UML의 주요 다이어그램들 간의 일관성 유지 기준

박지환 김수동
숭실대학교 컴퓨터학과
jihwan@sclab.soongsil.ac.kr sdkim@comp.ssu.ac.kr

Criteria for Maintaining Consistency Among UML Diagrams

Ji Hwan Park, Soo Dong Kim
School of Computing, Soongsil University

요 약

주어진 도메인에 대한 설계 및 구현 이전 단계인 분석 과정에서 객체지향의 방법을 적용할 수 있는 모델링 언어인 UML(Unified Modeling Language) 다이어그램 중 Use Case 다이어그램, 클래스 다이어그램, 순서도 등 3 가지 모델들 간의 일관성(consistency)을 유지하기 위한 기준이나 구체적인 지침에 관한 여러 가지 형태의 연구들이 이미 행해져 왔지만, 본 논문에서는 이러한 다이어그램들 간의 일관성을 유지하는데 있어서 지켜져야 하는 기준과 checklist를 예제를 통해서 제시한다.

1. 서 론

소프트웨어의 개발을 하는데 있어서 주어진 도메인에 대한 시스템을 개발할 때에 개발자들로 하여금 객체지향의 개념을 이용하여 분석 및 설계를 할 때 적용되는 것이 UML(Unified Modeling Language)이다. UML 의 전체 9 개 다이어그램 중 분석 및 설계 시에 사용되는 주요한 3 가지 다이어그램 즉, Use Case 다이어그램, 클래스 다이어그램 그리고 순서도는 핵심이 되는 다이어그램 들이다. 그런데 분석 단계에서의 이 세 가지 다이어그램을 적용하는데 있어서 주의할 것이 각 다이어그램 간의 일관성을 유지하는 것이다.

각 다이어그램들은 서로가 상호 밀접한 관계들을 가지고 있으며, 각 다이어그램의 세부적인 요소들은 다른 다이어그램에서 특정한 요소들로 매핑이 된다. 그래서 본 논문에서는 이러한 다이어그램들 간의 중요한 요소들을 다른 다이어그램에서 어떠한 요소들로 정확히 매핑 시키는지에 관한 기준을 예제를 통해서 제시하고자 한다.

2. 관련 연구

2.1. Use Case Diagram

Use Case 다이어그램은 전체 시스템을 클라이언트의 요청에 따라 시스템을 사용하려는 사용자가 시스템에 대하여 요청하는 기능들을 Use Case 라는 단위로 분할하여 각 기능들을 정의함으로써 시스템에 대한 전반적인 이해를 높이고, 문제 영역에 대해 개발자와 사용자 간의 의사 소통을 원활하게 하는데 도움을 준다. Use Case 다이어그램에서의 Actor 는 시스템, 서브시스템 또는 클래스와 상호 작용하는 외부의 사람, 프로세스 또는 어떠한 것이며, Use Case 는 하나 또는 그 이상의 Actor 에 의해 요청되어 시스템이 제공하게 되는 기능 자체를 나타낸다 [1].

Rational Unified Process(RUP)에서는 Use Case 로부터 시작하여 그 다음 개발 과정인 분석, 설계 그리고 구현 단계에 이르기까지, 각 단계 마다 이전 단계의 산출물 들로써 다음 단계에 적용시킨다. 그리하여, 좀 더 작은 서브시스템 또는 전 단계를 이루는 상세한 여러 가지 작은 요소들로 Traceability 를 통하여 그 구성을 세분화 시키면서 각 개발단계를 거치면서 일관성이 유지되도록 하고 있다.

2.2. Class Diagram

클래스 다이어그램은 Use Case 다이어그램을 통하여 나타냈던 수행하고자 하는 기능들을 위해 꼭 필요한 데이터와 연관된 함수들을 각각의 독립된 클래스에 나타내고 그들 클래스들 사이의 관계를 정적인 관점에서 나타낸 것이다. 클래스 다이어그램의 오퍼레이션은 어떤 클래스의 객체에 대한 특정한 행동을 나타내는 것이며 이 오퍼레이션을 통해서 객체에 대하여

어떠한 일들을 수행할 수 있는지를 알 수 있다[2].

