특성 지향의 제품계열분석 모델의 정형적 정의와 일관성 분석

(Formal Definition and Consistency Analysis of Feature-Oriented Product Line Analysis Model)

이 관 우 †

(Kwanwoo Lee)

요 약 제품계열분석(product line analysis)은 제품계열자산(product line asset)을 개발하기에 앞서, 제품계열 내에 속한 제품들의 다양한 요구사항과 이들 간의 관계 및 제약사항을 분석하는 활동을 말한다. 지금까지 특성모델링(feature modeling)이라 불리는 특성 지향의 공통성과 가변성 분석은 제품계열분석의 핵심적인 부분으로 간주되어 왔다. 비록 공통성과 가변성 분석이 제품계열분석의 핵심적인 요소이지만, 이것만으로는 재사용가능하고 적응성이 뛰어난 제품계열자산(예, 아키텍처와 컴포넌트) 개발에 한계가 있다. 특성 간의 의존성 및 특성결합시간도 제품계열자산 개발에 중대한 영향을 미치는 요소이다. 따라서 본 논문에서는 기존에 공통성과 가변성 관점에서 제품계열을 분석한 결과인 특성모델(feature model)을 세 가지의 특성 측면(즉, 제품특성의 공통성과 가변성, 특성간의 의존성, 그리고 특성결합시간)으로 확장한 특성 지향의 제품계열분석 모델을 제안한다. 특히, 세 가지 측면의 일관성을 검증하기 위해서, 특성 지향의 제품계열분석 모델을 정형적으로 정의하고, 모델의 일관성을 검사하는 규칙을 제공한다.

키워드 : 소프트웨어 제품계열 공학, 제품계열분석 모델, 특성 모델, 일관성 분석, 재사용

Abstract Product line analysis is an activity for analyzing requirements, their relationships, and constraints in a product line before engineering product line assets (e.g., architectures and components). A feature-oriented commonality and variability analysis (called feature modeling) has been considered an essential part of product line analysis. Commonality and variability analysis, although critical, is not sufficient to develop reusable and adaptable product line assets. Dependencies among features and feature binding time also have significant influences on the design of product line assets. In this paper, we propose a feature-oriented product line analysis model that extends the existing feature model in terms of three aspects (i.e., feature commonality and variability, feature dependency, and feature binding time). To validate the consistency among the three aspects we formally define the feature-oriented product line analysis model and provide rules for checking consistency.

Key words : product line engineering, product line analysis model, feature model, consistency analysis, software reuse

1. 서 론

제품계열공학(product line engineering)은 유사한 기능을 지닌 소프트웨어 제품들의 개발에 있어 생산성(productivity) 및 적시성(time-to-market)의 향상을 위해서 최근에 제안된 소프트웨어 재사용 패러다임이다[1]. 제품계열공학의 핵심은 동일 계열의 제품들 간에 재사용될 수 있는 제품계열자산(예, 아키텍처 및 컴포넌트)을 미리 개발한 후에, 이를 체계적이고 계획적으로 재사용함으로써 다양한 고객의 요구에 맞는 제품을 저비용으로 적기에 생산하는 데 있다.

제품계열공학은 크게 제품계열분석, 제품계열자산개발, 제품생산의 단계로 구성된다. 제품계열분석은 제품계열 내의 제품들의 다양한 요구사항과 이들 간의 관계 및 제약사항을 분석하는 활동을 말하고, 제품계열자산개발은 분석된 요구사항 및 제약사항을 만족시키면서 제

· 본 연구는 2003년도 한성대학교 교내연구비 지원과제 임
† 정 회 원 : 한성대학교 정보시스템공학과 교수
kwlee@hansung.ac.kr
논문접수 : 2004년 6월 26일
심사완료 : 2004년 12월 3일

품들의 생산에 재사용될 수 있는 아키텍처 및 컴포넌트를 개발하는 활동이다. 마지막으로 제품생산 활동은 미리 개발된 제품계열자산을 재사용하여 고객의 요구사항을 만족시키는 제품을 체계적으로 생산하는 것을 의미한다.

제품계열분석 활동은 성공적인 제품계열공학을 위한 중요한 첫 단계로서, 분석 결과는 재사용 가능한 제품계열자산 개발에 중요한 입력요소로서 활용될 뿐만 아니라, 제품계열자산을 이용하여 다양한 제품을 구성(configuration)하는데 핵심적인 역할을 한다. 지금까지 많은 제품계열공학 방법[1-4]에서는 제품 간의 공통성과 가변성을 분석하기 위해 특성 지향 분석 방법(feature-oriented analysis method)인 특성모델링(feature modeling)[2,5,6]을 널리 사용해 오고 있다. 이는 특성모델링이 제품 간의 공통성과 가변성을 찾아내고 구조화시키는데 효과적인 방법을 제공해 줄 뿐만 아니라, 분석결과인 특성모델(feature model)이 동일 계열 내의 여러 제품 개발에 재사용될 수 있는 자산 개발을 위해 중요한 정보를 제공해 주기 때문이다.

하지만, 공통성과 가변성 정보만으로는 재사용성과 적응성이 뛰어난 제품계열자산 개발에 한계가 있다. 공통성과 가변성의 단위인 제품특성(product feature)간의 의존성도 제품계열자산 개발을 위해 중요한 정보를 제공한다. 일반적으로 제품특성들은 상호 독립적이기 보다는 다른 특성들과 다양한 의존관계를 가진다. 가령, 제품특성 f1과 f2가 특정한 가변특성 f3에 의존적이고 이들 간의 의존관계가 f1과 f2를 구현한 컴포넌트들에 녹아져있다면, 가변특성인 f3의 선택여부는 f1과 f2를 구현한 모든 컴포넌트들의 변화를 야기 시키게 된다. 이는 하나의 가변특성의 선택여부가 그 특성을 구현한 컴포넌트에만 영향을 미치지 않고 다른 특성을 구현한 컴포넌트에 영향을 주게 되므로, 제품계열자산의 재사용성 및 적응성이 떨어지게 된다. 따라서 재사용성과 적응성이 뛰어난 제품계열자산을 개발하기 위해서는 제품계열의 공통성과 가변성뿐만 아니라 제품특성간의 의존성을 철저히 분석하고, 이러한 분석결과를 바탕으로 가변특성의 존재여부에 따른 제품계열자산의 변화가 지역화 될 수 있도록 제품계열자산을 개발해야 한다.

