



제2장 요구모델링



초기작업

- 초기에 해야할 작업
 - 개발하려는 시스템은 어떤 것인가?
 - 이번의 개발에서는 어디까지 구현할 것인가?
- 실제로, 이런 작업은 단순하지 않다.
 - 시스템의 범위 및 형태가 불명확, 계속적으로 수정될 수도...
- 그러나, 무계획적으로 개발할 수는 없다!



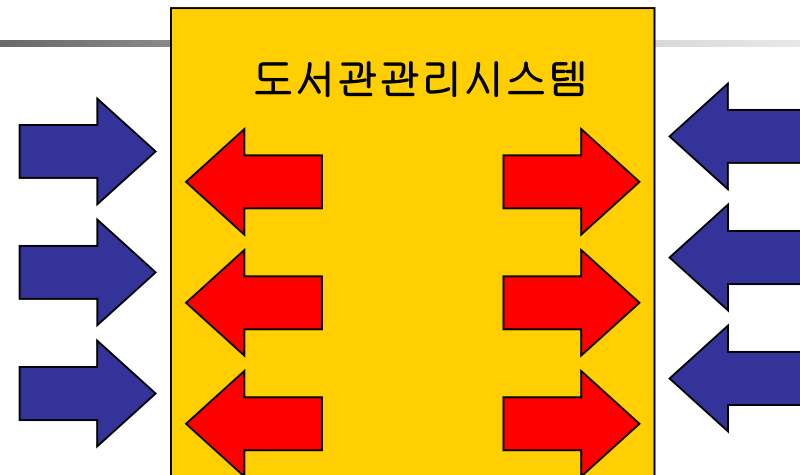
그래서...



도서관관리시스템

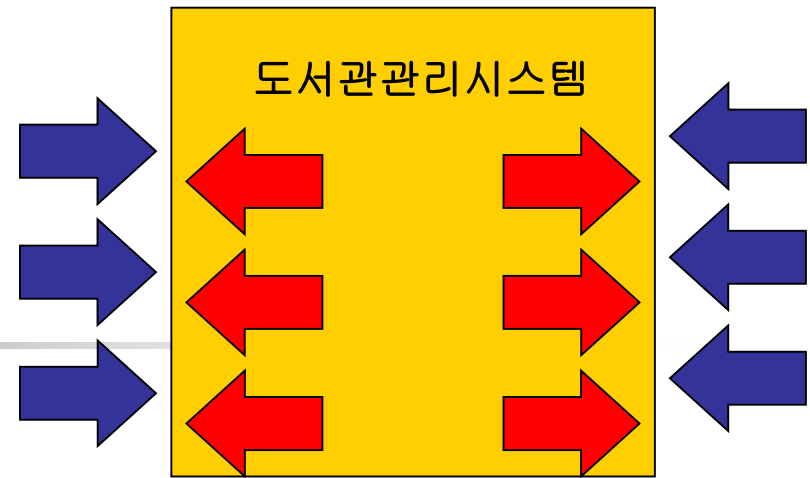
- 우선은, 개발하려는 시스템의 경계 표시
- 이런 사각형을 fractal로 변형→...→ 현실세계의 시스템으로 근접시킴.

시스템을 개발하는 2가지 힘



- 외부로부터의 힘(관점)
 - 이 시스템은 현실세계에서 어떻게 사용되나?
- 내부로부터의 힘(관점)
 - 이 시스템은 어떻게 구축하나?

Use Case View



- 외부로부터의 힘
 - 사용자 및 현실세계로부터의 관점
- 고려사항
 - 이 시스템에 관련있는 사용자는?
 - 그 사용자는 이 시스템을 어떻게 사용하나?
- 도면그리기
 - Use Case Diagram을 중심으로
 - Activity Diagram, Sequence Diagram으로 보조
- 누가 도면을 그리나?
 - 개발자, 사용자, 고객 등이 협력하여...



A시립도서관

■ 현재 : 비전산화 상태

- A시립도서관에서는, 아직 도서관직원이 카운터에서 서적 대출목록을 대조하면서 대출업무 수행

■ 우선, “收藏資料 貸出業務의 電算化”가 목표

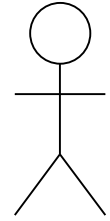
- 카운터에서 도서관 직원이 단말기 이용하여 대출업무 수행
- 추후에는, 이용자가 단말기 조작하여 원격예약도 가능하도록
- 대출에 관한 규칙 : 도중에 변경될 수도...
 - 한번에 대출가능한 수량 : 서적(10권이하), 음악 및 영상물(5개이하)
 - 대출기간 : 2주 이내
- 소장자료의 등록 및 이용자 등록도 필요

Actor in Real World

1. 배우, 남자 배우(opp. actress);
'연극'을 잘 하는 사람
a film actor 영화 배우
2. (사건의) 관여자, 관계자, 장본
인;행위자, 행동자



Actor in OO SD with UML



Actor Name

- 무엇을 Actor로 선정할까?
 - 개발시스템의 직접적인 사용자
 - 개발시스템의 간접적인 이해관계자
 - 개발시스템과 통신하는 다른 시스템
- Actor의 역할명 ➔ Actor Name
 - 모델링에서는 우수한 어휘사용능력이 필요
 - 일단은 간단히 명명,
 - 추후에 더좋은 이름이 떠오르면 변경가능

도서관관리시스템

문제 : 도서관관리시스템의 Actor는?

■ 후보Actor

■ 이용자

- 도서관을 실제 이용하는 사람
- 도서관의 소장자료를 열람/대출하는 사람

■ 도서관직원

- 이용자에게 서비스를 제공하는 사람
- 도서대출 및 검색을 도와주는 사람
- 소장자료의 등록 등의 내부작업을 수행하는 사람

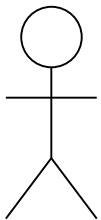
■ 도서관

- 이용자에게 서비스를 제공하는 조직
- 도서관직원은 도서관에 소속되어 있음.

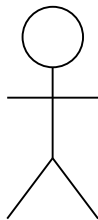
도서관관리시스템

Actor들을 표형태로 정리 & 표기

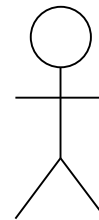
Actor #	이름	설명	비고
1	이용자	도서관을 실제로 이용하는 사람	
2	도서관직원	이용자에게 서비스를 제공하는 사람	도서관에 소속
3	도서관	이용자에게 서비스를 제공하는 조직	



이용자



도서관직원



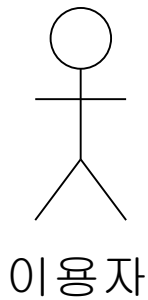
도서관



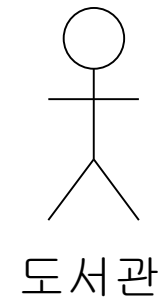
문제: “수장자료”는 Actor인가?

- No!
- “수장자료”는 개발시스템의 외부요소이긴 하지만, 비능동적!

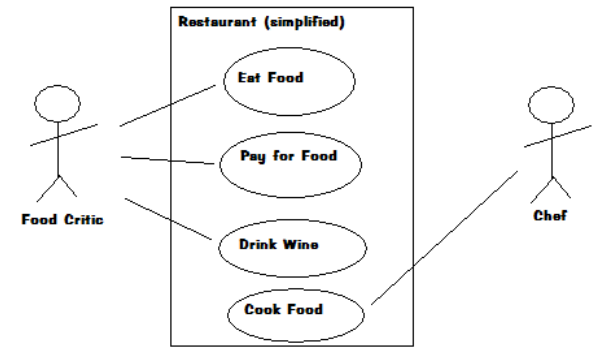
UC Dgm에 Actor추가



도서관관리시스템

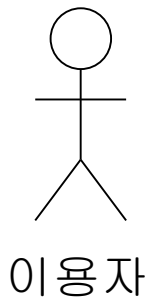


Use Case Diagram

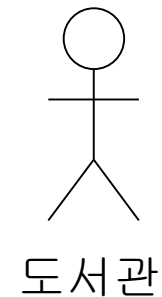


- Ivar Jacobson이 고안(1980년대 중반)
- 스웨덴語 → "anvendningsfall"
 - "사용상황"(situation of usage)
 - "사용사례"(usage case)
- 보다 자세한 사항은
 - http://alistair.cockburn.us/index.php/Use_cases%2C_ten_years_later

이것으로 “요구모델링” 종료??



도서관관리시스템

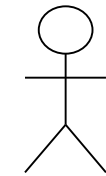


(1) 시스템 내부요소 파악/표현

(2) 요소들 사이의 관계를 파악하여 표현해야...

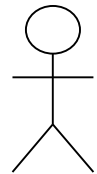
Actor들 사이의 관계파악

- 도서관 vs. 도서관직원
- 도서관직원 vs. 이용자
- 이용자 vs. 도서관

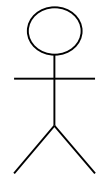


이용자

도서관관리시스템



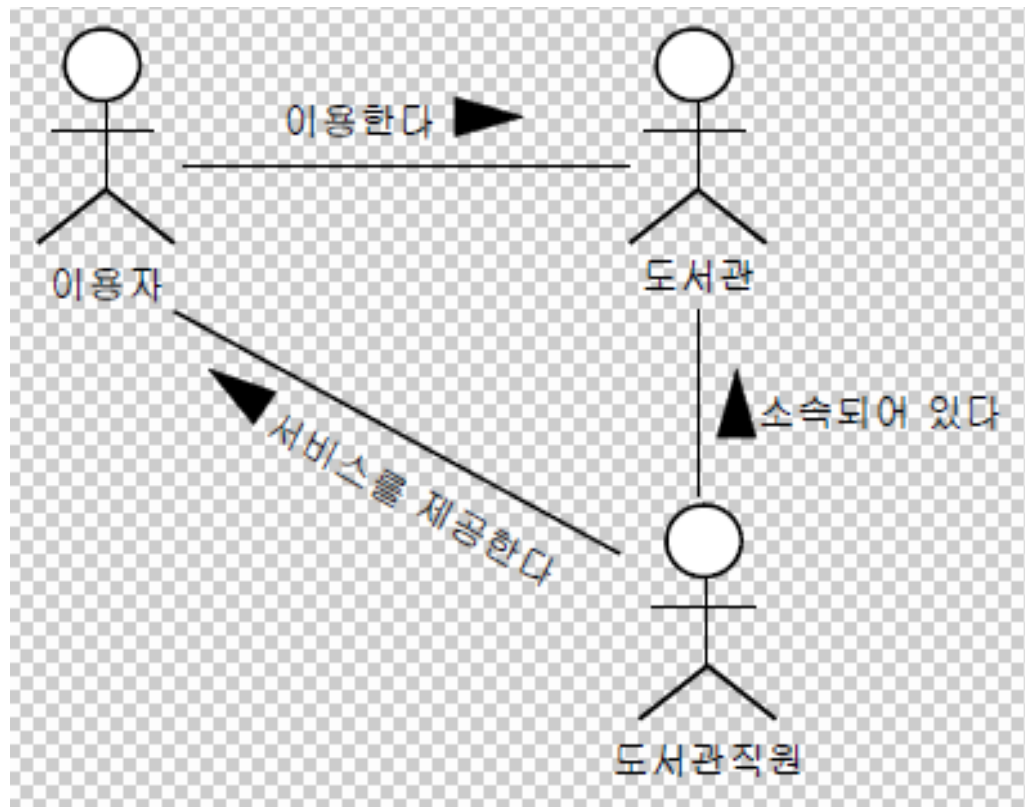
도서관직원



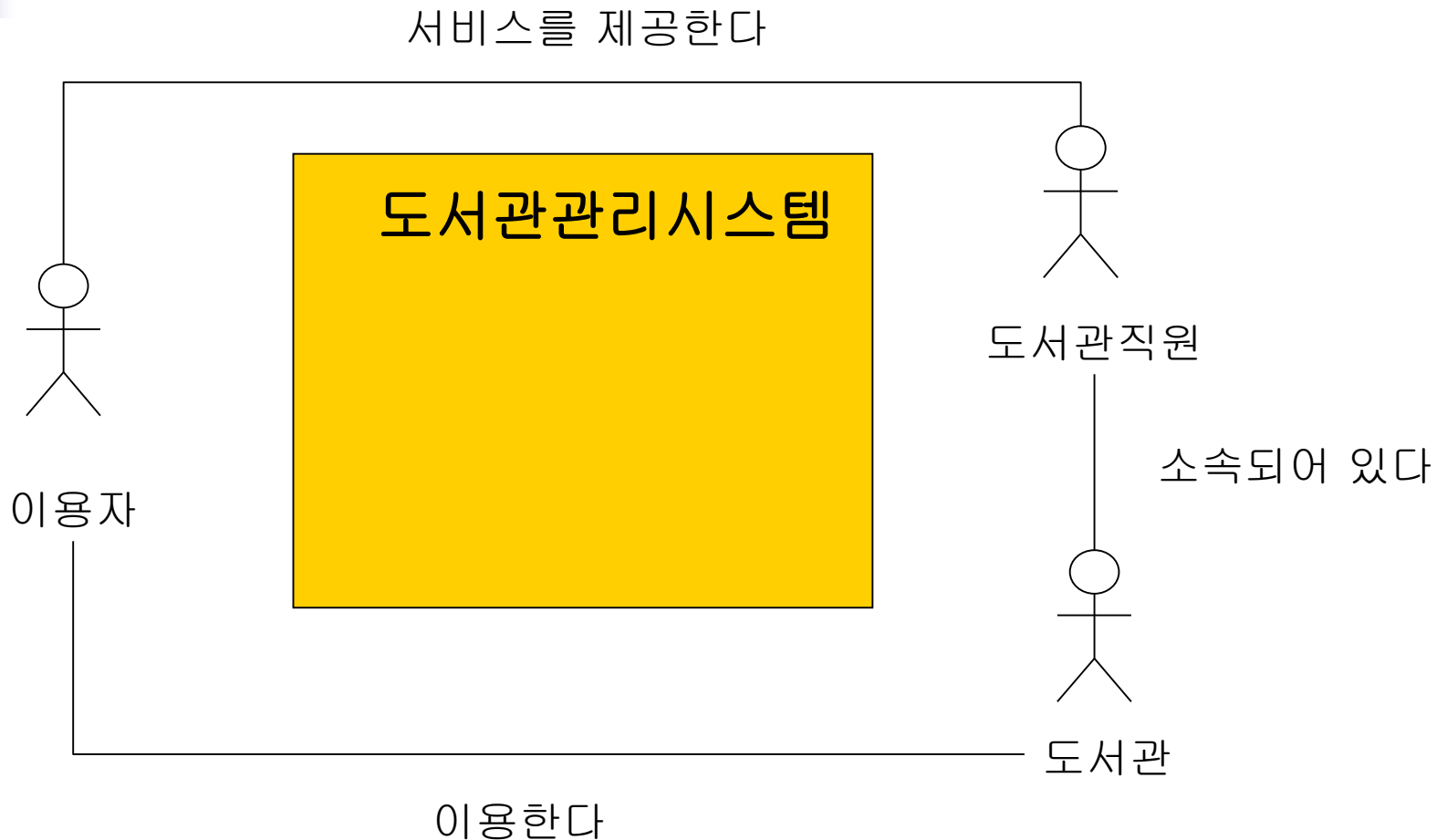
도서관

Actor들 사이의 관계

- 관련(Association)
 - 초기단계에서 너무 과다하게 표현하지 말 것.
모델중독증에 걸림!
 - Actor들 사이에 어떤 관계 존재 ➔ 선으로 연결

