01 UML의 용도와 특징

UML의 탄생과 특징

- UML(Unified Modeling Language)
  - 시스템 개발을 위한 시각적인 설계 표기 제공
  - 객체 지향 시스템을 개발할 때 산출물을 명세화, 시각화, 문서화하는 데 사용
  - 개발하는 시스템을 이해하기 쉬운 형태로 표현하여 분석가, 설계자, 의뢰인이 효율적으로 의사소통할 수 있게 해줌

➔ UML은 표준화된 통합 모델링 언어

UML의 탄생과 특징

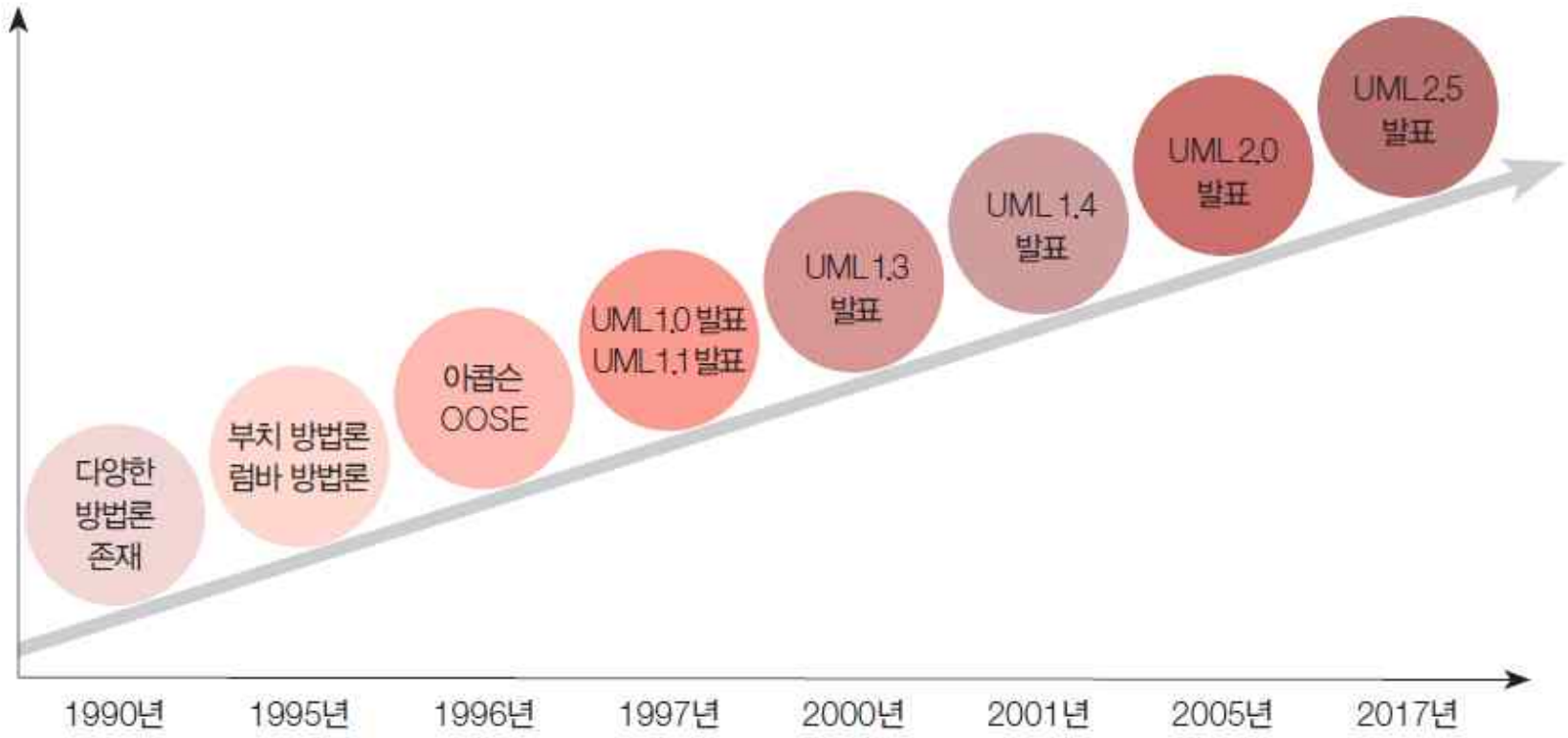


그림 1-2 UML의 발전 과정

01 UML의 용도와 특징

- **UML이 제공하는 표준화된 다이어그램**
  - 유스케이스 다이어그램Use-case Diagram
  - 클래스 다이어그램Class Diagram
  - 순차 다이어그램Sequence Diagram
  - 통신 다이어그램Communication Diagram
  - 활동 다이어그램Activity Diagram
  - 상태 다이어그램State Diagram
  - 컴포넌트 다이어그램Component Diagram
  - 배치 다이어그램Deployment Diagram
  - 패키지 다이어그램Package Diagram