뿐만 아니라, 가변특성이 언제 선택되어 특정 제품에 결합되는 가를 나타내는 특성결합시간(feature binding time)도 제품계열자산 개발에 중요한 영향을 미친다. 가령, 어떤 가변특성은 제품 생산 시에 특정 제품에 결합될 수도 있고, 다른 가변특성은 제품 생산 후인 운영시간에 특정 제품에 결합되어 사용가능하게 될 수도 있다. 따라서 제품 생산 시에 결합되는 가변특성은 이를 구현한 컴포넌트가 제품 생산 시에 쉽게 포함되거나 제거될 수 있는 방식(예, 매크로 프로세싱)으로 구현되어야 하고, 운영 중에 선택되어 특정 제품에 결합되어야 하는 가변특성은 동적으로 제품에 결합될 수 있는 방식(예, 플러그인)으로 구현되어야 한다. 즉, 특성결합시간도 제품계열자산 개발에 중대한 영향을 미치는 요소이다.

본 논문에서는 공통성과 가변성 관점에서만 제품계열을 분석한 기존의 특성모델[2,5,6]을 확장하여, 특성 간의 의존성 및 특성결합시간 관점의 분석을 추가한 특성 지향의 제품계열분석 모델을 제안한다. 특히, 세 가지 관점에서 분석된 모델의 일관성을 분석하기 위해 본 논문에서는 제품계열분석 모델을 정형적으로 정의하고, 각 구성 요소간의 일관성을 정의하는 규칙을 제안한다. 이러한 일관성 규칙을 통해 제품계열분석 모델의 유효성 검사를 수행할 수 있다. 유효성이 검증된 제품계열분석 모델은 제품계열자산 개발의 핵심적인 입력 역할을 수행한다.

본 논문의 구성은 다음과 같다. 2장에서는 본 논문에서 확장하고자 하는 기존의 특성모델에 대해 간단히 요약하고, 이로부터 확장된 제품계열분석 모델은 3장에서 정형적으로 정의된다. 4장에서는 제품계열분석 모델의 일관성을 검사하기 위한 규칙을 제안하고, 5장에서는 기존 연구와 본 연구와의 차이에 대해서 비교분석을 한다. 끝으로, 6장에서는 본 연구에 대한 토의와 향후 연구 과제에 대해 언급하면서 결론을 내린다.

2. 특성 모델의 요약

특성모델은 많은 제품계열 방법에서 분석 모델의 한 부분으로 사용하고 있는 모델로서, 동일 계열 내의 여러 제품 간의 공통성과 가변성을 제품특성 단위로 찾아내고, 이들 특성을 구조적인 관계를 통해 비순환적 그래프(acyclic graph) 형태로 구조화시킨 모델이다.

특성모델의 구조적인 측면은 집합화(aggregation)와 일반화(generalization)의 두 가지 관계로 표현된다. 특성 간의 집합화 관계는 하나의 특성이 여러 부분 특성으로 분해되거나, 여러 특성들이 하나의 집합 특성으로 묶일 수 있을 때 표현되고, 특성 간의 일반화 관계는 하나의 특성이 좀 더 구체화된 특성으로 상세화되거나, 특성들이 좀 더 포괄적인 특성으로 일반화 때 나타난다. 가령, 자동차는 *변속기, 에어콘, 엔진* 등의 세부 특성으로 분해 될 수 있고, *변속기*는 변속하는 방식에 따라 *수동변속기* 혹은 *자동변속기* 등으로 좀 더 상세화 될 수 있다.

이러한 두 가지 구조적 관계를 통해 특성모델에 구조화된 특성은 크게 공통특성과 가변특성의 두 가지 형태로 분류된다. 공통특성은 제품계열 내의 모든 제품에 공통으로 존재하는 제품특성을 의미하고, 가변특성은 제품

계열 내의 특정 제품 개발을 위해 선택될 수 있는 제품 특성을 의미한다. 이러한 가변특성들은 선택되는 형태에 따라 다시 택일특성, 다중특성, 그리고 선택특성으로 분류된다. 택일특성(alternative feature)은 집합 내의 한 특성만 한 제품에 포함될 수 있는 특성들의 집합을 의미하고, 다중특성(multiple feature)은 집합 내의 특성 중 적어도 하나 이상의 특성이 한 제품에 포함될 수 있는 특성들의 집합을 의미한다. 마지막으로 선택특성(optional feature)은 특정 제품에 포함될 수도 있고 포함되지 않을 수도 있는 특성들을 의미한다.

이와 더불어, 특성 모델에서는 가변특성의 선택을 제한하는 두 가지 구성규칙(configuration rule)을 제공한다. 요구구성(required configuration)은 한 특성이 선택되면 다른 특성도 반드시 선택되어야 할 때 정의되며, 배타적구성(excluded configuration)은 두 특성이 하나의 제품을 위해 동시에 선택될 수 없을 때 정의된다.

이와 같이 공통성과 가변성 관점에서 제품특성을 찾아내고 구조화한 특성모델은 제품계열의 분석모델로서 중요한 역할을 차지하지만, 이러한 정보만으로는 재사용 가능한 제품계열자산 개발에 한계가 있다. 다음 장에서는 앞서 설명한 제품특성 간의 의존성 및 특성결합시간을 포함하여 기존 특성모델을 확장한 특성 지향의 제품계열분석 모델을 정의한다.

3. 특성 지향의 제품계열분석 모델

Clements와 Northrop[1]은 제품계열을 "특정한 시장의 요구를 만족시키고 유사한 제품특성을 공유하면서, 공통의 제품계열자산으로부터 계획적으로 산출될 수 있는 제품들의 집합"이라고 정의하고 있다. 여기서 제품특성은 제품의 두드러진 혹은 제품 간의 구별되는 특징 및 속성을 의미하며, 소프트웨어 제품을 바라보는 사람의 관점에 따라, 다양한 추상화 수준으로 표현될 수 있다. 예를 들면, 제품의 사용자는 제품의 기능적인 특성(예, 서비스)과 비 기능적인 특성(예, 성능) 관점에서 제품특성을 찾아내고, 개발자는 제품의 구현 기술(예, 알고리즘) 관점에서 제품특성을 정의할 것이다. 본 논문에서는 제품계열분석 모델을 정의하기 위해서 사용자 관점에서 바라볼 수 있는 특성(즉, 기능 혹은 비 기능)에 초점을 두고 특성을 다음과 같이 정의한다.

정의 1. 특성F = (*name, type, spec*)은 다음과 같다.

