프로덕트 라인 공학의 체계적 비즈니스 케이스 분석 기법

박 신 영[†] · 김 수 동[††]

요 약

프로덕트 라인 공학(Product Line Engineering, PLE)은 도메인의 멤버간에 공통적인 휘처를 핵심 자산으로 만들고, 만들어진 핵심 자산을 이용해서 어플리케이션을 개발하는 방법론이다. 따라서 PLE 방법론을 사용해서 제품을 개발하면, 개발 비용은 감소시키고 재사용성은 증가시킬 수 있다. 핵심 자산의 재사용성을 최대화 하기 위해서는, 제품 계열 공학을 위한 비즈니스 케이스 분석이 요구된다. 제품 계열 공학의 핵심 자산 영역이 지나치게 광범위하면, 핵심 자산 개발 비용은 증가하지만 재사용성은 감소할 수 있는 반면, 핵심 자산의 영역이 지나치게 좁으면 핵심 자산을 개발하는 비용은 작지만 극히 일부의 멤버만이 사용할 수 있으므로 핵심 자산의 적용성은 감소된다.

이 논문에서는 프로덕트 라인을 적용하기 위해서 도메인을 분석한 후 비즈니스 케이스를 분석하는 프로세스를 제안한다. 그리고 프로세스의 각 활동을 위한 가이드라인을 제안해서 PLE 방법을 적용해서 시스템을 개발 할 경우 최대의 이익을 획득할 수 있도록 핵심 자산의 영역을 결정하는 방법을 제안한다. PLE에서 가변성은 중요한 개념일 뿐만 아니라, 핵심 자산 개발 비용에 영향을 미친다. 따라서 이 논문에서는 비즈니스 케이스 분석을 위한 프로세스에 가변성을 상세한 수준으로 반영한다. 우리가 제안한 프레임워크를 적용하면, PLE 방법론을 적용해서 핵심 자산을 만들고 시스템을 개발할 때, 최적의 이익을 얻을 수 있을 것으로 기대한다.

키워드 : 프로덕트 라인 공학, 비즈니스 케이스 분석, 핵심 자산 영역 설정, 가변성

A Systematic Method for Analyzing Business Cases in Product Line Engineering

Shin Young Park[†] · Soo Dong Kim[††]

ABSTRACT

Product Line Engineering (PLE) is an effective reuse methodology where common features among members are captured into core assets and applications are developed by reusing the core assets, reducing development cost while increasing productivity. To maximize benefits in developing systems, business case analysis for PLE is essential. If the scope for core assets is excessively broad, it will result in high cost of asset development while lowering reusability. On the other hand, if the scope is too narrow, it will result in a limited applicability which only supports a small number of members in the domain.

In this paper, we propose a process for business case analysis for PLE and for deciding economical analysis of core asset scope. Then, we define guidelines for each activity of the process. Since variability often occurs in PLE, we significantly treat the variability of features among members in detailed level. By applying our framework for business case analysis, one can develop core assets of which scope provide the most economical value with applying PLE.

Key Words : Product Line Engineering, Business Case Analysis, Core Asset Scoping, Variability

1. 서 론

프로덕트 라인 공학(Product Line Engineering, PLE)은 도메인의 멤버간에 공통적인 휘처를 핵심 자산으로 만들고, 만들어진 핵심 자산을 이용해서 어플리케이션을 개발하는 방법론이다 [1]. 따라서 PLE 방법론을 사용하면, 개발 비용은 감소시키며 재사용성은 증가시킬 수 있다. 핵심 자산을 개발하고 재사용하기 위해서는 멤버간에 공통적인 휘처를 추출하고, 공통적인 휘처 내에서 가변적인 부분을 분석하는 일이 중요하다. 이 가변적인 부분은 가변성(variability)이라고 하며, 핵심 자산을 어플리케이션에 맞게 개발하는 능력을 제공한다.

PLE 방법론을 적용해서 제품을 개발할 것인지 또는 시스템을 개별적으로 개발할 것인지 확인하고 결정하기 위해서는 비즈니스 케이스를 분석해야 한다. 그리고 PLE 방법론을 적용해서 시스템을 개발할 경우, 개발 이익을 최대화 하기 위해서는 핵심 자산의 영역을 설정하는 과정이 중요시된다. 비즈니스 케이스 분석은 이와 같이 PLE를 적용한 개발 방

※ 본 연구는 숭실대학교 교내연구비 지원으로 이루어졌음.
† 준 회 원 : 숭실대학교 대학원 컴퓨터학과
†† 종신회원 : 숭실대학교 대학원 컴퓨터학과 부교수
논문접수 : 2005년 12월 31일, 심사완료 : 2006년 6월 8일

법에서 재사용성을 최대화 할 수 있는 핵심 자산의 영역을 결정하는 과정에도 필요하다. 만약 핵심 자산의 영역이 지나치게 광범위하면, 핵심 자산 개발 비용은 증가하지만 특정 멤버가 핵심 자산의 모든 휘처를 사용할 수 없으므로 재사용성은 감소할 수 있는 반면, 핵심 자산의 영역이 지나치게 좁다면 핵심 자산 개발 비용은 감소하지만 핵심 자산을 사용할 수 있는 멤버는 극히 일부로 한정되므로 핵심 자산의 적용성은 역시 감소할 것이다.

