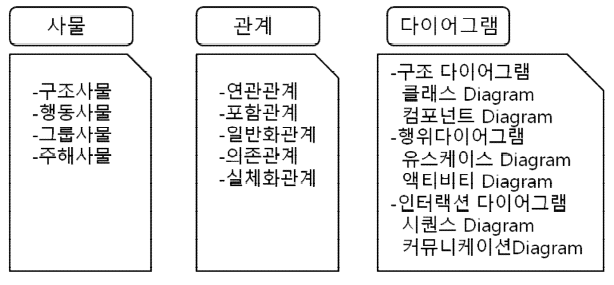
**정적모델 설계**

정적모델, 객체지향, UML 모델링, 설계패턴, 분석 클래스, 엔티티 클래스, 경계 클래스, 제어 클래스

* 1. 정적모델 상세분석

1. **UML(Unified Modeling Language)의 이해**

UML은 소프트웨어의 청사진을 작성하는 표준 언어로, 이를 이용하여 산출물을 통한 가시화, 명세화, 문서화가 가능한 도구이다. 소프트웨어 아키텍처 설계 소프트웨어에 대한 요구사항 정의, 분석과 설계 모델링 작업에서 널리 활용되고 있다.

1. UML의 구성 요소
2. 사물

모델 구성의 기본요소로 추상적인 개념. 시스템의 구조와 행위를 표현하고 개념들을 그룹화하기 위한 것들로구성되어 있다.

1. 관계

사물간의 연결관계를 추상화하여 표현.

(가) 연관관계 (Association) : 구조적 관점에서 사물들의 연결을 표현한다.

(나) 포함관계 (Composition) : 사물 사이의 포함관계를 표현한다.

(다) 일반화관계 (Generalization) : 사물 간의 특수화, 일반화 관계를 표현한다.

(라) 의존관계 (Dependency) : 사물 간의 서로 영향을 미치는 관계를 표현한다.

(마) 실체화관계 (Realization) : 사물이 수행하기로 되어 있는 제약을 명세화한 것이다.

1. 다이어그램(Diagram)

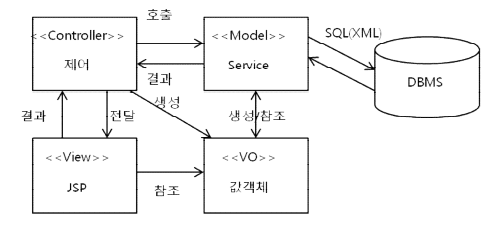
사물들 간의 관계를 도형으로 표현한 것. 여러관점에서 시스템을 가시화한 뷰를 제공함으로써 의사소통에 도움을 준다

1. 정적 다이어그램과 동적 다이어그램
2. 정적(구조) 다이어그램
   1. 클래스 다이어그램
   2. 객체 다이어그램
   3. 컴포넌트 다이어그램
      1. 실제 구현 모듈인 컴포넌트 간의 관계를 표현
   4. 배치 다이어그램
3. 동적 다이어그램
   1. 유스케이스 다이어그램
   2. 시퀀스 다이어그램
      1. 시스템의 내부적인 로직 흐름을 동적으로 표현한 다이어그램. 객체와 객체 사이의 관계와 객체들끼리 상호교환하는 메시지의 순서를 강조
   3. 상태 다이어그램
   4. 활동 다이어그램
4. UML을 이용한 모델링
5. 기능 모델링
6. 정적 모델링
7. 동적 모델링
8. 설계패턴, MVC(Model-View-Controller) 패턴 이해

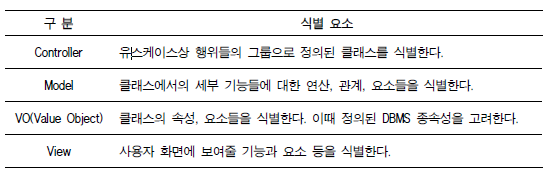


1. 컨트롤러 컴포넌트: 전체적인 흐름관리. 모델과 뷰 영역 간 흐름을 조정. 사용자 인터페이스에서 발생된 이벤트를 입력받아 처리
2. 모델 컴포넌트: 애플리케이션에서 비즈니스 핵심 기능 로직과 데이터를 다루는 영역. 데이터를 캡슐화하고 애플리케이션에 특화된 프로세스 로직을 수행
3. 뷰 컴포넌트: 클라이언트에게 보여지는 프레젠테이션 로직이 구현되는 영역. 다양한 방법으로 모델 컴포넌트 영역에서 처리한 정보를 보여줌. 주로 화면 출력과 관련된 로직으로 구성