- *name*: 특성의 고유한 이름이다.
- *type*: *기능* 혹은 *비 기능* 중의 하나로 정의되는 특성의 타입이다. *기능* 타입의 특성은 제품이 무엇을 해야 하는 지를 나타내고, *비 기능* 타입의 특성은 정량적 혹은 정성적으로 측정 가능한 제품의 속성을 의미한다.
- *spec*은 특성의 의미적 명세이다.

제품계열분석 모델이란 제품계열 내의 제품들을 사용자 요구(제품특성) 관점에서 분석한 결과를 모델화 한 것으로, 본 논문에서는 앞서 말한 제품특성의 공통성과 가변성, 제품특성 간의 의존성, 특성결합시간의 측면에서 제품계열분석 모델을 정의한다.

정의 2. 제품계열분석 모델$PLAM$ = (FS,CR,R_1,R_2, R_3,R_4,R_5,R_6,R_7,R_8, FBT)은 다음과 같다.

- FS: 제품계열을 구성하는 제품들의 특성 집합으로 공통특성 집합(F_C)와 가변특성 집합(F_V) 으로 나뉘어 진다(FS=F_C∪F_V).
- CR: F_V 내의 특성의 선택을 제한하는 *필수선택*, *배타선택*, *택일선택*, *다중선택*의 구성규칙들의 집합이다.
- R_1, R_2, R_3, R_4, R_5, R_6, R_7, R_8: FS에 포함된 특성 간의 관계로서, 각각은 집합(aggregation), 일반화(generalization), 사용(usage), 변경(modification), 배타적 활성화(exclusive-activation), 종속적 활성화(subordinate-activation), 병렬적 활성화(concurrent-activation)와 순차적 활성화(sequential-activation)를 나타낸다.
- FBT: (f, bt)의 집합으로, f는 FS 내의 특성을 나타내고, bt는 특성이 제품에 결합 되는 시간을 나타내는 것으로 *PDT(제품 개발 시간), PIT(제품 설치 시간), PRT(제품 운영 시간)* 중 하나의 값을 가진다.

제품계열분석 모델에서는 공통특성 집합(F_C), 가변특성 집합(F_V), 그리고 구성규칙을 사용하여 제품계열의 공통성과 가변성에 대한 정보를 정의한다. 여기서 공통특성은 제품계열을 구성하는 모든 제품들 간의 공통성을 의미하고, 가변특성은 제품 간의 구별되는 특징인 가변성을 의미한다. 가변특성은 특정한 제품을 위해 선택될 수 있는 특성으로서, 하나의 가변특성 선택은 다른 가변특성의 선택에 영향을 줄 수 있다. 따라서 제품계열분석 모델에서는 가변특성의 선택 간에 존재하는 의존성을 다음 네 가지의 구성규칙(configuration rule)으로 정의한다(여기서, $S(f)$는 제품특성 f가 특정한 제품을 위해 선택 되었으면 참이고, 그렇지 않으면 거짓인 것을 의미한다).

- (필수선택) $Req(f_1, f_2) \Leftrightarrow S(f_1) \Rightarrow S(f_2)$
  만약 특성 f_1이 특정한 제품을 위해 선택된다면, 특성 f_2도 반드시 선택되어야 한다.
- (배타선택) $Ex(f_1, f_2) \Leftrightarrow \neg (S(f_1) \wedge S(f_2))$
  두 특성 f_1과 f_2는 하나의 제품을 위해 동시에 선택되어서는 안 된다.
- (택일선택) $XOR(f, Z) \Leftrightarrow (S(f) \wedge (\exists z \in Z \cdot S(z) \wedge (\forall x$

$\in Z-\{z\} \cdot \neg S(x))) \vee (\neg S(f) \wedge \forall x \in Z \cdot \neg S(x))$

특성 f가 특정한 제품을 위해 선택된다면, $Z \subseteq F_V$내의 특성들 중에 단 하나의 특성만 선택되어야 하고, 특성 f가 선택되지 않는 다면 집합 Z내의 어떠한 특성도 선택되어서는 안 된다.

- (다중선택) $OR(f, Z) \Leftrightarrow (S(f) \wedge \exists z \in Z \cdot S(z)) \vee (\neg S(f) \wedge \forall x \in Z \cdot \neg S(x))$

특성 f가 특정한 제품을 위해 선택된다면, $Z \subseteq F_V$내의 특성들 중에 적어도 하나의 특성이 반드시 선택되어야 하고, 특성 f가 선택되지 않는 다면 집합 Z내의 어떠한 특성도 선택되어서는 안 된다.

이와 같은 공통성과 가변성에 대한 정보뿐만 아니라, 제품특성 간의 다양한 관계를 표현하기 위해 제품계열분석 모델에서는 R_1, R_2, R_3, R_4, R_5, R_6, R_7, R_8의 여덟 가지 관계를 정의한다. 특히, R_1과 R_2는 기존 특성 모델에서 정의된 집합화와 일반화 관계를 의미하고,R_3에서 R_8까지는 제품특성 간에 존재하는 의존관계를 나타낸다. 이에 대한 자세한 내용은 2.1 절에서 다룬다.

또한, 제품계열분석 모델에 포함된 특성들은 제품계열 범위 내에 있는 제품들을 생산(사례화)시키기 위한 파라미터 역할을 한다. 즉, 특정한 제품을 생산하기 위해서는 요구되는 모든 특성들을 이 제품과 결합시켜야 하는데, 요구되는 특성이 제품과 결합되는 시간이 제품 개발 시간, 제품 설치 시간, 혹은 제품 운영 시간 등으로 다양하게 나타날 수 있다. 예를 들면, 주문형 특성은 구체적인 내용이 고객의 요구에 따라 달라질 수 있으므로, 제품 개발 시간에 구현되어 제품과 결합되어야 하고, 고객이 선택할 수 있는 특성들은 제품계열자산 개발 시점에 컴포넌트로 구현되고 제품 설치 시간에 설치 파라미터에 따라 제품과 결합되거나 제품 운영 시간에 고객의 선택에 따라 동적으로 제품과 결합될 수 있다. 따라서 제품계열분석 모델은 각 특성들이 언제 제품에 결합 되는지를 나타내는 (f, bt)의 순서쌍 집합을 포함한다.