핵심 자산의 영역 설정은 제품 계열 영역 설정(product line scoping), 도메인 영역 설정(domain scoping), 핵심 자산 영역 설정(asset scoping)으로 분류된다 [2]. 비즈니스 케이스를 분석하기 위해서, 이 논문에서는 위에 언급한 세가지 영역 설정 중에서 핵심 자산 영역 설정만을 고려하며, 핵심 자산 영역 설정을 기반으로 비즈니스 케이스를 분석하기 위한 프로세스와 공식을 제안한다. 그리고 프로세스의 각 활동을 위한 가이드라인을 제안해서 비즈니스 케이스 분석을 최적화 할 수 있는 방법을 제공한다. PLE에서 가변성은 중요한 개념일 뿐만 아니라, 핵심 자산 개발 비용에 영향을 미친다. 따라서 이 논문에서는 비즈니스 케이스 분석을 위한 프로세스와 비즈니스 케이스 분석에 상세한 수준으로 가변성을 반영한다. 우리가 제안한 프레임워크를 적용하면, 가장 경제적인 가치를 가질 수 있는 핵심 자산을 실체화(realize)해서 시스템을 개발할 수 있을 것으로 기대한다.

2장에서는 관련 연구를 설명하며, 3장에서는 비즈니스 케이스를 분석하기 위한 다섯 단계의 활동을 정의한다. 활동 1은 공통성을 분석하며, 활동 2는 가변성을 분석한다. 활동 3은 가변성간 의존성을 분석하며, 활동 4는 도메인 모델을 정제한다. 마지막으로 활동 5는 비즈니스 케이스를 분석한다. 4장에서는 비즈니스 케이스 분석을 위한 프로세스를 적용해서 사례 연구를 하며, 5장에서는 결론을 맺는다.

2. 관련 연구

2장에서는 PLE 방법론 중에서 프로덕트 라인 공학을 위한 비즈니스 케이스 분석과 관련된 대표적인 연구를 조사한다.

2.1 Bosch의 프로세스

Bosch는 소프트웨어 아키텍처 설계를 강조하는 PLE 프로세스를 제안한다 [3]. Bosch가 제안한 프로세스는 비즈니스 케이스 분석, 영역 설정, 휘처와 프로젝트에 대한 계획 수립 등의 단계를 포함한다. 비즈니스 케이스 분석은 PLE 방법론을 적용해서 제품을 개발하는 경우와 PLE 방법론을 적용하지 않고 제품을 개발하는 방법을 비교해서 전자의 접근 방식을 채택했을 때 획득할 수 있는 이익을 예측하는 목적으로 수행된다.

비즈니스 케이스를 분석하기 위한 활동을 구체적으로 살펴보면, 우선 개별적으로 제품을 개발하는 방법을 확인하고, 이 방법으로 소프트웨어를 개발할 경우 취할 수 있는 이익을 예측한다. 그리고 PLE 기반의 개발 방법을 적용하기 위한 투자 비용을 계산하고, 마지막으로 PLE 기반의 개발 방법을 적용했을 때, 향후 획득할 수 있는 이익을 계산하다.

Bosch의 프로세스는 비즈니스 케이스를 분석하는 프로세스를 제안한다. 그러나 구체적인 지침이나 산출물 등은 제안하지 않으며, PLE 방법론을 적용해서 애플리케이션을 개발하기 위한 비용을 계산하는 방법은 제안하지 않기 때문에 적용하는데 한계가 있다.

2.2 ESAPS

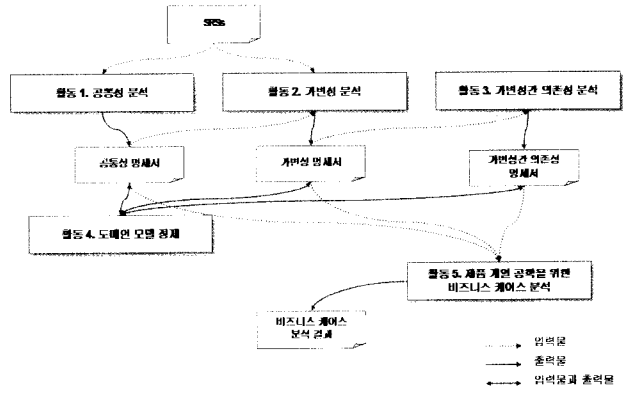
ESAPS(Engineering Software Architectures, Processes and Platforms for System-Families) 는 ITEA가 PLE 개념을 적용해서 수행한 프로젝트며 방법론이다[3]. ESAPS 프로젝트는 유럽의 기업과 학계가 가진 기술을 기반으로, 향상된 시스템 패밀리를 개발하기 위해서 수행되었다. ESAPS 프로젝트는 영역 설정과 관련된 산출물을 제공하며, 영역 설정 방법을 다음과 같은 세 단계로 분류한다. PL 영역 설정(PL scoping)은 제품 포트폴리오를 체계적으로 정의하기 위한 방법이며, 도메인 영역 설정(Domain scoping)은 PL 도메인의 적합한 경계를 결정하기 위한 방법이다. 핵심 자산 영역 설정(Asset scoping)은 재사용 가능한 구성 요소들을 결정하는 방법이다.

ESAPS는 PLE 기반으로 제품을 개발하기 위한 영역 설정 방법보다는, 도메인 영역 설정 방법과 핵심 자산 영역 설정 방법을 더 강조한다. 이 중에서 핵심 자산의 영역을 설정하는 방법은 비즈니스 케이스와 밀접한 관련이 있는데, 비즈니스 케이스를 분석하기 위한 핵심 자산의 영역을 설정하는 과정이나 방법은 설명이 충분하지 않다.