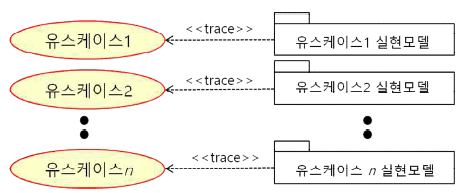
<실제 설계 과정>



1. **소프트웨어 아키텍처 설계 가이드라인 확인**
   1. 소프트웨어 설계 가이드라인에 정의된 설계패턴(MVC패턴, Layer계층패턴 등)과 개발 플랫폼, DBMS 확인
   2. 소프트웨어 설계를 위한 표준 설계 지침 등을 확인
2. **정적모델 상세분석에 필요한 요소들을 확인**
   1. 설계 가이드라인에서 정의된 설계패턴의 요소별 정적모델 요소를 식별



* 1. 유스케이스 분석을 통해 정적모델을 상세분석



* + 1. 유스케이스의 각 행위(동작)에 할당된 정적 분석 클래스를 확인
    2. 유스케이스 행위(동작)를 기준으로 하여 클래스 내의 속성 요소를 확인
    3. 유스케이스 행위(동작)를 기준으로 클래스상의 오퍼레이션이 도출되었는지 확인
    4. 유스케이스 전체 기능이 각각 정적모델 간의 관계로 도출되었는지 확인
    5. 식별된 정적모델 요소로 상관 분석한다.

1. **분석 단계에서 도출된 분석 클래스 요소들을 상세분석한다.**
2. 클래스와 해당 클래스 구성 요소에 대해 상세분석
3. 경계 클래스를 상세분석(외부 액터와 시스템 간의 상호작용 제공 클래스)
   1. 액터와의 상호 작용은 경계 클래스만 제공하고 있는지 확인
   2. 유스케이스 기능 중 입력과 출력에 대한 기능만 제공하고 있는지 확인
   3. 액터와의 상호작용을 위한 경계 클래스가 정의 되어있는지 확인
4. 제어 클래스 상세분석
5. 엔티티 클래스 상세분석 (영속적인 정보의 저장 등 정보의 관리 기능을 제공하기 위한 클래스)
6. 분석 클래스 간의 관계를 상세분석
   1. 제어 클래스 간의 관계는 비즈니스 로직 및 제어 로직간의 관계를 표현
   2. 엔티티 클래스 간의 관계는 지속적인 데이터 간의 관계
   3. 분석 클래스 간의 협력이 필요하며 이를 위해 분석 클래스 간의 관계가 정의되어야 함.
   4. 제어 클래스가 경계 클래스에서 반드시 하나 이상 사용되었는지 확인
   5. 엔티티 클래스가 제어 클래스에서 반드시 하나 이상 사용되었는지 확인
   6. 클래스의 계층 구조에서 상위 클래스와 하위 클래스는 Is a의 관계를 만족하고 의미적 차이가 적절한지 확인
   7. **정적 분석 모델 정합성 검증**
7. 정적 분석모델의 이해
   1. 소프트웨어 분석 단계와 설계 단계에서의 정적모델

분석단계 클래스는 기능적 요구사항의 분석인 반면, 설계 단계에서의 클래스는 기능적 요구사항 뿐 아니라 비기능적 요구사항까지 고려해야함. 무엇보다 개발 플랫폼을 고려하여 설계



1. 정적모델의 구성 요소

2-1. 클래스

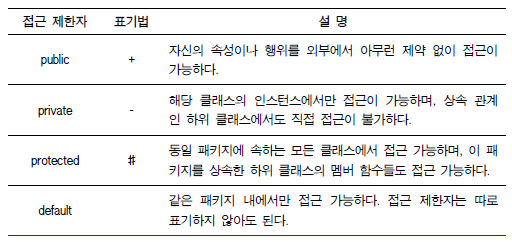
2-2. 속성: 클래스가 가지는 구조적 특성에 이름을 부여한 것으로 인스턴스가 가질 수 있는 값의 범위와 타입을 기술. 보통 영문 소문자로 시작