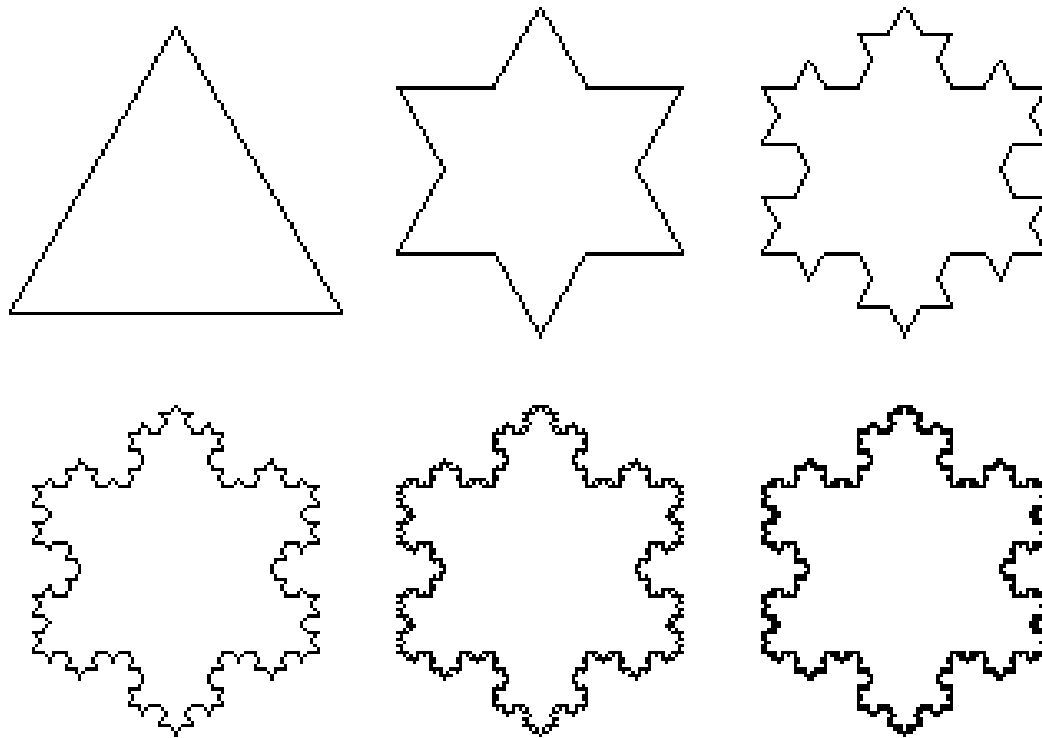


지금까지 완성된 UC Dgm

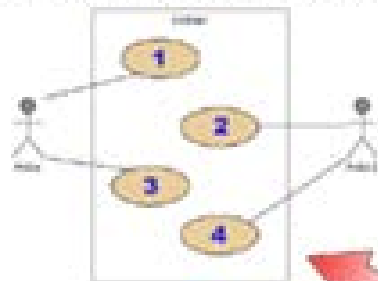




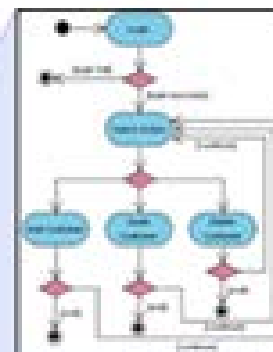
No “OneShot”, Step by Step



Prioritize Use Case

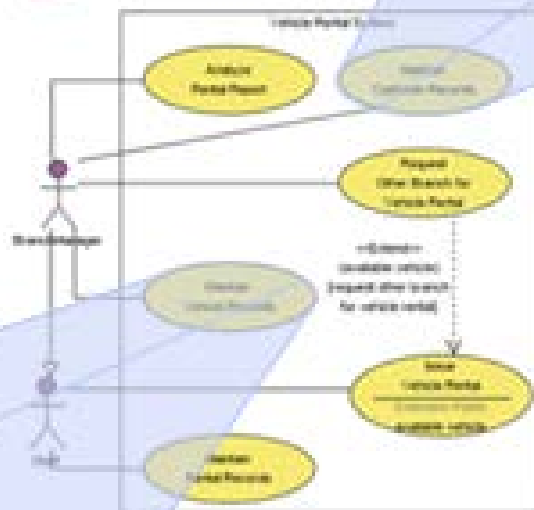


Elaborate Use Case using Activity Diagram



Detail Use Case

Use Case ID:	UC001
Use Case Name:	Vehicle Rental Request
Version:	1.0
Created By:	John Doe
Created Date:	2022-01-01
Modified By:	John Doe
Modified Date:	2022-01-01
Use Case Description:	The user requests a vehicle rental. The system checks the availability of the vehicle and the user's rental history. If the vehicle is available and the user has no outstanding rentals, the system approves the request. Otherwise, the system denies the request.
Use Case Flow:	<pre> graph TD Start((Start)) --> Request[Request Vehicle Rental] Request --> CheckAvailability[Check Vehicle Availability] CheckAvailability --> Availability{Availability} Availability -- Yes --> CheckHistory[Check Rental History] Availability -- No --> Deny[Deny Request] CheckHistory -- No Outstanding --> Approve[Approve Request] CheckHistory -- Outstanding --> Deny Deny --> End((End)) Approve --> End </pre>
Use Case Requirements:	<ul style="list-style-type: none"> The system must check the availability of the vehicle. The system must check the user's rental history. The system must approve the request if the vehicle is available and the user has no outstanding rentals. The system must deny the request if the vehicle is not available or the user has outstanding rentals.
Use Case Acceptance Criteria:	<ul style="list-style-type: none"> The system must return a 'Success' message when the request is approved. The system must return a 'Failure' message when the request is denied.

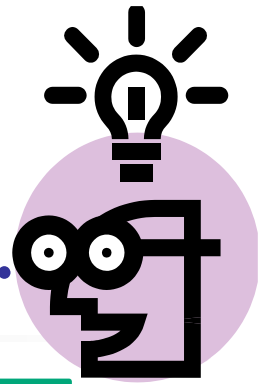


Generate Document

Use Case Modeling



시작하기에 앞서, 한마디...



들으면, 잊는다.
보면, 기억한다.
행동하면, 이해한다.

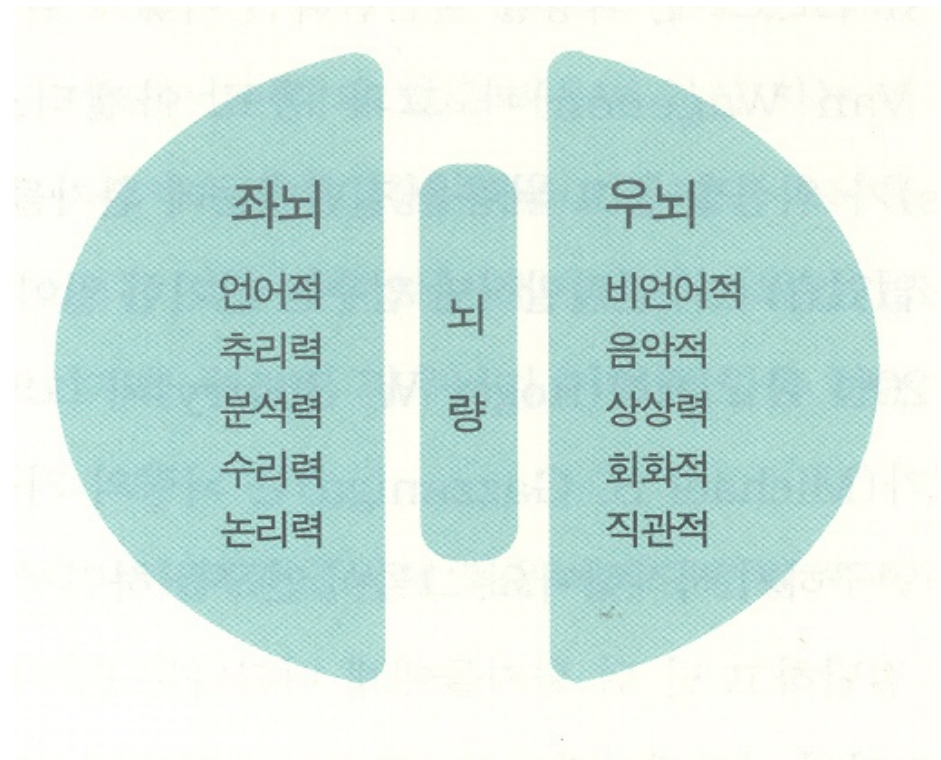
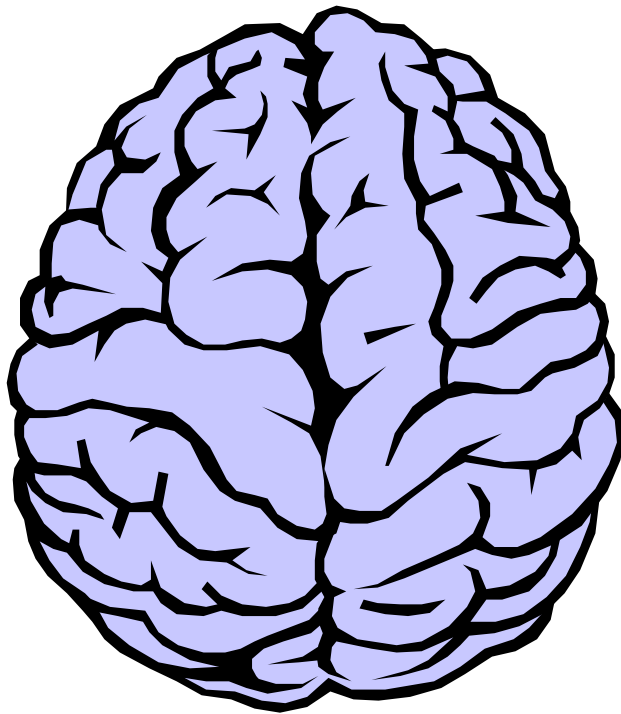
-공자-

하나 더!

- 읽는 것의 10~20%를 기억한다.
 - 듣는 것의 20~30%를 기억한다.
 - 보는 것의 30~50%를 기억한다.
-
- 보고 듣는 것을 동시에 하면, 50~60%를 기억
 - 머릿속에서 정보를 재구성하면, 60~80%를 기억
 - 실제로 관련된 행동으로 옮기면, 80~100%를 기억



뇌 腦 Brain



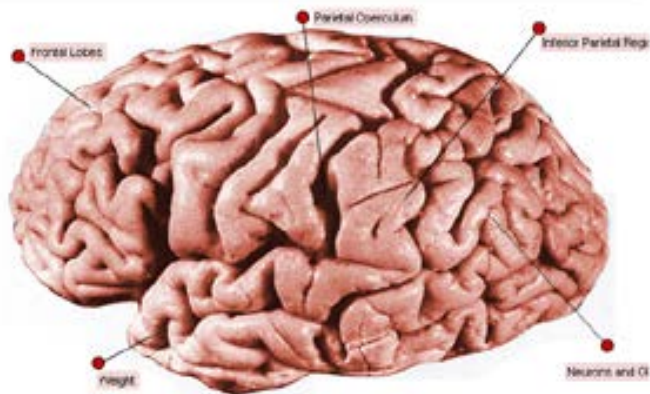
Neuron = 뇌세포

DISCOVER

Science, Technology, and The Future

Health & Medicine / Mind & Brain / Technology / Space / Human Origins / Living World / Environment / Physics & Math / Video / Photos / Podcast

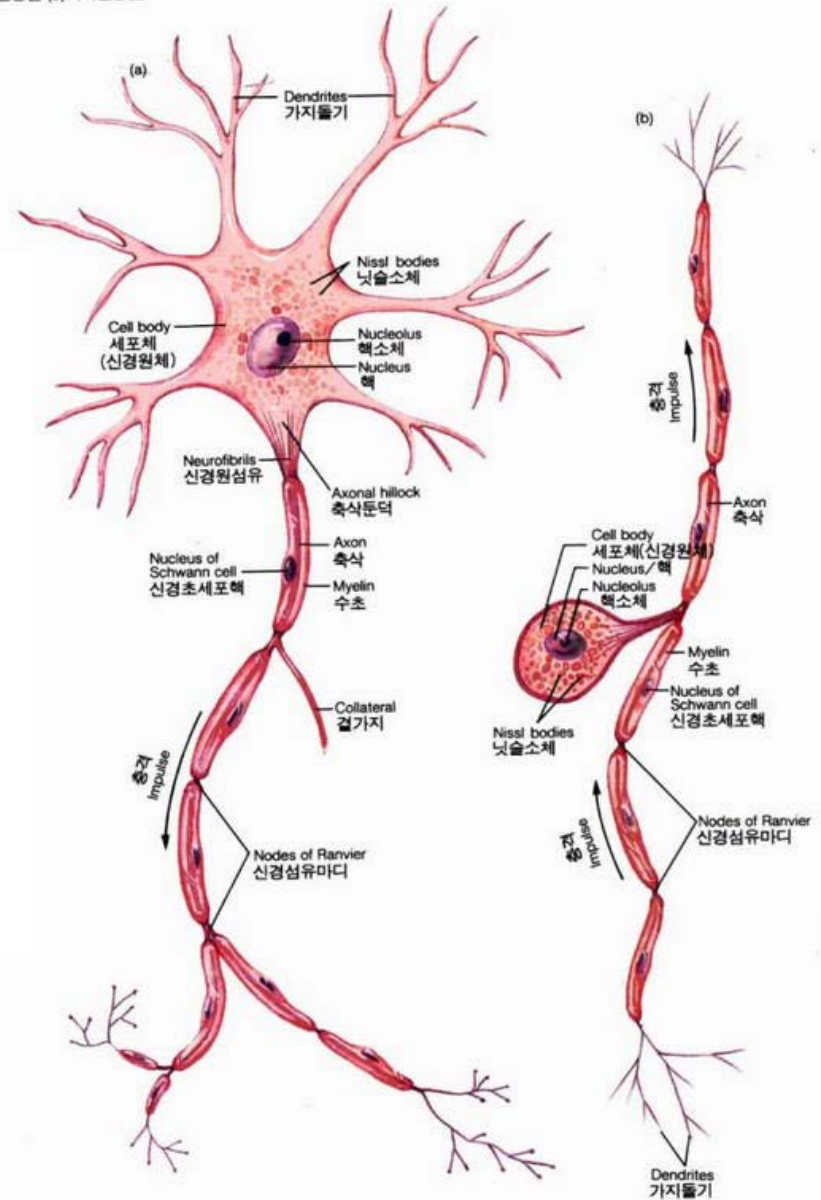
Galleries / Sliced: Einstein's Brain



Reproduced from Wilson et al., Lancet, 1998, with permission

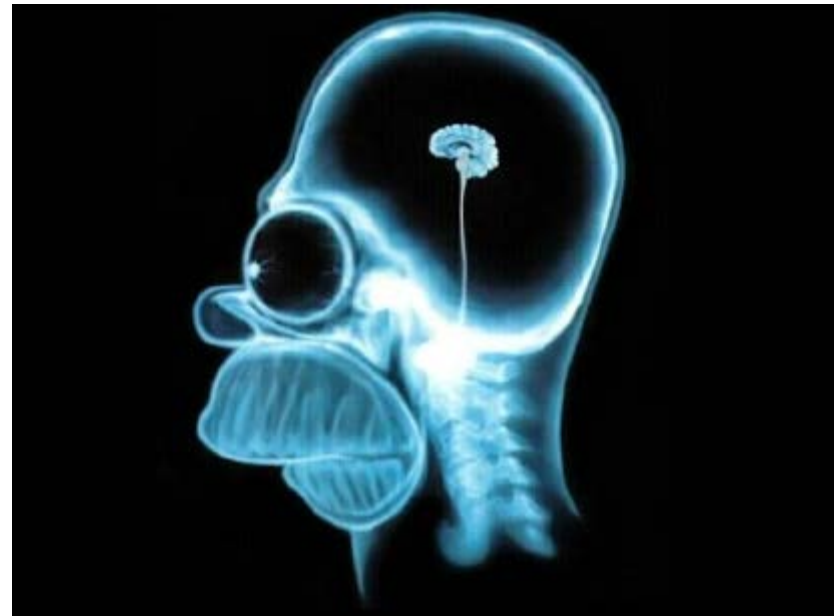
discovermagazine.com

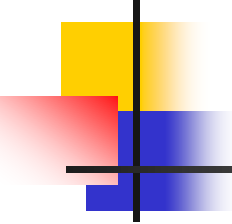
(a)운동신경원 (b)지각신경원



(c)시

용불용설.....





어떤 유명한 분(?)의 말씀

...

Don't ever give up yourself, because when you give up on yourself, you give up on your country.

...

If you get a bad grade, that doesn't mean you're stupid, it just means you need to spend more time studying.

...