UML의 특징

- UML은 시각화Visualization 언어다.
  - 소프트웨어의 개념 모델을 시각적인 형태로 표현하며 명확히 정의된 표준화된 다이어그램을 제공
  - 이를 이용해 오류 없는 원활한 의사소통 가능
- UML은 명세화Specification 언어다.
  - 소프트웨어 개발 과정인 분석, 설계 단계의 각 과정에서 필요한 모델을 정확하고 완전하게 명세화하여 만들 수 있음
  - 명세화에서 각 다이어그램의 기호는 의미를 담고 있으며 추상적이지만 고유의 특성을 갖고 있음
- UML은 구축Construction 언어다.
  - UML은 자바Java, C++, 비주얼 베이직Visual Basic, C# 같은 다양한 프로그래밍 언어로 표현가능
  - UML로 설계된 모델을 프로그램 코드로 자동 변환할 수 있으며, 이미 구축된 소스 코드를 UML로 역변환하여 분석하는 역공학Reverse Engineering도 가능
- UML은 문서화Documentation 언어다.
  - UML은 StarUML, 투게더Together 등 케이스 툴CASE Tool을 이용하여 설계한 내용을 자동으로 문서화 가능

UML과 모델링

```c
#include <stdio.h>
int main() {
      int a, b;
      scanf("%d %d", &a, &b);
      printf("a + b = %d", a + b);
 }
```

(a) C로 구현한 덧셈 프로그램

```java
import java.util.Scanner;
public class AddDemo {
      public static void main(String[] args) {
            Scanner in = new Scanner(System.in );
            int a = in.nextInt();
            int b = in.nextInt();
            System.out .printf("%d + %d = %d", a, b, a+b);
    }
}
```

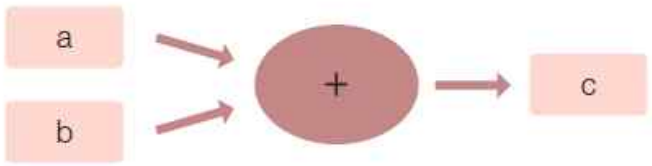
(b) 자바로 구현한 덧셈 프로그램

```python
def add():
      num = int(input("num1:"))
      num2 = int(input("num2:"))
      result = num + num2
      print("두수의 합은 " result)
```

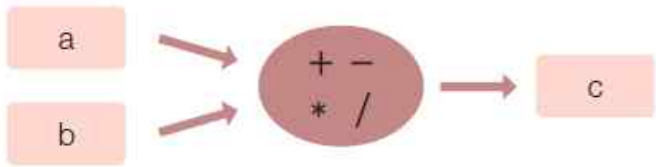
(c) 파이썬으로 구현한 덧셈 프로그램

그림 1-3 다양한 언어로 구현한 덧셈 프로그램

개발하고자 하는 프로그램을 시각적으로 표현하는 것이며,
이때 의뢰자의 요구에 맞게 쉽게 수정해서
결과적으로 유지보수 기간을 줄여 생산성을 높일 수 있음



(a) 덧셈 연산 프로그램의 시각적 표현



(b) 사칙연산 프로그램의 시각적 표현

그림 1-4 프로그램의 시각적 표현

- **모델링이 필요한 이유**

(a) 개인이 제어할 수 있는 작업 : 개집 짓기

(b) 개인이 제어할 수 없는 작업 : 큰 건축물 짓기

그림 1-5 모델링이 필요한 이유

모델링 개념

모델링

- 시스템을 구축할 때 개발자가 고민하고 결정하는 모든 활동
- 구현 단계 이전의 요구 사항 정의, 분석, 설계에서 수행하는 활동
- 모델 : 모델링의 결과물
- CASE 툴: 모델링을 전문적으로 지원하는 툴
  - Ex) StarUML, 로즈Rose, 투게더Together 등