2-3. 메소드

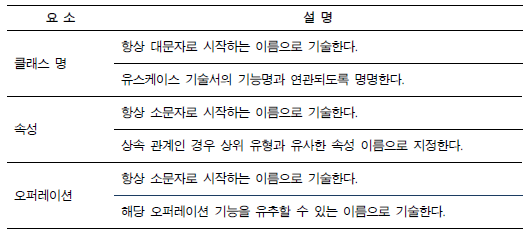
2-4. 관계



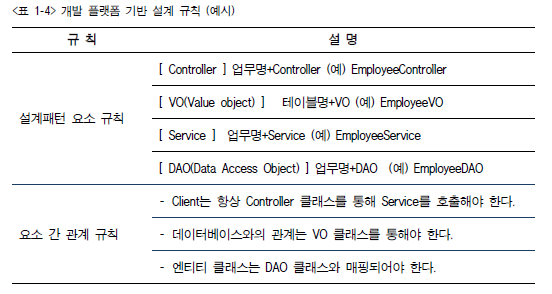
2-5. 접근 제한자



1. 정적 모델 검증에 필요한 정적 분석 규칙
   1. 정적모델 명명 규칙



* 1. 개발 플랫폼 기반 설계패턴 규칙

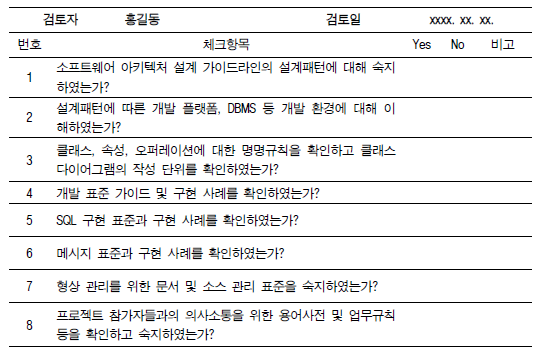


Model – VO

Controller - DAO

<실제 설계>

1. 소프트웨어 설계 가이드라인에서 정의된 설계 지침을 준수하였는지 확인
   1. 설계 가이드라인의 설계 지침 준수사항을 체크리스트로 작성



1. 유스케이스를 기반으로 정적모델 비즈니스 설계 관점에서 정합성을 검증



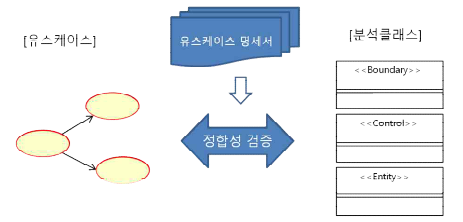
1. 유스케이스와 정적 분석모델 사이의 모든 요소들이 도출되었는지 검증
2. 유스케이스와 정적 분석모델 사이의 CRUD(create, read, update, delete) 매트릭스를 작성
3. 작성된 CRUD 매트릭스를 이용하여 정적모델 정합성을 검증



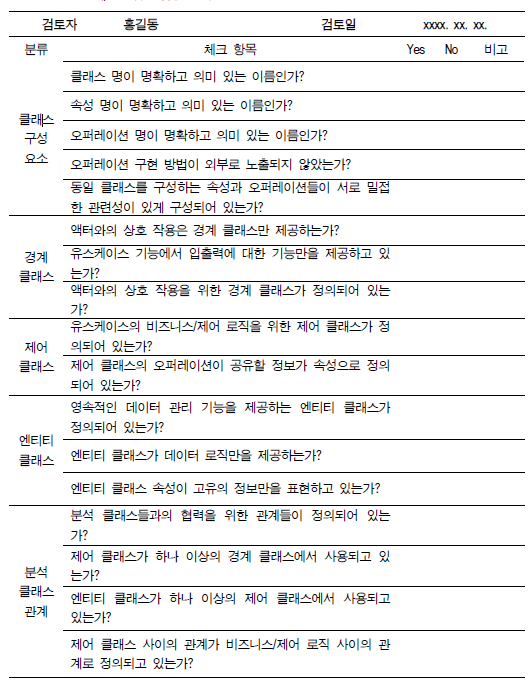
**3.분석단계에서 도출된 분석 클래스의 요소 관점에서 정적 분석모델의 정합성을 검증한다.**

- 클래스 요소 확인, 속성 요소 확인, 오퍼레이션 요소를 확인,

- 각 분석 클래스별로 정적 분석모델의 정합성을 검증



-정합성 검증을 위한 체크리스트 작성



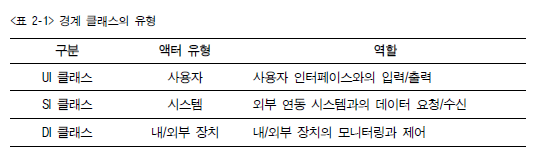
-작성된 체크리스트를 이용하여 리뷰를 통해 검증

4. 개발 플랫폼에서의 기술적인 관점에서 정적 분석모델을 검증한다.