어느 대학생의 뇌 구조



믿거나 말거나...^^;



“앞쪽형 인간”

- TV 끄고 책-신문 읽으면
앞쪽뇌 발달... 정보 종합
능력 앞서
- 주변 환경에 수동적인 ‘
뒤쪽형 인간’과 대비
- 충동조절 잘해... “많이
걸고 대화하면 효과”

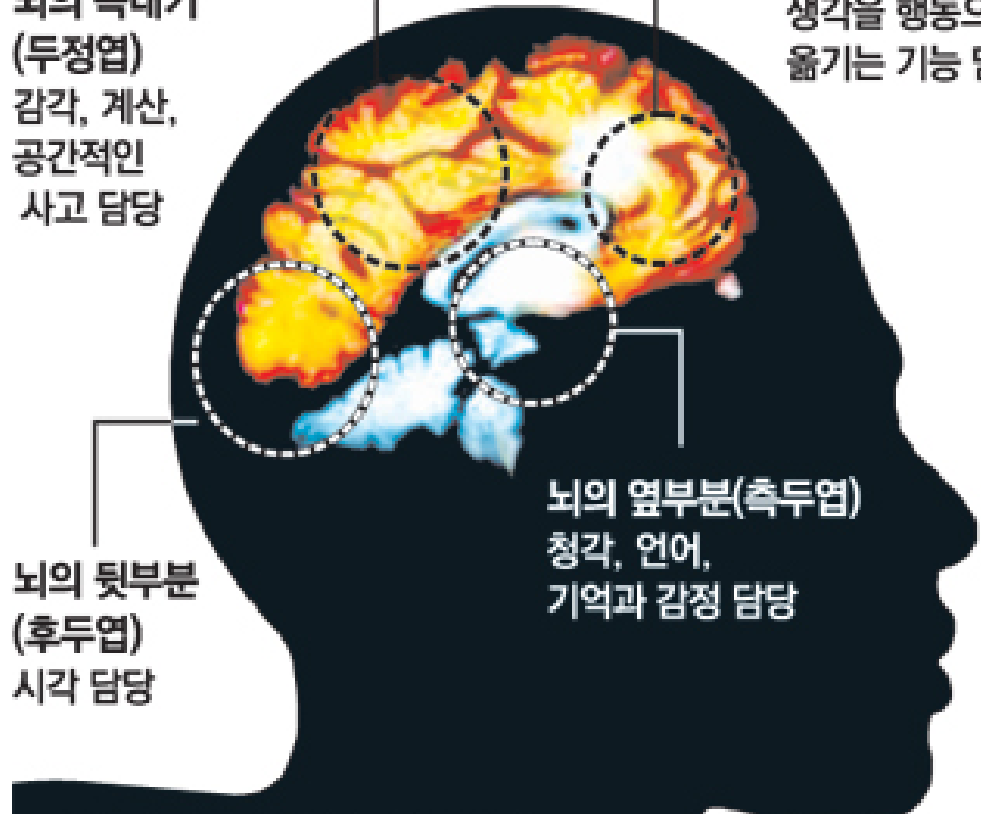
뇌의 부위별 역할

뇌의 꼭대기
(두정엽)
감각, 계산,
공간적인
사고 담당

뇌의 앞부분(전두엽)
판단력, 집중력,
생각을 행동으로
옮기는 기능 담당

뇌의 뒷부분
(후두엽)
시각 담당

뇌의 옆부분(측두엽)
청각, 언어,
기억과 감정 담당



2008.9.17 동아일보



“앞쪽형 인간”이 되려면...

1. TV를 끄고 신문이나 책을 읽어라
2. 읽기보다는 써라.
3. 듣기보다는 발표를 하라.
4. 외국어나 수학을 배워라.
5. 평소 적절한 단어와 표현을 찾는 노력을 하라.
6. 글쓰기, 그림그리기, 조립하기 등 창작활동을 하라.
7. 평소 시간관리를 하라.
8. 사소한 일의 경우, 틀리든 맞는 무조건 하나를 결정하는 습관을 들여라.
9. 논리적인 사고를 하라.
10. 예측하고 계획하는 습관을 들여라.



또 다른 유명한 분의 말씀

스펙 (학점, 토익/토플, ...)	전문능력	결과
Good	있음	당연히 취업성공!
Not Good	있음	창업/취업 가능성 높음
Good	없음	백수가능성 높음
Not Good	없음	100% 백수!

Must Read!!!

IT 취업 그것이 궁금하다



IT 전문가 김중태의 리얼한 취업 상담

김중태 지음

저는 이번에 서울 중위권 대학 컴퓨터학과를 졸업 예정이고 지금은 구직 상태입니다. 먼저



26월, 보직 800원, 자격증은 정보처리기사와 OCP가 있고 9개월간의 어학연수

보사다시의 저의 학점이 너무 낮습니다. 학점을 보완하려면 무엇을 준비해야 할

에서 일하는 사람들이 박사 학위까지



할 필요가 있나요? 스물일곱이고 여행업계에서 근무하고 있습니다. 2년

만에 멀티태이킹을 하려다 말았는데

지금도 멀티태이킹은 코화상때까지 살았으므로 입사하기가 힘들다고 들었습니다.

비전공자라도 저급부터 새롭게 할 수 있는 작업을 해서 팔리상이 돋보이는



작업을 만들 수 있는 IT 직종이 될 디자인 외

에도 있나요? 이직을



원하던 회사로 과감히 그만두기에는..... 조금



경기도 안 종에서 이직지도 저리지도 못하고 고

민 중입니다. 이직을



할 때 기업에서 제직 중인 사람보다 퇴직한 사람을 원한다고 하는데



제직 중이면서 이직서를

제출하는 것이 회사를 하고 지원하는 것보다 불합리할까요?





Workflow

- Actor가 어떤 목적을 달성하기 위하여, 실제로 어떻게 행동하는가를 나타낸 것
- Activity Diagram으로 표현
- 도서관관리시스템의 Workflow
 - 어떤 이용자가 서적을 검색, 대출,..., 반납
 - 이용자가 이용자등록하고, ...변경..., 말소
 - 어떤 서적에 대하여, 발주, 납품, 이용,..., 폐기

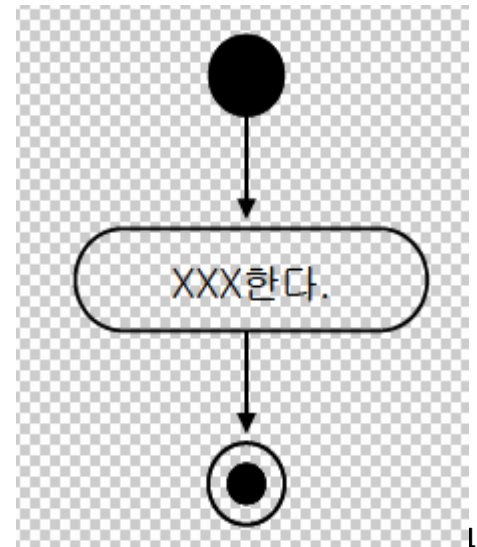
1st Workflow

: 서적의 검색-대출-반납

1. 이용자가 도서관에 도착한다.
2. 대출하려는 서적을 검색한다.
3. 만약, 해당서적이 도서관에 있으면 대출
4. 집으로 가져가서 읽는다.
5. 대출기한 전에 반납하기 위해 도서관에 간다.
6. 서적을 반납한다.

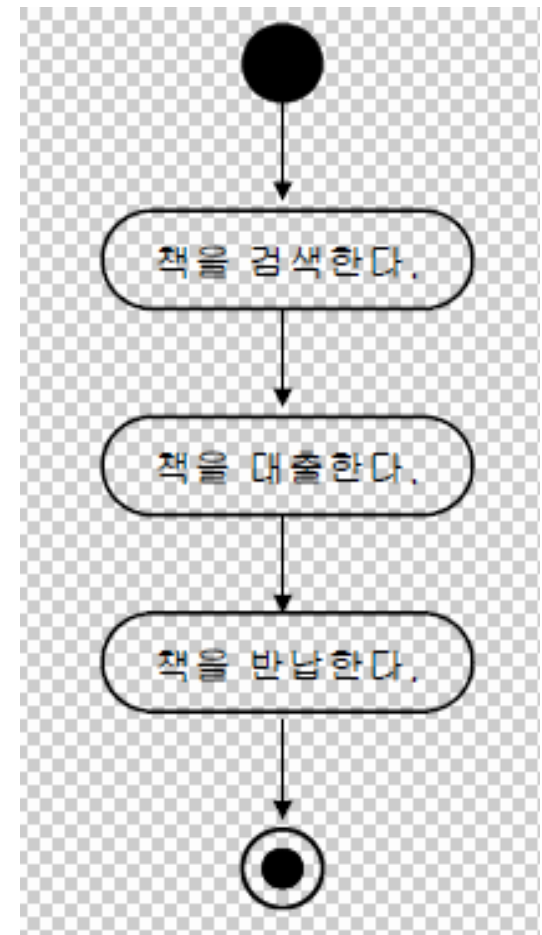
Activity Dgm 작성방법

- 각 업무단위를 Activity 또는 Action상태로 표현
- 실제수행되는 행위(동작 or 작업)을 간결하게 기입 ("XXX한다")
- 시간경과순으로 화살표로 연결("遷移")
- 시작과 끝을 나타내는 정점 사용



문제: Activity Dgm작성

1. 이용자가 도서관에 도착한다.
2. 대출하려는 서적을 검색한다.
3. 만약, 해당서적이 도서관에 있으면 대출한다.
4. 집으로 가져가서 읽는다.
5. 대출기한전에 반납하기 위해 도서관에 간다.
6. 서적을 반납한다.



분기와 Guard

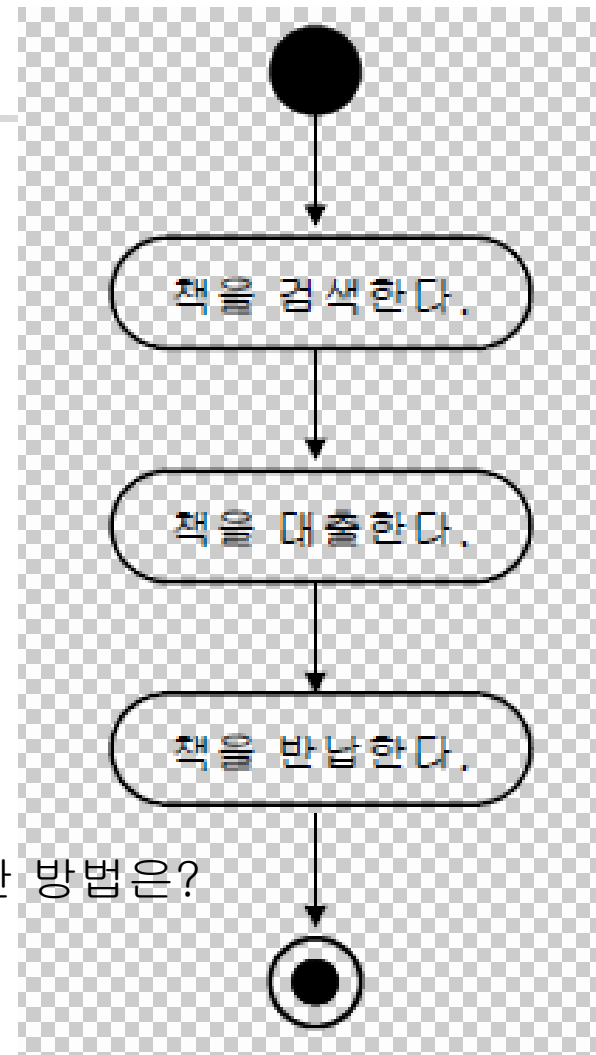
“책을 검색한 결과, 찾는 책이 없다면?”

대출중 ➔ 도서예약

미소장 ➔ 인근도서관에 문의 or 구입신청

이와같은, “조건에 따른 분기구조”를 나타내기 위한 방법은?

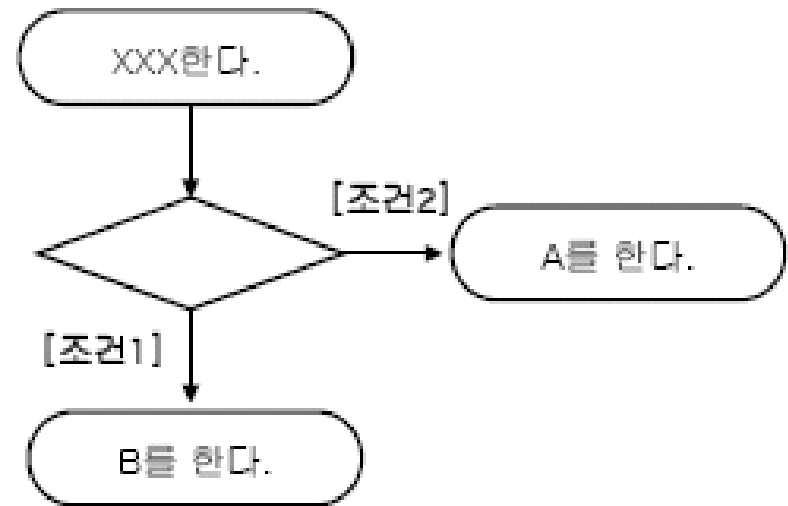
➔ “[조건]”을 기입 ← Guard



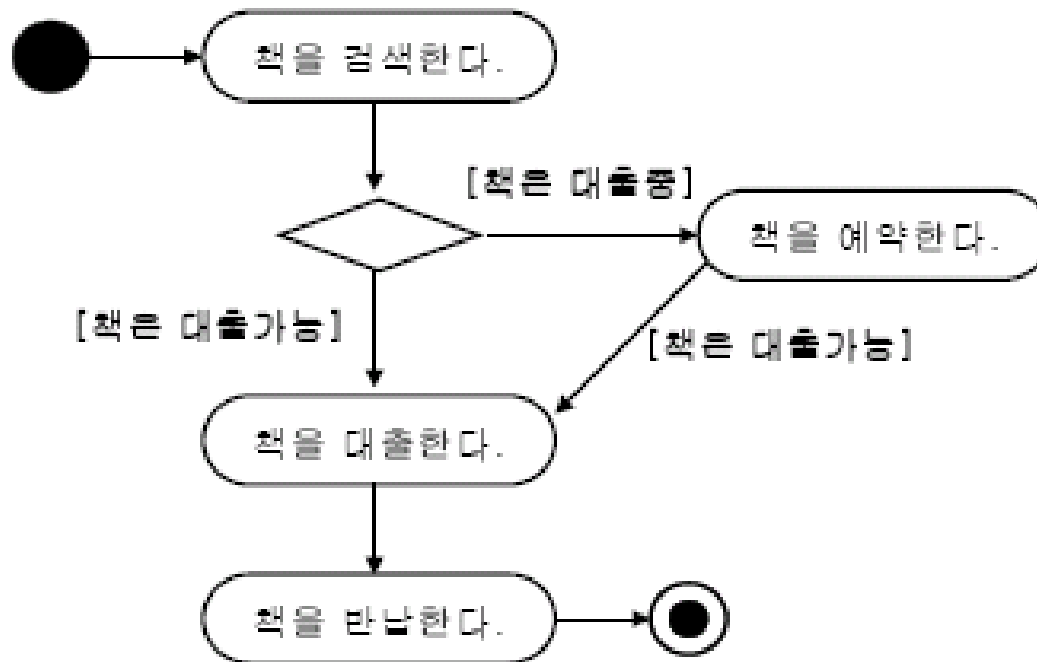
분기, Guard

일단은, 자연언어로 기입
“[책이 대출중]”
“[책이 대출가능]”

보다 구체적으로 정확히 기입하기 위해서는,
OCL(Object Constraint Language)를 사용

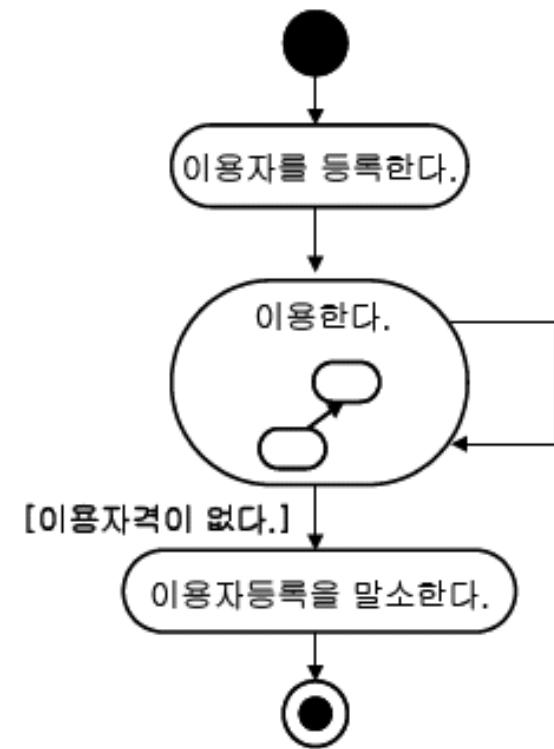


Guard조건 기입 후



■ 내장된 형태의 Activity

- 서적대출은 도서관이용자만 가능!
- 도서관이용자 등록에 관한 별도의 Workflow가 필요.
 1. 이용자등록을 수행한다.
 2. 도서관을 이용한다.(반복)
 3. 이용자격이 상실되면 등록을 말소한다.