2.3 비즈니스 케이스 분석을 위한 프로세스

3장에서는 비즈니스 케이스 분석을 수행하기 위한 프로세스를 제안한다. 이 프로세스는 다섯 단계의 활동으로 구성되며, 각 활동에 대한 작업 지침과 산출물은 (그림 1)과 같다.

이 프로세스는 비즈니스 케이스에 적용할 도메인을 분석해서 핵심 자산에 포함될 휘처를 결정하고 포함된 휘처를 개발하기 위한 비용을 계산해서 비즈니스 케이스에 적용하기 위한 단계를 포함한다.



(그림 1) 비즈니스 케이스를 분석하기 위한 프로세스

다음에 제시하는 각각의 활동은, 활동의 목적에 대해서 설명한 후 입력물, 세부 단계와 지침, 가이드라인, 산출물, 산출물이 다음 활동에서 어떻게 활용되는지에 대한 설명 순으로 제시된다.

[활동 1] 공통성 분석

프로세스의 첫 번째 활동은 도메인을 분석한 후 제품 계열(Product Line, PL)의 멤버간 공통 휘처를 확인하기 위해서 수행한다. 공통성이 핵심 자산의 재사용성에 가장 큰 영향을 미치므로, 공통 휘처를 확인하고 실체화하는 활동은 중요하다.

이 활동의 입력물은 도메인의 소프트웨어 요구사항 명세서(Software Requirement Specifications, SRSs)이다.

SRS 작성 방법과 SRS에 사용된 용어는 멤버간에 차이가 크므로, 작성법과 용어를 일반화시키고 표준화해서 SRS를 재작성 할 필요가 있다. 작성 방법과 용어를 표준화해야만 SRS를 비교하기 쉽기 때문이다.

PL의 규모가 큰 경우에는, 전체 도메인을 서브 도메인으로 분해하면 좀 더 효과적으로 휘처를 비교할 수 있다.

휘처를 체계적으로 비교해서 공통성을 식별하기 위해서는 <표 1>과 같은 **공통성 명세서**를 사용할 수 있다[5]. 도메인 분석가는 표에 휘처를 열거한 후에, 얼마나 많은 멤버들이 각 휘처를 필요로 하는지 확인해서 공통 휘처를 결정한다.

공통성 분석 활동의 결과는 **공통성 명세서**에 작성하며, 공통성 명세서는 가변성 분석 활동에 사용된다.

〈표 1〉 공통성 명세서

공통성 번호	서브 도메인	공통 휘처	멤버			공통성 정도	의사결정 (Y/N)
			A	B	C		

[활동 2] 가변성 분석

가변성은 하나의 공통성을 사용하는 멤버가 갖는 작은 차이로, 속성(attribute), 로직(logic), 워크플로우(workflow), 영속성(persistence), 인터페이스(interface) [6]에서 발생된다. 핵심 자산에 있는 가변성을 설계하고 실체화하는 것은 재사용성 측면에서 중요하다.

활동 2는 PL의 멤버간 가변성을 식별하고 분석하기 위해서 수행한다.

이 활동의 입력물은 멤버들의 *SRS*와 **공통성 명세서**이다.

가변성은 가변성이 발생하는 장소인 가변점과, 가변점에 바운드 될 수 있는 값인 가변치로 이루어진다 [7]. 가변점, 가변성 타입, 바인딩 타입, 가변치, 기본 가변치 등을 명세하기 위해서 <표 2>와 같은 **가변성 명세서**를 사용한다.

바인딩 타입은 옵션(optional) 타입, 이진(binary) 타입, 선택(selection) 타입, 오픈(open) 타입으로 분류된다. 옵션 바인딩 타입은 하나의 가변치가 사용되거나 사용 되지 않는 두 가지 경우를 가지며, 이진 바인딩 타입의 가변점은 오직 두 개의 가변치만 가질 수 있다. 옵션 바인딩 타입의 가변점은 두 개 이상의 가변치를 가지며, 오픈 바인딩 타입의 가변점은 가변치를 추가할 수 있는 타입이다.

〈표 2〉 가변성 명세서

가변성 번호	공통 휘처	가변점	가변성 타입	바인딩 타입	가변치 (V)			기본 가변치
					멤버 A	멤버 B	멤버 C	

가변성 명세서는 도메인 분석 수준의 의사 결정 모델이며, 가변성 의존성을 분석하는 입력물로써 사용된다.

[활동 3] 가변성간 의존성 분석

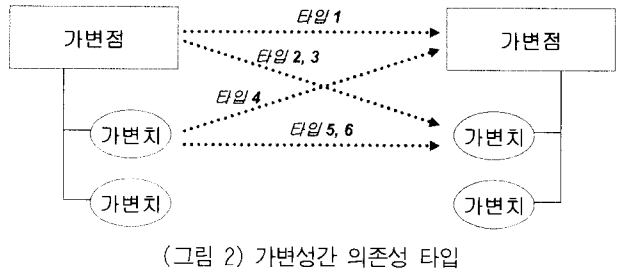
가변성간 의존성은 두 개의 가변성이 밀접한 관련을 갖는 경우에 발생하며, 가변점과 가변치 모두 가변성간 의존성과 관련 있다. 특정 가변성을 바인딩하면 다른 가변성을 만족시켜야 하는 어떤 활동이나 조건이 발생하는데, 유용하고 신뢰할 수 있는 핵심 자산을 개발하기 위해서는 가변성간 의존성이 핵심 자산에서 분석되고 실체화 되어야 한다. 또한 가변성과 가변성간 의존성이 비즈니스 케이스 분석에 적용되면, 더 높은 정확도를 갖는 분석이 수행될 수 있다.