이와 같이 정의된 제품계열분석 모델은 제품계열을 구성하는 모든 제품에 대한 특성을 포함하고 있으므로, 제품계열 내의 특정 제품만을 위한 특성과 이들 간의 관계가 제품계열분석 모델로부터 도출될 수 있어야 한다. 따라서 특정 제품을 위한 분석 모델은 제품계열분석 모델에 정의된 특성 집합의 유효한 부분집합과 집합 내의 특성들 간의 관계로 정의된다.

정의 3. 제품계열분석 모델 $PLAM$ =(FS,CR,R_1,R_2, R_3,R_4,R_5,R_6,R_7,R_8, FBT)가 주어지면, 제품 분석 모델 PAM=(FS_P, R_{1P}, R_{2P}, R_{3P}, R_{4P}, R_{5P}, R_{6P}, R_{7P}, R_{8P})는 다음과 같다.

- FS_P: 공통특성 집합 (F_C)내의 모든 공통특성과 가변특성 집합 (F_V)내의 특성 중에 CR에 포함된 모든 구성 규칙들을 만족하는 가변 특성들로 구성된 집합이다.
- R_{1P}, R_{2P}, R_{3P}, R_{4P}, R_{5P}, R_{6P}, R_{7P}, R_{8P}: R_1, R_2, R_3, R_4, R_5, R_6, R_7, R_8에서 FS_P 내의 특성들 간의 모든 관계를 포함하는 R_1, R_2, R_3, R_4, R_5, R_6, R_7, R_8의 부분집합이다.

이와 같은 제품계열분석 모델과 제품분석모델의 정의를 보다 구체적으로 설명하기 위해서, 특성 간의 관계 및 특성결합시간에 대해서는 이어지는 절에서 자세히 알아본다.

3.1 특성 간의 관계

특성은 다른 특성과 독립적으로 존재한 다기 보다 다른 특성과 다양한 관계로 연관되어 있다. 특성 간의 관계는 크게 구조적 관계(structural relationship)와 수행적 의존 관계(operational dependency relationship)로 구분된다. 특성간의 구조적 관계는 특성 간의 추상화 계층을 나타내는 것으로, 집합 관계(aggregation relationship)와 일반화 관계(generalization relationship)가 구조적 관계로 분류된다. 이들 각각에 대해 자세히 살펴보면 다음과 같다.

하나의 특성은 여러 부분 특성으로 분해되거나, 여러 특성들이 하나의 집합 특성으로 묶일 수 있다. 예를 들면, ATM(Automated Teller Machine) 시스템에서, *예금 인출* 서비스는 *고객 인증*, *계좌 검증*, *현금 인출* 등의 세부 기능으로 구성되어 있다. 즉, *고객 인증*, *계좌 검증*, *현금 인출* 각각은 *예금 인출*의 한 부분 특성이고, *예금 인출*은 *고객 인증*, *계좌 검증*, *현금 인출*의 집합 특성을 의미한다. 이들 부분 특성과 집합 특성과의 관계는 집합 관계를 의미하며, 다음과 같이 정의 된다.

정의 4. 집합 관계(R_1)는 $FS \times FS$에 대한 이진 관계로서, FS에 포함된 두 개의 구별되는 특성 $f1$과 $f2$에 대해서 다음과 같이 정의 된다.

{($f1$,$f2$) | $f1$은 $f2$를 포함하는 여러 특성의 집합체이거나, $f2$는 $f1$의 한 부분이다}

주의할 점은 하나의 특성이 하나 이상의 여러 특성의 부분이 될 수 있다는 것이다. 예를 들면, *고객 인증*은 *예금 인출* 서비스나 *계좌 이체* 서비스 모두의 부분 특성이 될 수 있다. 따라서 집합 관계로 특성들을 구조화하면, 트리 형태라기보다 그래프 형태로 구성된다.

하나의 특성이 좀더 상세화된 특성으로 구체화 되거나, 여러 특성이 좀더 포괄적인 특성으로 일반화 때, 이들 특성 간의 관계를 일반화 관계라 한다. 예를 들면, *고객 인증* 서비스는 *카드 고객 인증* 혹은 *패스워드고객 인증* 등으로 좀 더 상세화 될 수 있다. 특성 간의 일반화 관계는 다음과 같이 정의 된다

정의 5. 일반화 관계(R_2)는 $FS \times FS$에 대한 이진 관계로서, FS에 포함된 두 개의 구별되는 특성 *f1*과 *f2*에 대해서 다음과 같이 정의 된다.

{(*f1*,*f2*) | *f1*은 *f2*의 일반화 이거나, *f2*는 *f1*의 상세화를 나타낸다}

특성 간의 수행적 의존 관계는 한 특성의 기능 수행이 다른 특성의 기능 수행에 영향을 받는다는 것을 의미한다. 본 논문에서는 특성 간의 수행적 의존 관계로 사용(usage), 변경(modification), 배타적 활성화(exclusive-activation), 종속적 활성화(subordinate-activation), 병렬적 활성화(concurrent-activation)와 순차적 활성화(sequential-activation)의 여섯 가지 의존 관계를 정의한다. 이들 각각에 대해 자세히 살펴보면 다음과 같다.

어떤 특성이 정의된 기능을 올바로 수행하기 위해서는 다른 특성의 올바른 기능 수행에 의존해야 할지도 모른다. 예를 들면, 엘리베이터 제어 소프트웨어에서 *방향 결정* 특성은 현재 위치 정보와 방향 정보로부터 엘리베이터의 다음 운행 방향을 계산하는 기능을 수행한다. 그런데, 현재 위치 정보는 *위치 제어* 특성의 수행 결과로부터 얻어지므로, 만약 *위치 제어* 특성이 정상적으로 수행되지 못하여 올바른 현재 위치 정보가 사용 가능하지 않는다면, *방향 결정* 특성은 정의된 기능을 올바로 수행할 수 없다. 따라서 *방향 결정* 특성은 *위치제어* 특성의 올바른 기능수행에 의존적인 관계에 있다. 이와 같이, 한 특성이 자신의 정의된 기능을 올바로 수행하기 위해서 다른 특성의 올바른 기능 수행을 필요로 한다면, 전자는 후자와 사용(usage) 의존 관계에 있다고 정의한다.

정의 6. 사용 관계(R_3)는 $FS \times FS$에 대한 이진 관계로서, FS에 포함된 두 개의 구별되는 특성 *f1*과 *f2*에 대해서 다음과 같이 정의 된다.