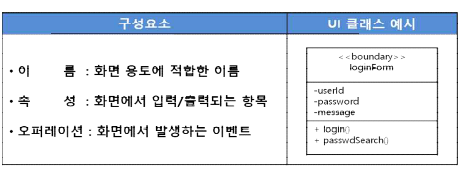
**2-1. 정적 설계모델 상세화**

1, 분석 클래스 모델의 이해

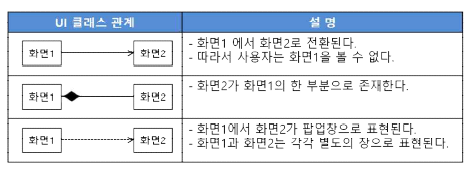
* 1. 경계 클래스(boundaty Class)



1. UI클래스 (ui Boundary class): 사용자 인터페이스를 모델링할 때 사용하는 경계 클래스 유형으로 사용자와 UI 화면의 인터페이스를 위한 역할과 관계를 표현
   1. UI 클래스의 구성 요소

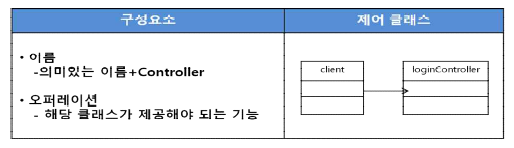


* 1. UI 클래스의 관계

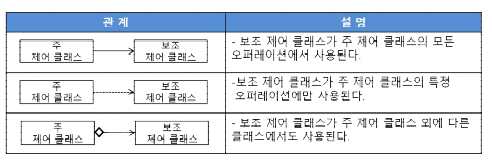


* 1. SI 클래스 (System Boundaty Class) : 외부 시스템과의 상호작용을 위한 기능을 정의, 외부 시스템과 통신할 때 어떤 기능을 사용하고 그 기능의 파라미터는 어떤 것인지 구체적인 프로토콜을 기술
  2. DI 클래스 (Device Boundary Class): 내부, 외부 장치와의 상호작용

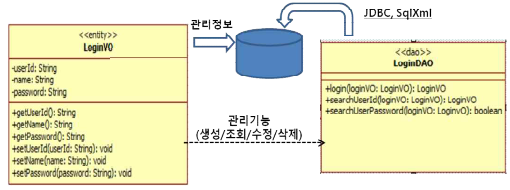
1. 제어 클래스(control clasS): 비즈니스 로직의 처리 흐름을 제어 또는 조정하거나 트랜잭션을 처리할 때 사용되는 클래스. 경계 클래스에서 입력된 값을 전달받아 제어 클래스에서 정의한 비즈니스 로직에 다라 출력값을 제공
   1. 제어 클래스의 구성 요소



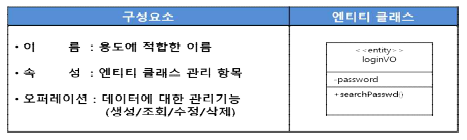
* 1. 제어 클래스의 관계



* 1. 엔티티 클래스(Entity Class): 엔티티 클래스의 정보는 시스템이 종료된다 해도 정보 값이 유지되어야 하므로 영구적인 데이터가 파일 또는 데이터베이스로 구현된다.

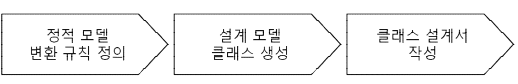


* + 1. 엔티티 클래스 구성요소: 엔티티 클래스는 클래스 이름과 엔티티 클래스가 관리해야하는 항복을 구성하는 속성과 오퍼레이션으로 표현
    2. 엔티티 클래스 관계: 데이터를 관리하는 로직을 제어하는 DAO 클래스와 연관 관계를 가지며 다른 클래스와 협력 관계를 표현한다.