활동 3은 가변성간 의존성을 분석하기 위해서 수행하며, 이 활동의 입력물은 *SRS*와 **가변성 명세서**이다.

(그림 2)와 같이, 가변성간 의존성은 여섯 가지 타입으로 분류된다[7]. 점선 화살표는 의존성이 발생되는 방향을 의미한다.

가변점은 'VP'로, 가변치는 'VP.V'로써 표시한다. 즉, 'i'번째 가변점은 'VP_i' 로 표기되며, 'x' 번째 가변치는 '$VP_i.V_x$.' 로써 사용된다.

- 타입 1. $VP_i \rightarrow VP_j$
  가변점 'i'가 바운드되면, 가변점 'j'가 바운드 되어야 한다.
- 타입 2. $VP_i \rightarrow VP_j.V_n$
  가변점 'i'가 바운드되면, 가변치 'n'이 바운드 되어야 한다.
- 타입 3. $VP_i \rightarrow \neg VP_j.V_n$
  가변점 'i'가 바운드되면, 가변치 'n'은 바운드 되어서는 안된다.
- 타입 4. $VP_i.V_m \rightarrow VP_j$
  가변치 'm'이 바운드되면, 가변점 'j'는 바운드 되어야 한다.
- 타입 5. $VP_i.V_m \rightarrow VP_j.V_n$
  가변치 'm'이 바운드되면, 가변치 'n'이 바운드 되어야 한다.



(그림 2) 가변성간 의존성 타입

〈표 3〉 가변성간 의존성 명세서

의존성 번호	의존성 타입	영향을 주는 가변성		영향을 받는 가변성	
		가변점	가변치	가변점	가변치
	타입 1	VPi		VPj	
	타입 4	VPi	Vm	VPj	
…	…	…		…	

• 타입 6. $VP_i.V_m \rightarrow \neg VP_j.V_n$
가변치 'm'이 바운드되면, 가변치 'n' 은 바운드 되어서는 안된다.

<표 3>의 **가변성간 의존성 명세서**는 의존성 타입, 영향을 주는 가변성, 영향을 받는 가변성에 대해서 명세하기 위해서 사용한다.

이와 같은 방법으로 가변성간 의존성을 분석해서, 가변성간 충돌이나 불일치와 같은 문제를 사전에 예방할 수 있다.

[활동 4] 도메인 모델 정제

도메인 모델을 정제하기 위한 활동은, 활동 1부터 활동 3까지의 산출물인 **공통성 명세서**, **가변성 명세서**, **가변성간 의존성 명세서**를 검증하고 정제하기 위해서 수행한다.

산출물을 정제하기 위해서는 공식적인 기술 검토(formal technical review)와 같은 기술이나 체크 리스트를 사용하면 유용하다. 예를 들어, 체크리스트에 있는 질문들을 사용해서 SRS에 있는 휘처가 정확히 분석되었는지 확인한다. 만일 체크 리스트에 있는 질문에 대답하는데 주저한다거나 답할 수 없다면, 그 휘처를 확인하고 다시 분석해야 한다.

명세를 정제하기 위한 체크리스트의 항목 중 일부를 기술하면 다음과 같다. 한 가변점은 최소 두 개 이상의 가변치를 가진다. 각 가변성의 가변치는 둘 이상의 멤버에서 추출되어야 하며, 멤버들은 가변점에서 반드시 하나 이상의 가변치를 가져야 한다.

[활동 5] 프로덕트 라인 공학을 위한 비즈니스 케이스 분석

다섯 번째 활동은 PLE 기반으로 제품을 개발하기 위해서 비즈니스 케이스를 분석하며, 가장 합리적이며 재사용성을 최대화 하는 핵심 자산의 영역을 선택하기 위해서 수행한다. **공통성 명세서**, **가변성 명세서**, **가변성 의존성 명세서**가 이 활동의 입력물로 사용된다.

PLE 기반으로 제품을 개발하기 위해서 비즈니스 케이스를 분석하고, 재사용성을 최대화하는 핵심 자산의 영역을 선택하기 위해서는 다음과 같은 메트릭이 사용된다.

- 비용(핵심 자산) - 핵심 자산을 실체화 하기 위한 비용
- 비용(가변성 없는 휘처) - 가변성이 없는 모든 휘처를 핵심 자산으로 실체화 하기 위한 비용
- VP_i - i 번째 가변점
- VP_iV_j - i 번째 가변점의 j 번째 가변치
- $nv(VP_i)$ - i 번째 가변점에 있는 가변치의 수

〈표 4〉 가변성간 의존성 정도

가변성 의존성	의존성 정도
$VP_i \rightarrow VP_j$	1
$VP_i \rightarrow VP_j.V_n$	$1/\ nv(VP_j)$
$VP_i \rightarrow \neg VP_j.V_n$	
$VP_i.V_m \rightarrow VP_j$	
$VP_i.V_m \rightarrow VP_j.V_n$	$1/\ nv(VP_i) * 1/\ nv(VP_j)$
$VP_i.V_m \rightarrow \neg\ VP_j.V_n$	