{(*f1*,*f2*) | *f1*은 그의 기능을 올바로 수행하기 위해 *f2*의 올바른 기능 수행을 필요로 한다.}

어떤 특성의 행위는 다른 특성의 수행에 따라 변경될 수도 있다. 가령, 예를 들면, 엘리베이터 제어 소프트웨어에서, *등록층 정지* 특성은 운행 중인 엘리베이터가 내부 혹은 각 층에서 등록한 서비스 층에 도달하였을 때 정지하도록 하는 기능을 수행한다. 하지만, *과부하 통과* 특성은 엘리베이터가 지정된 용량을 초과하면 각 층에서 등록한 서비스 등록 층에 도달하더라도 이를 무시하고 통과하는 기능을 수행한다. 즉, *과부하 통과* 특성의 수행은 모든 서비스 등록 층에 정지해야 하는 *등록층 정지* 특성의 행위를 변경하여 엘리베이터 내부에서 등록한 서비스 층에만 정지하도록 하는 역할을 한다. 이와 같이, 하나의 특성의 수행이 다른 특성의 행위를 변경한다면, 전자는 후자와 변경 의존 관계에 있다고 정의한다.

정의 7. 변경 관계(R_4)는 $FS \times FS$에 대한 이진 관계로서, FS에 포함된 두 개의 구별되는 특성 *f1*과 *f2*에 대해서 다음과 같이 정의 된다.

{(*f1*,*f2*) | *f1*의 수행은 *f2*의 행위의 변경을 야기 시킨다.}

특성이 정의된 기능을 수행하기 위해서는 먼저 활성화되어야 한다. 특성의 활성화는 다른 특성의 활성화에 의존적일 수 있는데, 본 논문에서는 배타적 활성화, 종속적 활성화, 병행적 활성화, 순차적 활성화의 네 가지 종류로 활성화 의존 관계를 정의한다.

배타적 활성화란 특성들이 같이 활성화 되어서는 안 되는 의존 관계를 의미한다. 예를 들면, 엘리베이터 제어 소프트웨어에서 *승객용 운전 서비스* 특성과 *소방관 운전 서비스* 특성은 같이 활성화 되어서는 안 되고, 한 순간에 한 가지 특성만 활성화되어야 한다. 따라서 배타적 활성화 관계는 다음과 같이 정의된다. (여기서, *Active(f,t1,t2)*는 특성 *f*가 *t1* 시간에서부터 *t2* 시간까지 활성화되면 참을 의미한다.)

정의 8. 배타적 활성화 관계(R_5)는 $FS \times FS$에 대한 이진 관계로서, FS에 포함된 두 개의 구별되는 특성 *f1*과 *f2*에 대해서 다음과 같이 정의 된다.

{(*f1*,*f2*) | $Active(f1,t1,t2) \wedge Active(f2,t3,t4) \Rightarrow x \neq y$, where $t1 \leq x \leq t2,\ t3 \leq y \leq t4$}

종속적 활성화란 한 특성의 활성화가 다른 특성이 활성화될 때만 이루어지는 의존 관계를 말한다. 예를 들면, *부름 요청 취소* 특성은 *승객용 운전 서비스* 특성이 활성화된 상태에서만 활성화 될 수 있고, *승객용 운전 서비스* 특성이 비활성화 되는 동안은 비활성화 되어야 한다. 종속적 활성화 관계는 다음과 같이 정의된다.

정의 9. 종속적 활성화 관계(R_6)는 $FS \times FS$에 대한 이진 관계로서, FS에 포함된 두 개의 구별되는 특성 *f1*과 *f2*에 대해서 다음과 같이 정의 된다.

{(*f1*,*f2*) | $Active(f2,t3,t4) \Rightarrow Active(f1,t1,t2) \wedge (t1 \leq t3 \leq t4 \leq t2)$}

한 특성에 대해 종속적 활성화 관계에 있는 여러 특성들 간에는 병행 혹은 순차 관점에서 의존 관계를 더 생각해 볼 수 있다.

병행적 활성화는 한 특성에 대해 종속적 활성화 관계에 있는 특성(종속특성)들 간에 활성화가 병행적으로 일어나야 한다는 것을 의미한다. 예를 들면, *부름 처리* 특성과 *운전 제어* 특성은 *승객용 운전 서비스* 특성이 활성화 되어 있는 동안에 활성화될 수 있는 종속 특성들인데, 이 두 특성은 승객용 운전 서비스가 제공되고 있는 동안에 서로 병행적으로 활성화되어야 한다. 하지만, *부름 처리* 특성과 *운전 제어* 특성은 다른 운전 서비스

(예, *소방관 운전 서비스*) 가 활성화 되어 있는 경우에는 병행적으로 활성화되어서는 안 된다. 따라서 병행적 활성화는 다음과 같이 정의된다.

정의 10. 병행적 활성화 관계(R_7)는 $FS\times FS\times FS$에 대한 삼진 관계로서, FS에 포함된 세 개의 특성 *f1*, *f2*, *f3*에 대해서 다음과 같이 정의 된다.

$\{(f1,f2,f3) \mid (f1,f2)\in R_6 \wedge (f1,f3)\in R_6 \wedge Active(f2,t1,t2) \wedge Active(f3,t1,t2) \wedge (t1 \le t2)\}$

순차적 활성화는 종속특성들이 순차적으로 활성화 되어야 한다는 것을 의미한다. 예를 들면, *부름 처리*, *운전 제어*, *도어 제어* 특성들은 *소방관 운전 서비스* 특성이 활성화된 동안에 순차적으로 활성화되어야 한다. 즉, 한 특성이 수행을 마친 후에 순서상으로 다음에 있는 특성이 활성화 되어 수행을 하게 된다. 따라서 순차적 활성화는 다음과 같이 정의된다.

정의 11. 순차적 활성화 관계(R_8)는 $FS\times FS\times FS$에 대한 삼진 관계로서, FS에 포함된 세 개의 특성 *f1*, *f2*, *f3*에 대해서 다음과 같이 정의 된다.

$\{(f1,f2,f3) \mid (f1,f2)\in R_6 \wedge (f1,f3)\in R_6 \wedge (Active(f2,t1,t2) \Rightarrow Active(f3,t2,t3) \wedge (t1 \le t2 \le t3))\}$

병행적 활성화와 순차적 활성화는 다른 활성화 의존관계와 달리, 세 특성 간의 관계로 정의된다. 이는 두 특성이 어떤 특성의 종속특성인가에 따라, 병행적으로 활성화되어야 할 수도 있고 순차적으로 활성화되어야 할 수도 있기 때문이다. 앞에서도 예시하였듯이, *부름 처리*와 *운전 제어* 특성은 *승객용 운전 서비스* 특성이 활성화 된 동안에는 병행적으로, *소방관 운전 서비스* 특성이 활성화 된 동안에는 순차적으로 활성화되어야 한다.

