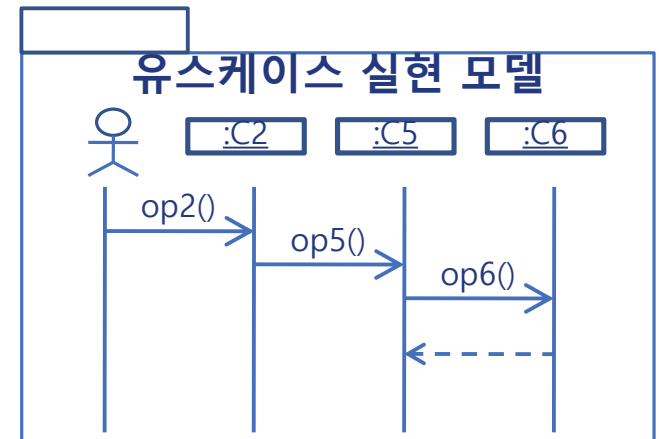
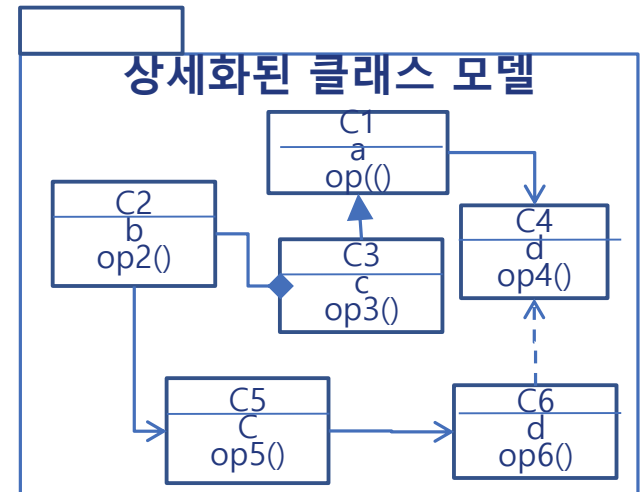
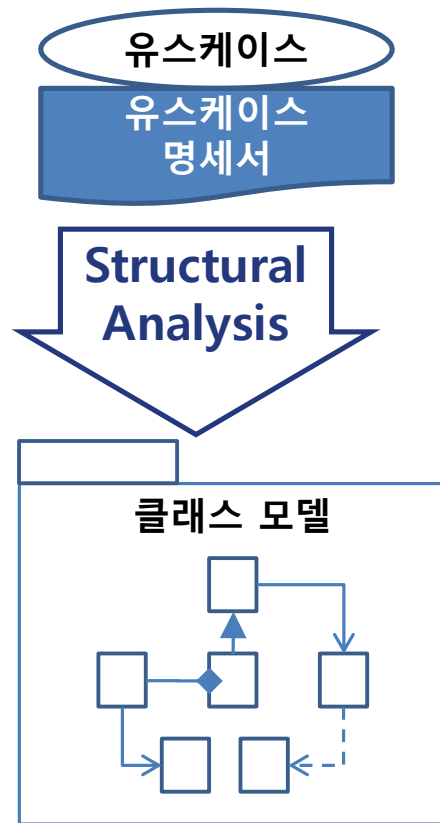


