# 유스케이스 확장 관계의 개선 모델링 기법

조준수*, 정기원**
*SK C&C 개발담당
**숭실대학교 컴퓨터학부
e-mail: bcto@naver.com

# An Improved Modeling Technique of Use Case Extend Relationship

Junsoo Cho*, Kiwon Chong**
*Dept. of Development, SK C&C
**School of Computing, Soongsil University

요 약

확장 관계는 유스케이스 수행 중 조건에 따라 기반 유스케이스의 기능을 확장하고자 할 때 활용된다. 그러나 현재 확장 관계가 조건에 따른 분기를 위해 활용하는 확장점은 레이블 형태를 가지며, 이는 객체지향 프로그램의 작성을 어렵게 만드는 요인이 된다. 또한 확장기능 수행 후 기반 유스케이스로의 복귀점에 대한 명확한 모델링 기준이 부족하여 확장 관계의 활용이 어려운 것이 현실이다. 본 논문에서는 유스케이스 확장 관계의 개선 모델링 기법을 제시한다. 개선 모델링은 확장 관계의 분기점으로 기존 레이블 형태의 확장점 대신 액티비티 노드를 설정한다. 또한 분기점과 쌍으로 복귀점에 해당하는 액티비티 노드를 명시적으로 지정 가능하다. 이를 통해 확장 관계의 분기 및 복귀의 보다 명확한 명세가 가능하다.

## 1. 서론

유스케이스(Use Case)는 시스템의 기능을 표현하기 위한 수단으로 널리 사용되고 있으며, 사용자 요구사항을 매우 효과적으로 표현할 수 있다[2][5]. 유스케이스 간 관계 설정을 위해서는 포함(Include) 및 확장(Extend) 관계가 활용된다.

확장(<<extend>>) 관계는 유스케이스 수행 중 조건에 따라 기반 유스케이스(Base Use Case)의 기능을 확장하고자 할 때 활용된다. 그러나 현재 확장 관계가 조건에 따른 분기를 위해 활용하는 확장점(Extension Point)은 레이블(Label) 형태를 가지며, 이는 객체지향 프로그램의 작성을 어렵게 만드는 요인이 된다. 또한 포함 관계와 달리 확장기능 수행 후 기반 유스케이스로의 복귀점(Rejoin Point)에 대한 명확한 모델링 기준이 부족하여 확장 관계의 활용이 어려운 것이 현실이다[3][4].

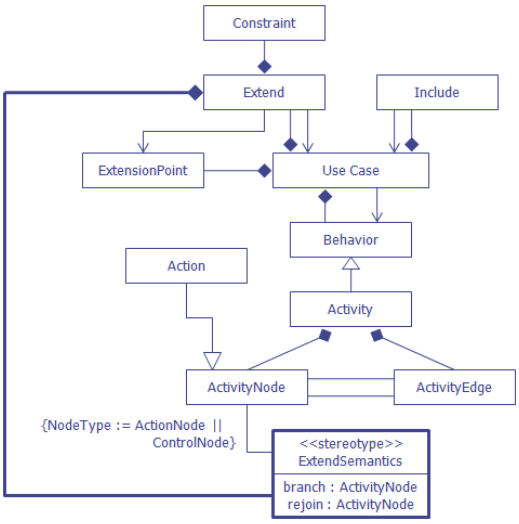
본 논문에서는 유스케이스 확장 관계의 개선 모델링 기법을 제시한다. 개선 모델링은 기존의 확장점 대신 액티비티 노드(Activity Node)를 활용하며, 확장 관계가 갖는 분기점 및 복귀점은 액티비티 노드의 집합인 액티비티 그룹(Activity Group)을 통해 정의된다.

## 2. 관련 연구

Metz 등은 확장 관계의 복귀점과 관련한 모호성 제거를 위하여 확장하는 유스케이스의 실행 후 기반 유스케이스로 복귀하는 경우를 5개의 경로로 세분하여 제시하였다. 그러나 이들은 여전히 복귀점과 관련하여 확장점을 그대로 활용함으로써 확장점과 관련한 문제점을 해결하지 못하고 있으며, 복귀점에 대한 명확한 모델링 기법을 제시하지 않고 있다[4].

Braganca 등은 [4]를 기반으로 기존 유스케이스 모델에 메타모델을 추가 적용한 모델링 기법을 제안하였다. 이들은 유스케이스의 개념적 단편으로서 "ExtensionFragment"를 제안하고, "RejoinPoint" 등을 통해 확장 관계의 명확한 모델링을 주장하였다. 그러나 메타모델의 추가는 모델의 복잡성을 증가시키는 요인이 되며, 특히 "ExtensionFragment"와 같이 새로운 개념의 도입은 기존 모델과의 호환성 문제를 야기한다[1].

## 3. 확장 관계의 개선 모델링 기법




(그림 1) 개선된 유스케이스 메타모델

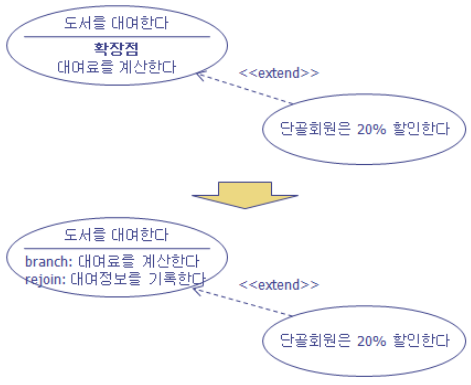
확장 관계의 개선 모델링을 위한 유스케이스 메타모델은 그림 1과 같다. 기존 유스케이스 모델로부터 확장된 요소는 굵은 선으로 표시되어 있다. 분기점과 복귀점을 표현하기 위하여 액티비티 그룹(ActivityGroup) 메타모델을 확장한 <<ExtendSemantics>> 스테레오타입(Stereotype)을 정의한다.