제품계열분석 모델을 이루는 모든 요소들(즉, 특성 집합, 특성 간의 관계, 구성 규칙, 결합 시간)은 서로 일관성을 가져야 한다. 다음 장에서는 제품계열분석 모델의 일관성을 유지시키기 위해 만족 되어야 하는 규칙을 살펴본다.

4. 제품계열분석 모델의 일관성

제품계열분석 모델의 일관성은 구성규칙의 일관성, 특성 관계의 일관성, 그리고 특성결합시간의 일관성 관점에서 살펴볼 수 있다.

4.1 구성 규칙의 일관성

제품계열에 정의된 모든 구성규칙은 서로 상충되는 면이 없이 일관적으로 정의되어 있어야 한다. 다음은 구성 규칙 간의 일관성을 정의하기 위해서 유용하게 사용되는 집합을 정의한 것이다.

- Require$^+$(f): 특성 f와 함께 같은 제품에 존재해야 하는 특성들의 집합을 의미하며, Req(필수선택) 규칙에 의해 직, 간접적으로 연관된 모든 특성을 포함한다. 이 집합은 다음과 같이 정형적으로 정의된다.

  $Require^+(f) \triangleq \cdot \mu Z.Require(\{f\}) \cup Require(Z)$, where $Require(X) \triangleq \{y \mid Req(x,y)\in CR,\ where\ x \in X \subseteq F, y\in F\}$
- Excluded(f): 특성 f와 함께 같은 제품에 존재해서는 안 되는 특성들의 집합을 의미하며, f와 함께 Ex(배타선택) 규칙에 의해 연관된 모든 특성과 f와 함께 XOR 규칙에 의해 연관된 모든 특성을 포함한다. 이 집합은 다음과 같이 정형적으로 정의된다.

  Excluded(f) $\triangleq$

  $\{x \mid (Ex(f,x)\in CR \vee Ex(x,f)\in CR)$, where $x \in F\} \cup$

  $\{y \mid (XOR(z,S) \in CR) \wedge (f\in S) \wedge (y\in S-\{f\})$,

  where $y\in F$, $z\in F$, $S\subseteq F\}$
- OR(f): 특성 f가 선택되었을 때 함께 선택되어야 하는 다중특성[1]의 집합을 의미하며 , 이것은 특성 f와 함께 존재해야 하는 모든 특성 각각에 대해서 OR 규칙에 의해 연관된 모든 다중특성을 포함한다. 이 집합은 다음과 같이 정형적으로 정의된다.

  $OR(f) \triangleq$

  $\{S \mid OR(f,S)\in CR$, where $S\subseteq F\} \cup$

  $\{S \mid OR(x,S)\in CR$, where $x\in Require^+(f)$, $S\subseteq F\}$
- XOR(f): 특성 f가 선택되었을 때 함께 선택되어야 하는 택일특성[2]의 집합을 의미하며, 이것은 특성 f와 함께 존재해야 하는 모든 특성 각각에 대해서 XOR 규칙에 의해 연관된 모든 택일특성을 포함한다. 이 집합은 다음과 같이 정형적으로 정의된다.

  $XOR(f) \triangleq$

  $\{S \mid XOR(f,S)\in CR$, where $S\subseteq F\} \cup$

  $\{S \mid XOR(x,S)\in CR$, where $x\in Require^+(f)$, $S\subseteq F\}$

이상의 집합을 이용하여 구성 규칙간의 일관성을 정의하면 다음과 같다.

규칙 1: 필수선택(Req 구성규칙)의 일관성

$\forall f\in F \cdot Require^+(f) \cap Excluded(f) = \emptyset$

특성 f와 함께 선택되어야 하는 특성들은 f에 의해 선택이 제한되는 특성 집합에 속해서는 안 된다.

규칙 2: 다중선택(OR 구성규칙)의 일관성

$\forall A\in OR(f) \cdot A \not\subset Excluded(f)$

특성 f와 함께 선택되어야 하는 다중특성은 적어도 그 특성집합내의 한 특성이 선택되어야 하므로, 이 다중특성은 f에 의해 선택이 제한되는 특성 집합의 부분 집

1) 다중특성: 특성의 집합으로 집합 내의 특성 중 적어도 하나가 반드시 선택되어야 한다.

2) 택일특성: 특성의 집합으로 집합 내의 특성 중 오로지 하나만 선택되어야 한다.

합 이어서는 안 된다. 만약 부분 집합이라면, 다중특성의 어떤 특성도 선택될 수 없으므로, 이는 OR 구성규칙을 위반하게 된다.

규칙 3: 택일선택(XOR 구성)의 일관성

$\forall A \in XOR(f) \cdot A \not\subset Require^{+}(f)$

특성 f와 함께 선택되어야 하는 택일특성은 그 집합 내의 단 하나의 특성만 선택되어야 하므로, 이 택일특성은 f와 함께 선택되어야 하는 특성 집합의 부분집합이어서는 안 된다. 만약 부분 집합이라면, 택일특성의 모든 특성이 선택될 수 있으므로, 이는 XOR 구성규칙을 위반하게 된다.

4.2 특성 관계의 일관성

앞 장에서 특성 간의 관계로서 구조적 관계와 수행적 의존 관계에 대해서 정의하였다. 이들 특성 간의 관계에 있어서도 서로 상충되는 면이 없이 일관적으로 정의되어야 한다. 다음은 특성 관계 사이의 일관성을 정의하기 위해서 유용하게 사용되는 집합이다.

- Child(X): 구조적 관계에 의해 구조화된 특성 그래프에서 특성 집합 X의 자식 특성 집합을 의미하며, 다음과 같이 정의된다.
  $Child(X) \triangleq \{f \mid (x,f) \in R_1 \cup R_2,\ \text{where } x \in X \subseteq F,\ f \in F\}$
- Descendant(f): 구조적 관계에 의해 구조화된 특성 그래프에서 특성 f의 자손 특성 집합을 의미하며, 다음과 같이 정의된다.
  $Descendant(f) \triangleq \mu Z.Child(\{f\}) \cup Child(Z)$
- Parent(X): 구조적 관계에 의해 구조화된 특성 그래프에서 특성 집합 X의 부모 특성 집합을 의미하며, 다음과 같이 정의된다.
  $Parent(X) \triangleq \{f \mid (f,x) \in R_1 \cup R_2,\ \text{where } x \in X \subseteq F,\ f \in F\}$
- Ancestor(f): 구조적 관계에 의해 구조화된 특성 그래프에서 특성 f의 조상 특성 집합을 의미하며, 다음과 같이 정의된다.
  $Ancestor(f) \triangleq \mu Z.Parent(\{f\}) \cup Parent(Z)$

이상의 집합을 이용하여 특성 관계 간의 일관성을 정의하면 다음과 같다.