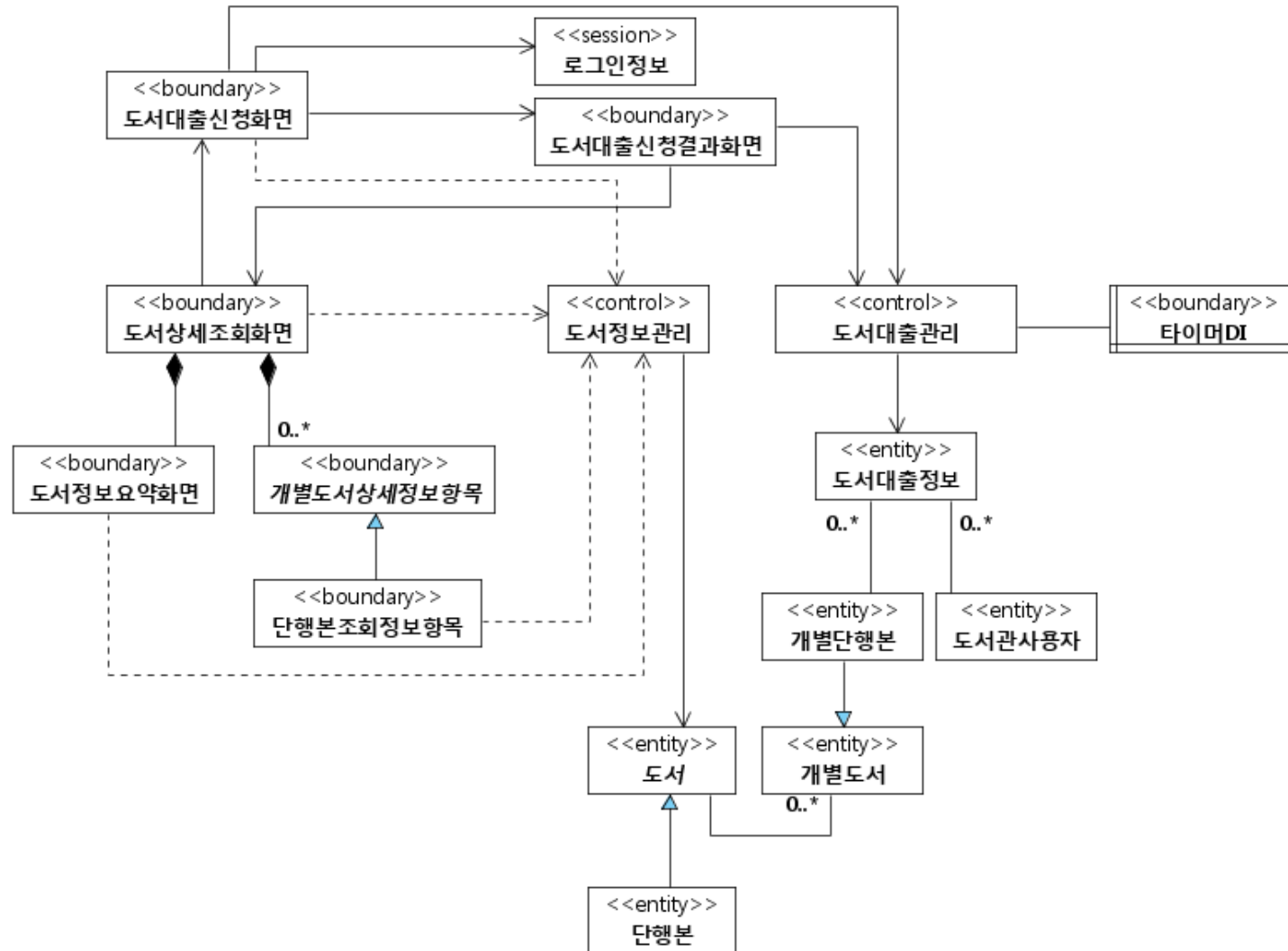
Object-oriented Analysis



Object-oriented Analysis

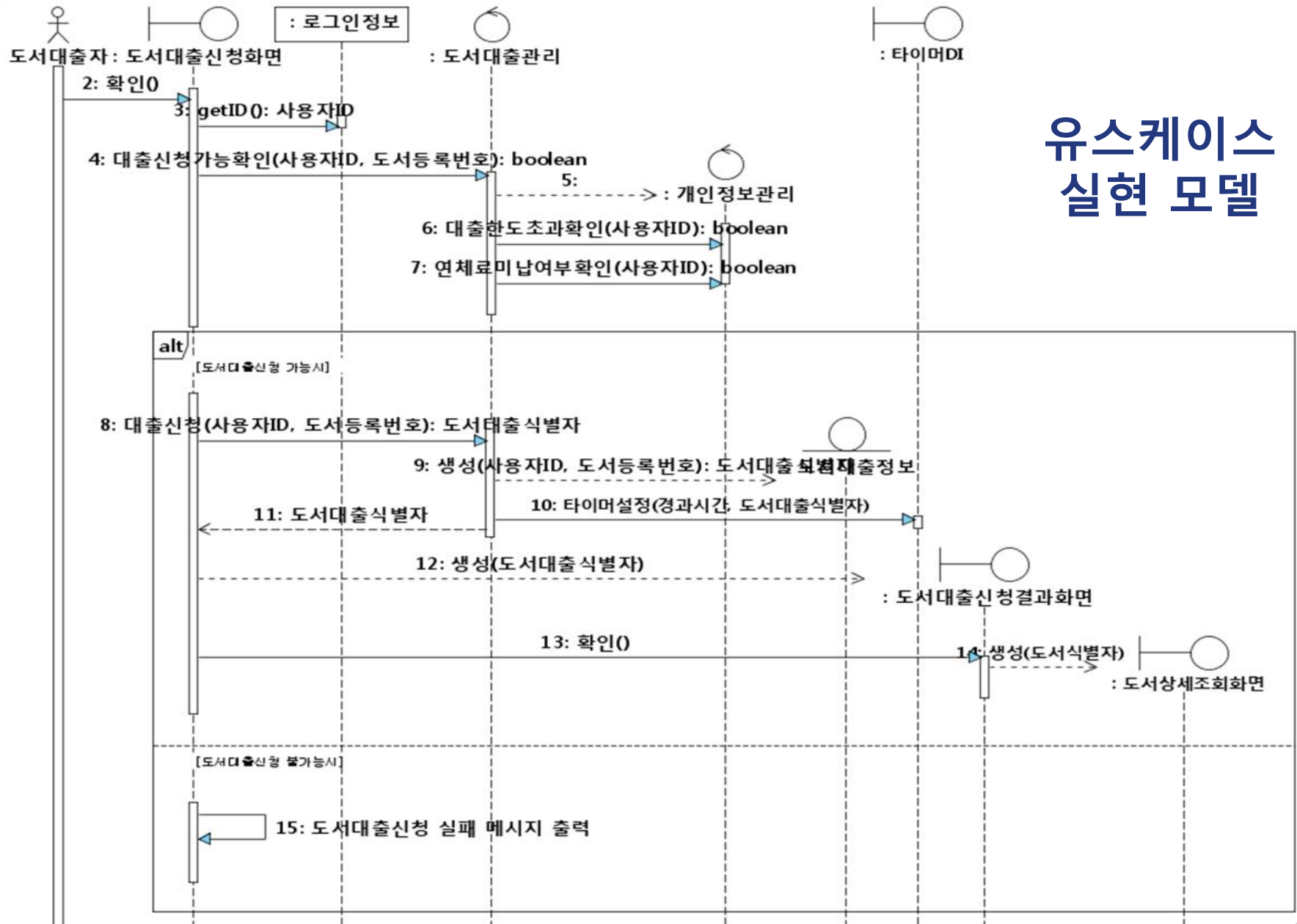


클래스 모델: 도서대출신청 유스케이스

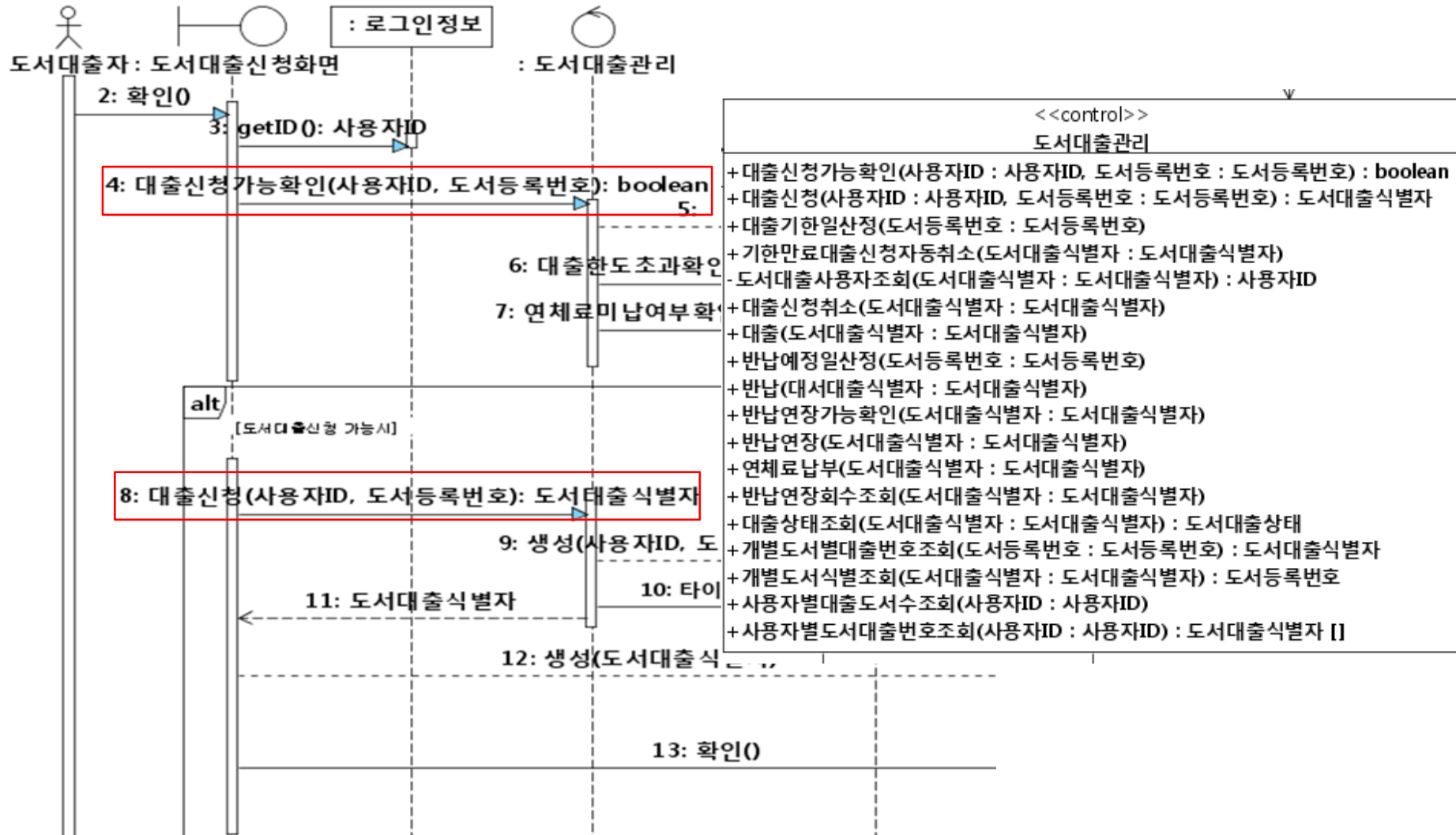




유스케이스 실행 모델



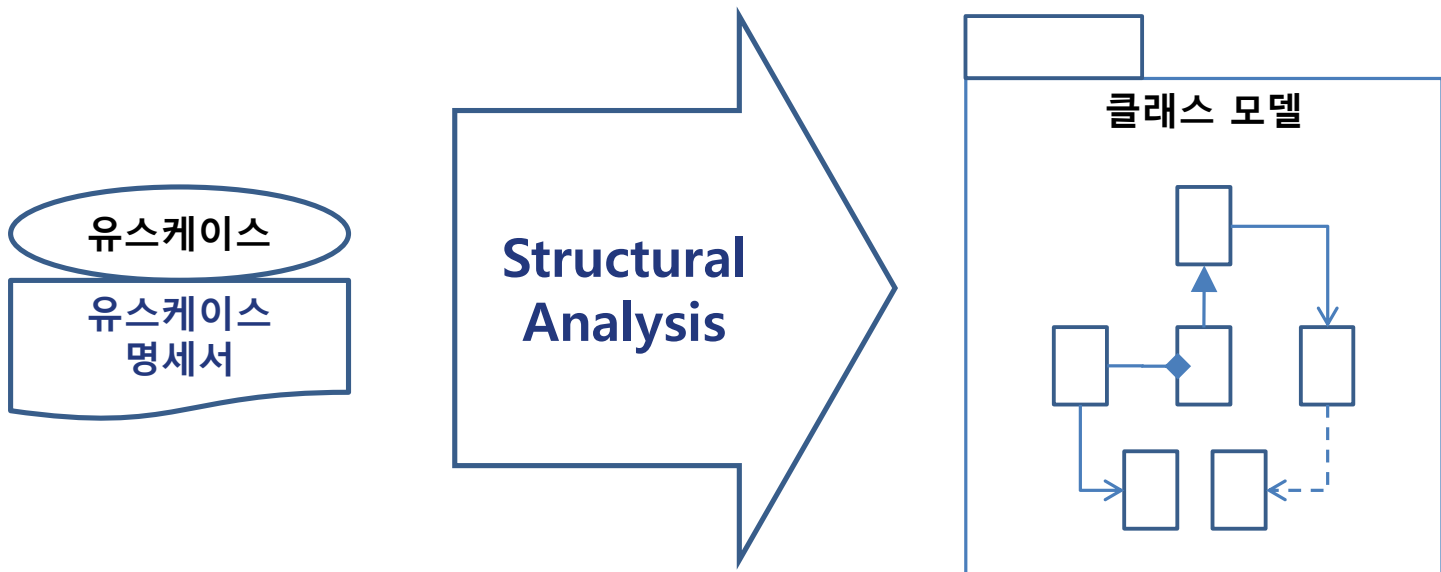
상세화된 클래스 모델 vs 유스케이스 실현 모델



STEP 1. STRUCTURAL ANALYSIS



Step 1. Structural Analysis

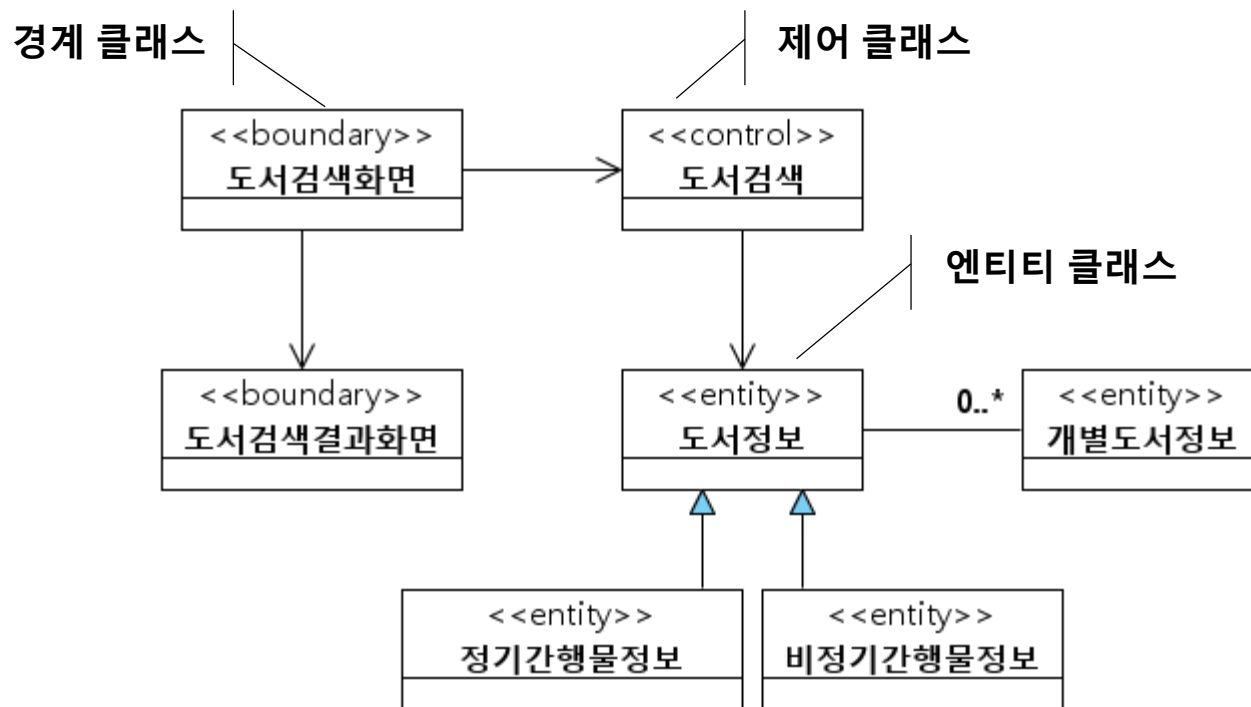


분석 클래스 모델

- ❖ 분석 클래스 모델을 구성하는 분석 클래스들은 그 역할에 따라서 분류됨
- ❖ 경계 클래스(boundary class)
 - 시스템과 외부 액터와의 상호작용을 전담하는 클래스이다
 - 시스템의 기능 중에서 입력과 출력만을 전담하는 클래스이다
- ❖ 제어 클래스(control class)
 - 시스템이 실제로 제공하는 비즈니스 로직 및 제어 로직을 전담하는 클래스이다.
- ❖ 엔티티 클래스(entity class)
 - 시스템이 유지해야 하는 persistent 데이터를 관리하는 기능을 전담하는 클래스이다.
 - Persistent 데이터는 시스템이 종료되어도 그 값이 유지되어야 하는 데이터를 말하며 파일 또는 데이터베이스 등으로 구현된다