- VD_k - k 번째 가변성간 의존성
- DE_m - m 번재 가변성간 의존성의 정도
- DW_n - n 번째 가변성간 의존성의 가중치
DW_n은 가변점과 가변치의 크기에 따라서 가변성간 의존성을 실체화 하는 비용에 차이가 발생하기 때문에 고려한다. DW는 0과 1 사이 값으로써, 도메인 전문가 등의 시스템 분석가에 의해서 계산된다.
- 비용(VP_i) - i 번째 가변점을 실체화 하기 위한 비용
가변치를 추가 또는 삭제 할 수 있는 오픈 가변성은 이 메트릭에서는 고려하지 않는다.
- 비용(VP_iV_j) - i 번째 가변점의 j 번째 가변치를 실체화 하는 비용
- 비용(VD_k) - k 번째 가변성 의존성을 실체화 하기 위한 비용
비용(VD_k)는 비용(VP_i), DE_m, DW_n값에서 유도되는 비용이다.
- 비용(가변적 휘처) - 가변적인 휘처를 실체화 하기 위한 비용
비용(가변적 휘처)는 비용(VP_i), 비용(VP_iV_j), 비용(VD_k)에서 유도되는 비용이다.

본 모델에서는 DE_m 이 <표 4>와 같이 계산된다.

가변성간 의존성 정도를 기반으로, 가변성간 의존성 실체화 비용인 비용(VD_k)을 다음과 같은 공식으로 계산한다.

$$\textbf{비용(VDk)} = \text{비용}(VP_i) * (1/\ DE_m * DW_n)$$

앞에 정의된 기본 공식을 기반으로, 가변성 실체화 비용을 위한 공식이 다음과 같이 정의된다.

$$\textbf{비용(가변적 휘처)} = \sum_{i=1}^{n} \text{비용}(VP_i) + \sum_{i=1}^{m}\sum_{j=1}^{n} \text{비용}(VP_iV_j) + \sum_{i=1}^{n} \text{비용}(VD_k)$$

가변적인 휘처를 개발하기 위한 비용은, 가변점을 실체화 하기 위한 비용, 가변치를 실체화 하기 위한 비용, 가변성간 의존성을 실체화 하기 위한 비용의 합으로 계산된다.

핵심 자산을 개발하기 위한 비용은 비용(가변성 없는 휘처)과 비용(가변적 휘처)의 합으로 계산된다.

$$\textbf{비용(핵심 자산)} = \text{비용(가변성 없는 휘처)} + \text{비용(가변적 휘처)}$$

만일 핵심 자산의 영역이 너무 광범위하면, 많은 멤버들이 핵심 자산을 사용할 수 있지만 극히 일부의 자산만을 사용할 것이다. 반대로 핵심 자산의 영역이 너무 좁으면, 적은 수의 멤버가 핵심 자산을 사용할 것이다. 즉 핵심 자산의 영역이 너무 크거나 너무 작을 경우 모두 핵심 자산의 재사용성은 극대화 되지 않는 경향이 있다.

기대 수익을 최대화하는 핵심 자산의 영역은 다음과 같은 메트릭을 적용해서 확인할 수 있다.

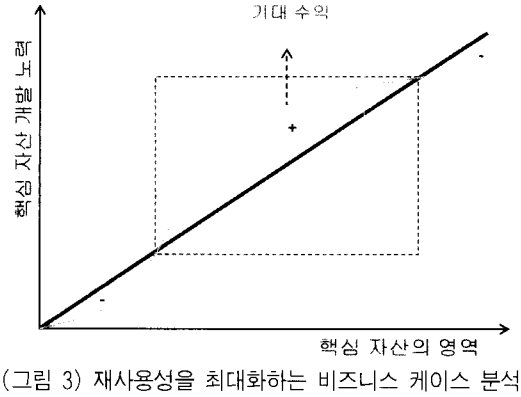
- n비용(PLE를 적용하지 않은 개발) = 핵심 자산을 사용하지 않고 n개의 애플리케이션을 개발하기 위한 비용
- n비용(핵심 자산을 사용한 개발) = 개발된 핵심 자산을 사용해서 n개의 애플리케이션을 개발하기 위한 비용

위에 정의된 메트릭을 사용한 다음과 같은 공식은 프로덕트 라인 공학을 적용하기 위한 비즈니스 케이스 분석에 사용될 수 있다.

기대 수익 = n비용(PLE를 적용하지 않은 개발)
- {비용(핵심 자산) + n 비용(PLE를 적용한 개발)}

이 공식을 적용해서, 핵심 자산을 사용해서 n개의 시스템을 개발하기 위한 비용과 핵심 자산을 사용하지 않고 n개의 시스템을 개발하기 위한 비용을 비교한다. 그리고 PLE를 적용한 개발 비용이 최대의 이익을 낼 수 있도록 핵심 자산의 영역을 조절해서 기대 수익이 가장 큰 값을 갖도록 해야 한다. 비즈니스 케이스 분석을 통해서, PLE 기반으로 제품을 개발했을 경우에 획득할 수 있는 이윤이 최대가 되도록 하기 위해서는 핵심 자산의 영역을 변경한 후 핵심 자산을 개발하는 비용을 여러 번 계산해서 최대의 수익을 얻을 수 있도록 비용을 조절해야 한다.

핵심 자산의 영역이 증가하면, 핵심 자산을 개발하는 비용 역시 증가한다. 그러나 핵심 자산을 개발하는 비용과 기대 수익은 정비례 관계에 있지 않다. 그러므로 가장 큰 수익을 얻기 위한 핵심 자산의 영역은 (그림 3)의 점선 사각형 내부 값과 같이 투자 비용보다는 기대 수익이 큰 범위에서 선택되어야 한다.