규칙 4: (특성 관계 간의 일관성)

$$\forall f \in F \cdot \bigwedge_{x \in Excluded(f)} \left((x,f) \notin \bigcup_{1 \le i \le 6} R_i \wedge (f,x) \notin \bigcup_{1 \le i \le 6} R_i \right)$$

어떤 특성 f에 의해 선택이 제한되는 특성들 간에는 어떠한 관계도 정의되어서는 안 된다. 만약 같은 제품에 존재할 수 없는 특성들 간에 정의된 특성 관계가 있다면, 이는 무의미하기 때문이다.

규칙 5: (구조적 관계의 비 순환성)

$\forall f \in F \cdot Descendant(f) \cap Ancestor(f) = \emptyset$

구조적 관계로 구조화된 특성들 간에는 순환이 없어야 한다. 즉, 어떤 특성의 자손 특성이 그 특성의 조상 특성이 되어서는 안 된다.

규칙 6: (병행적과 배타적 활성화 관계 간의 일관성)

$\forall (f_1 f_2) \in R_5 \cdot (f_x f_1 f_2) \notin R_7 \wedge \forall (f_1 f_2 f_3) \in R_7 \cdot (f_2 f_3) \notin R_5$

배타적 활성화 관계에 있는 특성들은 서로 같이 활성화 될 수 없는 관계를 말하므로, 특성들이 서로 같이 활성화 되어야 하는 병행적 활성화 관계와는 동시에 관련되어서는 안 된다.

규칙 7: (구조적 관계와 가변성 간의 일관성)

$\forall f \in F \cdot Parent(\{f\}) \subseteq F_V \Rightarrow (f \in F_V)$

특성이 한 제품에 존재하기 위한 조건으로는 그 특성의 조상이 같은 제품에 존재해야 한다. 만약 어떤 특성의 모든 부모 특성들이 가변특성 집합에 속한다면, 그 특성 자체도 가변특성 집합에 속해야 한다. 그렇지 않다면 부모 특성들이 모두 선택되지 않은 상황에서 그 특성 자체가 제품에 존재할 수 있게 되기 때문이다.

규칙 8: (사용 관계와 가변성 간의 일관성)

$\forall (f_1 f_2) \in R_3 \cdot (f_1 \in F_C) \Rightarrow (f_2 \in F_C)$

만약 특성 f1이 다른 특성 f2를 사용한다면, 이 두 특성은 같은 제품에 존재해야 한다. 하지만, 만약 f1은 공통특성 이지만 f2가 가변특성 라면, f1과 f2가 같은 제품에 존재하지 않을 수 있다는 것을 의미하므로, 공통특성이 가변특성을 사용하는 관계는 특성 간의 관계에 있어서 불일치를 나타낸다.

4.3 특성 결합 시간의 일관성

다음은 각 특성의 결합 시간을 결정할 때 지켜져야 하는 규칙을 말한다(여기서, FBT(f)는 특성 f의 결합시간을 나타내며 {PBT, PIT, PRT}중의 하나의 값을 나타낸다. 그리고 PBT<PIT<PRT의 관계가 있다).

규칙 9: (구조적 관계와 특성 결합 시간의 일관성)

$\forall (f_1 f_2) \in R_1 \cup R_2 \cdot \mathrm{FBT}(f_1) \le \mathrm{FBT}(f_2)$

구조적 관계에 있는 두 특성에서 자식 특성은 부모 특성이 같은 제품에 존재해야 존재할 수 있다. 따라서 자식 특성이 제품과 결합되는 시간이 부모 특성이 제품과 결합되는 시간보다 먼저일 수는 없다.

규칙 10: (구성 규칙과 특성 결합 시간의 일관성)

$\forall f_x \in Require^{+}(f) \cdot FBT(f) = FBT(f_x)$

구성 규칙 중 Req(필수선택) 규칙에 의해 연관된 두 특성의 결합은 서로 같은 시간에 이루어져야 한다.

이상과 같은 일관성 규칙은 제품계열분석 모델이 유효한 제품 분석 모델을 사례화 할 수 있기 위해 지켜져야 하는 선행 조건이다.

5. 관련연구

특성 지향의 영역 분석인 FODA(feature-oriented domain analysis)[5]는 영역 관점에서 소프트웨어 제품

의 공통성과 가변성을 분석하기 위해서 1990년에 제안되었다. 그 이후로 여러 영역 공학 방법뿐만 아니라, 제품계열공학 방법에서도 특성 지향의 분석 방법을 제품 간의 공통성과 가변성을 분석하는 수단으로 사용하거나 자신들의 방법에 맞게 적응시키거나 확장해왔다. 그 대표적인 예로서 Generative Programming[2], Feature-RSEB[3], FORM[4] 등이 있다. 이들은 특성 지향의 공통성과 가변성 분석을 통해 영역 혹은 제품계열 내의 제품들 간에 재사용가능한 아키텍처나 컴포넌트를 개발하는 방법을 제공해 왔다. 하지만 이들 방법에서는 제품 간의 공통성과 가변성 분석에만 초점을 두었지, 제품특성 간에 존재하는 다양한 의존관계는 명시적으로 분석하지 않았다.

최근에는 특성간의 의존 관계 및 특성결합시간에 대한 연구가 활발히 진행되고 있다. Ferber[7]와 Fey[8]는 제품특성 간에 존재하는 의존성을 명시적으로 분석하기 위해 기존 특성모델을 확장하였고, 저자[9]는 이들의 연구결과를 확장하여 특성 간의 의존성이 제품계열자산 개발에 어떻게 도움이 되는가를 발표하였다. 또한 Lee[10]는 특성결합시간이 제품계열자산에 미치는 영향에 대한 연구를 수행하였다. 본 논문에서는 이러한 기존의 연구결과를 바탕으로 공통성과 가변성, 특성간의 (의존) 관계, 그리고 특성결합시간의 세 가지 관점을 포함하는 제품계열분석 모델을 정형적으로 정의하였다.