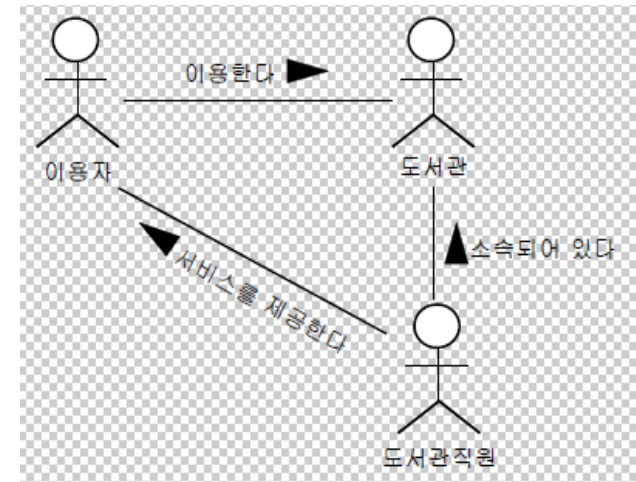
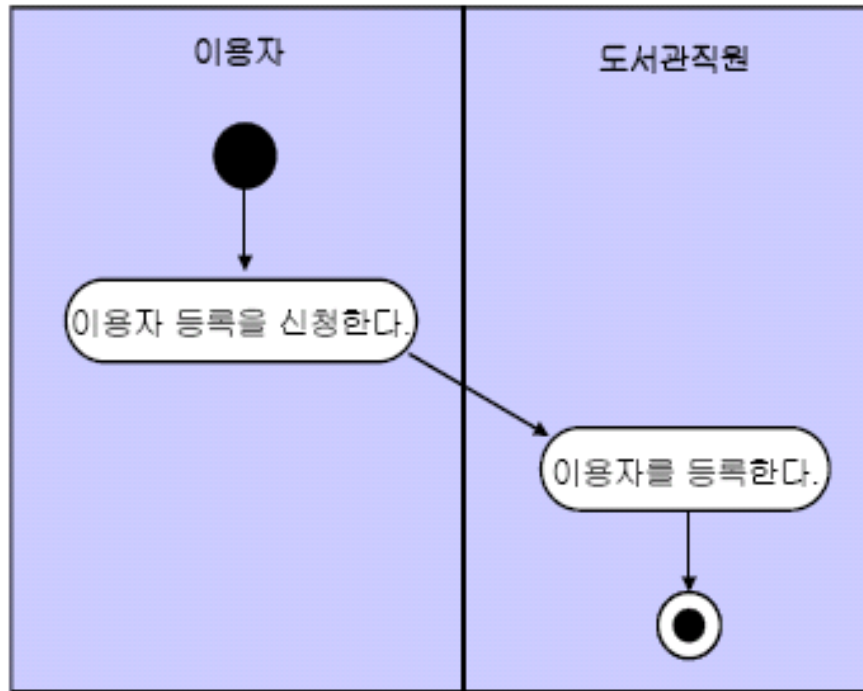




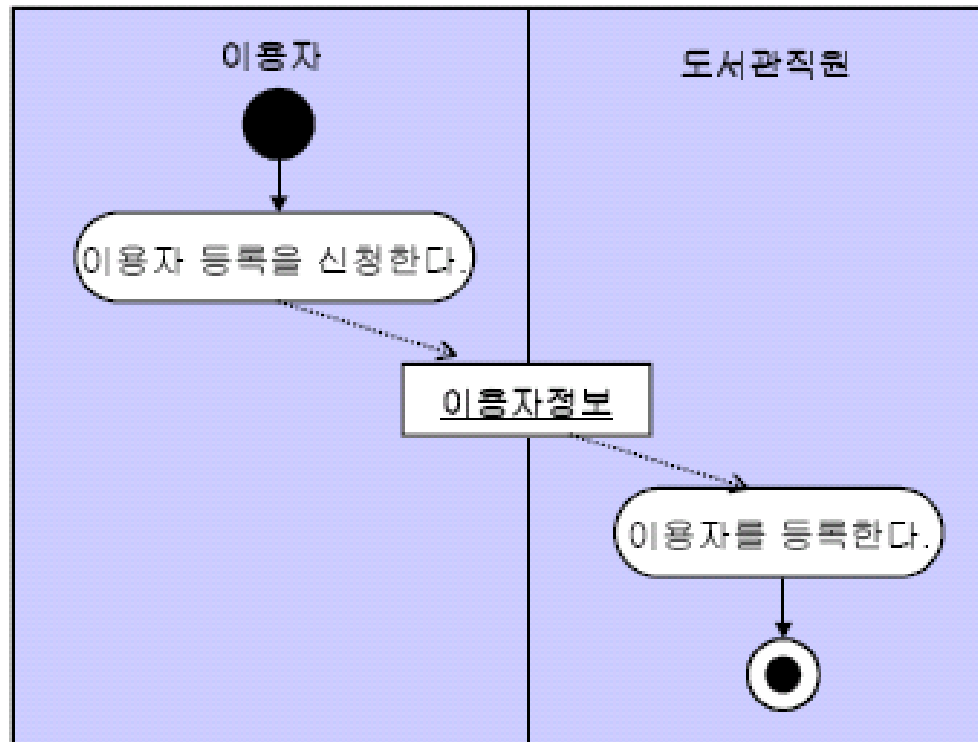
SwimLane

- “이용자를 등록한다”Workflow
 1. 이용하려는 사람이 이용자등록을 신청한다.
 2. 도서관직원이 신청을 받아서 실제로 등록한다.
- 2개의 Actor가 관여
 - 이용자, 도서관직원
- 이런 Workflow를 Activity Dgm으로 그릴 때에는 “SwimLane”사용

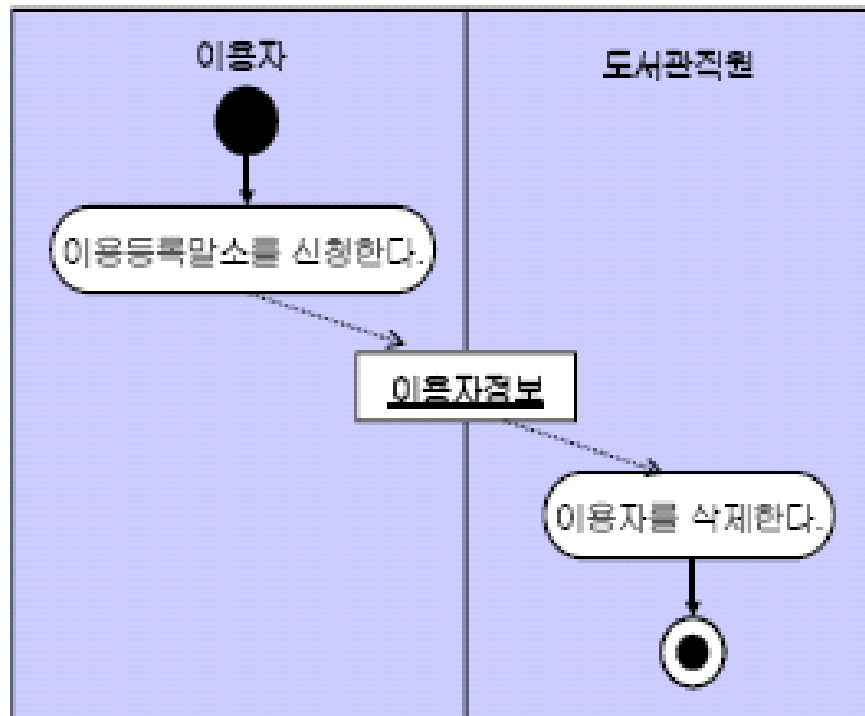
SwimLane을 추가한 Activity Dgm



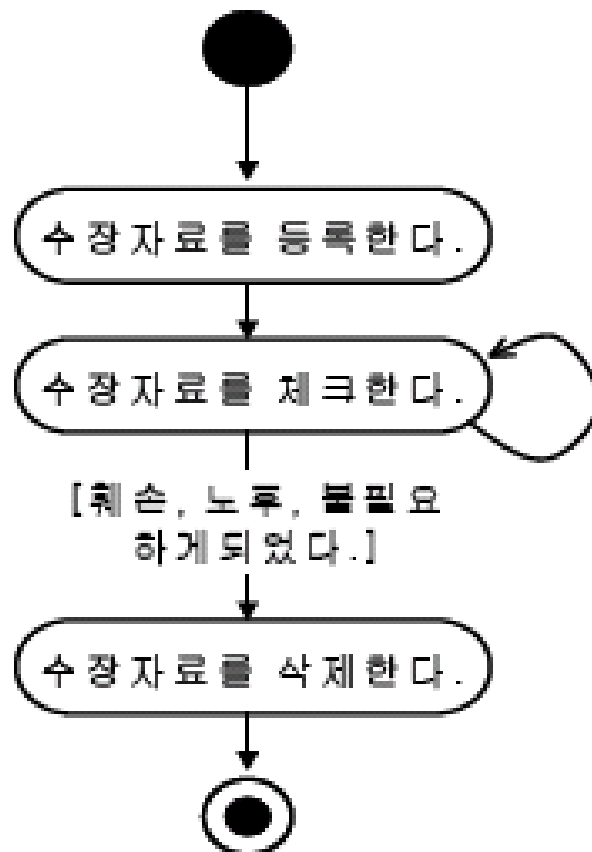
Object Flow



이용자등록을 말소한다



收藏資料를 등록한다





3.6 조작 및 동작의 완전성

- 앞서 작성한 Workflow는 완전한가?
- 확인방법
 - Walk through
 - CRUD



Walk through

- Role Play 방식

- 각각 역할 결정
- Activity Dgm의 Workflow대로 수행, 체크
- 대규모 Workflow의 경우, 세밀한 부분까지 수행 곤란
- 중요한 부분만 수행

CRUD

- Create, Retrieve, Update, Delete와 같은 속성을 만족하는지를 체크
- 예: 收藏資料, 利用者

이용자정보에 관한
Workflow를 재검토하여,
관련된 Activity Diagram을
수정 or 신규추가

속성	Activity	비고
Create	수장자료를 등록한다.	-
Retrieve	수장자료를 검색한다.	-
Update	없음.	수장자료정보는 한번 등록되면 변경시키지 않을 것이다.
Delete	수장자료를 삭제한다.	-

속성	Activity	비고
Create	이용자를 등록한다.	-
Retrieve	없음.	서적 대출에 의해 묵시적으로 사용되며, 그 이외의 경우에는 개인정보보호를 위해 참조할 수 없다.
Update	???	-
Delete	이용자를 삭제한다.	-



"수장자료"의 CRUD속성

속성	Activity	비고
Create	수장자료를 등록한다.	-
Retrieve	수장자료를 검색한다.	-
Update	없음.	수장자료정보는 한번 등록되면 변경시키지 않을 것이다.
Delete	수장자료를 삭제한다.	-



"이용자"의 CRUD속성

속성	Activity	비고
Create	이용자를 등록한다.	-
Retrieve	없음.	서적 대출에 의해 묵시적으로 사용되며, 그 이외의 경우에는 개인정보보호를 위해 참조할 수 없다.
Update	???	-
Delete	이용자를 삭제한다.	-



3.7 Use Case

- 사용자는 개발시스템을 어떻게 사용하나?
➔ Use Case로 표현



XXX가 YYY를 ZZZ한다

개발시스템이 Actor에게 제공해야되는 기능을 표현
사용자에게 있어서 시스템이 어떻게 느껴지는가를 기능적 측면에서 표현



Use Case

- 고려해야할 사항
 - 반드시 Actor와 관계
 - Actor에게 의미있는/어떤 가치를 부여하는 기능
- 예: ATM
 - “현금을 출금한다” ➔ 사용자에게 의미있음.
 - “화면의 현금인출버튼을 체크한다” ➔ 사용자에게 어떤 가치도 부여하지 못함.
 - “지폐를 센다” ➔ 사용자와 직접적인 관계없음.



Use Case의 이름 정하기

- 형식

- “XXX가 YYY를 ZZZ한다”
 - XXX == Actor
 - (YYY == Actor)가 되는 경우도 있음.
 - 명확히 Actor가 주어인 경우에는, XXX를 생략

문제

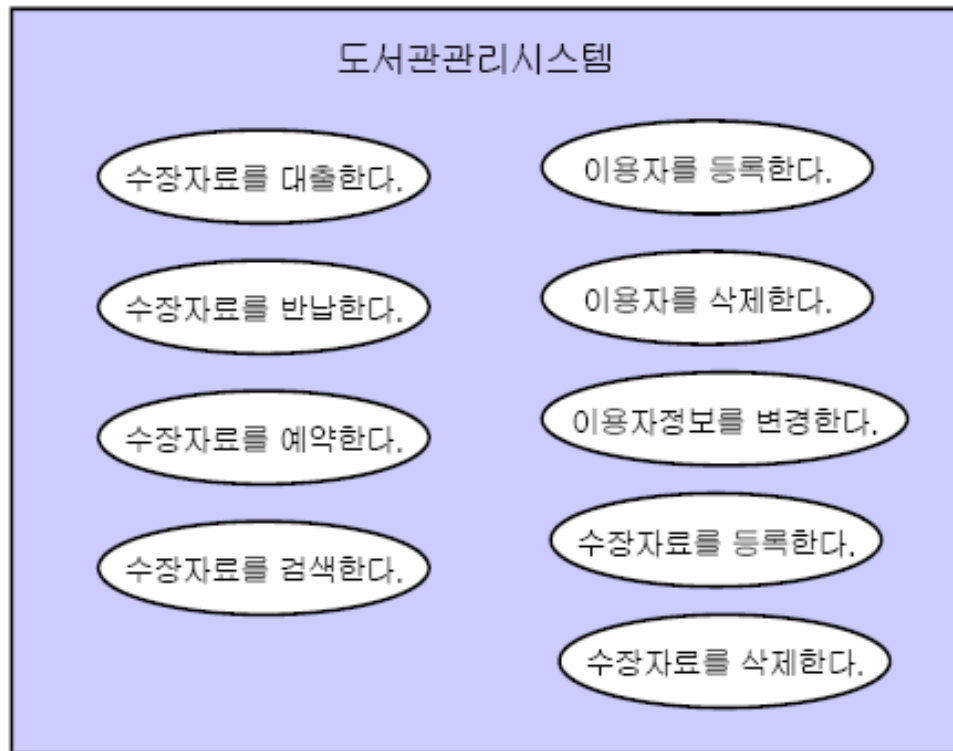
■ 도서관관리시스템의 Use Case는?

이용자 Actor가,

- 수장자료를
- 수장자료를
- 수장자료를
- 수장자료를

도서관직원 Actor

- 이용자를
- 이용자를
- 이용자정보
- 수장자료를
- 수장자료를



Main Actor
이용자
이용자
이용자
이용자
도서관직원
도서관직원
도서관직원
도서관직원
도서관직원



Activity vs. Use Case

- Activity(그림2-8~15)과 Use Case(그림2-17)는 매우 유사
 - Use Case는 시스템과 Actor들의 상호수수작용 표현
 - Workflow는 Actor들의 목적달성을 위한 일련의 행동 표현
- UC와 Activity는 중복될 수도...
 - Workflow에 출현하지 않는 UC는 Actor에게 무가치

Activity vs. Use Case

WorkFlow	Activity	Use Case	주Actor
전체	이용자를 등록한다	-	-
전체	이용한다	-	-
전체	이용자등록을 말소한다.	-	-
수장자료를 이용한다.	책을 검색한다.	수장자료를 검색한다.	이용자
수장자료를 이용한다.	책을 예약한다.	수장자료를 예약한다.	이용자
수장자료를 이용한다.	책을 대출한다.	수장자료를 대출한다.	이용자
수장자료를 이용한다.	책을 반납한다.	수장자료를 반납한다.	이용자
이용자를 등록한다.	이용자등록을 신청한다.	-	-
이용자를 등록한다.	이용자를 등록한다.	이용자를 등록한다.	도서관직원
이용자등록을 말소한다.	이용자등록말소를 신청한다.	-	-
이용자등록을 말소한다.	이용자를 삭제한다.	이용자를 삭제한다.	도서관직원
수장자료를 관리한다.	수장자료를 등록한다.	수장자료를 등록한다.	도서관직원
수장자료를 관리한다.	수장자료를 체크한다.	-	-
수장자료를 관리한다.	수장자료를 삭제한다.	수장자료를 삭제한다.	도서관직원
(CRUD속성에 의하여)	사용자정보를 갱신한다.	사용자정보를 갱신한다.	도서관직원



UC가 되지 못한 Activity ???