표 1-1 모델링과 프로그래밍

구분	모델링	프로그래밍
목적	구축할 시스템의 모습 정의	시스템의 실제 구현
세부 수행 활동	요구 사항 정의, 분석, 설계	소스 코드 편집, 컴파일, 시험, 디버깅
결과물	모델	소스 코드를 포함한 구현된 시스템
표기법	모델링 언어(UML, ERD, DFD)	프로그래밍 언어(자바 , C++)
지원 툴	CASE 툴(StarUML, 로즈Rose, 투게더Together)	개발 툴(Jbuilder, Visual Studio, .Net)

- **다이어그램**
  - 클래스 다이어그램
    - 클래스, 인터페이스, 통신과 함께 이들의 관계를 나타냄
  - 컴포넌트 다이어그램
    - 컴포넌트 사이의 구성과 의존을 표현
  - 배치 다이어그램
    - 실행 시 처리하는 노드와 그 노드에 있는 컴포넌트들의 구성을 표현
  - 패키지 다이어그램
    - 여러 모델 요소를 그룹화하여 패키지를 구성하고, 이들 패키지 사이를 관계로 표현
  - 유스케이스 다이어그램
    - 유스케이스와 액터의 관계를 구조적으로 표현
  - 순차 다이어그램과 통신 다이어그램
    - 교류 다이어그램의 한 종류
  - 활동 다이어그램
    - 시스템 내부에 있는 활동의 흐름을 표현한 것
  - 상태 다이어그램
    - 시스템의 동적뷰를 나타냄

- 다이어그램

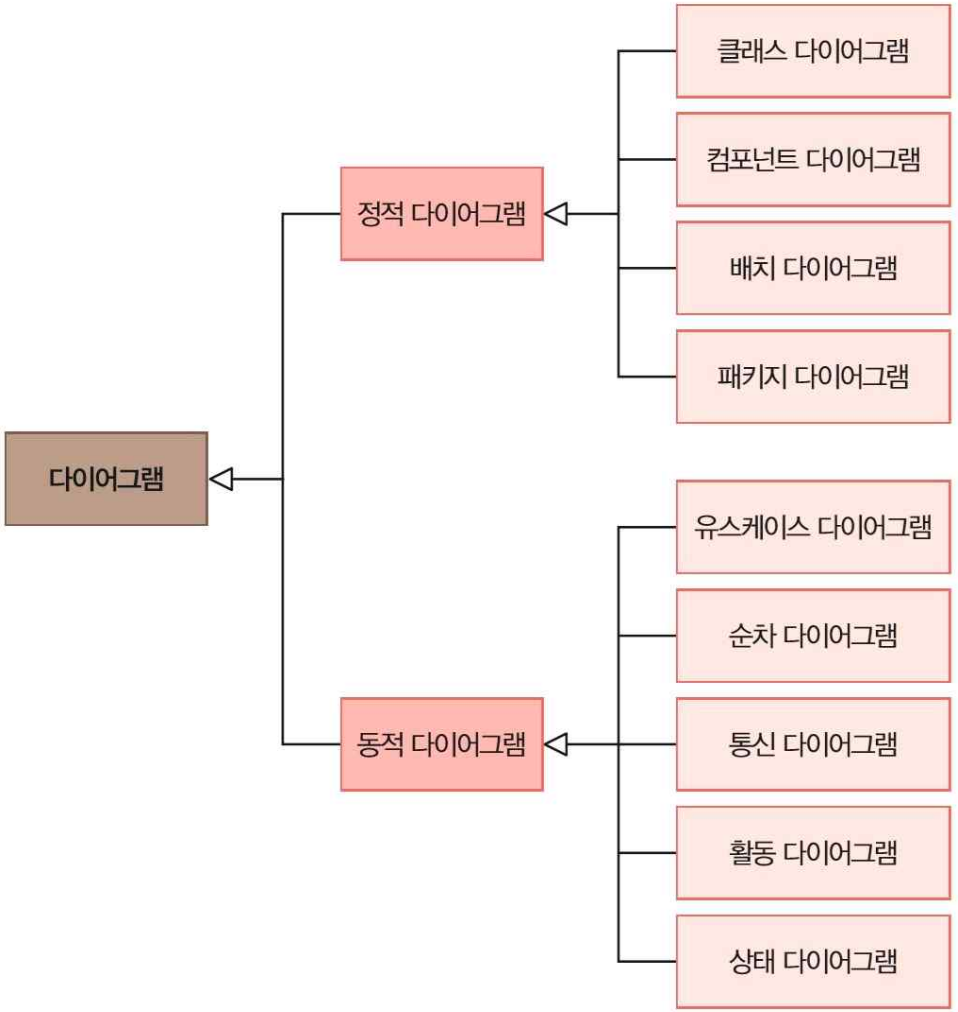


그림 2-21 정적 다이어그램과 동적 다이어그램

UML뷰의 개념과 종류

유스케이스 뷰 (요구사항 뷰)