속성은 클래스의 인스턴스인 객체들간에 서로 공유할 수 있는 데이터에 해당되는 것이다. 마지막으로 클래스와 클래스간의 의미적인 연결관계를 나타내는 Relationship 에는 세 가지가 있는데, 첫째 의존관계(Dependency)는 관계가 어느 한 방향에 대해서만 존재하는 것을 나타내며 둘째 상속관계(Generalization)는 Supertype 의 기능을 Subtype 에서 대체할 수 있도록 한 것이며 셋째, 연관관계(Association)는 각 클래스들간의 복잡도나 어떠한 역할을 가지고 서로간의 관계들을 형성하는지를 나타낸다.

2.3. Sequence Diagram

순서도는 클래스 다이어그램에서 나타냈던 여러 가지 객체와 객체들 사이에서 어떤 기능을 수행하기 위해 주고받는 메시지 즉, 객체들간의 상호 호출하게 되는 오퍼레이션 들을 순서에 입각해서, Use Case 다이어그램에서 나타냈던 Use Case 하나하나에 대응시킴으로써 객체들이 Use Case 의 행동을 어떻게 수행하는지를 표현한다[3].

구성요소의 하나인 객체는 다이어그램 상에서 Actor 혹은 하나의 객체가 다른 객체의 오퍼레이션 들을 수행할 때 그 주체가 되는 것을 나타내며, Flow of Control 은 객체와 객체간의 경계를 구분 지으면서 다이어그램 상에서는 한 객체에서 다른 객체로 보낼 메시지의 생명주기를 그 길이로써 표현하게 된다. 메시지는 Actor 또는 한 객체가 다른 객체의 오퍼레이션을 부를 때 해당되는 오퍼레이션이다.

3. 다이어그램들 간의 Consistency 기준

3.1. Actor 와 Class 간의 매핑관계

Use Case 다이어그램에서 시스템과 상호작용 하는 각 Actor 는 클래스 다이어그램에서 이름이나 속성 등 Actor 자체의 정보가 목표 시스템에서 관리대상이 되는 경우 하나의 독립된 클래스가 될 수 있다. 또한 이 Actor 는 상호작용 하는 대상이나 주체가 되어 중요한 기능을 수행한다. 이들 여러 기능들의 상호작용 중에서 Actor 가 여러 Use Case 들과 연관 되어 질 때, 이것은 클래스 다이어그램의 클래스가 어떤 기능을 수행함에 있어서 다른 클래스의 오퍼레이션들과 연관이 됨을 의미한다. 하나의 Actor 는 여러 Use Case 와 연관될 수 있으며 각각의 Use Case 또한 하나 또는 그 이상의 클래스들과 연관될 수 있다.

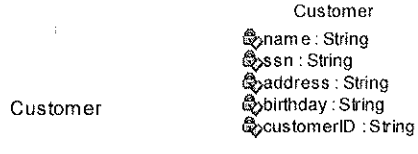

그림 1. Actor 와 Class

3.2. Use Case 의 Workflow 와 Sequence Diagram

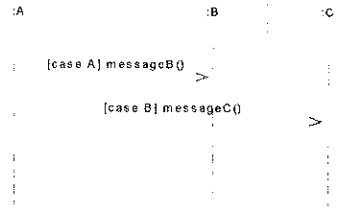

그림 2. Sequence Diagram 에서의 Guard Condition

Use Case 하나의 기능을 상세하게 설명하는 Use Case 명세서의 주 흐름에는 기본 흐름, 선택 흐름 그리고 예외 흐름 등이 있다. 이러한 흐름들은 순서도 상에서 나타나는 메시지들 즉, 오퍼레이션들의 일련의 순차적인 흐름으로써 나타난다. 기본 흐름의 경우는 순서도에서 단지 해당 메시지에 대해서 이름만으로 그것을 나타내지만, 선택 흐름과 예외 흐름의 경우에는 Guard Condition 의 형태로 표현된다. 그림 2 에서 처럼 클래스 B 와 C 에 대하여 각각 messageB()와 messageC()를 수행하게 되는데 이것은 '[]' 으로써 명시되어 있는 guard condition 으로 나타난다.