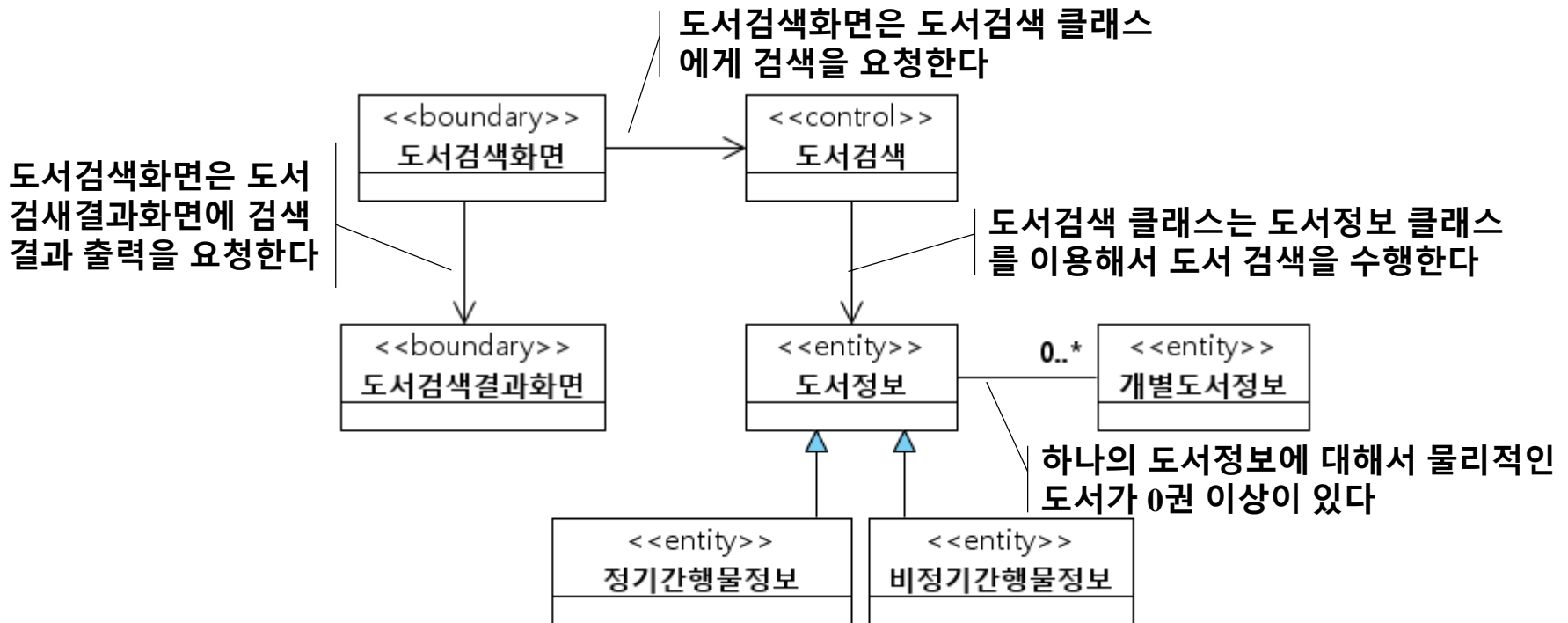
클래스 모델

❖ 소장도서검색 유스케이스



클래스 모델

❖ 클래스 간의 관계

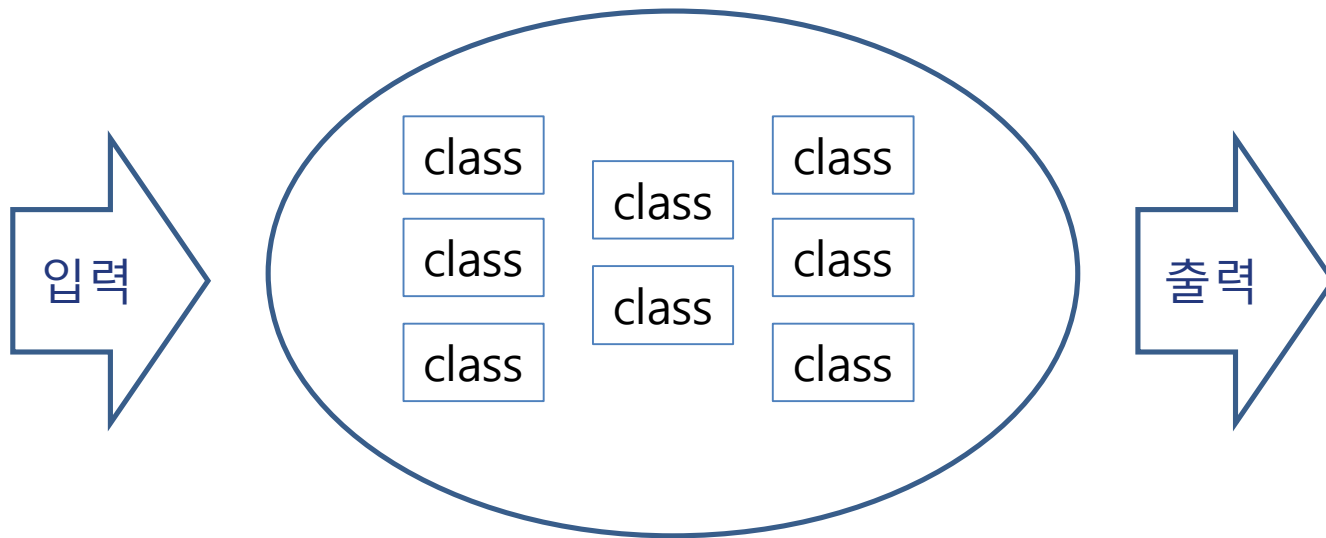


기본 개념



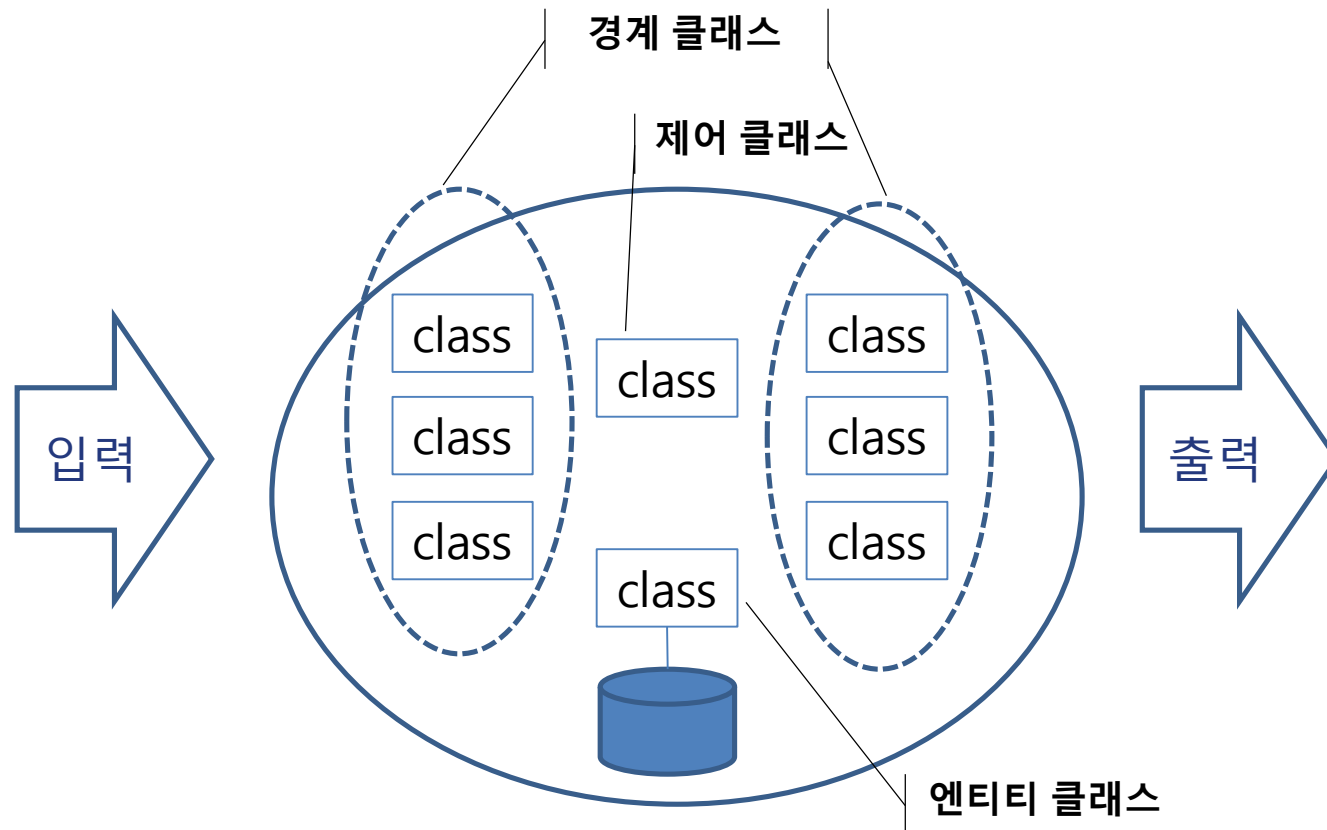
클래스

- ❖ 클래스는 시스템을 구성하는 실체로서 외부 입력으로부터 출력을 산출하는 실질적인 기능을 제공한다.

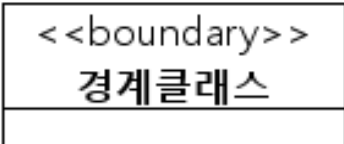
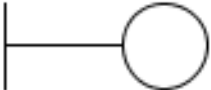
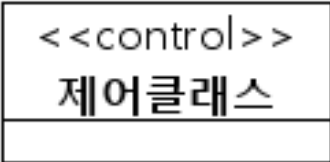

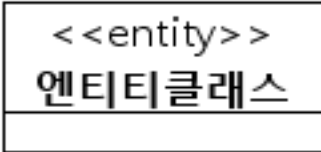


분석 클래스의 유형

- ❖ 시스템의 기능은 경계 클래스, 제어 클래스, 엔티티 클래스를 통하여 제공된다.

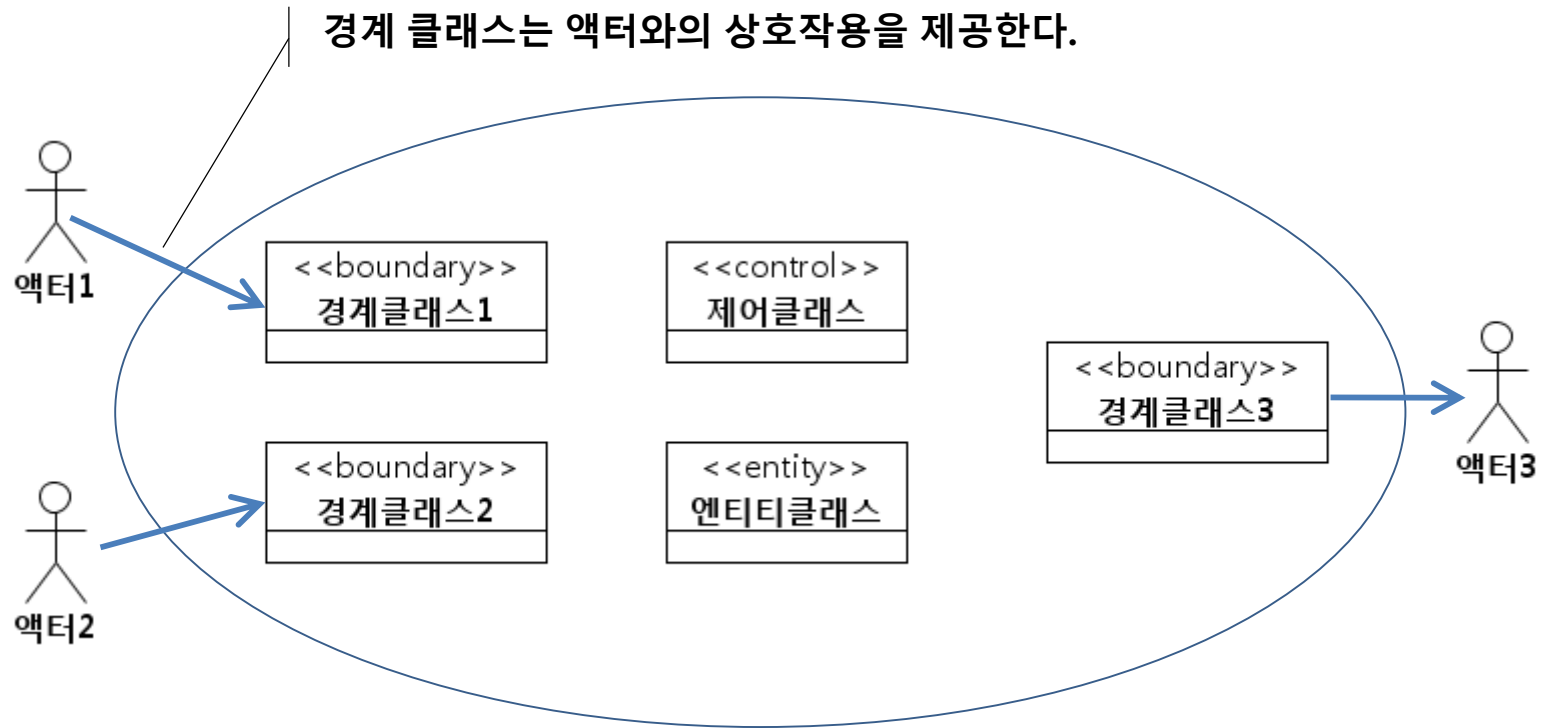


분석 클래스 표현법

분석 클래스 유형	스테레오타입	아이콘
경계 클래스		 경계클래스
제어 클래스		 제어클래스
엔티티 클래스		 엔티티클래스

경계(boundary) 클래스

❖ 액터와의 상호작용을 제공하는 클래스이다.

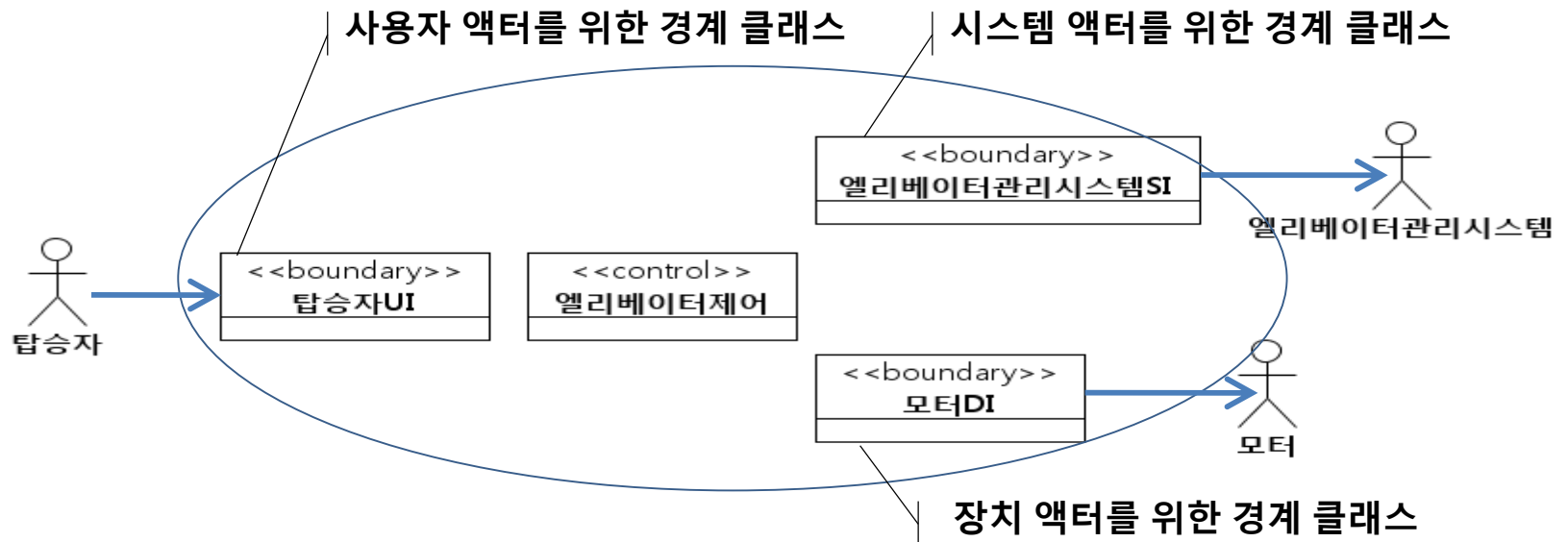
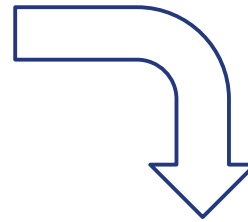
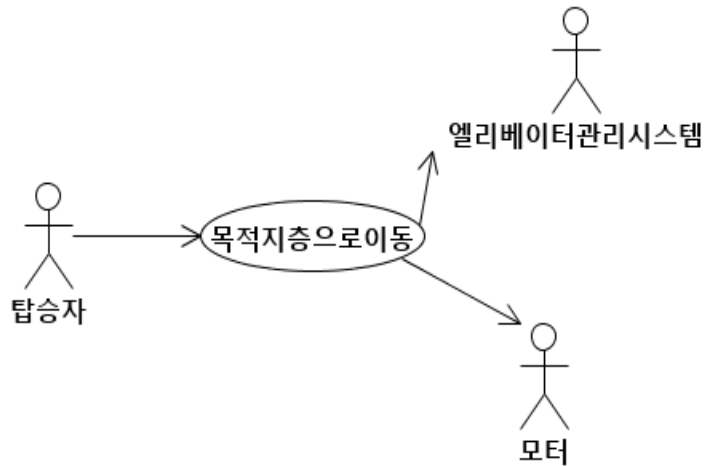


경계 클래스의 유형

❖ 액터의 유형에 따라서 경계 클래스는 세분화될 수 있다.