- 외부 액터에 의해 인식되는 시스템의 기능 요구 사항을 보여주는 관점
- 사용자가 시스템으로부터 원하는 기능이 '무엇'인지를 정의
- 다른 뷰를 유도하는 중심 역할
- 시스템을 하나의 블랙박스 Black Box로 바라봄

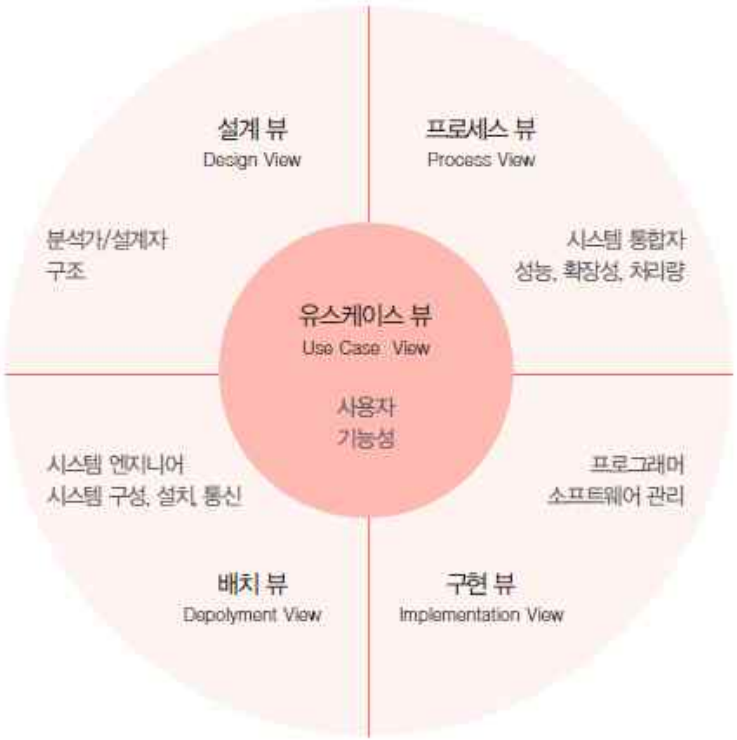


그림 2-22 소프트웨어 아키텍처의 5개 뷰

표 2-1 유스케이스 뷰에 이용되는 UML 다이어그램

구분	UML 다이어그램
정적 측면	유스케이스 다이어그램
동적 측면	상태 다이어그램, 순차 다이어그램 통신 다이어그램, 활동 다이어그램

UML뷰의 개념과 종류

설계 뷰

- 시스템 내부의 클래스와 컴포넌트를 파악해 기술

표 2-2 설계 뷰에 이용되는 UML 다이어그램

구분	관심 사항	이용되는 UML 다이어그램
정적 측면	클래스 및 클래스 사이의 관계	클래스 다이어그램, 객체 다이어그램
동적 측면	클래스 내의 동작	상태 다이어그램
	클래스 간의 상호작용	순차 다이어그램, 통신 다이어그램
	클래스의 연산 동작	활동 다이어그램