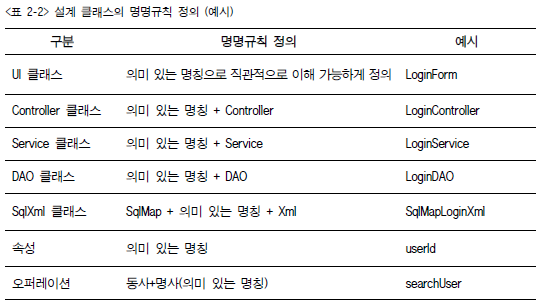


<실제 수행 순서>

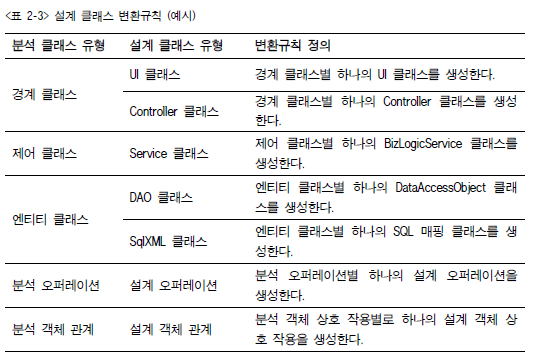
* + - 1. 정적 분석모델 상세화는 다음과 같은 순서로 진행



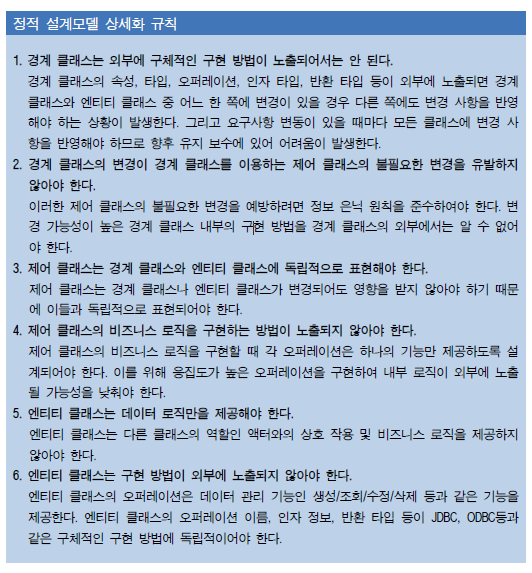
* + - 1. 정적 분석모델을 정적 설계모델로 변환하기 위한 규칙을 정의



+ 정적 분석모델과 설계 클래스 변환규칙을 정의

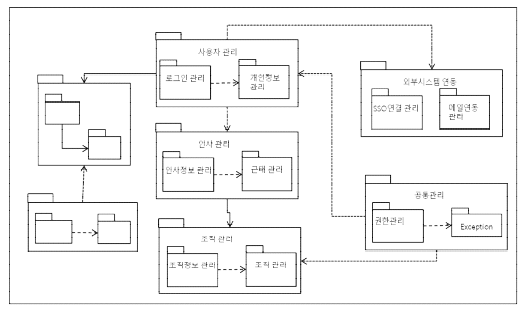


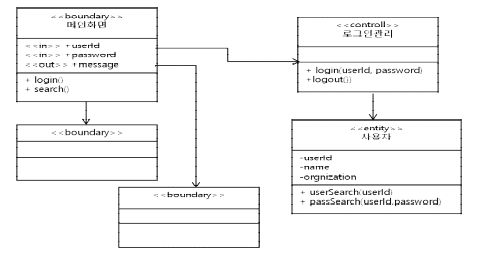
+정적 설계모델 상세화 규칙을 숙지하고 이를 기준으로 상세화 진행

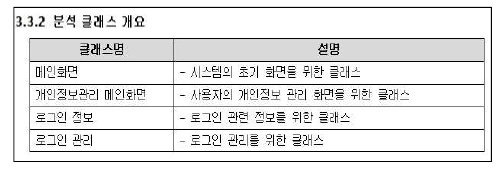


정적 설계모델 클래스별 변환규칙을 정의

**2-2. 정적 설계모델 명세화**







2-3. 정적 설계 모델 도식화

-클래스, 속성, 오퍼레이션, 관계,

정적 설계모델 설계서의 작성 방법

-설계 클래스 목록, 시퀀스 다이어그램, 설꼐 클래스 다이어그램, 설계 클래스 상세 정의

<실제 예>