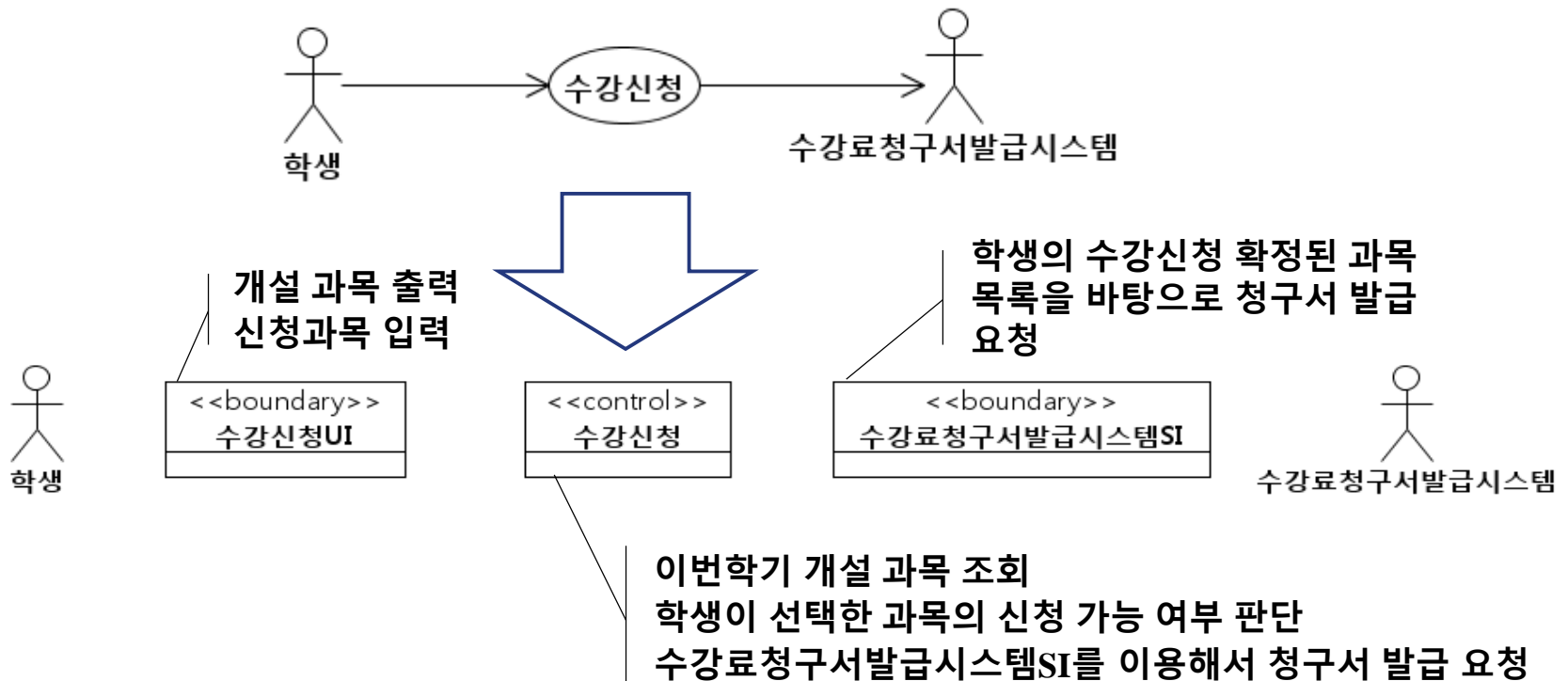
경계 클래스 유형	액터 유형	기능/역할
UI 클래스	사용자	사용자와의 입/출력
SI 클래스	시스템	외부 시스템과의 데이터 송/수신
DI 클래스	장치	장치의 모니터링 및 제어

경계 클래스의 유형



제어(control) 클래스

- ❖ 유스케이스의 비즈니스/제어 로직을 제공한다.
 - 경계 클래스를 통하여 입력 받은 값을 정의된 요구사항에 따라서 적절한 출력 값을 산출하는 기능을 제공한다.



엔티티(entity) 클래스

- ❖ 오랜 시간 동안(long-lived) 또는 영속적(persistent)으로 그 값이 유지되어야 하는 데이터에 대한 관리

시스템의 유형	영속적인 데이터의 예
수강신청 시스템	학생 정보 교수 정보 개설된 교과목 정보 학생이 수강 신청한 교과목
도서관리 시스템	각 도서에 대한 정보 사서 정보 대출자 정보 대출 정보



엔티티 클래스

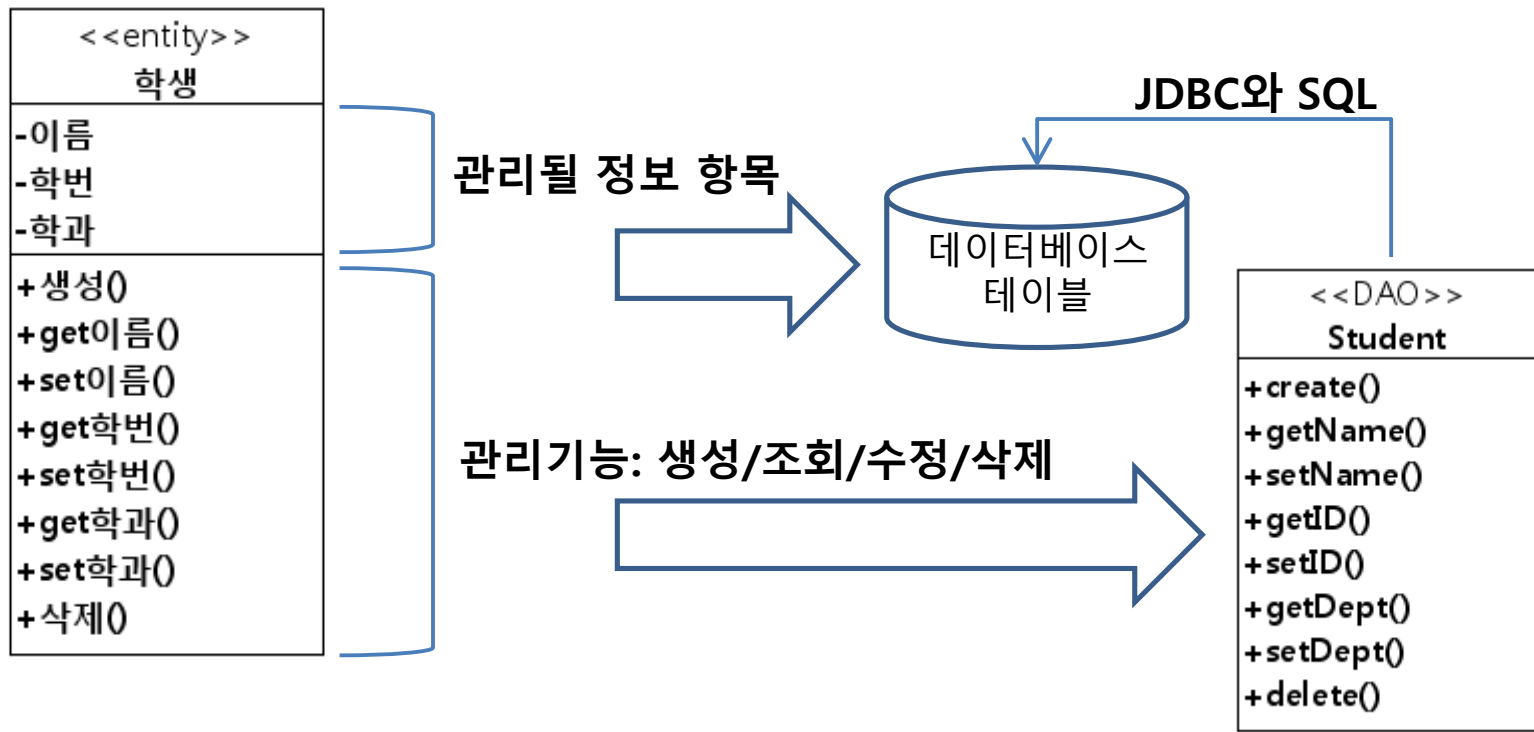
- ❖ 분석 단계에서는 영속적인 데이터를 도출하여 엔티티 클래스로 표현한다.

시스템의 유형	엔티티 클래스			
수강신청 시스템	<div><<entity>> 학생</div>	<div><<entity>> 교수</div>	<div><<entity>> 교과목</div>	<div><<entity>> 수강신청과목</div>
도서관리 시스템	<div><<entity>> 도서</div>	<div><<entity>> 사서</div>	<div><<entity>> 대출자</div>	<div><<entity>> 대출</div>



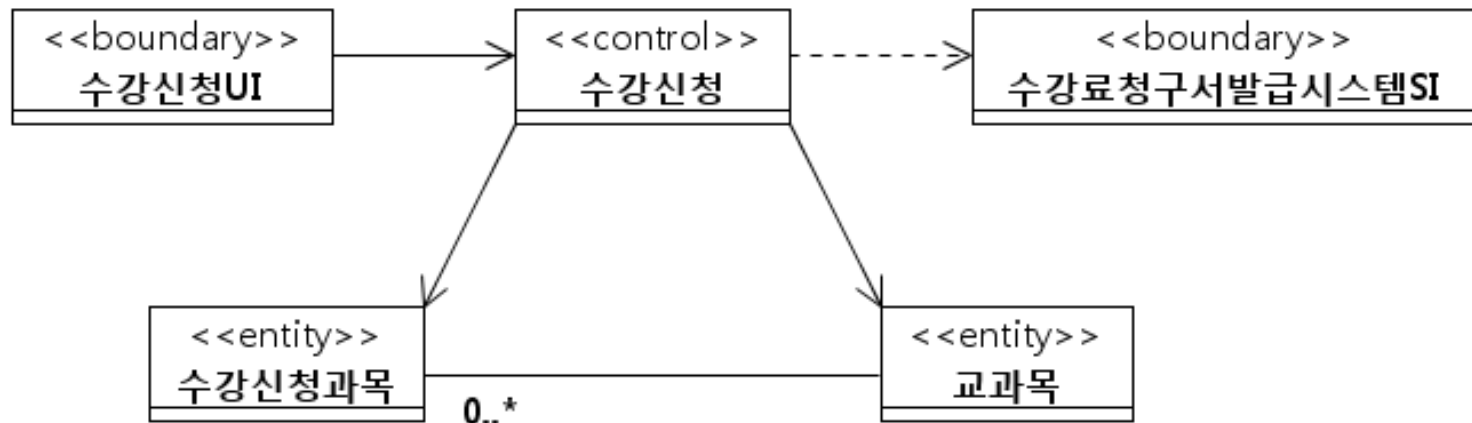
엔티티 클래스의 속성/연산

- ❖ 속성: 엔티티 클래스가 관리할 정보 항목
- ❖ 연산: 데이터 관리 기능



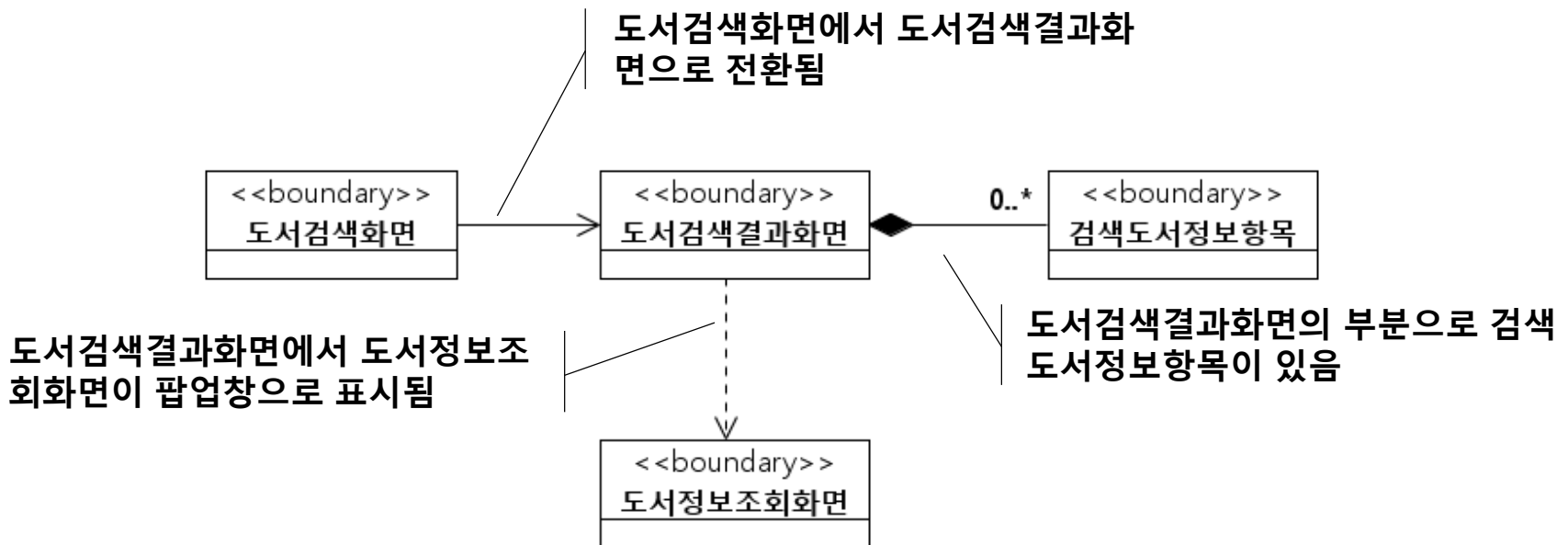
클래스 간의 관계

- ❖ 각 클래스는 유스케이스의 전체 기능의 일부만을 제공하므로 여러 클래스와의 협력(collaboration)이 필요
- ❖ 클래스 간의 협력을 위해서는 클래스 간에 메시지 전달을 위한 관계(relationship)(association or dependency)가 필요



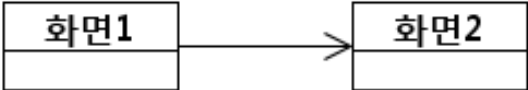

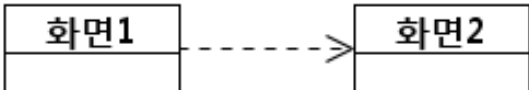
UI 클래스 간의 관계

- ❖ UI 클래스는 사용자와의 인터페이스를 위한 각 화면에 해당된다.
- ❖ UI 클래스 간의 관계는 사용자 인터페이스 화면 간의 관계를 표현한다.