(그림 3) 재사용성을 최대화하는 비즈니스 케이스 분석

만약 도메인이 충분히 분석되지 않았다면, 기대 수익이 너무 작을 수 있으며 또는 기대 수익에 만족하지 못할 수도 있다. 이와 같은 경우에는 공통성 분석이 다시 수행되어야 한다. 그러나 여러 번 계산을 했지만 핵심 자산을 적용하지 않고 애플리케이션을 개발하는 비용이 PLE를 적용해서 애플리케이션을 개발하는 비용보다 적게 소비될 것으로 예상되면, PLE를 적용하지 않을 수도 있다.

핵심 자산의 영역은 비즈니스 케이스 분석 결과로 결정되며, 핵심 자산 개발자들은 이 영역에서 선택된 휘처들을 개발할 것이다.

4. 사례 연구

본 절에서는 정통부 산하의 컴포넌트 컨소시엄에서 시행했던 CBD 시범적용 프로젝트의 요구사항과 분석 모델을 기반으로 파일럿 프로젝트를 수행해서 본 논문에서 제안한 기법을 적용한 사례연구를 다룬다. 도메인은 대여(Rental) 업무이며, 다음의 세 개 제품 요구사항들을 기반으로 사례 적용을 한다.

- 베스트 비디오 대여점 (Best Movie Rental, BMR)
- 드림 도서관 (Dream Library, DL)
- 월드 자동차 대여사 (World Car Rental Corporation, WCRC)

대여 도메인은 대여, 회원 관리, 회계, 대여 예약의 네 개의 서브 도메인으로 분류되며, 이 장의 사례 연구는 대여 서브 도메인만을 보여준다.

[활동 1] 공통성 분석

SRS는 표준화된 용어로 재작성 된 후에, 공통성 분석에 사용된다.

<표 5>는 대여 도메인에 대한 사례 연구의 일부이며 멤버간 공통성을 비교하기 위해서 수행되었다. SRS를 사용해서 서브 도메인의 휘처를 분석한다. 대여 서브 도메인에 있는 휘처들은 '아이템 대여', '대여 정보 검색', '포인트 합산', '지난 대여 기록 검색' 등과 같이 열거된다. 그리고 각 멤버별로 개발 예정 휘처에 표기한다. WCRC 멤버가 '대여 아이

〈표 5〉 대여 서브 도메인의 공통성 명세서

공통 번호	서브 도메인	공통 휘처	멤버			공통성 정도	의사 결정 (Y/N)
			BMR	DL	WC RC		
…							
15	대여	1. 아이템 대여	√	√	√	3/3	Y
16		2. 대여 정보 검색	√	√	√	3/3	Y
17		3. 포인트 합산	√			1/3	N
18		4. 지난 대여 기록 검색	√		√	2/3	Y
19		…	…	…	…	…	…
…							

〈표 6〉 대여 서브 도메인의 가변성 명세서

가변성 번호	공통 휘처	가변점	가변성 타입	바인딩 타입	가변치 (V)			기본 가변치
					1. BMR	2. DL	3. WCRC	
…								
21	아이템 대여	대여 정보	속성	선택	대여아이디, 멤버아이디, 아이템아이디, 대여일, 대여기간, 반납일, 대여료	대여아이디, 멤버아이디, 아이템아이디, 대여일, 대여기간, 반납일, 대여료	대여아이디, 멤버아이디, 아이템아이디, 대여일, 보험 대여기간, 반납일, 대여료	대여아이디, 멤버아이디, 아이템아이디, 대여일, 대여기간, 반납일, 대여료
22		대여 흐름	워크 플로우	선택	멤버쉽 체크 → 이전 대여기록 체크 → 연체 기록 체크 → 대여료 계산 → 대여 아이디 생성 → 저장	멤버쉽 체크 → 이전 대여기록 체크 → 연체 기록 체크 → 대여료 계산 → 대여 아이디 생성 → 저장	멤버쉽 체크 → 이전 대여기록 체크 → 대여료 계산 → 대여 아이디 생성 → 저장	멤버쉽 체크 → 이전 대여기록 체크 → 연체 기록 체크 → 대여료 계산 → 대여 아이디 생성 → 저장
	…							
…								

〈표 7〉 대여 서브 도메인의 가변성간 의존성 명세서

의존성 번호	의존성 타입	영향을 주는 가변성		영향을 받는 가변성	
		가변점	가변치	가변점	가변치
…					
31	타입 5	VP_{21}	V_1	VP_{22}	V_1
32	타입 5	VP_{21}	V_2	VP_{22}	V_2
…					
38	타입 5	VP_{22}	V_3	VP_{23}	V_3
39	타입 1	VP_{24}		VP_{25}	
…					

템', '대여 정보 검색', '지난 대여 기록 검색' 기능을 요구하므로 '√' 기호를 사용해서 체크한다. 또한 얼마나 많은 SRS가 각 휘처를 요구하는지 공통성의 정도를 확인한 후에 공통 휘처로 결정한다. 이와 같은 방법으로 '아이템 대여', '대여 정보 검색', '지난 대여 기록 검색' 등의 기능이 공통 휘처로 결정되었다.

[활동 2] 가변성 분석

가변성은 공통성의 부분으로써 멤버간 차이가 발생하는 부분이다. 그래서 '아이템 대여'와 같은 공통성 내에서 식별된다. 가변성을 식별하기 위해서 <표 6>과 같은 가변성 명세서를 사용한다.