3.3. Use Case Description 과 Sequence Diagram

Use Case 명세서에 나타나는 Actor 와 시스템과의 상호작용은 그 둘 사이에 일정한 순서와 규칙을 가지고 메시지를 주고 받는 것이다. 이 때 주고받는 메시지의 흐름에 해당하는 것은 순서도에서 Actor 와 Controller 클래스간에 주고받는 메시지로 나타난다. 그림 3 에서처럼 Use Case 명세서에는 Actor 로부터 시스템으로 그리고 다시 시스템에서 Actor 로의 메시지의 흐름이 순서도에서 Actor 가 시스템에 대한 어떤 기능을 요청할 때 Controller 클래스에 대하여 요청을 하게 되므로 결국 Actor 와 Controller 클래스간의 관계로 매핑된다.

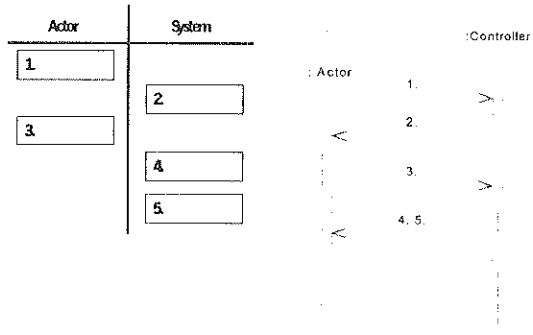

그림 3. Actor 와 시스템간의 메시지 흐름

3.4. Use Case 와 Sequence Diagram

Use Case 다이어그램에서 하나의 독립된 기능을 나타내는 Use Case 는 여러 객체들간에 주고받는 메시지의 시간흐름을 나타낸 하나의 독립된 순서도가 될 수 있다. 이 순서도는 하나

의 기능적인 단위를 나타내는 Use Case 에 대하여 해당 Use Case 가 어떠한 기능을 수행하는지를 상세하게 나타낸다. 해당 기능을 수행하기 위해 참여하는 클래스의 인스턴스인 객체들 간에 어떠한 오퍼레이션 들을 수행하며 그러한 오퍼레이션 들을 호출하는 객체는 어느 것인지, 호출된 오퍼레이션을 수행하는 객체는 어떠한 것인지, 어떠한 순서로 호출하게 되는지, 오퍼레이션의 호출에 필요한 데이터는 어떠한 것들이 있는지 상세하게 나타낸다. 그림 4 의 경우는 조건에 따라 각각 독립된 Workflow 를 가지며 그 각각의 독립된 Workflow 는 또 다시 독립된 순서도로 나타난다. 이와 같이 Use Case 다이어그램 상의 각 Use Case 는 반드시 하나의 순서도로만 나타나지 않고 자체가 몇 개의 독립된 기능을 갖는 Use Case 로 나뉘질 경우 각각에 대하여 순서도를 나타낸다.