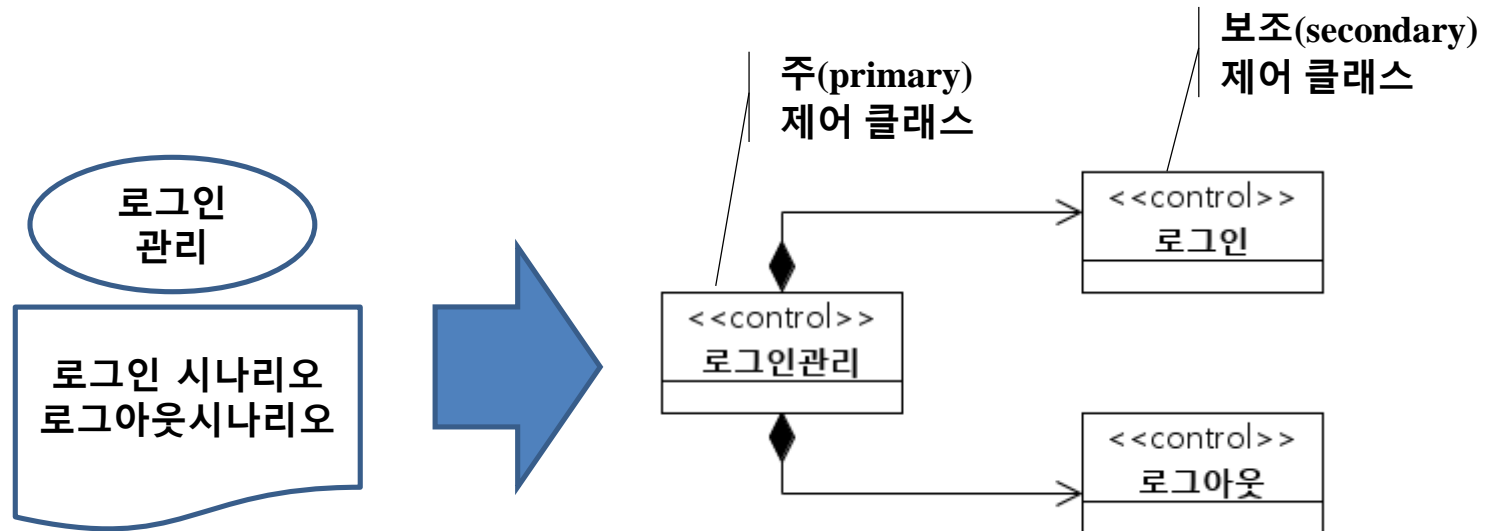
UI 클래스 간의 관계

❖ UI 클래스 간의 관계 요약

UI 클래스 간의 관계	의미
	화면1에서 화면2로 전환됨을 뜻한다. 즉 사용자는 더 이상 화면1을 볼 수 없다.
	화면2가 화면1의 한 부분임을 뜻한다.
	화면1 상에서 화면2가 팝업창으로 표시됨을 뜻한다. 즉 화면1도 출력되고 있다. 그러나 화면2가 닫히기 전에는 화면1을 사용할 수가 없다. 즉 화면2는 모달(modal) 대화상자에 해당된다.

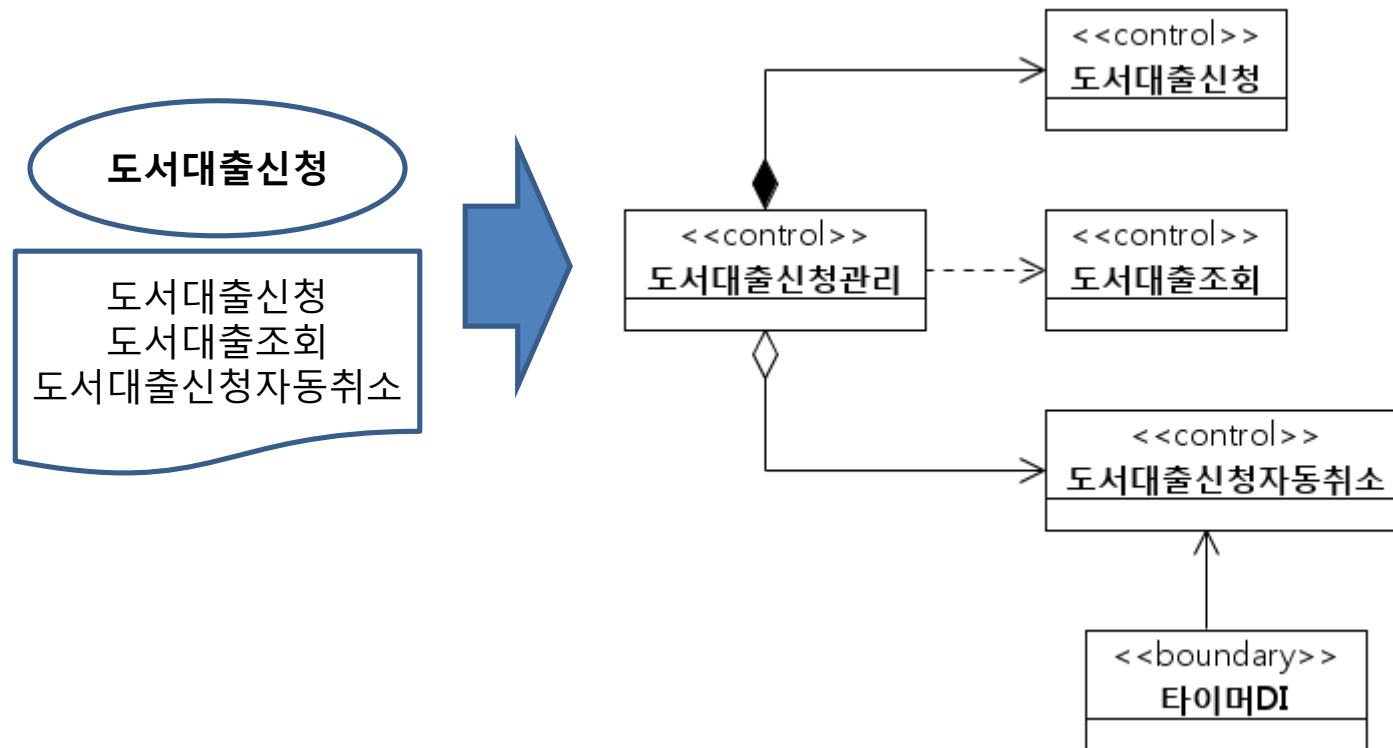
제어 클래스 간의 관계

- ❖ 비즈니스/제어 로직 간의 관계를 표현한다.
- ❖ 원래 하나의 유스케이스의 부분 기능을 보조 제어 클래스가 담당하므로 주 제어 클래스와 보조 제어 클래스 간에는 포함 관계를 이용하여 표현할 수 있다.



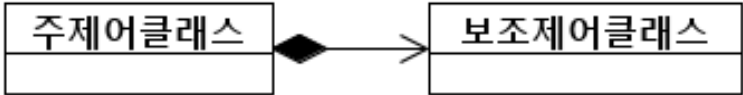
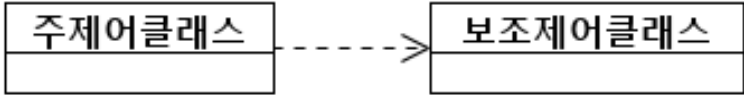
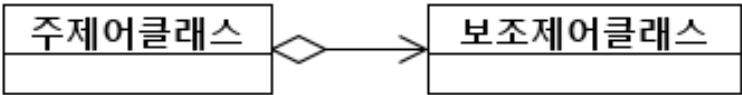
제어 클래스 간의 관계

- ❖ 주 제어 클래스와 보조 제어 클래스 간에는 포함 관계뿐만 아니라, 의존 관계, 집합 관계를 사용될 수 있다.



제어 클래스 간의 관계

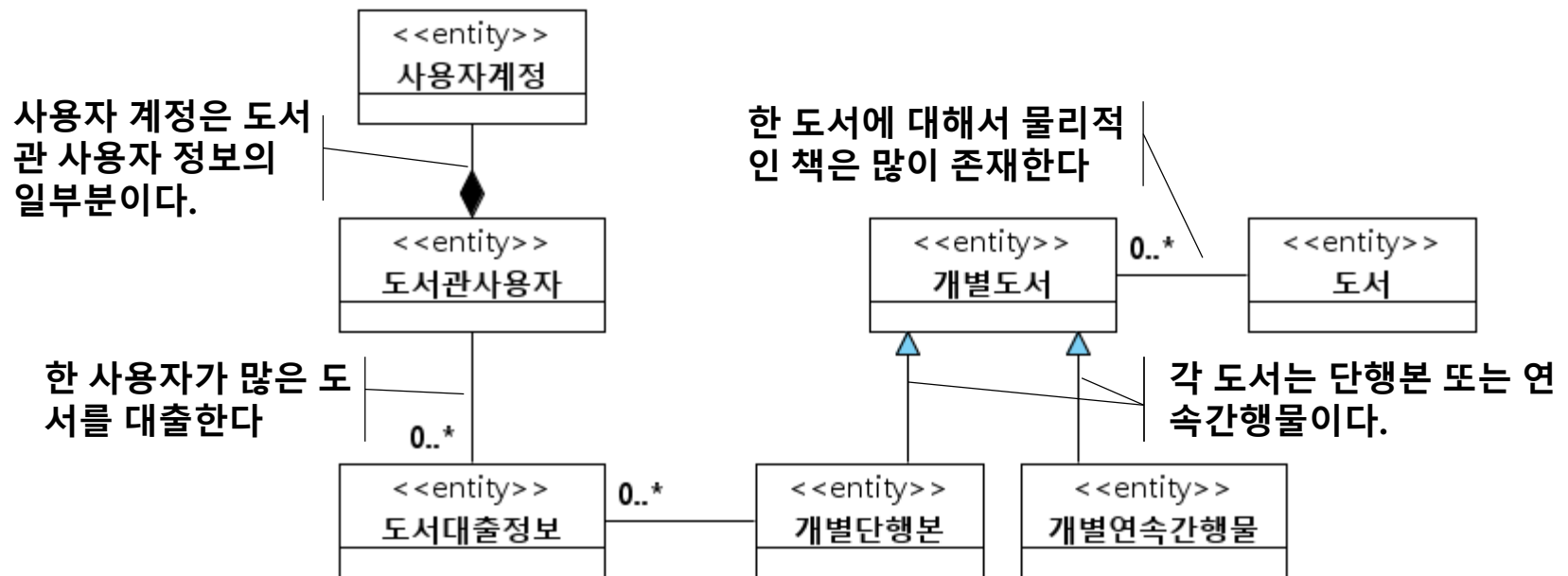
❖ 제어 클래스 간의 관계 요약

제어 클래스 간의 관계	의미
	보조제어클래스가 주제어클래스의 모든 연산에서 사용된다.
	보조제어클래스가 주제어클래스의 특정 연산 속에서만 사용된다.
	보조제어클래스가 주제어클래스 이외의 다른 클래스에서도 사용된다.



엔티티 클래스 간의 관계

- ❖ 영속적인 데이터 간의 관계를 표현한다.
- ❖ 기존의 ER 다이어그램을 이용하여 표현하던 데이터 모델을 클래스 다이어그램을 이용하여 표현할 수가 있다.



엔티티 클래스의 속성 도출

- ❖ 이어서 수행되는 “**Behavioral Analysis**”에서 각 클래스 별 속성과 연산이 정의된다

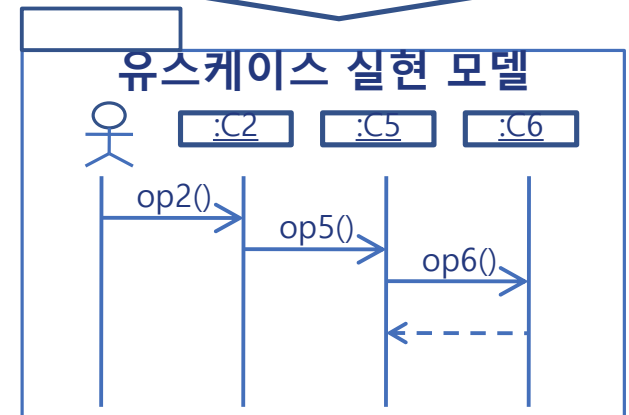
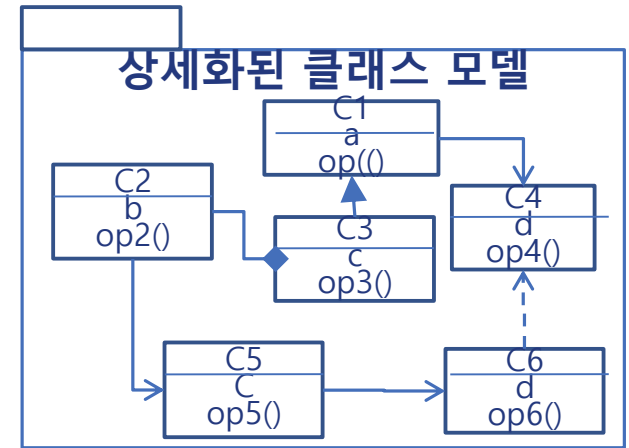
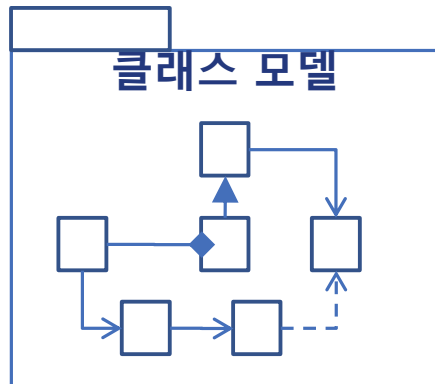
엔티티 클래스	대표적인 속성
도서관사용자	이름, 소속, 연락처
도서	도서식별자, 도서명, 출판사, 출판년도
개별도서	등록번호, 소장처명, 등록일, 폐기일
개별단행본	구입일, 파손여부, 대출여부
개별연속간행물	권호, 발행일
도서대출정보	대출신청일, 대출일, 반납일

