2.2. UML

2.2.1. UML의 개요

UML은 시스템 분석, 설계, 구현 등 시스템 개발 과정에서 시스템 개발자와 고객 또는 개발자 상호 간의 의사소통이 원할하게 이루어지도록 표준화한 대표적인 **객체지향 모델링 언어**이다.

- UML을 이용하여 시스템의 구조를 표현하는 6개의 **구조 다이어그램**과 시스템의 동작을 표현하는 7개의 **행위 다이어그램**을 작성할 수 있다.
- 각각의 다이어그램은 사물과 사물 간의 관계를 용도에 맞게 표현한다.
- UML의 구성 요소에는 **사물**, **관계**, **다이어그램** 등이 있다.

2.2.2. 사물

사물은 모델을 구성하는 가장 중요한 요소로, 다이어그램 안에서 관계가 형성될 수 있는 대상들을 말한다.

- 사물에는 **구조 사물**, **행동 사물**, **그룹 사물**, **주해 사물**이 있다.
  i. **구조 사물**
     - 시스템의 개념적, 물리적 요소를 표현
     - 클래스, 유스케이스, 컴포넌트, 노드 등
  ii. **행동 사물**
     - 시간과 공간에 따른 요소들의 행위를 표현
     - 상호작용, 상태 머신 등
  iii. **그룹 사물**
     - 요소들을 그룹으로 묶어서 표현
     - 패키지
  iv. **주해 사물**
     - 부가적인 설명이나 제약조건 등을 표현
     - 노트

2.2.3. 관계

관계는 사물과 사물 사이의 연관성을 표현하는 것으로 **연관 관계**, **집합 관계**, **포함 관계**, **일반화 관계**, **의존 관계**, **실체화 관계** 등이 있다.

2.2.3.1. 연관 관계

연관 관계는 2개 이상의 사물이 서로 관련되어 있음을 표현한다.

- 사물과 사물 사이를 **실선**으로 연결하여 표현하며, 방향성은 **화살표**로 표현한다.
- 서로에게 영향을 주는 양방향 관계의 경우, 화살표를 생략하고 실선으로만 연결한다.
- 연관에 참여하는 객체의 개수를 의미하는 **다중도**를 선 위에 표기한다.

2.2.3.2. 집합 관계

집합 관계는 하나의 사물이 다른 사물에 포함되어 있는 관계를 표현한다.

- 포함하는 쪽과 포함되는 쪽은 서로 독립적이다.
- 포함되는 쪽에서 포함하는 쪽으로 **속이 빈 마름모**를 연결하여 표현한다.

2.2.3.3. 포함 관계

포함 관계는 집합 관계릐 특수한 형태로, 포함하는 사물의 변화가 포함되는 사물에게 영향을 미치는 관계를 표현한다.

- 포함하는 쪽과 포함되는 쪽은 서로 독립될 수 없고, 생명주기를 함께한다.
- 포함되는 쪽에서 포함하는 쪽으로 **속이 채워진 마름모**를 연결하여 표현한다.

2.2.3.4. 일반화 관계

일반화 관계는 하나의 사물이 다른 사물에 비해 더 일반적인지 구체적인지를 표현한다.

- 보다 일반적인 개념을 상위(부모), 보다 구체적인 개념은 하위(자식)라고 부른다.
- 구체적인 사물에서 일반적인 사물 쪽으로 **속이 빈 화살표**를 연결하여 표현한다.

2.2.3.5. 의존 관계

의존 관계는 연관 관계와 같이 사물 사이에 연관은 있으나, 필요에 의해 서로에게 영향을 주는 짧은 시간 동안만 연관을 유지하는 관계를 표현한다.

- 하나의 사물과 다른 사물이 소유 관계는 아니지만 사물의 변화가 다른 사물에도 영향을 미치는 관계이다.
- 영향을 주는 사물이 영향을 받는 사물 쪽으로 **점선 화살표**를 연결하여 표현한다.

2.2.3.6. 실체화 관계

실체화 관계는 사물이 할 수 있거나 해야 하는 기능으로, 서로 그룹화 할 수 있는 관계를 표현한다.

- 사물에서 기능 쪽으로 **속이 빈 점선 화살표**를 연결하여 표현한다.

2.2.4. 다이어그램

다이어그램은 사물과 관계를 도형으로 표현한 것이다.

- 여러 관점에서 시스템을 가시화한 뷰를 제공함으로써, 의사소통에 도움을 준다.
- 정적 모델링에서는 **구조적 다이어그램**을 사용하고, 동적 모델링에서는 주로 **행위 다이어그램**을 사용한다.

2.2.4.1. 구조적 다이어그램의 종류

1. **클래스 다이어그램**
   - 클래스와 클래스가 가지는 속성, 클래스 사이의 관계를 표현한다.
   - 시스템의 구조를 파악하고, 구조상의 문제점을 도출할 수 있다.
2. **객체 다이어그램**
   - 클래스에 속한 사물등, 즉 인스턴스를 특정 시점의 객체와 객체 사이의 관계로 표현한다.
3. **컴포넌트 다이어그램**
   - 실제 구현 모듈인 컴포넌트 간의 관계나 컴포넌트 간의 인터페이스를 표현한다.
   - 구현 단계에서 사용되는 다이어그램이다.
4. **배치 다이어그램**
   - 결과물, 프로세스, 컴포넌트 등 물리적 요소들의 위치를 표현한다.
   - 노드와 의사소통 경로를 표현한다.
   - 구현 단계에서 사용되는 다이어그램이다.
5. **복합체 구조 다이어그램**
   - 클래스나 컴포넌트가 복합 구조를 갖는 경우, 그 내부 구조를 표현한다.
6. **패키지 다이어그램**
   - 유스케이스나 클래스 등의 모델 요소들을 그룹화한 패키지들의 관계를 표현한다.

2.2.4.2. 행위 다이어그램의 종류

1. **유스케이스 다이어그램**
    - 사용자의 요구를 분석하는 것으로, 기능 모델링 작업에 사용한다.
    - 사용자와 사용 사례로 구성되며, 사용 사례 간에는 여러 형태의 관계로 이루어진다.
2. **시퀀스 다이어그램**
    - 상호 작용하는 시스템이나 객체들이 주고받는 메시지를 표현한다.
3. **커뮤니케이션 다이어그램**
    - 시퀀스 다이어그램과 같이 동작에 참여하는 객체들이 주고받는 메시지를 표현하는데, 메시지 뿐만 아니라 객체들 관의 연관까지 표현한다.
4. **상태 다이어그램**
    - 하나의 객체가 자신이 속한 큰래스의 상태 변화 혹은 다른 객체와 상호 작용에 따라 상태가 어떻게 변화하는지를 표현한다.
5. **활동 다이어그램**
    - 시스템이 어떤 기능을 수행하는지 객체의 처리 로직이나 조건에 따른 처리의 흐름을 순서에 따라 표현한다.
6. **상호작용 개요 다이어그램**
    - 상호작용 다이어그램 간에 제어 흐름을 표현한다.
7. **타이밍 다이어그램**
    - 객체 상태 변화와 시간 제약을 명시적으로 표현한다.