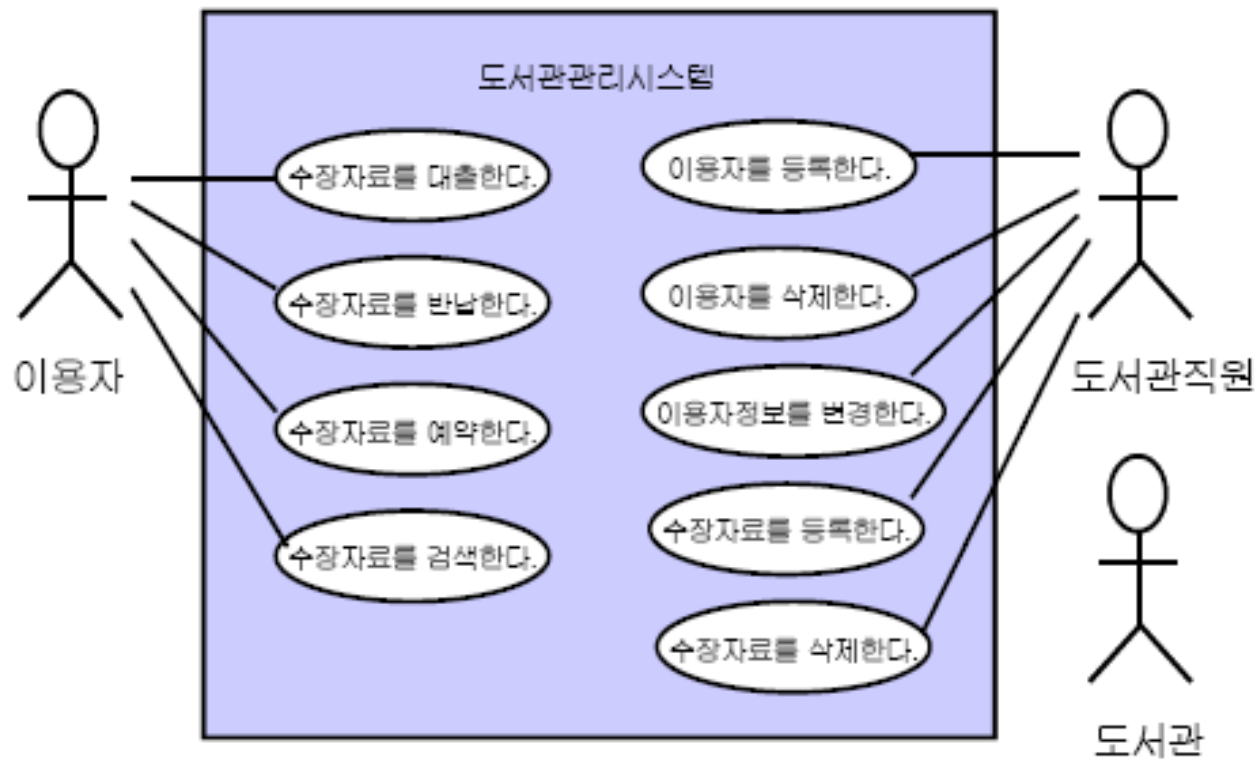
- UC가 될 수 없는 Activity는 시스템화되지 않는 기능을 나타내고 있다고 생각하십시오.



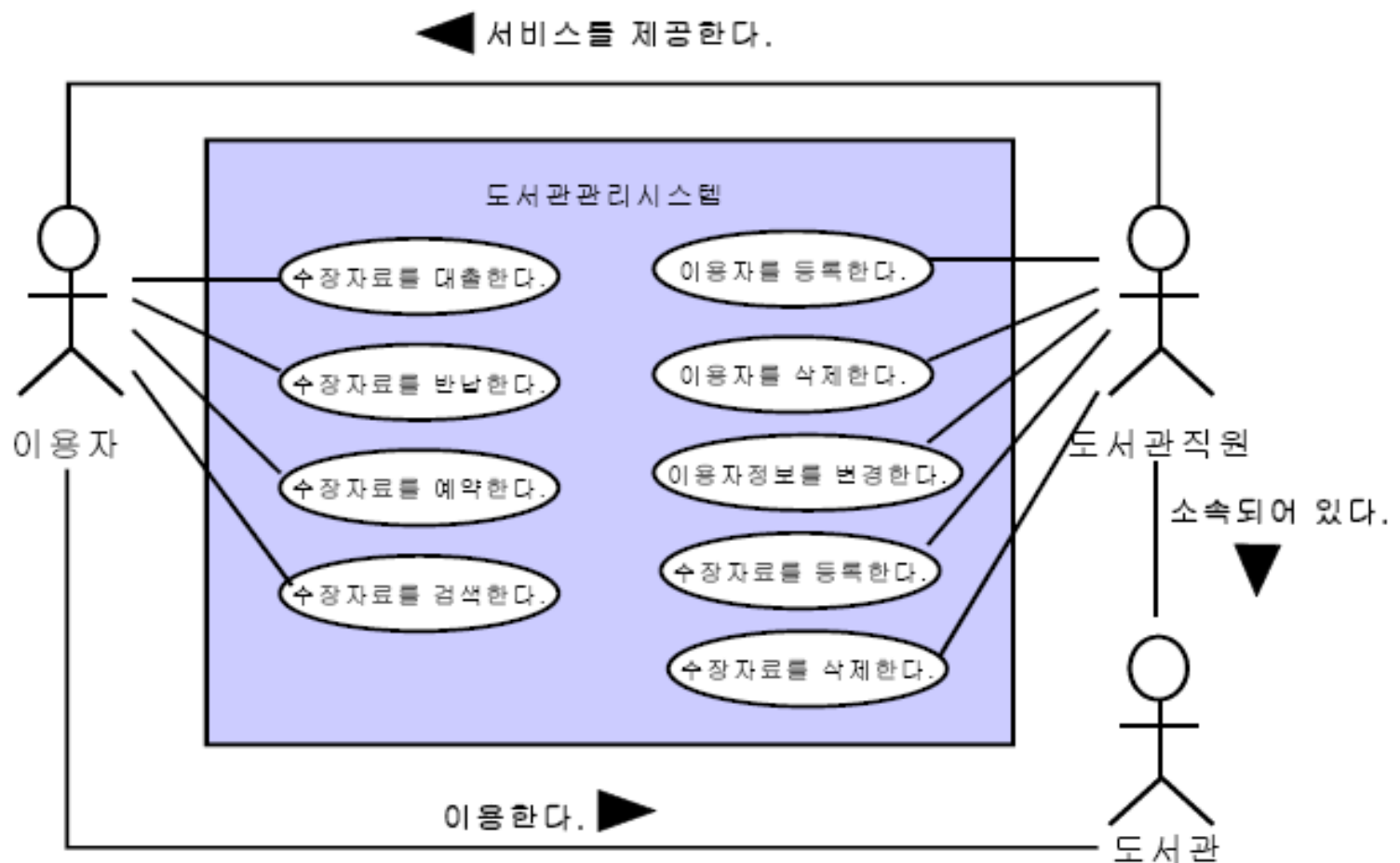
Workflow vs. Use Case Dgm

WorkFlow	Use Case View
SwimLane	Actor
Activity(증의 일부)	Use Case
Activity들 사이의 천이	Actor들 사이의 관련
Object Flow	?

UC Diagram



UC Diagram

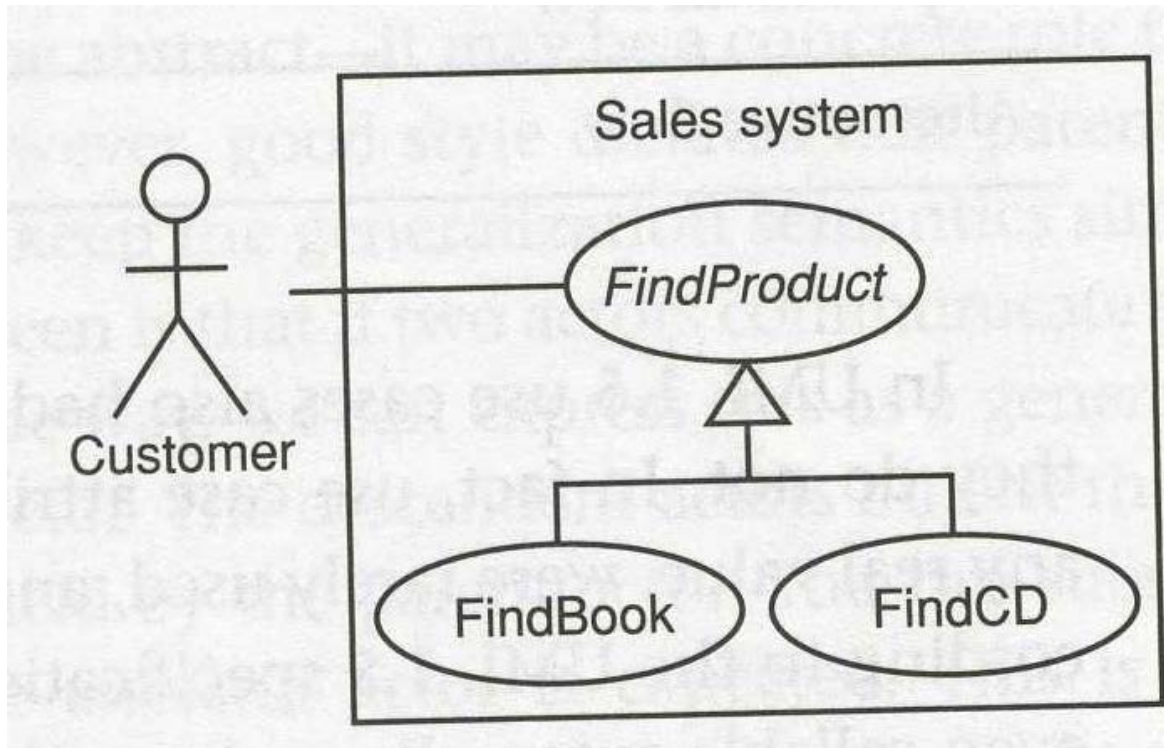




UC들 사이의 관계

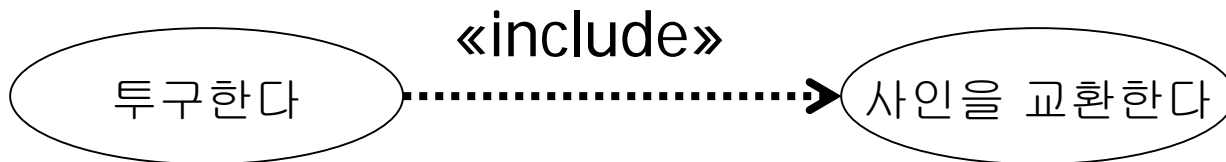
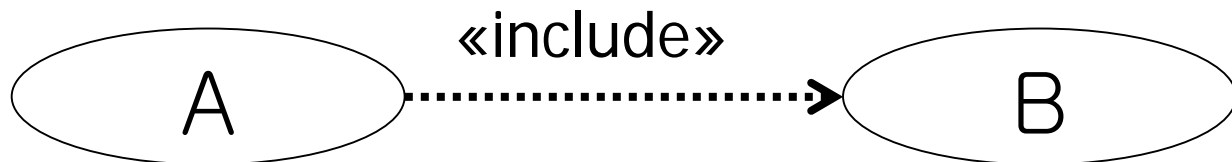
- 포함
- 확장
- 일반화