프로세스 뷰

- 설계 뷰와 마찬가지로 시스템 내부의 구조에 중점을 두고 기술
- 그러나 독자적인 제어 스레드를 가질 수 있는 클래스를 중점으로 함

표 2-3 프로세스 뷰에 이용되는 UML 다이어그램

구분	UML 다이어그램
정적 측면	클래스 다이어그램, 활성 클래스 다이어그램
동적 측면	상태 다이어그램, 통신 다이어그램, 활동 다이어그램

UML뷰의 개념과 종류

구현 뷰

- 시스템 구현 형태를 나타내기 위해 구현 모듈과 그들의 관계 각종 파일의 의존 관계 등을 보여줌
- 컴포넌트 다이어그램으로 표현

표 2-4 구현 뷰에 이용되는 UML 다이어그램

구분	UML 다이어그램
정적 측면	컴포넌트 다이어그램
동적 측면	상태 다이어그램, 순차 다이어그램, 통신 다이어그램, 활동 다이어그램

배치 뷰

- 컴퓨터와 컴퓨터 간의 통신 방법에 중점을 둠
- 배치 다이어그램으로 표현

표 2-5 배치 뷰에 이용되는 UML 다이어그램

구분	UML 다이어그램
정적 측면	배치 다이어그램
동적 측면	상태 다이어그램, 순차 다이어그램, 통신 다이어그램, 활동 다이어그램

- 개발 활동과 UML뷰

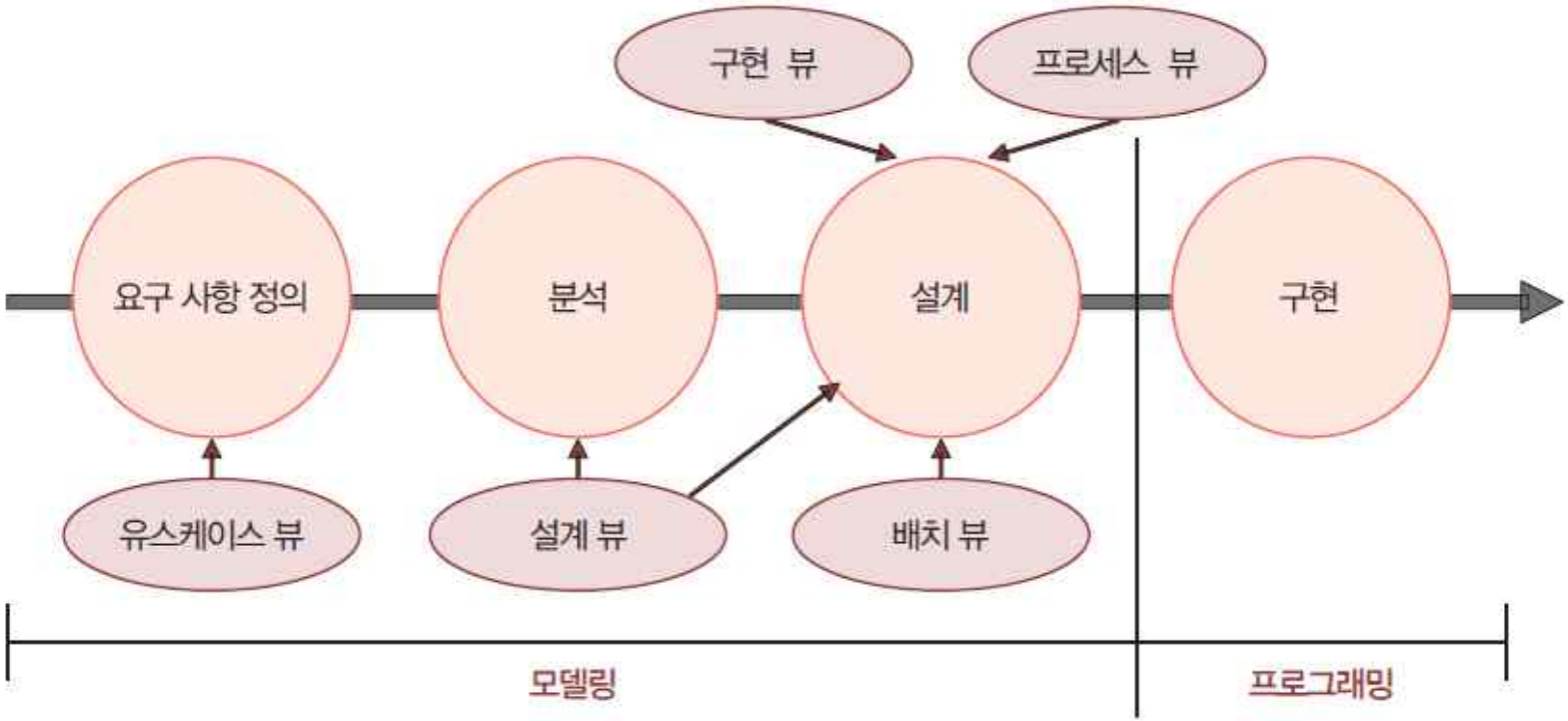


그림 2-23 개발 활동과 UML 뷰

■ 개발 활동과 UML뷰

표 2-6 시스템 유형별 뷰와 UML 다이어그램

뷰	UML 다이어그램	시스템 유형		
		간단한 시스템	반응적 시스템	분산 시스템
유스케이스 뷰	유스케이스 다이어그램	V	V	V
설계 뷰	클래스 다이어그램	V	V	V
	순차 다이어그램	V	V	V
	통신 다이어그램	V	V	V
	상태 다이어그램		V	V
프로세스 뷰	클래스 다이어그램			V
	순차 다이어그램			V
	통신 다이어그램			V
구현 뷰	컴포넌트 다이어그램			V
배치 뷰	배치 다이어그램			V

Thank You