STEP 2. BEHAVIORAL ANALYSIS



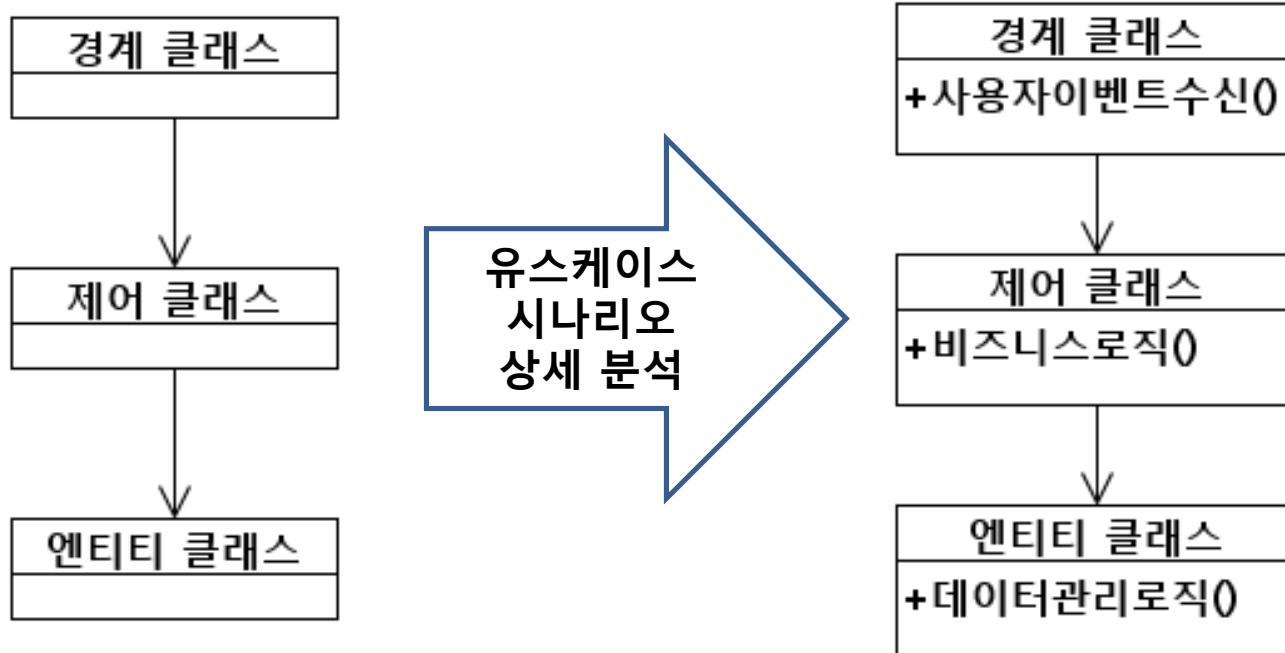
Step 2. Behavioral Analysis

- ❖ 클래스 모델의 구체화: 속성과 연산의 완성
- ❖ 각 유스케이스 별로 실현 모델

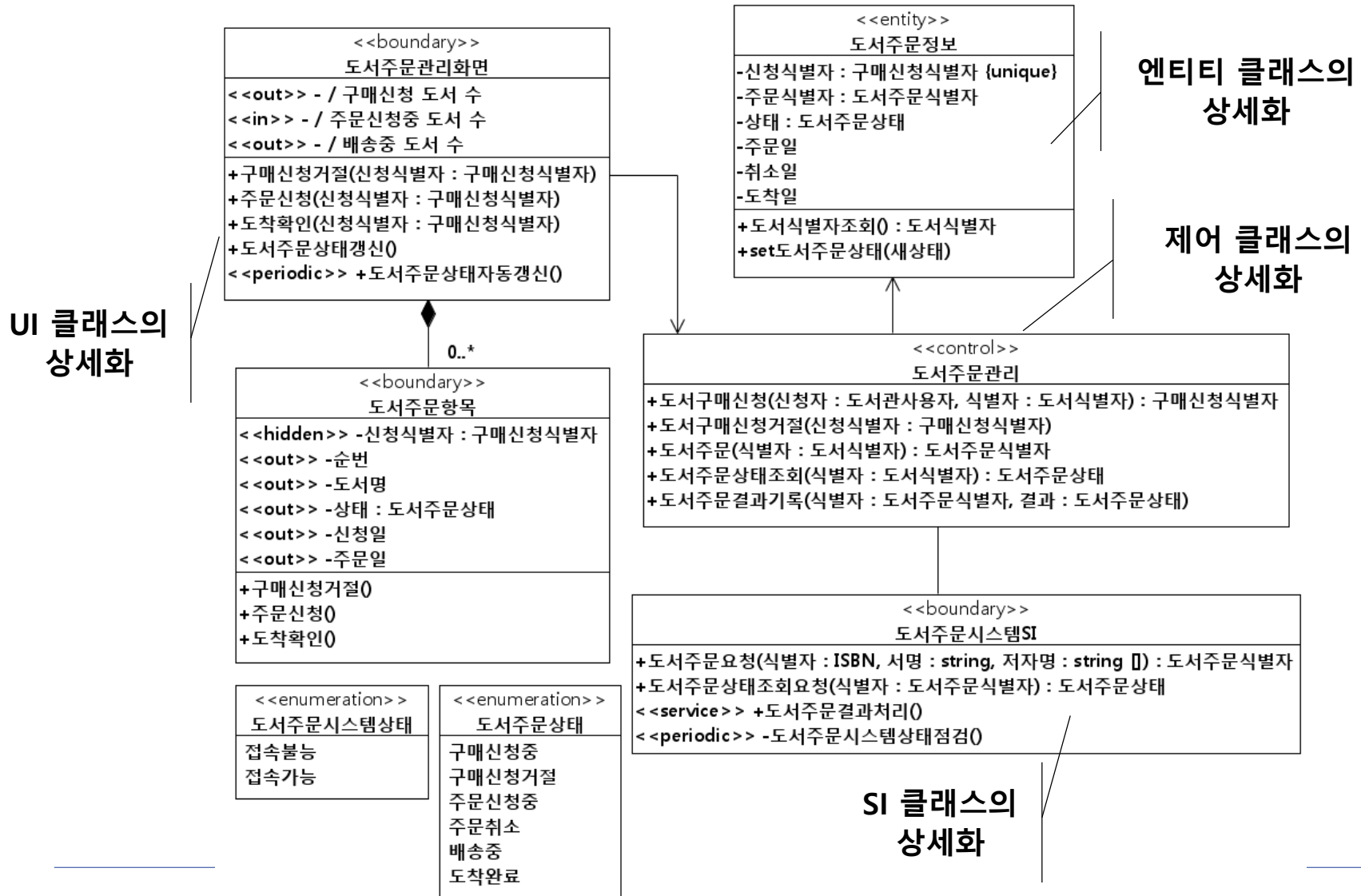


상세화된 클래스 모델

- ❖ 클래스의 유형(boundary, control, entity) 별로 적절한 데이터/기능을 속성/연산으로 정의

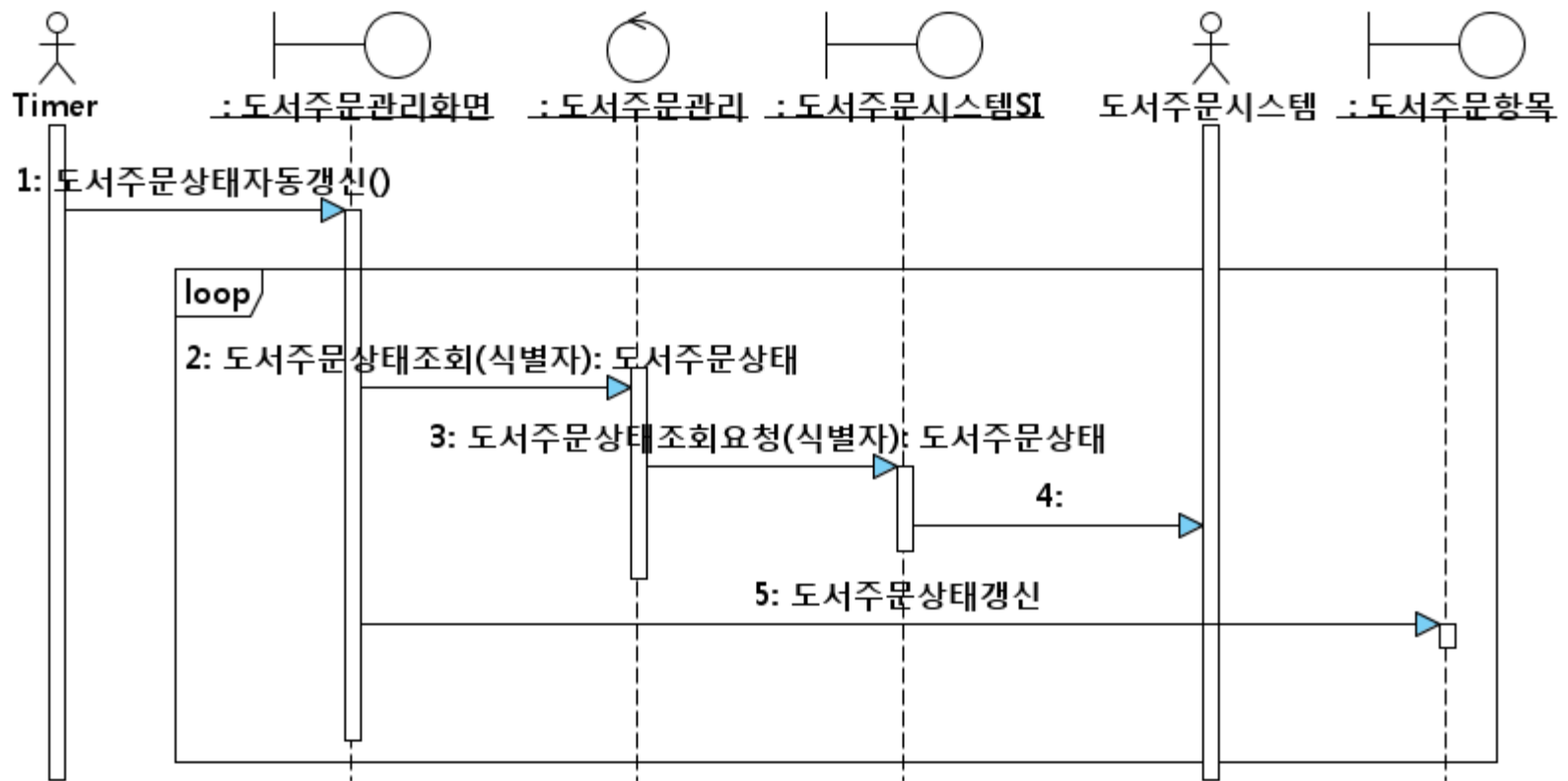


상세화된 클래스 모델 – 도서주문관리 UC



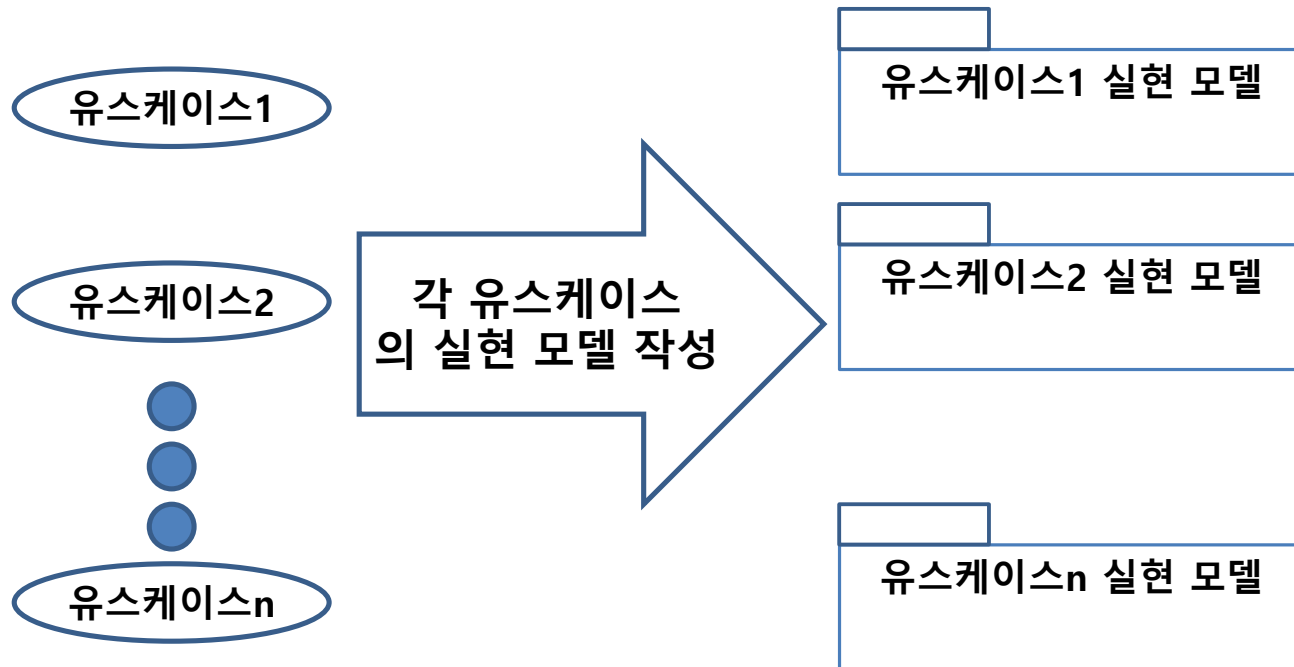
유스케이스 실현 모델 - 도서주문관리 UC

❖ 도서주문상태 자동 갱신 시나리오



유스케이스 실현 모델

❖ 유스케이스의 각 시나리오 별로 실현 모델이 작성



유스케이스 실현 모델

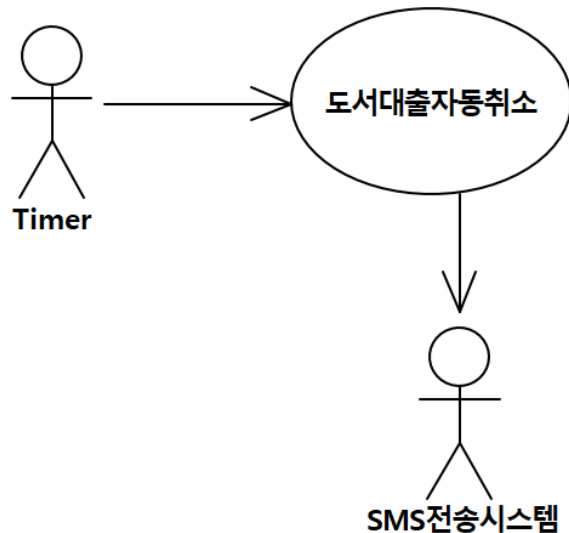
- ❖ 유스케이스 시나리오를 바탕으로 유스케이스 실현 모델을 작성한다.
 - 액터가 주어일 때는 이 액터를 담당하는 경계 클래스를 정의하고 액터로부터의 입력을 수신하는 연산을 정의하도록 한다.
 - 시스템이 주어일 때는 시스템의 제공하는 기능의 유형에 따라서 경계 클래스, 제어 클래스, 엔티티 클래스를 정의한다.
 - 액터와의 출력 기능이면 경계 클래스를 정의하고,
 - 비즈니스/제어 로직이면 제어 클래스를 정의한다.
 - 영속적인 정보를 생성/조회/수정/삭제할 필요가 있으면 엔티티 클래스를 정의한다.



CASE STUDY



예제 – 도서대출자동취소 UC

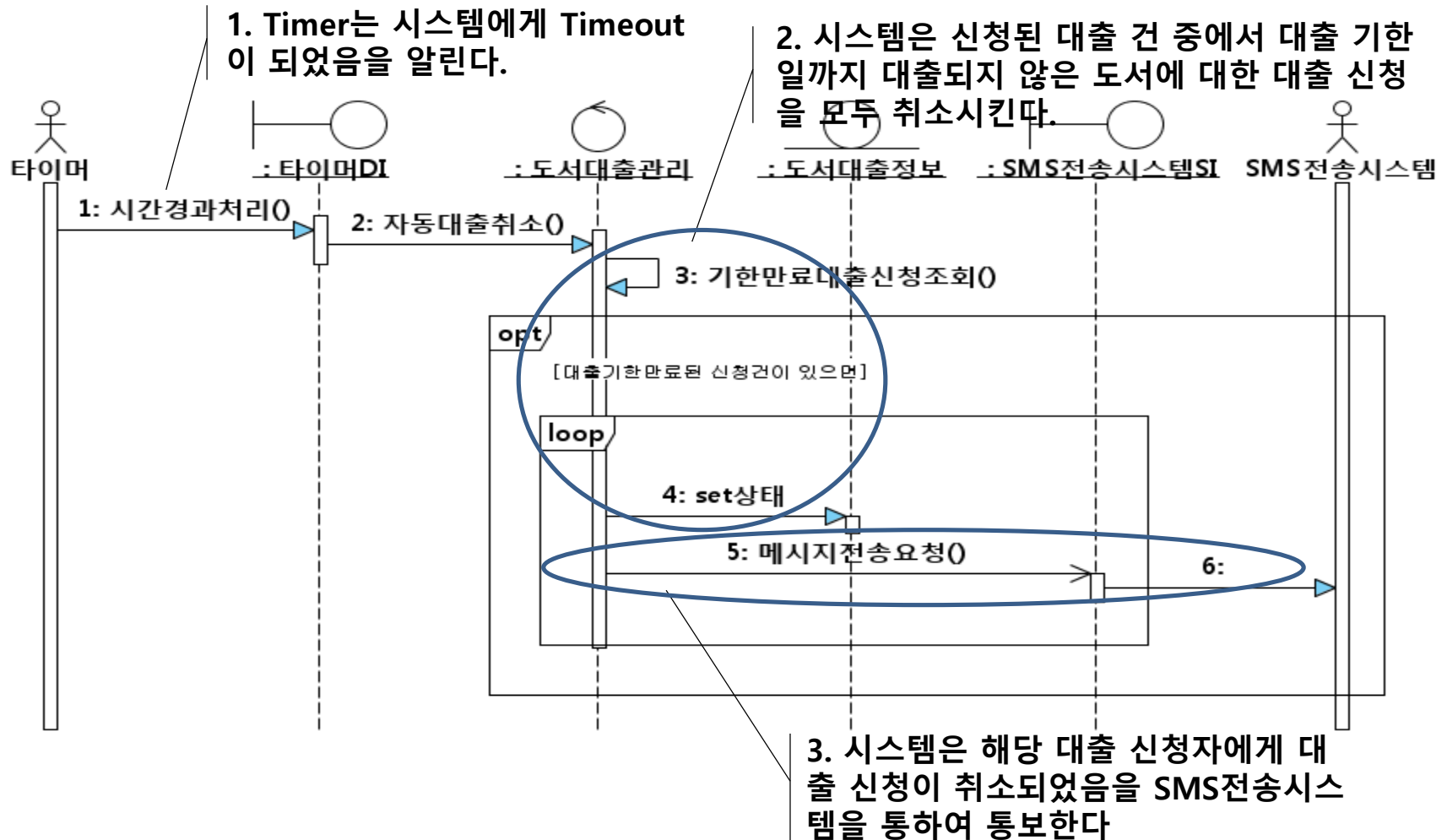


기본 시나리오

1. Timer는 시스템에게 Timeout이 되었음을 알린다.
2. 시스템은 신청된 대출 건 중에서 대출 기한 일까지 대출되지 않은 도서에 대한 대출 신청을 모두 취소시킨다.
3. 시스템은 해당 대출 신청자에게 대출 신청이 취소되었음을 SMS전송시스템을 통하여 통보한다.