An example of UC generalization

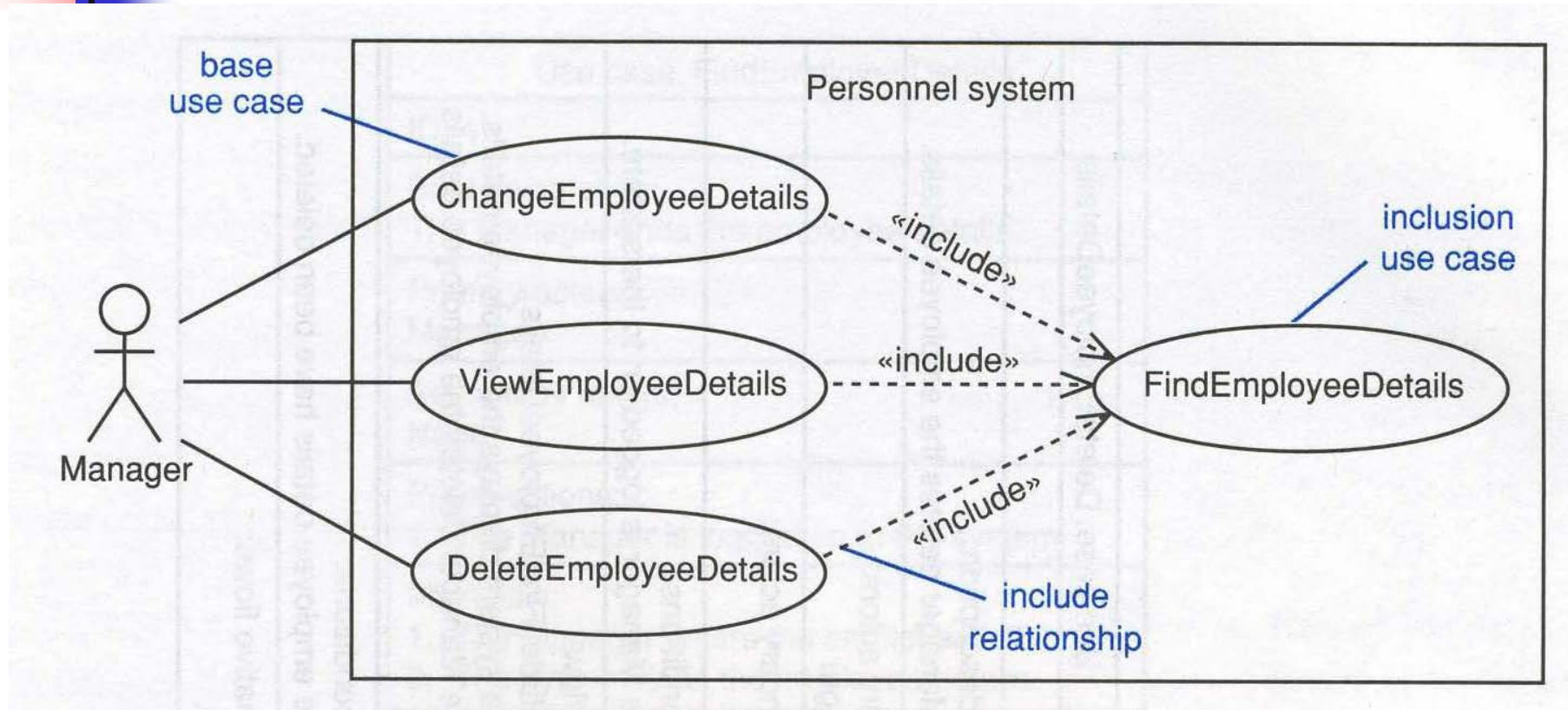


포함관계

- use case A가 다른 use case B를 사용하고 있는 관계
- 표기방법

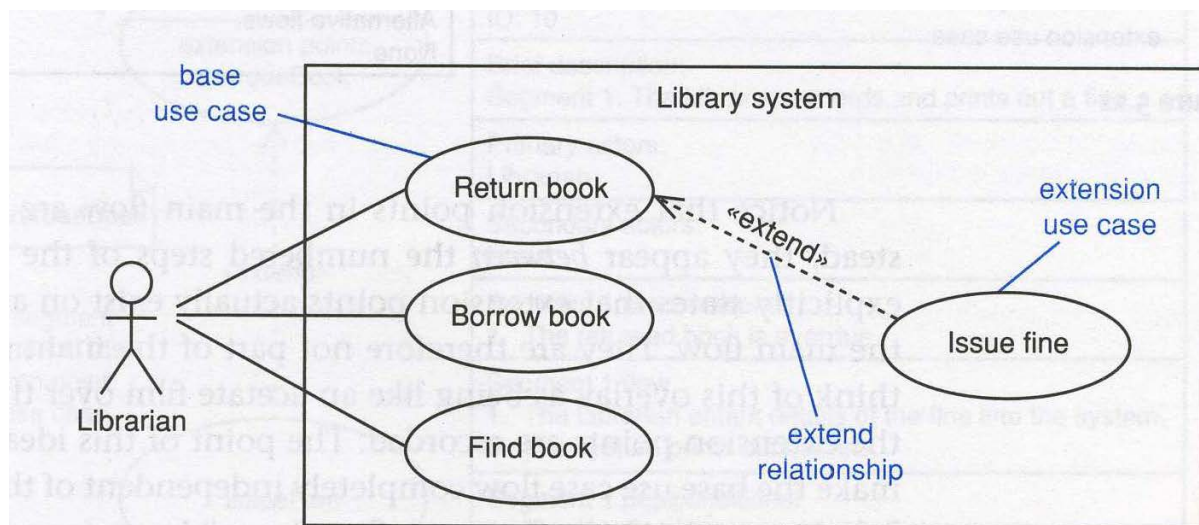


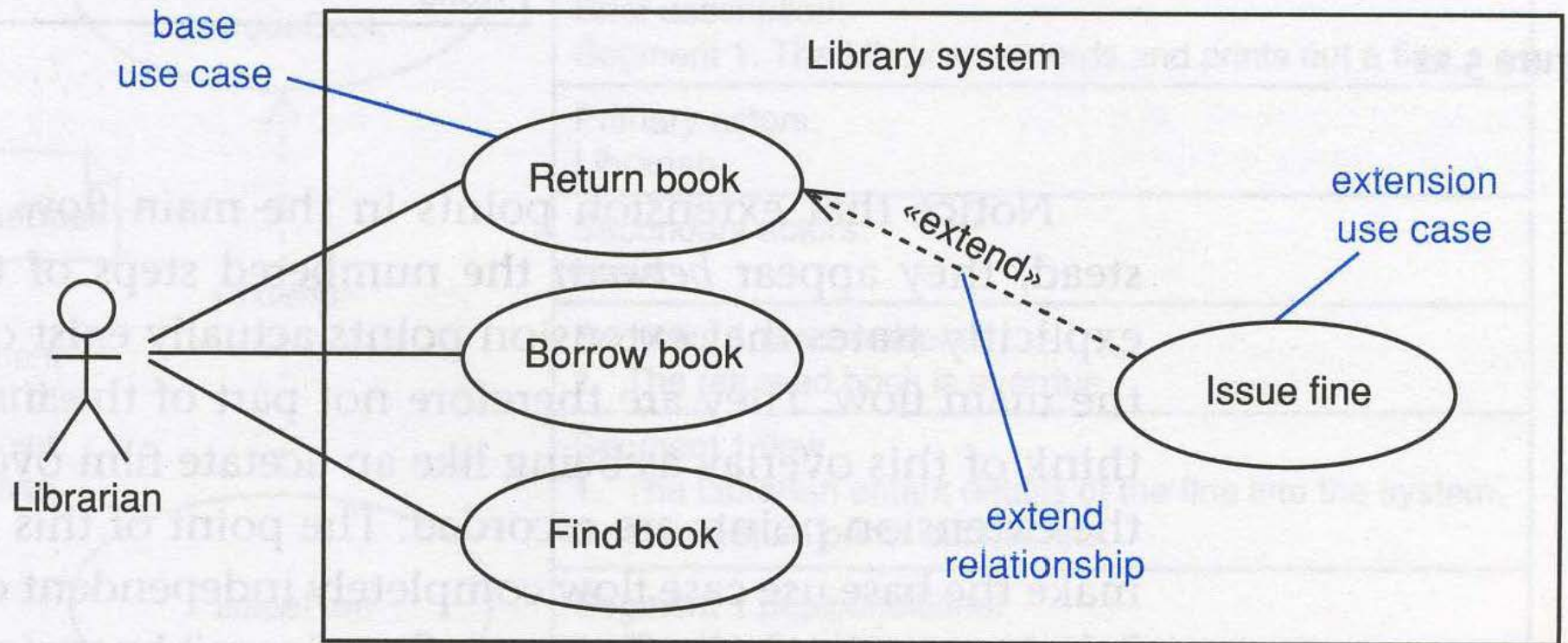
An example of <<include>>



확장 <<extend>>

- 일반화관계와 매우 유사
- UC의 선택사항을 보다 명확하게 표현가능
 - 어떤 특정한 조건(상황)하에서만 발생하는 동작을 표현
 - 토대가 되는 UC로 향하는 점선화살표
 - 동작이 발생하는 조건 ➔ "확장점"
 - 기본UC의 순서가 확장점에 도달, "조건"만족 ➔ 확장UC 실행





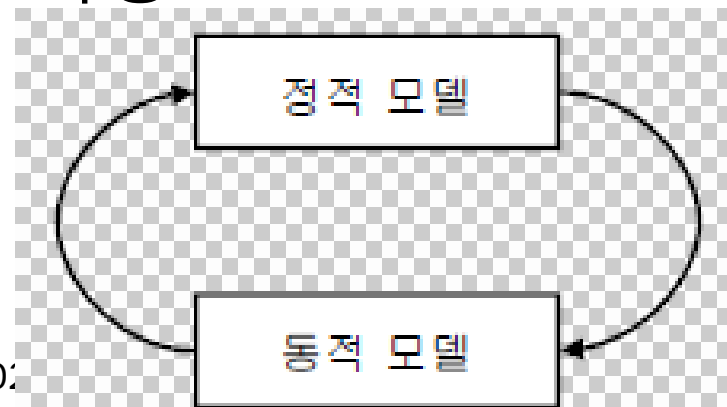


Use Case 추출방법

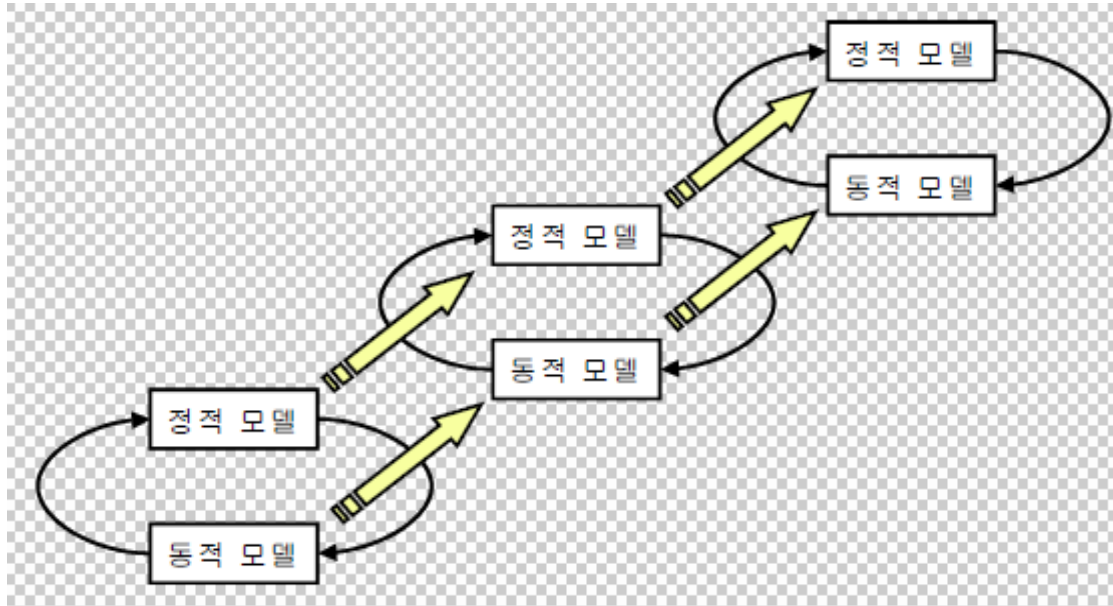
- 도서관의 업무흐름을 관찰
- 도서관 직원을 인터뷰
- 상상력 발휘, 고안, ...
- 주의사항
 - 가능한 한 단순하게 생각할 것.
 - 본질에 충실할 것.

3.8 靜的모델과 動的모델

- 정적모델
 - 시간의 개념이 불포함, 정적 구조 표현
- 동적모델
 - 시간의 개념 포함, 동적 동작 표현
- UML모델링에서는 상호반복 사용
 - 상호간의 정당성 검증
 - 상호 보완



정적모델 & 동적모델



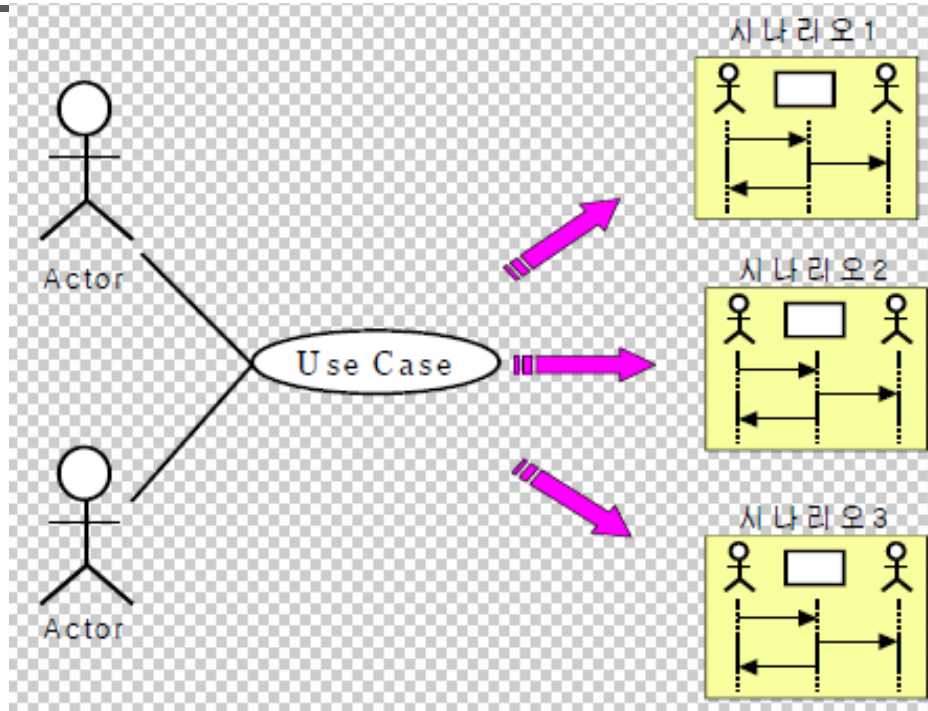
- Actor찾기 ➔ 정적 모델링
- Workflow표현(Activity Dgm) ➔ 동적 모델링
- Activity를 토대로 Use Case찾기 ➔ 정적 모델링
- ...



3.9 Use Case시나리오

- Workflow
 - Actor들이 정보(Object Flow)를 상호수수하면서 업무 수행
- 개발시스템은 이와같은 수수작용과 어떤 관계?
➔ 어떻게 모델링?
 - Activity Dgm로 표현
 - "시스템"에 해당하는 SwimLane추가
 - Use Case Scenario로 표현
 - 자연어(한글, 영어)로 표현 ➔ Use Case Description
 - UML 도면으로 표현 ➔ Sequence Diagram

Use Case와 Scenario



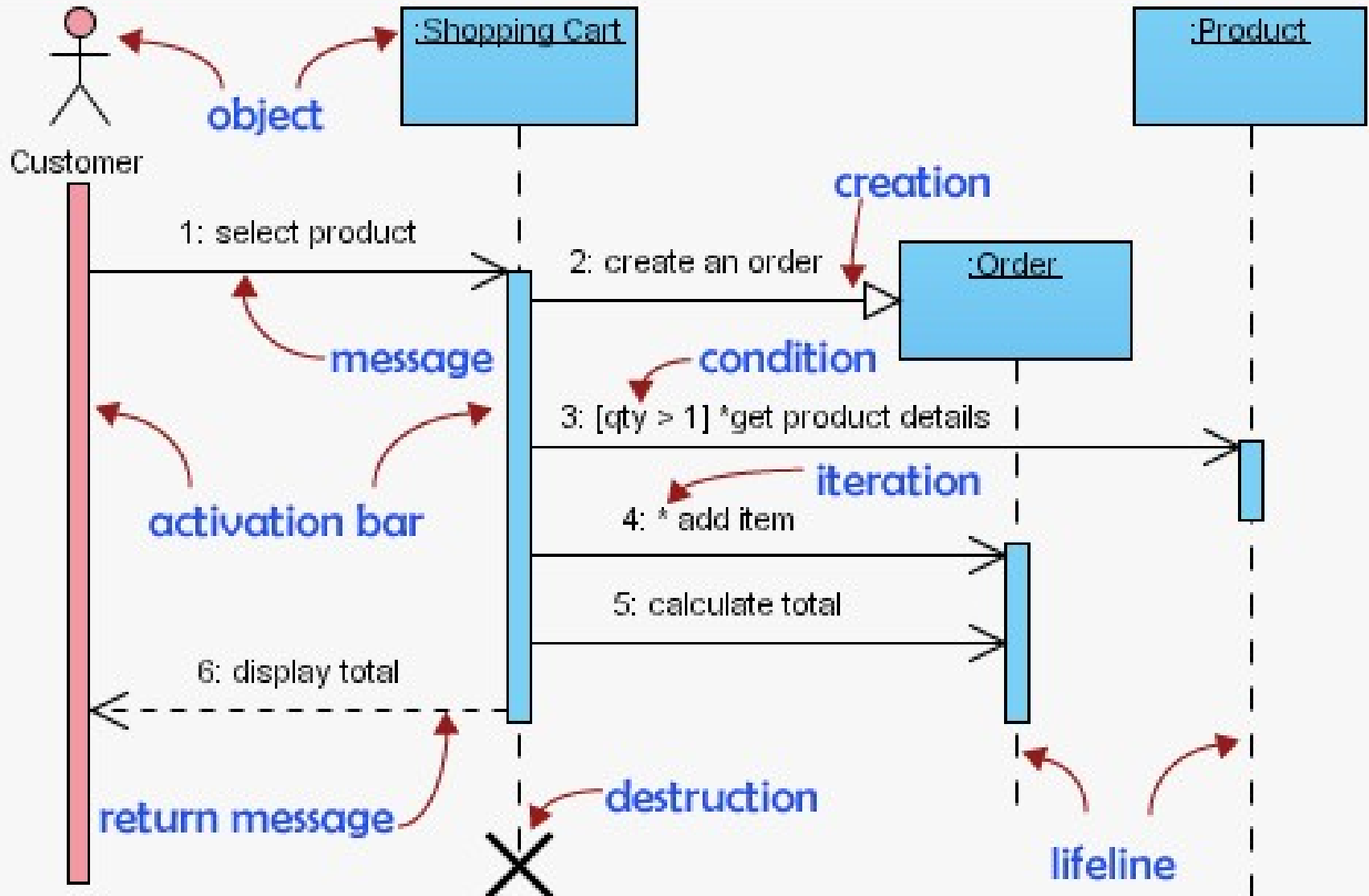
- 1 Use Case → many Scenarios
 - 일반적 처리 → "Main Scenario"
 - 에러발생시 처리
 - 특별한 경우의 처리 } "Sub Scenario"



3.10 Sequence Diagram

- 객체들사이의 상호수수작용을 시간순으로 표현
- Activity Dgm과의 차이점
 - AD :
 - 객체내부의 action 표현
 - 일반적인 상황 표현
 - SD :
 - 다른객체와의 상호수수작용표현
 - 특정한 상황에서 특정한 객체가 어떻게 동작해왔는가를 표현

Sequence Diagram





Sequence Diagram의 구성요소

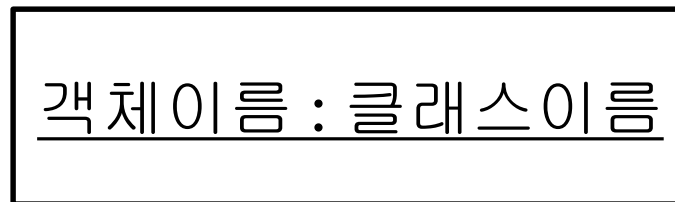
- 객체(Object)