'아이템 대여' 공통 휘처에서는 '대여 정보', '대여 흐름' 등의 가변점이 식별되었고, 각 가변점 별로 가변성 타입과 바인딩 타입이 결정되었다. 21번째 가변점의 가변성 타입은 속성 타입이며, 바인딩 타입은 선택 타입이다. 그리고 하나의 가변점에서 세 개의 가변치가 SRS를 기반으로 작성된다. 또한 기본 가변치가 선택된다.

[활동 3] 가변성간 의존성 분석

<표 7>은 대여 도메인에서 가변성간 의존성을 분석하기 위해서 작성된다. <표 7>에서 31번째 가변성간 의존성은 타입 5번에 해당하며, VP_{21}번 가변점의 V_1 가변치는 VP_{22}번 가변점의 V_1 가변치에 영향을 준다. 32 번째 가변성간 의존성은 타입 5번에 해당하며, VP_{21}번 가변점의 V_2 가변치는 VP_{22} 가변점의 V_2 가변치에 영향을 준다.

[활동 4] 도메인 모델 정제

도메인 모델 정제는 SRS가 충분히 분석되었는지 또는 잘못 분석된 내용이 없는지를 확인하기 위해서 수행된다.

다음과 같은 질문은 체크리스트의 일부이며, 모델 정제를 위해서 체크리스트에 있는 질문에 대답한다.

- SRS에 있는 모든 기능성이 공통성 분석 활동 동안에 고려되었나?
- 모든 휘처가 의사 결정 규칙에 준거해서 공통성으로 선택되었나?
- 가변성은 공통 휘처 내에서 식별되었나?
- 모든 가변점이 2개 이상의 가변치를 포함하나?
- 선택된 기본 가변치가 가변성과 충돌되는 부분은 없는가?

[활동 5] 프로덕트 라인 공학을 위한 비즈니스 케이스 분석

활동 5에서는 대여 도메인에서 PLE를 적용하기 위한 비즈니스 케이스를 분석하고, 가장 경제성이 좋은 핵심 자산의 영역을 결정한다. 영역을 결정하기 위해서는 위에서 분석한 자료를 기반으로 몇 가지 사례를 조사한 후에, 비용이나 노력 측면에서 가장 타당한 사례를 선택한다. 핵심 자산으로 개발 가능한 몇 가지 영역을 조사한 후, 가장 합리적

〈표 8〉 영역으로 결정된 기능적 요구사항

기능성 번호	서브 도메인	기능적 요구사항
FR_Inv01	제품 관리	제품 등록
FR_Inv02		제품 삭제
FR_Inv03		제품 검색
…		…
FR_Mem01	멤버쉽 관리	멤버 정보 등록
FR_Mem02		멤버 정보 수정
FR_Mem03		VIP로 멤버 업데이트
…		…
FR_Ren01	대여 관리	제품 대여
FR_Ren02		대품 반납
…		…
FR_Res01	예약 관리	제품 예약
…		…

〈표 9〉 영역으로 결정된 가변적 요구사항

가변성 번호	서브 도메인	기능적 요구사항	가변점
FVP_Inv01	제품 관리	FR_Inv01. 제품 관리	제품 정보
FVP_Inv02			제품 등록 로직
…		…	…
FVP_Mem01	멤버쉽 관리	FR_Mem01. 멤버쉽 관리	멤버 아이디 생성 로직
…		…	…
FVP_Ren01	대여 관리	FR_Ren01. 대여 관리	대여의 로직 흐름
…		..	..

〈표 10〉 영역으로 결정된 가변성간 의존성

의존성 번호	가변성 번호	영향을 주는 가변성 (VPiVj)	영향을 받는 가변성 (VPiVj)
VD_01	FVP_Inv01	FVP_Inv01.v1	FVP _Inv02.v1
VD_02	FVP_Inv01	FVP _Inv01.v2	FVP _Inv02.v2

이라고 판단되는 핵심 자산의 영역은 다음과 같았다.

결정된 핵심 자산의 영역 중 기능적 요구사항은 <표 8>과 같다.

결정된 핵심 자산의 영역 중 가변적 요구사항은 <표 9>와 같다.

결정된 핵심 자산의 영역 중 가변성간 의존성 요구사항은 <표 10>과 같다.

결정된 영역을 개발하기 위한 비용을 계산하기 위해서는 본 사례연구서는 pd(person-days) 노력 단위를 사용한다. 이는 각 회사가 가지고 있는 비용 자료를 이용해서 실제 비용으로 쉽게 변환될 수 있다.

BMR, DL, WCRC 세 개의 애플리케이션을 단독으로 개발하기 위한 노력은 각각 5pd씩 총 15pd의 노력이 소요될 것으로 예상된다.

대여 도메인에서 제품 삭제, 아이템 삭제, 대여 정보 검색 등 가변적이지 않은 공통 휘처는 총 12개가 있으며, 실체화하는 노력을 합산하면 4pd가 예상된다.

가변적인 휘처를 실체화 하기 위한 노력은 다음과 같이 확인된다. 우선 가변점은 총 29개가 있으며, 29 개의 가변점을 실체화 하기 위한 노력은 1.5pd가 소요된다. 가변치는 총 72개가 있으며, 72개의 가변치를 실체화 하기 위한 노력은 1.7pd가 소요된다. 또한 가변성간 의존성을 실체화하기 위한 노력은 1.9pd가 소요된다. 즉, 가변적인 휘처를 실체화 하기 위한 노력은 총 5.1pd가 소요된다.