모델의 정형적 정의는 모델의 일관성 검사를 위해 필수적인 요소이다. 지금까지 연구들은 요구 명세의 일관성 검사[11,12], 요구 명세와 설계 명세 간의 일관성 검사[13], 설계 모델의 일관성 검사[14] 등의 다양한 형태로 이루어져왔다. 이러한 기존 연구에서 사용된 모델의 정형적 정의 및 일관성 검사 방법은 제품계열모델의 정형적 정의 및 일관성 검사에 활용될 수 있다. 최근에 [15]는 제품계열의 가변성에 대한 일관성을 검사하기 위해 논리 식을 이용하여 제품계열모델을 정의하고, 이 모델의 일관성을 검사하는 방법을 제안하였다. 본 논문에서는 제품계열의 가변성뿐만 아니라, 제품계열 내의 특성 간의 의존 관계 및 특성결합시간 관점을 포함한 제품계열분석 모델에 대한 일관성 검사를 제공했다는 점이 기존 연구와 차이가 난다.

6. 결 론

최근 들어 소프트웨어 제품계열의 방법 개발 및 실제 응용 소프트웨어로의 적용은 활발히 진행되어 오고 있다. 하지만, 제품계열의 개념에 대해 정형적으로 정의하고 분석하려는 노력은 거의 없어 왔다. 본 논문에서는 소프트웨어 제품계열의 중요한 세 가지 활동 중 제품계열분석 활동의 결과인 제품계열분석 모델을 정형적으로 정의하고, 이 분석 모델의 구성요소들이 서로 일관성을 가지는 지를 검사하기 위한 규칙을 정의하였다.

제품계열분석 모델이 일관성을 가진다는 의미는 이 모델로부터 개별적인 제품을 위한 유효한 분석 모델을 유도할 수 있다는 의미를 가진다. 이러한 일관성을 지닌 제품계열분석 모델은 제품계열 아키텍처 및 컴포넌트 모델로부터 특정 제품을 위한 유효한 모델의 사례화에 초석을 마련한다는 점에서 의의가 있다.

특히, 본 논문에서는 기존의 특성 모델을 특성 간의 의존관계 및 특성 결합 시간 관점에서 확장하여 제품계열분석 모델을 정의하였다. 하지만, 본 논문에서 정의된 특성 간의 의존관계는 기능 특성간의 관계에만 초점을 두었다. 비 기능 특성간의 혹은 비 기능 특성과 기능 특성와의 관계는 본 논문에서 다루지 않았다. 향후에 이들 간의 관계에 대한 연구가 수행되어야 할 것이다.

참 고 문 헌

[1] Clements, P. and Northrop, L., "Software Product Lines: Practices and Patterns," Addison-Wesley, Upper Saddle River, NJ, 2002.

[2] Czarnecki, K. and Eisenecker, U., "Generative Programming: Methods, Tools, and Applications," Addison-Wesley, New York, 2000.

[3] Griss, M., Favaro, J., d'Alessandro, M., "Integrating Feature Modeling with the RSEB," In Proceedings of Fifth International Conference on Software Reuse, pp. 76-85, 1998.

[4] Kang, K. C., Kim, S., Lee, J., Shin, E., Huh, M., "FORM: A Feature-Oriented Reuse Method with Domain Specific Reference Architectures," Annals of Software Engineering, Vol. 5, pp. 143-168, 1998.

[5] Kang, K. C., Cohen, S., Hess, J., Nowak, W., Peterson, S., "Feature-Oriented Domain Analysis (FODA) Feasibility Study," Technical Report CMU/SEI-90-TR-21, Pittsburgh, PA, Software Engineering Institute, Carnegie Mellon University, November 1990.

[6] Lee, K., Kang, K. C., Lee, J., "Concepts and Guidelines of Feature Modeling for Product Line Software Engineering," C. Gacek (ed.), Software Reuse: Methods, Techniques, and Tools, Berlin, Springer-Verlag, pp. 62-77, 2002.

[7] Ferber, S., Haag, J., Savolainen, J., "Feature Interaction and Dependencies: Modeling Features for Reengineering a Legacy Product Line," G. Chastek (ed.), Software Product lines, Springer-Verlag, Berlin, pp. 235-256, 2002.

[8] Fey, D., Fajta, R., Boros, A., "Feature Modeling: A Meta-model to Enhance Usability and Usefulness," G. Chastek (ed.), Software Product lines, Springer-Verlag, Berlin, pp. 198-216, 2002.

[9] Lee, K. and Kang, K. C., "Feature Dependency Analysis for Product Line Component Design," J. Bosch, C. Krueger (eds.), Software Reuse: Methods, Techniques, and Tools, Berlin, Springer-Verlag, pp. 69-85, 2004.
[10] Lee, J. and Kang, K. C., "Feature Binding Issues in Variability Analysis for Product Line Engineering," Workshop on "Modeling Variability for Object-Oriented Product Lines" at ECOOP 2003, Darmstadt, Germany, July 21, pp. 77-82, 2003.
[11] Heimdahi, M. P. E., Leveson, N. G., "Completeness and Consistency Analysis of State-Based Requirements," In Proceedings of the 17th International Conference on Software Engineering, pp. 3-14, 1995.
[12] Heitmeyer, C. L., Jeffords, R. D., Labaw, B. G., "Automated Consistency Checking of Requirements Specifications," ACM Transactions on Software Engineering and Methodology, Vol. 5, No. 3, pp. 231-261, July 1996.
[13] Chechik, M., Gannon, J., "Verification of Consistency Between Concurrent Program Designs and Their Requirements," In Proceedings of COMPASS'96, June, 1996.
[14] Allen, R. J., "A Formal Approach to Software Architecture," Ph.D. thesis, Carnegie Mellon University, Pittsburgh, PA, 1997.
[15] Mannion, M., "Using First-Order Logic for Product Line Model Validation," LNCS2379, pp. 176-187, 2002.

이 관 우

1994년 포항공과대학교 컴퓨터공학과(학사). 1996년 포항공과대학교 컴퓨터공학과(공학석사). 2003년 포항공과대학교 컴퓨터공학과(공학박사). 2003년 9월~현재 한성대학교 정보시스템공학과 전임강사 관심분야는 제품계열공학, 가상현실 시각화, 실시간 시스템 개발, CASE 도구 개발