<<ExtendSemantics>> 액티비티 그룹은 액티비티 노드(Node)를 갖는다. 노드는 액션 노드이거나 컨트롤 노드이며, 객체 노드는 허용되지 않는다. 또한 기존 액티비티 그룹과 달리 액티비티 에지(Edge)를 갖지 않는다. 이는 <<ExtendSemantics>> 액티비티 그룹이 분기점 및 복귀점만을 명시하기 위해 사용되기 때문이다.

<<ExtendSemantics>>로 정의된 액티비티 그룹은 두 개의 꼬리표(Tag)를 갖는다. 하나는 분기점 지정을 위한 "branch"이며, 다른 하나는 복귀점을 위한 "rejoin"이다. 이들은 모두 액티비티 노드 형태로 정의된다.

모든 확장 관계는 <<ExtendSemantics>> 액티비티 그룹을 가지며, 기존의 확장점은 활용하지 않는다. 따라서 확장에 따른 분기점과 복귀점이 명확하게 정의되며, 복귀 경로의 수 및 복귀 위치 등과 관계없이 복귀점을 명확하게 정의할 수 있다.

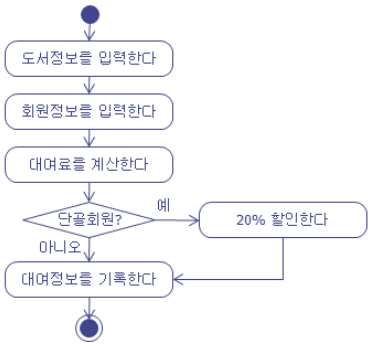
그림 2는 <<ExtendSemantics>> 액티비티 그룹을 적용하여 유스케이스를 모델링한 사례를 보이고 있다. 그림에서 상단의 모델은 기존 확장점을 이용하여 표현된 모델이며, 확장을 위한 분기점은 레이블 형태를 갖는다. 또한 복귀점에 대해서는 명확히 정의되어 있지 않다. 반면 하단 모델은 확장 유스케이스의 분기 및 복귀 지점이 기반 유스케이스에 명확하게 정의되어 있음을 볼 수 있다.




(그림 2) 개선된 유스케이스 모델 (예시)

분기점은 조건이 충족되는 경우 기반 유스케이스로부터 분기하기 직전의 액티비티를 나타내며, 복귀점은 확장 유스케이스 실행 후 기반 유스케이스로 복귀한 다음 실행하게 될 액티비티를 지정한다. 따라서 순차적인 실행순서는 "액티비티(branch) -> 확장 유스케이스 -> 액티비티(rejoin)"가 된다.

그림 3은 "도서를 대여한다" 유스케이스를 명세한 것으로, 분기점과 복귀점 정의에 따라 "대여료를 계산한다 -> 단골회원의 경우, 20% 할인한다 -> 대여정보를 기록한다" 순으로 실행순서가 명확하게 명세됨을 알 수 있다.




(그림 3) "도서를 대여한다" 유스케이스 명세

## 4. 결론

유스케이스 확장 관계는 조건에 따라 기반 유스케이스의 기능을 확장하고자 할 때 활용 가능한 메커니즘이며, 특히 프로덕트 라인(Product Line) 등에서 시스템 간 가변성(variability)을 모델링 할 때 매우 유용하게 활용 가능하다. 그러나 현재 기반 유스케이스로의 복귀점과 관련한 모호성이 존재하고, 또한 레이블 형태의 확장점의 활용은 객체지향 프로그램 작성을 어렵게 만드는 요인이 된다.

본 논문에서 제안하는 확장 관계의 개선 모델링은 확장 관계의 분기점으로 기존 레이블 형태의 확장점 대신 액티비티 노드를 설정한다. 또한 분기점과 쌍으로 복귀점에 해당하는 액티비티 노드를 명시적으로 지정 가능하다. 이를 통해 확장 관계의 분기 및 복귀의 보다 명확한 명세가 가능함은 물론, UML 스테레오타입 확장 메커니즘을 활용함으로써 기존 유스케이스 모델과의 호환성을 그대로 유지 가능하다.

## 참고문헌

[1] Braganca, Alexandre and Machado, Ricardo J., "Extending UML 2.0 Metamodel for Complementary Usages of the <<extend>> Relationship within Use Case Variability Specification", SPLC'06, 2006

[2] Cockburn, Alistair, "Writing Effective Use Cases", Addison-Wesley, 2000

[3] Lilly, Susan, "Use Case Pitfalls: Top 10 Problems from Real Projects Using Use Cases", Proceedings of the Technology of Object-Oriented Languages and Systems, 1999

[4] Metz, Pierre, O'Brien, John and Weber, Wolfgang, "Specifying Use Case Interaction: Types of Alternative Courses", Journal of Object Technology, Vol. 2, No. 2, Mar 2003

[5] Rumbaugh, James, Jacobson, Ivar, and Booch, Grady, "The Unified Modeling Language: Reference Manual", 2nd Ed., 2005