" illustrate the maximum *OIV* and attribute "min_oiv" illustrates the minimum *OIV* among all the valid selections.

Once we have the representation schema of the measured interdependency between a quality attribute feature and its contributors, we can assess the level of quality attribute for any configured product. A configured product can be represented as a 2-tuple of the form (S, R) where S is the set of features to be included and R is the set of features to be removed. For a specific configured product $PC = (S, R)$, to assess its level on quality attribute QA, a set of steps need to be followed based on the representation schema of quality attributes.

```
<QA name="Data Transfer Speed">
  <Relevant Features>
      <Feature name="Modem 19200", RIV="9.87" ></Feature>
      <Feature name="Modem 9600", RIV="6.88" ></Feature>
      <Feature name="PDA", RIV="3.1" ></Feature>
      <Feature name="Mobile", RIV="2.76" ></Feature>
      <Feature name="LAN", RIV="33.37" ></Feature>
      <Feature name="WAN", RIV="28.56" ></Feature>
      <Feature name="¬Encryption", RIV="15.45" ></Feature>
  </Relevant Features>
  <Feature Groups>
      <FG type="SumGp", included features="¬Encryption "></FG>
      <FG type="AvgGp", included features="Modem 19200","Modem 9600"></FG>
      <FG type="AvgGp", included features="PDA", "Mobile"></FG>
      <FG type="MinGp", included features="LAN","WAN"></FG>
  </Feature Groups>
  <VS_OIV max_oiv="58.69", min_oiv="31.66"></VS_OIV>
</QA>
```

Figure 3: The representation schema for DTS.

- ***Identify Valid Selection.*** The first step is to identify the valid selection *VS* with respect to *QA* from the configured product *PC* using the formula $VS = (RF\ (QA) \cap S) \cup (RF\ (QA) \cap R)$. The feature set *RF (QA)* and the relative importance value of each feature can be found from the attribute "name" and "RIV" of element "Feature" in the representation schema.
- ***Calculate OIV for Valid Selection.*** The second step is to calculate the overall importance value of the identified valid selection *VS* on quality attribute *QA* using the formula $OIV\ (QA, VS) = Sum\ (OIV\ (QA, fg_i \cap VS))$ where *fgi* illustrates one of the feature groups in *RF (QA)*. The feature groups can be found from the element "Feature Groups". The included features of a feature group fg_i and its type can be found from the attribute "included features" and "type" of the element "FG" in the representation schema. To calculate $OIV\ (QA, fg_i \cap VS)$, we assume that $vfg_i = fg_i \cap VS$ and use the following formulas to calculate the *OIVs*.
  $OIV\ (QA, vfg_i) = \text{Sum}\ (RIV\ (QA, f_j) \mid fj \in vfg_i \wedge vfg_i \subseteq fg_i)$, if fg_i is a *SumGp*.
  $OIV\ (QA, vfg_i) = \text{Average}\ (RIV\ (QA, f_j) \mid fj \in vfg_i \wedge vfg_i \subseteq fg_i)$, if fg_i is an *AvgGp*.
  $OIV\ (QA, vfg_i) = \text{Maximum}\ (RIV\ (QA, f_j) \mid fj \in vfg_i \wedge vfg_i \subseteq fg_i)$, if fg_i is a *MaxGp*.

OIV (QA, vfg_i) = Minimum (*RIV (QA, f_j)* | $fj \in vfg_i \wedge vfg_i \subseteq fg_i$), if fg_i is a *MinGp*.

- ***Normalize OIV into NOIV.*** The third step is to calculate the normalized overall importance value (*NOIV*) of the valid selection *VS* with respect to quality attribute *QA* using the following formula (2). The expression *MIN* (*OIV* (VS_i)) can be found from the attribute "min_oiv" of the element "VS_OIV" while the expression *MAX* (*OIV* (VS_i)) can be found from the attribute "max_oiv" of the element "*VS_OIV*" in the representation schema.

$$NOIV(QA,VS) = \frac{OIV(VS) - MIN(OIV(VS_i))}{MAX(OIV(VS_i)) - MIN(OIV(VS_i))} \quad (2)$$

Following the above three steps, we can assess the level of *QA* for any configured product based on the representation schema of *QA*. Assume that we have a configured product: *PC* = ({network connection, tourist guide, operating environment, service, position detection, satellite, WAN, terminal device, Mobile, PDA}, {LAN, Encryption, authentication, modem, modem19200, modem 9600}). The valid selection *VS* with respect to DTS can be identified from *PC* as {WAN, Mobile, PDA, ¬Encryption}. Then we divide *VS* into four groups: vfg_1, vfg_2, vfg_3, vfg_4 where $vfg_i = fg_i \cap VS$. Based on the formulas in step two, we can calculate the *OIV* of each group as well as the *OIV* of *VS* as 46.94. In the final step, we normalize the calculated overall importance value into *NOIV* using the formula in step three as: *NOIV* (*DTS*, *VS*) = (*46.94-31.66*) / (*58.69-31.66*) = *0.56* which represents a relative medium DTS level.

6 CONCLUSIONS AND FUTURE WORK

In this paper, we improve our previous work (Zhang, 2010) from two aspects: first, we improve the completeness of our previous work by identifying and representing the quality attributes of a software product line using an adapted non-functional requirement framework; second, we improve the effectiveness of our previous work by developing a method to check the correctness of domain experts' judgments. After these two supplements, our approach provides more efficient and precise quality attribute assessments.

- The assessment is more efficient as we can easily predict or assess the impact on a quality attribute made by any combination of its contributors without involving human effort to assess the combinations one by one.
- The assessment is more precise than existing approaches as domain experts can provide more precise judgments in the pair-wise comparisons of AHP method. The quality attribute level for a configured product which is calculated based on the pair-wise comparison results is more precise than other approaches.

One limitation of our approach is that we cannot identify the relationships between conflicting quality attributes. During product configuration, quality attributes can never be achieved in isolation. The achievement of any one will have impact, sometimes positive and sometimes negative, on the achievement of others. The relationships between conflicting quality attributes play an important role when we aim to derive a product with desired quality attributes. In the future, we aim to understand the relationships between conflicting quality attributes and concentrate on how to identify the conflicting quality attributes and how to measure their relationships.

REFERENCES

Klaus Pohl, Gunter Bockle and Frank van der Linden., 2005. Software Product Line Engineering: Foundations, Principles, and Techniques. *Springer*

Jules White and Douglas C. Schmidt, 2007. Automating Product-Line Variant Selection for Mobile Devices. In *the 11th International Software Product Line Conference*.

Sonia Montagud and Silvia Abrahao, 2009. Gathering Current Knowledge about Quality Evaluation in Software Product Lines. In *Software Product Line Conference 2009*.

Guoheng Zhang, Huilin Ye and Yuqing Lin, 2010. Quality Attribute Assessment for Feature-Based Product Configuration in Software Product Line, in Asian Pacific Software Engineering Conference, Sydney, Australia.

Lawrence Chung, Brian A Nixon, Eric Yu, John Mylopoulos, 2000. *Non-Functional Requirements in Software Engineering*. Kluwer Academic.

Kyo Chul Kang, G.C.S., J.A.Hess, W.E.Novak and A.S.Petersem, 1990. Feature-Oriented Domain Analysis (FODA) Feasibility Study, in *Technical Report CMU/SEI 90-TR-21*.

David L. Hallowell, 2007. Analytical Hierarchical Process (AHP)-Getting Oriented. ISixSigma.com Retrieved 2007-08-21.