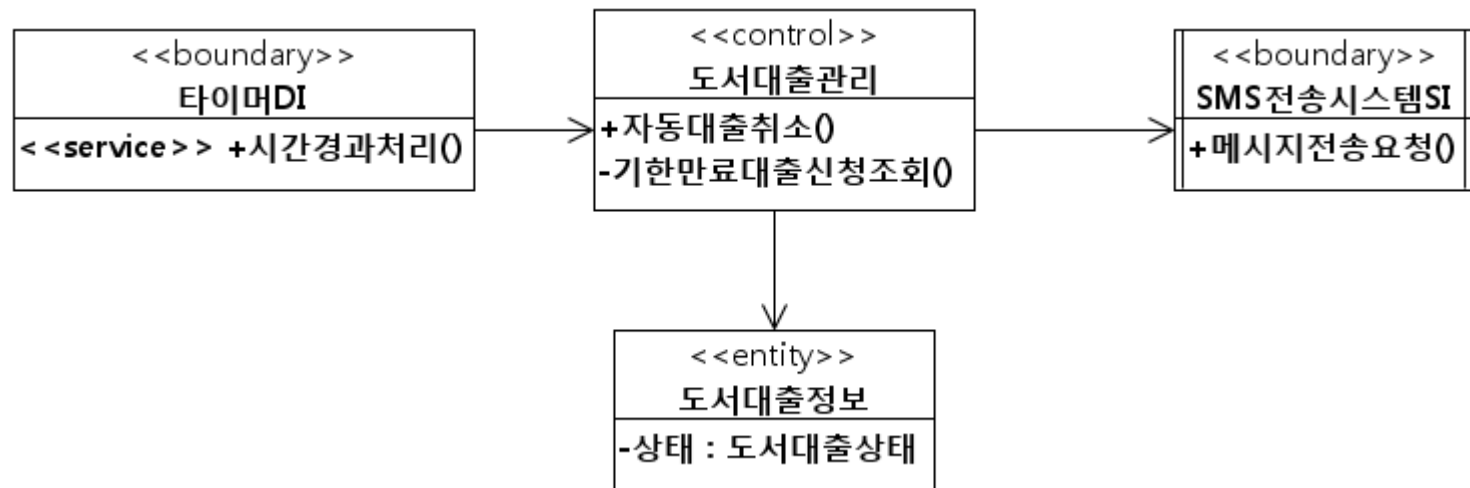


유스케이스 실현 모델: 버전 1



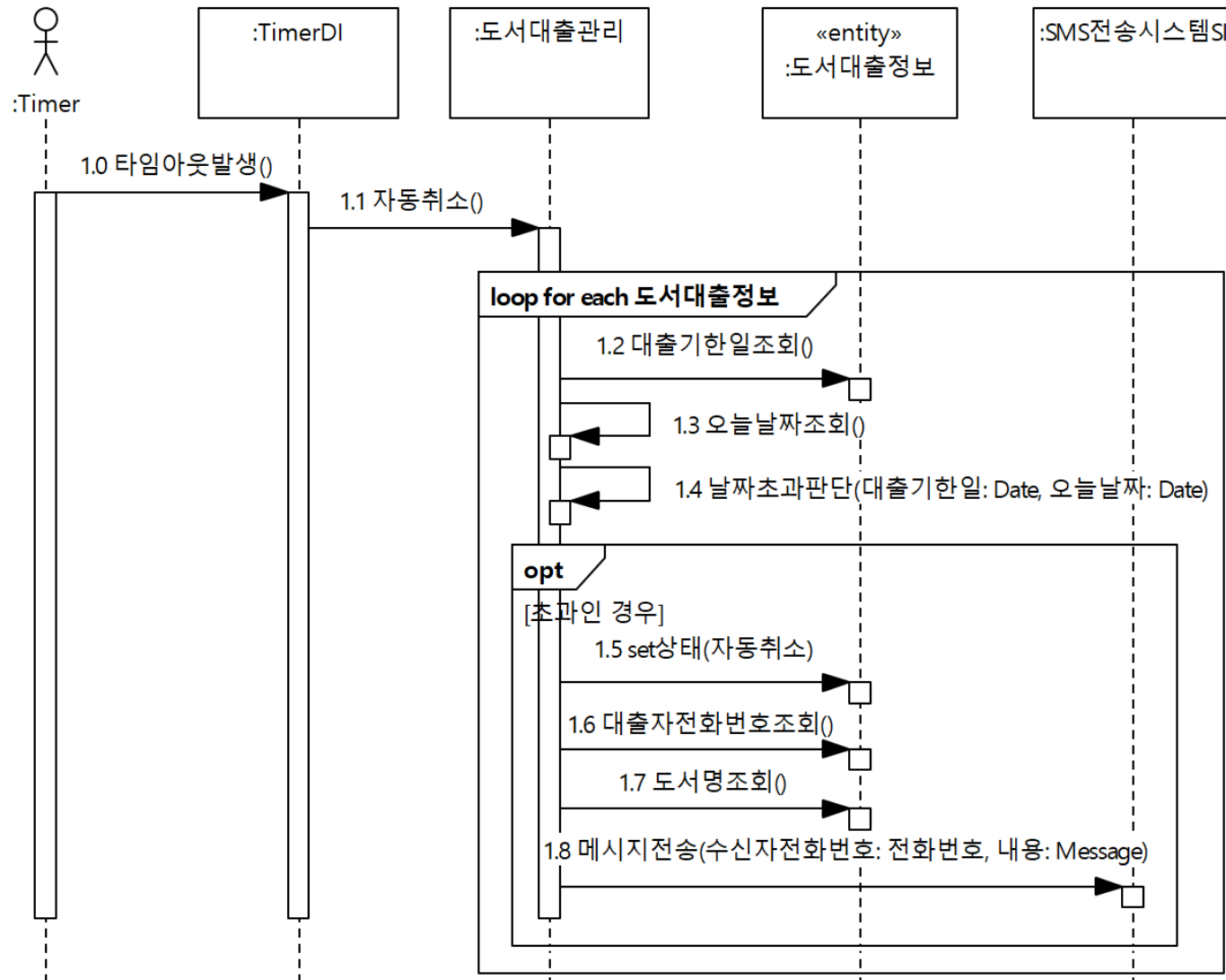
클래스 모델: 버전 1

❖ 도서대출자동취소의 클래스 모델

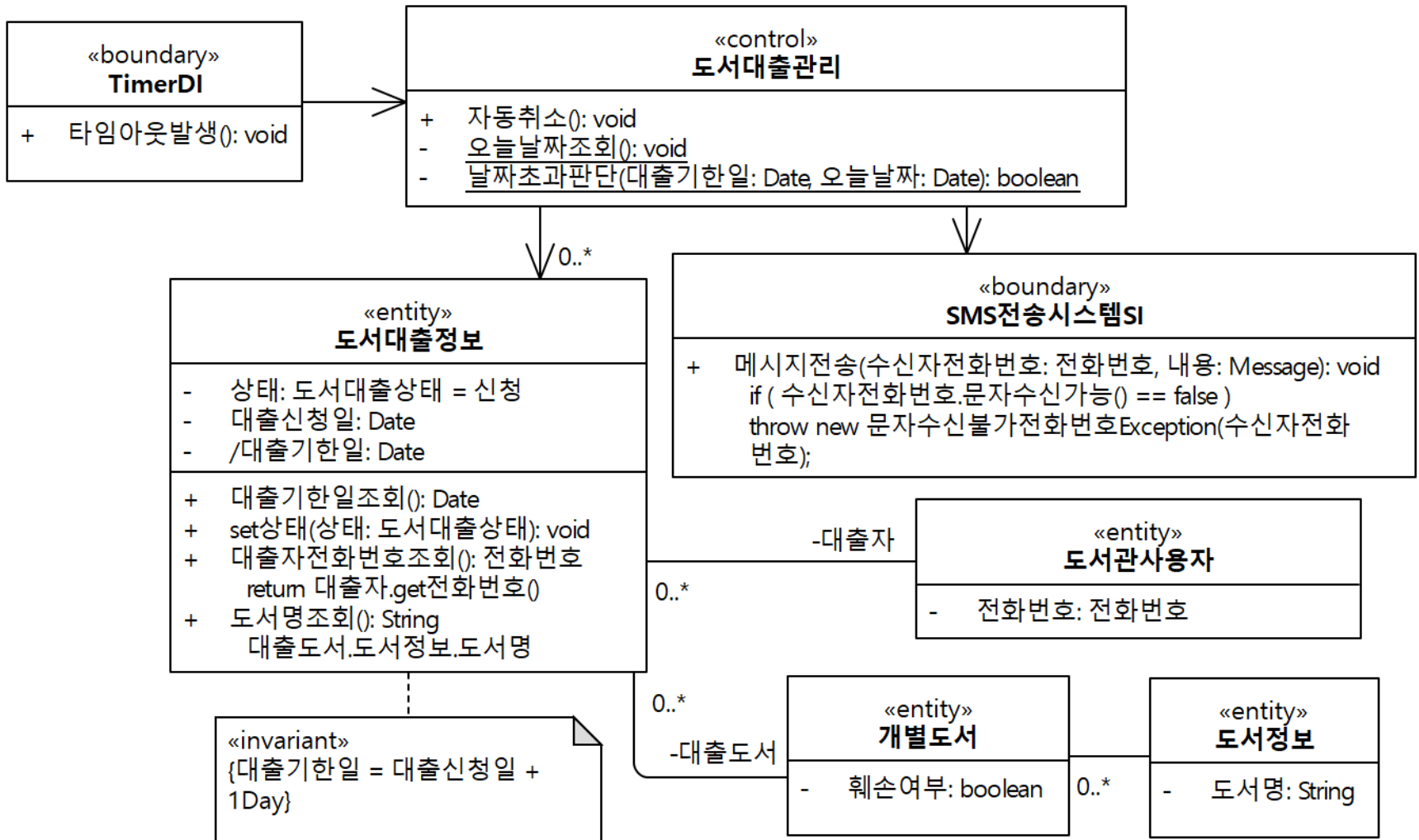


VERSION 2

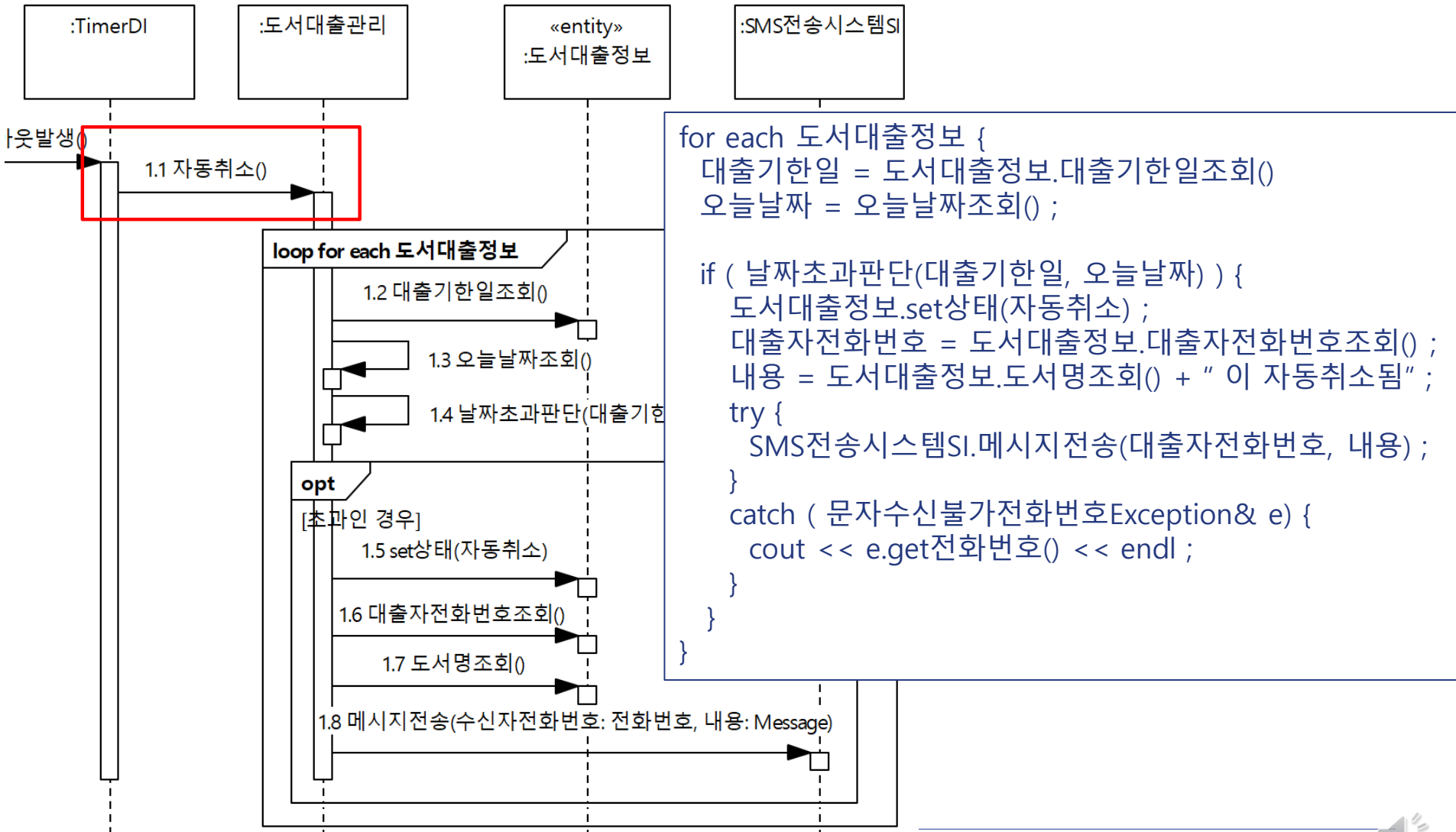
유스케이스 실현 모델



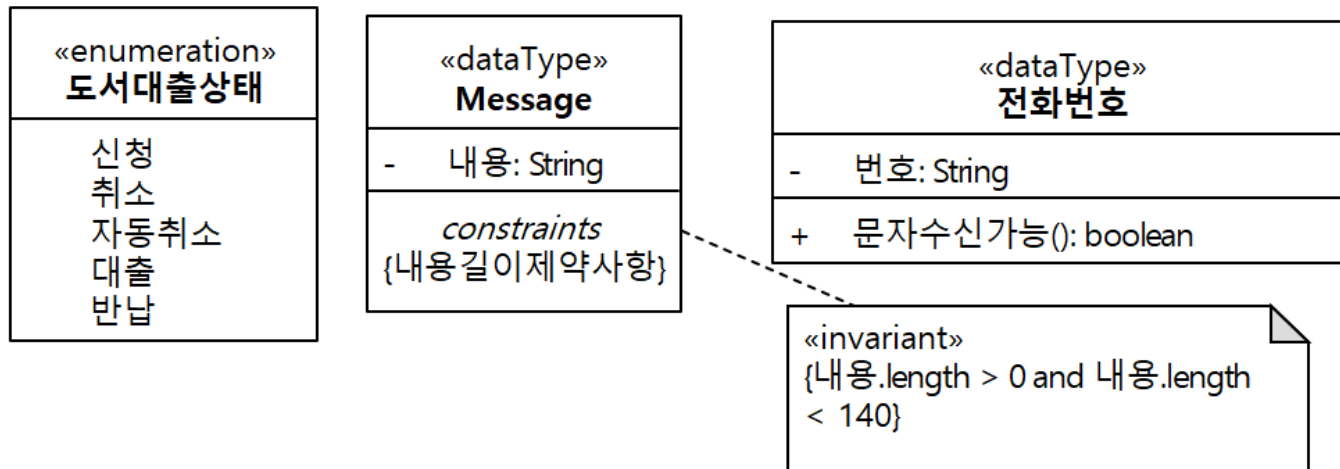
클래스 모델