객체이름 : 클래스이름



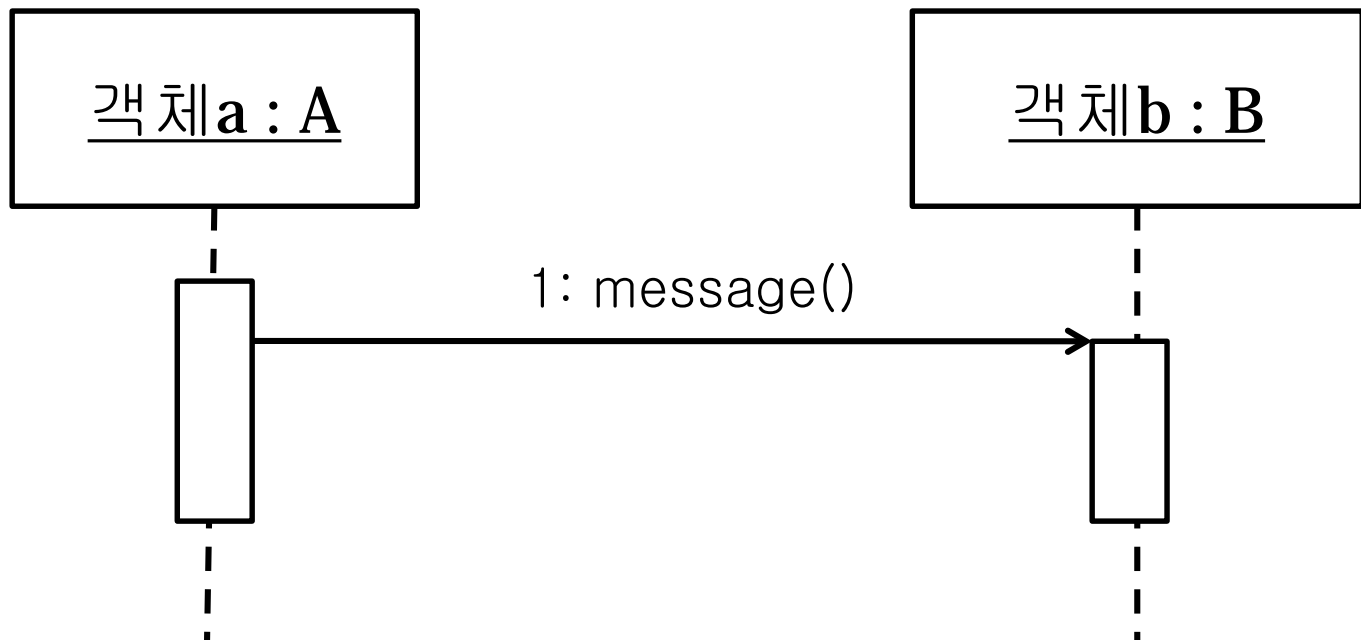
Sequence Diagram의 구성요소

- 생존선(生存線, lifeline)
 - 객체가 시스템내에서 생존하고 있는 기간을 표현



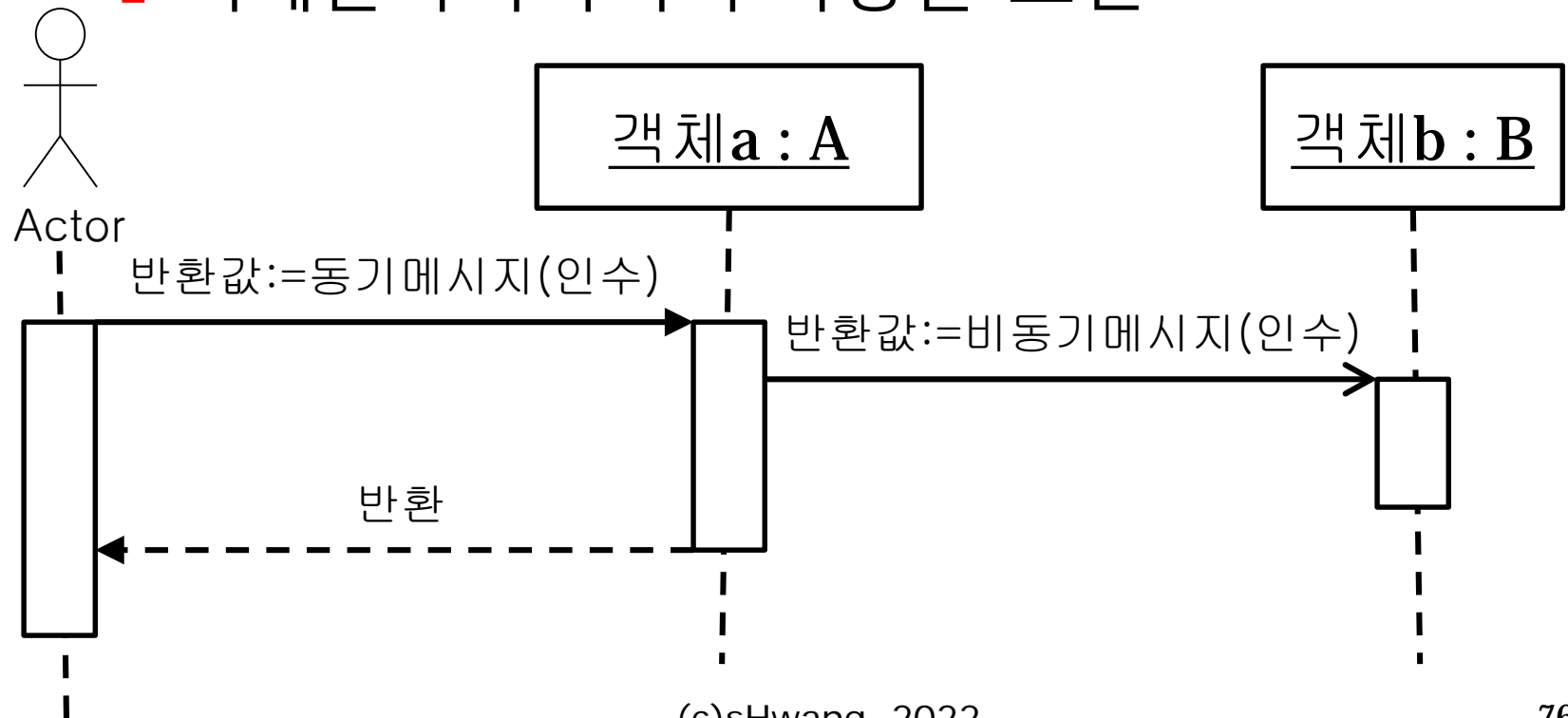
Sequence Diagram의 구성요소

- 활성화구간(Activation bar)
 - 객체가 동작을 실행하고 있는 기간을 표현



Sequence Diagram의 구성요소

- 메시지(Message)
 - 객체들사이의 수수작용을 표현



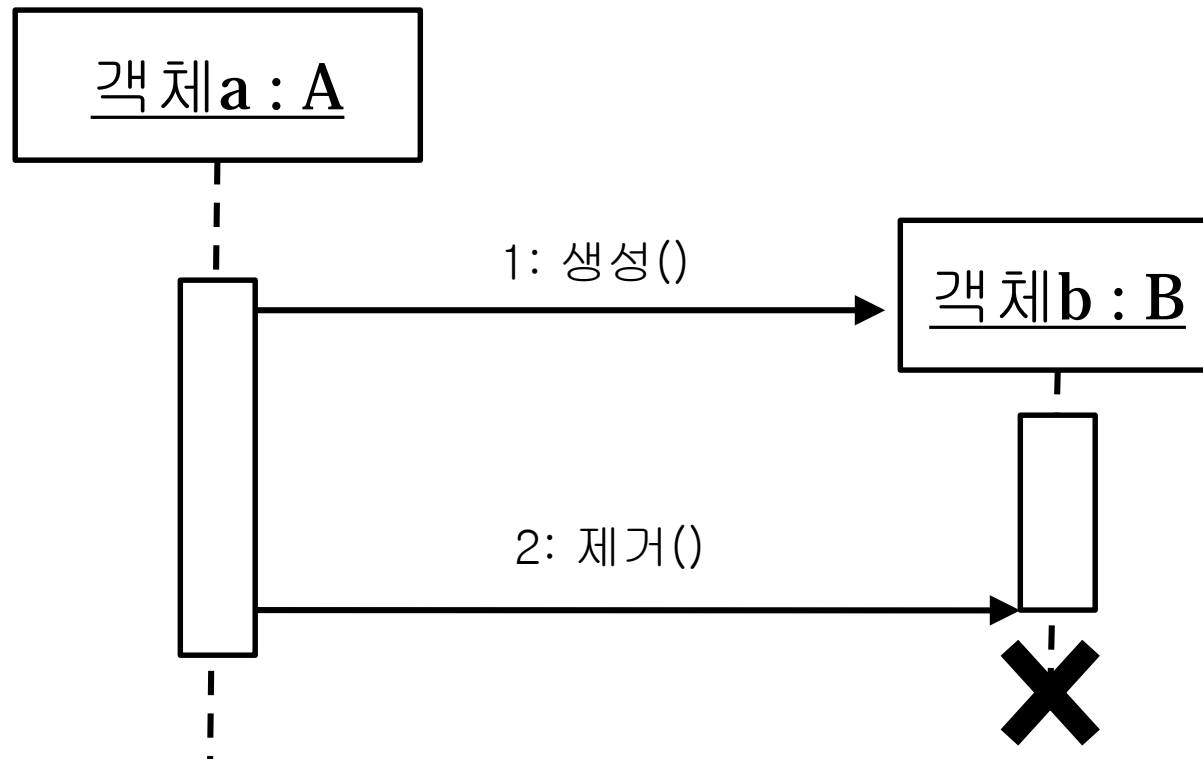


동기메세지 vs. 비동기메시지

- 동기메시지(Synchronous message)
 - 송신측은 메시지 수신측의 처리가 종료될 때까지 기다려야함.
 - “반환”을 나타내는 화살표 표시가능!
- 비동기메시지(Asynchronous message)
 - 송신측은 메시지 수신측의 처리가 종료될 때까지 기다리지 않아도 됨.

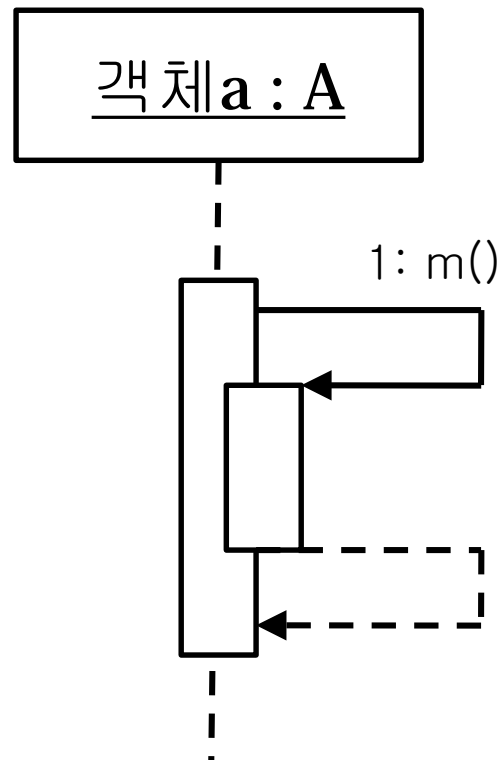
Sequence Diagram의 구성요소

- 객체의 생성 & 제거

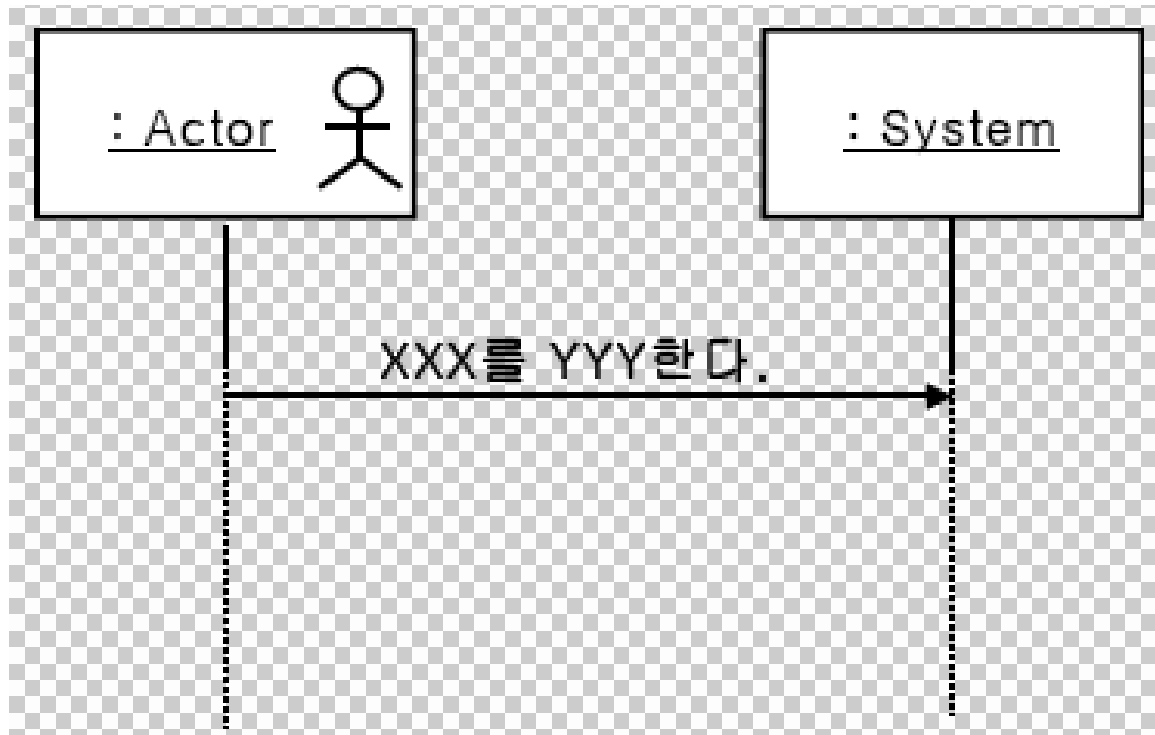


Sequence Diagram의 구성요소

- 자기호출



Sequence Diagram



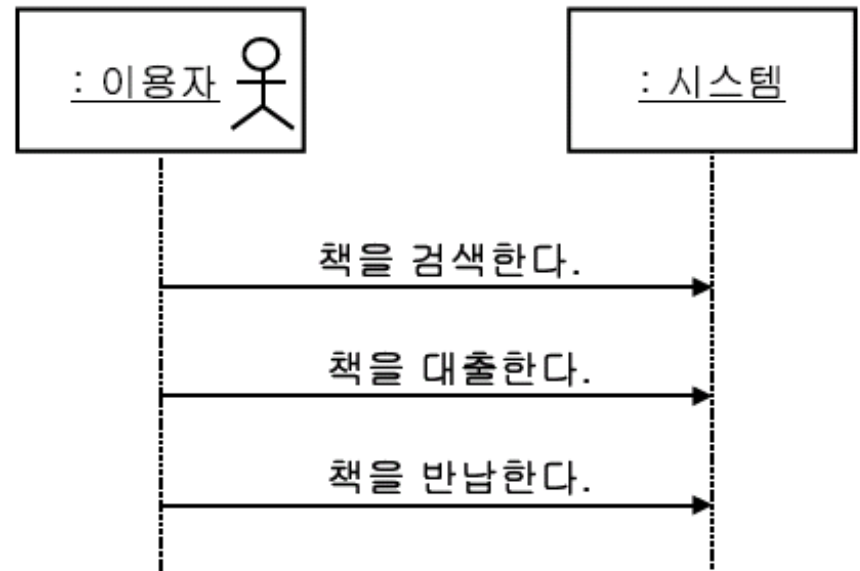
- Actor로부터 시스템으로의 요구 및 입력
- 시스템으로부터 Actor로의 응답 또는 출력

책을 이용한다 UC시나리오

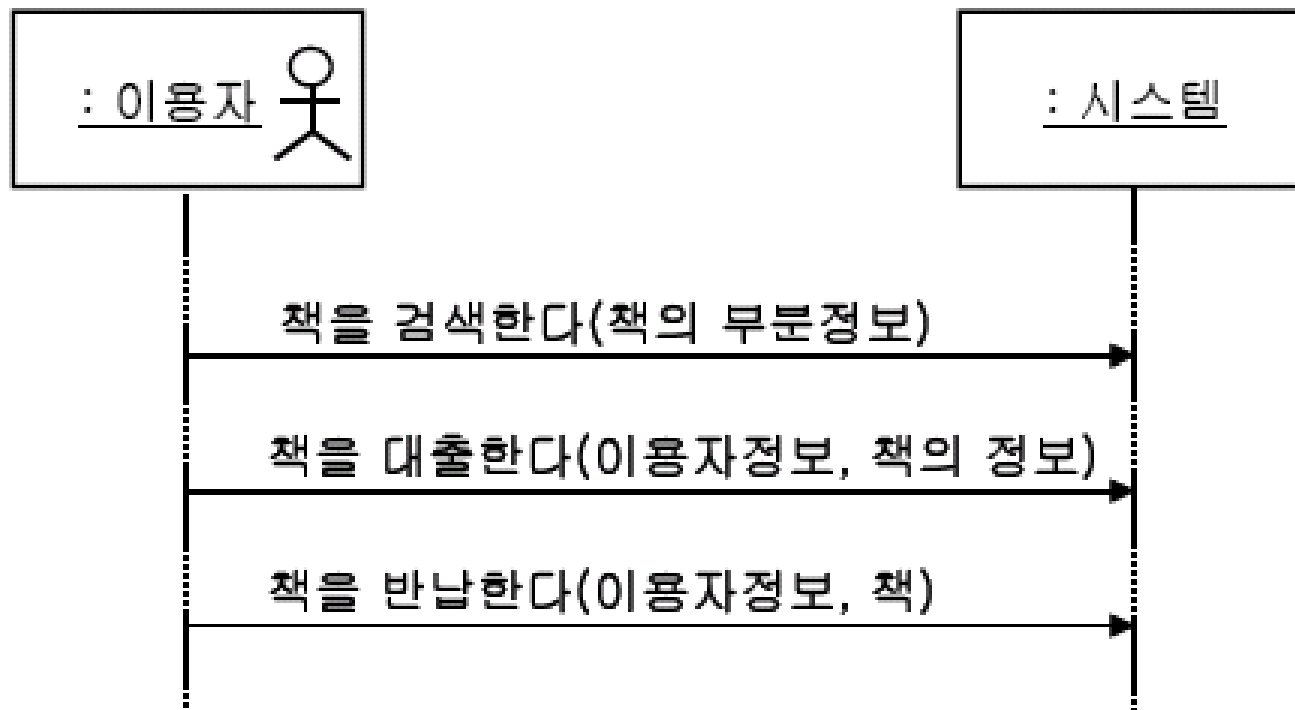
■ Main Scenario

1. (이용자 --> 시스템) 책을 검색한다.
2. (이용자 --> 시스템) 책을 대출한다.
3. (이용자 --> 시스템) 책을 반납한다.