또한 실체화된 핵심 자산을 기반으로 BMR, DL, WCRC 총 세 개의 애플리케이션을 개발하는 노력은 각각 2pd씩 총 6pd의 노력이 소요될 것으로 예상된다. 즉, PLE 방법론을 적용해서 세 개의 애플리케이션을 개발하기 위한 노력은 총 11.9pd의 노력이 소요될 것으로 예상된다.

사례 연구 결과, 핵심 자산을 사용하지 않고 세 개의 애플리케이션을 개발하는 노력은 핵심자산을 사용해서 애플리케이션을 개발하는 것보다 많은 노력이 든다는 결론이 나왔다. 즉, 프로덕트 라인 공학을 위한 비즈니스 케이스 분석 결과, 이 핵심 자산의 영역을 개발하는 비용과 개발된 핵심 자산을 사용해서 시스템을 개발하는 비용이 가장 큰 경제성을 줄 수 있는 비용으로 결정되었다.

5. 결 론

PLE 방법론은 제품 계열 멤버들이 공통적으로 가지고 있는 휘처를 효과적으로 재사용하기 위한 방법론으로써, 개발된 핵심 자산의 재사용성을 최대화 하기 위해서는 핵심 자산의 영역 설정이 중요하다.

이 논문에서는 우선 PLE 방법론을 적용하기 위한 비즈니스 케이스를 분석하는 프로세스를 제안하고 핵심 자산의 영역을 분석하며, 사례 연구를 보여준다. 또한 프로세스의 각 활동을 위한 가이드 라인과 산출물 양식을 제안하며, 비즈니스 케이스를 분석하는 과정에서 가변성과 가변성 의존성이 고려된다는 점이 특징적이다.

이 논문에서 제안한 프로세스는 가이드라인과 산출물을 구체적으로 제안하고 있으므로 프로젝트에 쉽게 적용할 수 있다고 생각되며, 개발자들은 우리가 제안한 비즈니스 케이스 분석 방법과 핵심 자산의 영역을 설정하는 프레임 워크를 적용해서 가장 효과적인 가치를 제공하는 핵심 자산의 영역을 개발할 것으로 기대한다.

우리는 프로덕트 라인 공학을 위한 비즈니스 케이스 분석 방법을 사례 연구에 적용하면서 몇 가지 교훈을 얻었다.

- PLE를 위한 비즈니스 케이스를 분석하기 위해서는 핵심 자산의 영역을 여러 번 변경해서 최적의 핵심 자산 개발 비용을 계산한 후에, PLE를 적용하지 않은 애플리케이션 개발에 소비되는 투자액과 비교해야 한다.
- 투자 비용이 기대 수익을 항상 보장하는 것은 아니다. 핵심 자산의 영역이 너무 광범위하면 많은 멤버가 핵심 자산을 적용할 수는 있지만, 적용할 수 있는 자산의

범위는 작아진다. 반대로 핵심 자산의 영역이 너무 협소하면, 일부 멤버만이 핵심 자산을 사용할 수 있을 것이다.

- 멤버간 가변성과 가변성간 의존성을 실체화 하는 비용 역시 PLE 방법론을 적용하기 위한 비즈니스 케이스 분석(핵심 자산을 실체화 하는 비용)에 포함될 수 있다.
- 가변성간 의존성의 정도는 가변성 타입에 따라 달라지며, 가변성을 실체화하는 비용에 영향을 준다.

참 고 문 헌

[1] D. M. Weiss, C. T. R. Lai, Software Product-Line Engineering: A Family-Based Software Development Process, Addison-Wesley, 1999.

[2] Clements, P. and Northrop, L., Software Product Lines: Practices and Patterns, Addison Wesley, Aug., 2001.

[3] Bosch, J., Design and use of software architectures, Addison-Wesley, 2000.

[4] Schmid, K., Scoping, Fraunhofer IESE, Analysis of, and Modeling for System-Families, June., 8, 2001.

[5] Choi, C., et al, "A Systematic Methodology for Developing Component Frameworks," M. Wermelinger and T. Margaria-Steffen(Eds.), FASE 2004, LNCS 2984, pp.359 - 373, 2004.

[6] Soo Dong Kim, Jin Sun Her, and Soo Ho Chang, "A Theoretical Foundation of Variability in Component-Based Development," Information and Software Technology (IST), Vol.47, p.663-673, July, 2005.

[7] Sinnema, M., et al., "COVAMOF: A Framework for Modeling Variability in Software Product Families," SPLC2004, LNCS3154, pp.197-213, 2004.

박 신 영

e-mail : sypark@otlab.ssu.ac.kr

1999년 3월~2003년 2월 동덕여자대학교 국어국문학과, 데이터정보학과 복수전공(학사)

2004년 3월~2006년 2월 숭실대학교 대학원 컴퓨터학과(공학석사)

2006월 3월~현재 현대정보기술 재직

관심분야 : 컴포넌트 기반 개발(CBD), 제품 계열 공학(PLE), 모델 기반 아키텍처(MDA)

김 수 동

e-mail : sdkim@comp.ssu.ac.kr

1984년 Northeast Missouri State University 전산학(학사)

1988년/1991년 The University of Iowa 전산학(석사/박사)

1991년~1993년 한국통신 연구개발단 선임연구원

1994년~1995년 현대전자 소프트웨어연구소 책임연구원

1995년 9월~현재 숭실대학교 컴퓨터학부 부교수

관심분야 : 객체지향 S/W공학, 컴포넌트 기반 개발(CBD), 제품계열 공학(PLE), 모델 기반 아키텍처(MDA)