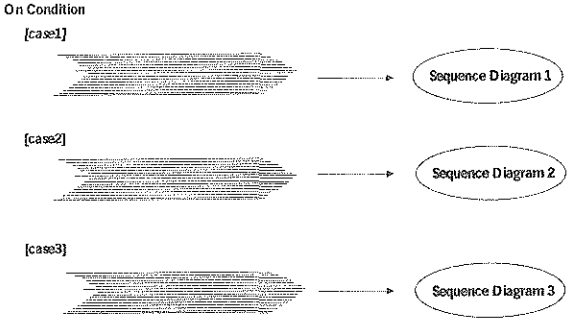

그림 4. 선택적인 Workflow 에 대한 Sequence Diagram

3.5. CD 의 Operation 과 SD 의 Message

클래스 다이어그램의 오퍼레이션은 순서도에서 메시지로 나타난다. Use Case 의 해당 기능을 수행하기 위해서 클래스들의 각 인스턴스 간에 주고받는 메시지에 해당되는 오퍼레이션 들은 Actor 에서 다른 객체로 또는 하나의 객체에서 다른 객체로의 메시지의 흐름이 된다.

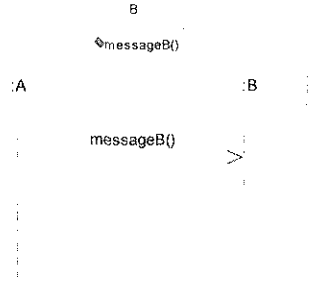

그림 5. Sequence Diagram 에서의 메시지 전달

그림 4 는 클래스 A 가 클래스 A 자신에 있는 오퍼레이션이 아니라 바로 그 메시지가 전달되는 객체인 클래스 B 에 있는 messageB() 라는 오퍼레이션을 호출하는 것을 나타내고 있다. 이렇게 순서도 상에 나타나는 오퍼레이션은 해당 메시지를 호출하는 객체의 오퍼레이션이 아니라, 그 메시지가 전달되어 받게 되는 객체가 수행하게 되는 세부적인 기능을 나타낸다. 그림 5 에서와 같이 클래스 A 의 객체가 클래스 B 의 객체에 대한 오퍼레이션을 호출하게 되는 경우 해당 오퍼레이션이 매개변수를 갖는 경우는 오퍼레이션은 B 클래스 객체의 것이지만 그에 대한 매개변수는 반드시 B 클래스 객체의 것만은 아니고 다른 객체의 속성들을 이용해서 해당 오퍼레이션을 호출할 수 있다.

4. 결론 및 향후 연구과제

본 논문에서는 UML 의 주요한 3 가지 다이어그램들 간의 일관성을 확인하는 기준에 대한 checklist 를 제시하였다. 지금까지의 여러 가지 연구들을 통해서도 이런 다이어그램들 간의 일관성에 관해서 논의된 바 있으나 본 논문에서 제시하는 일관성 확인 checklist 를 통해 다이어그램들 간에 이전보다 나은 좀 더 체계적이고 정확한 방법의 검증이 이루어질 것이다.

앞으로는 이러한 방법을 통해서 다이어그램들 간에 일관성을 확인하는 과정에서 더욱 더 상세하고 규칙화 되며 좀 더 높은 수준의 일관성을 유지할 수 있어야 할 것이다. 그러기 위해서 기존의 방법들 뿐 아니라 더욱 새롭고 진보된 기술의 개발도 필요할 것이다.

마지막으로 이러한 UML 의 다이어그램들 간의 일관성을 checklist 를 통해서가 아니라 마치 모델링 언어인 UML 을 사용할 수 있는 개발툴과 같이 손쉽게 검증해 줄 수 있는 자동화 도구나 소프트웨어의 개발이 필요할 것이다.

참 고 문 헌

[1] 김수동, 실무자를 위한 소프트웨어 공학, 에드텍, 1999

[2] Grady Booch, James Rumbaugh, Ivar Jacobson, *The Unified Modeling Language User Guide*, Addison Wesley, 2000

[3] Grady Booch, James Rumbaugh, Ivar Jacobson, *UML Distilled Second Edition*, Addison Wesley, 2000

[4] James Martin, James J.Odell, *Object-Oriented Methods*, Prentice-Hall, 1998

[5] Grady Booch, *Object-Oriented Analysis and Design*, The Benjamin/Cummings Publishing Company, 1994