도서대출관리 Pseudo Code



클래스 모델: Type

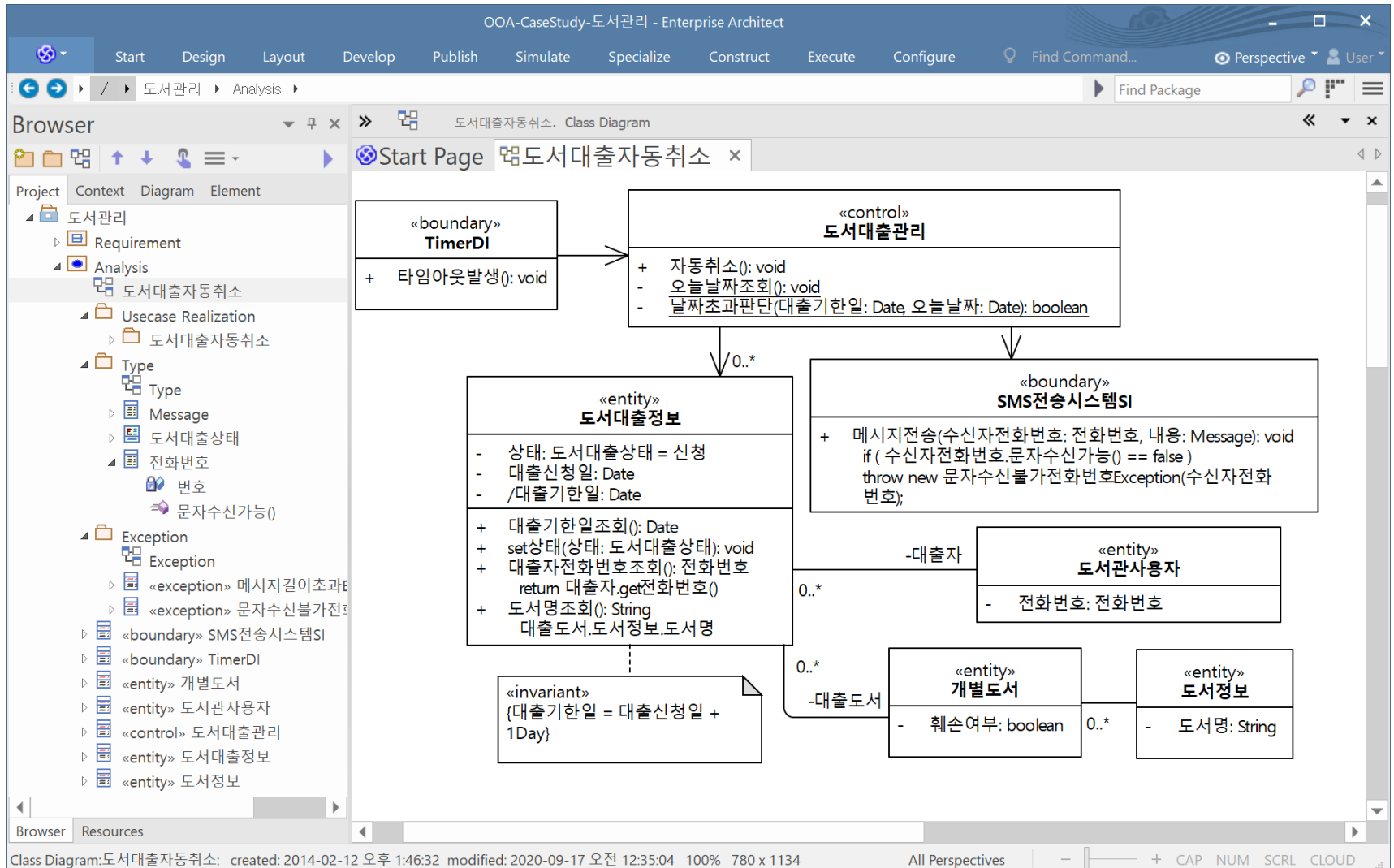


클래스 모델: Exception

«exception» 문자수신불가전화번호Exception	
-	전화번호: 전화번호 {readOnly}
+	<u>문자수신불가전화번호Exception(전화번호): void</u>

«exception» 메시지길이초과Exception	
-	길이: int {readOnly}
+	<u>메시지길이초과Exception(int): void</u>

Model Structure



Q&A