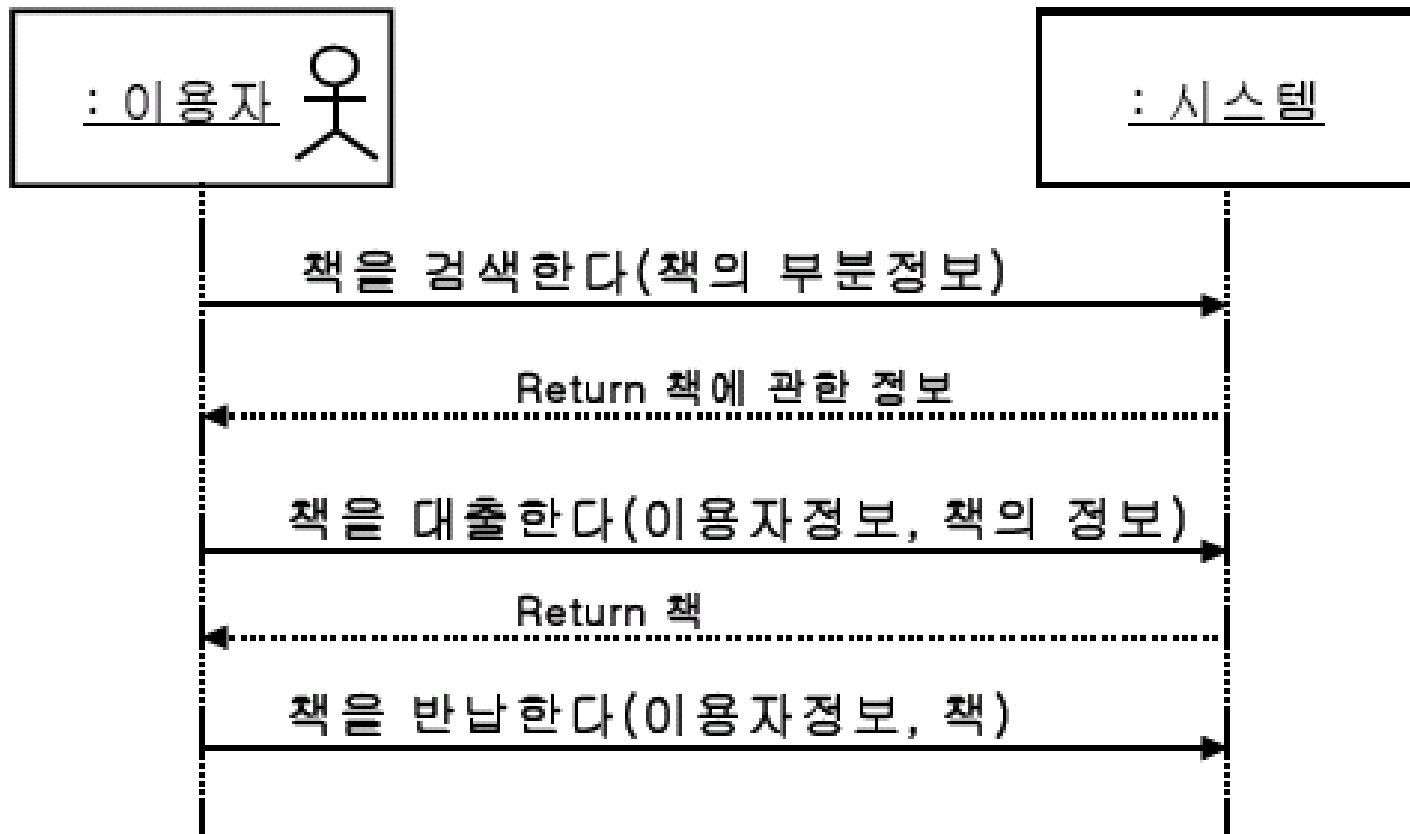
■ Sequence Diagram



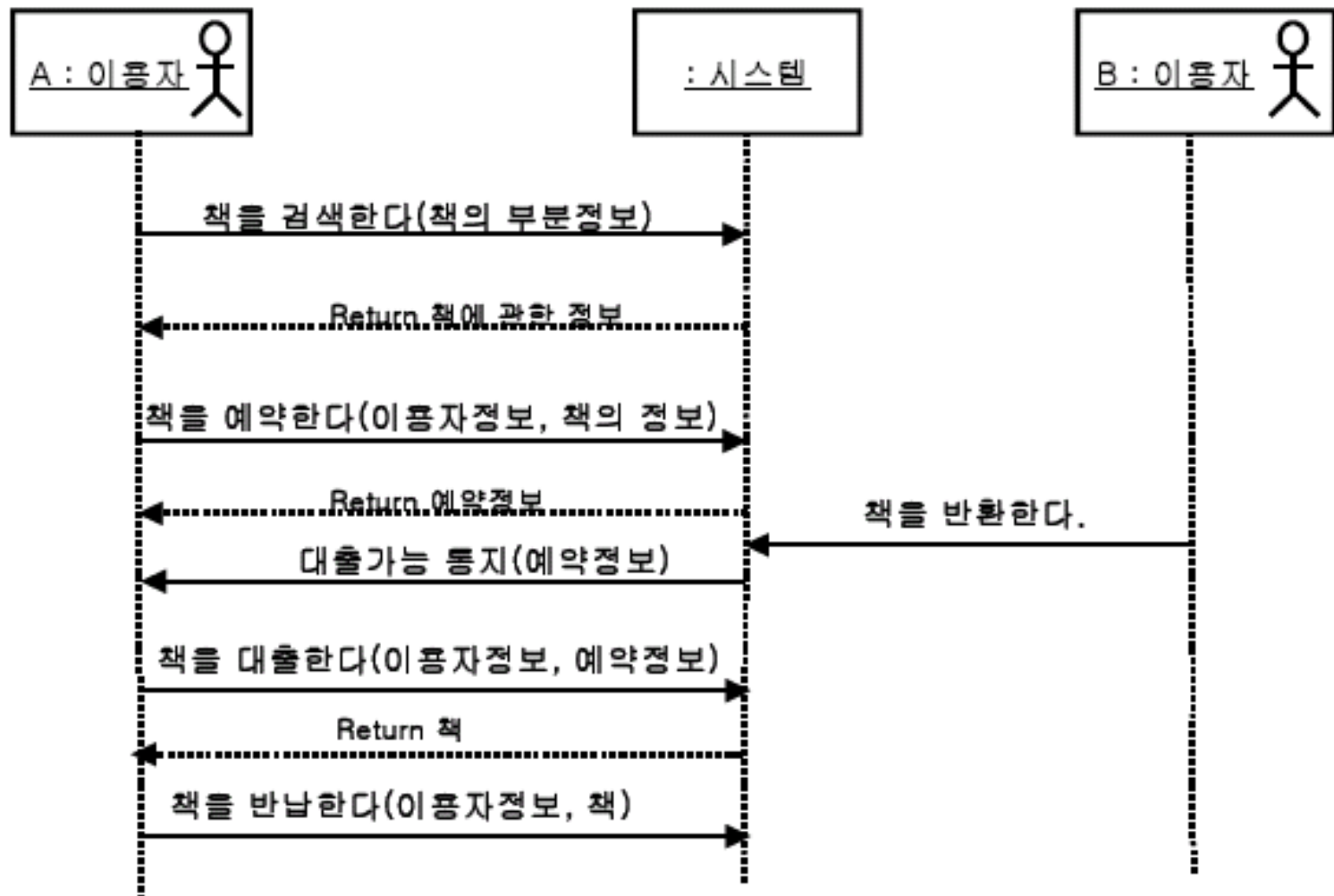
객체들 사이의 메시지 수수작용 (전달되는 정보의 표현)



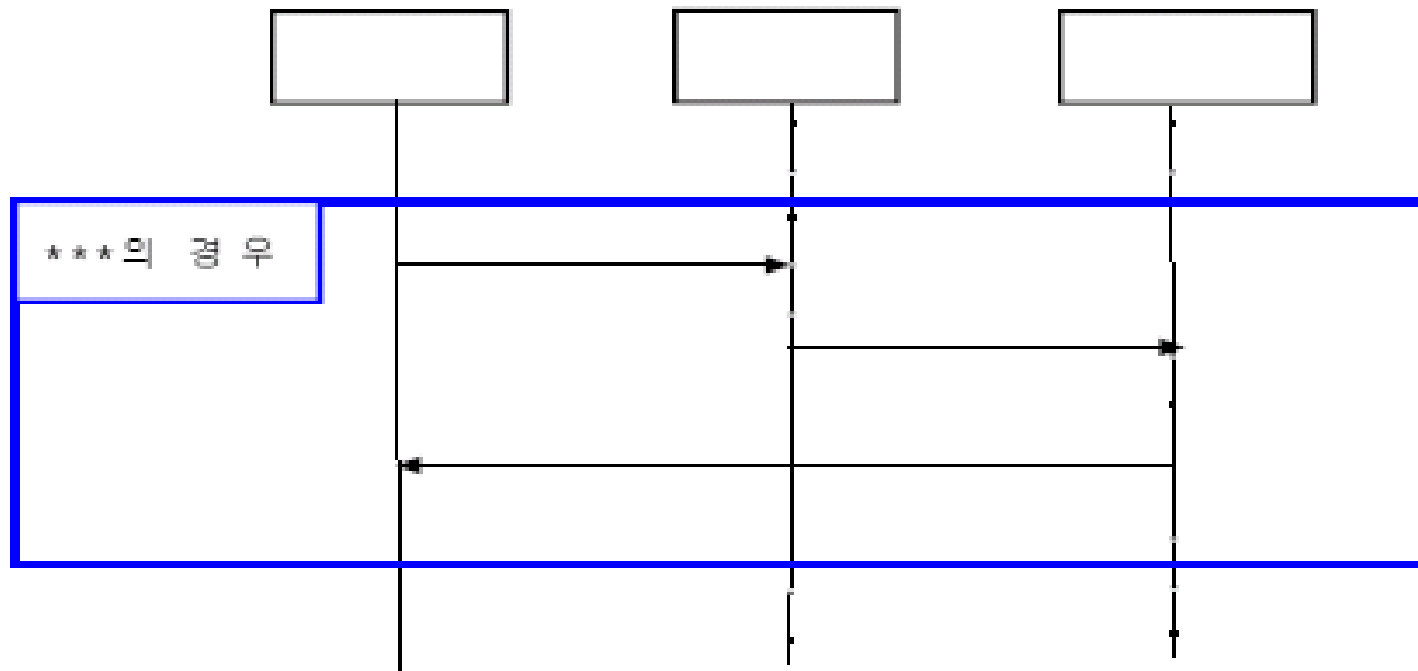
반환되는 정보 표현



"예약"에 관한 내용 추가

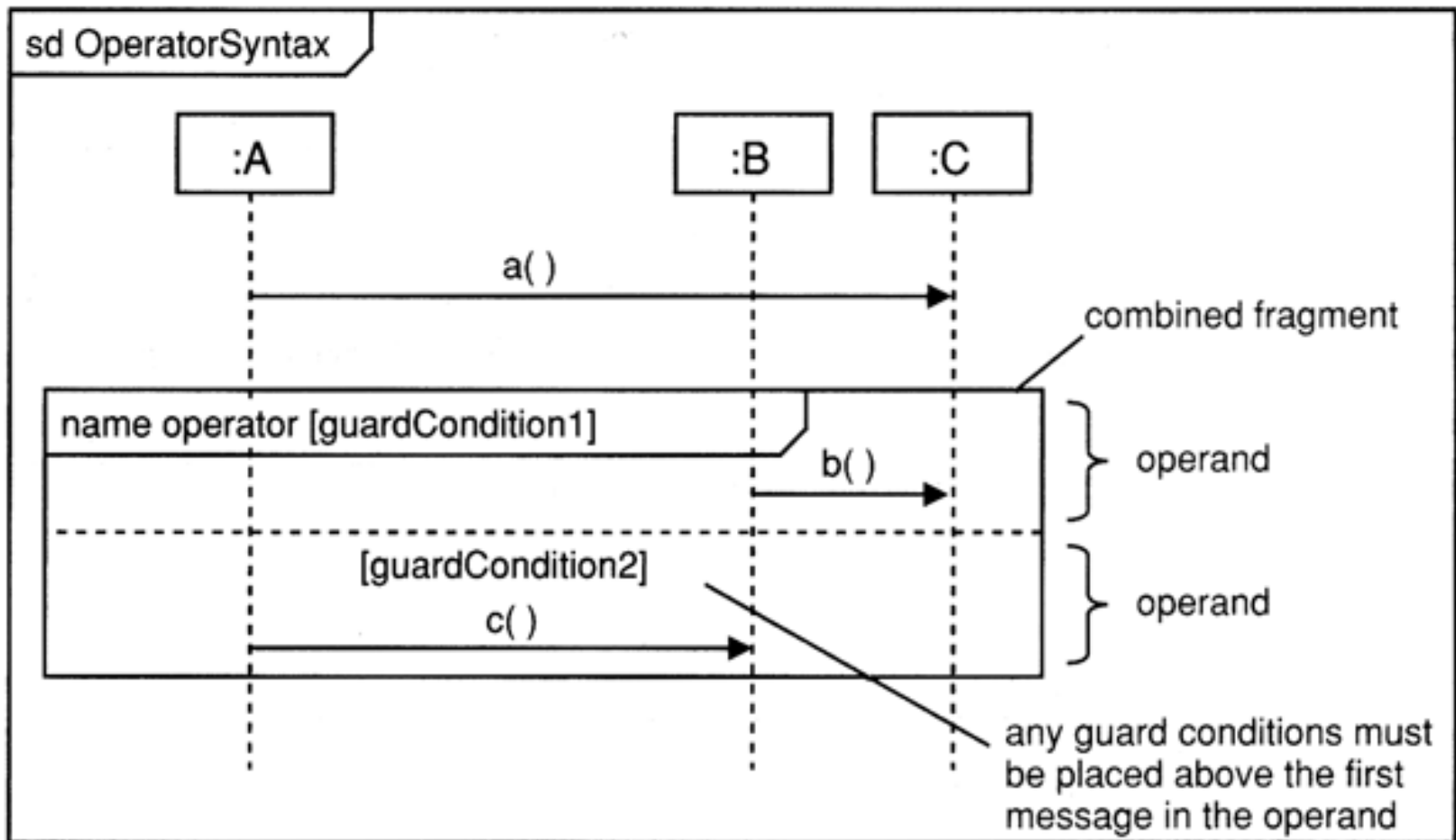


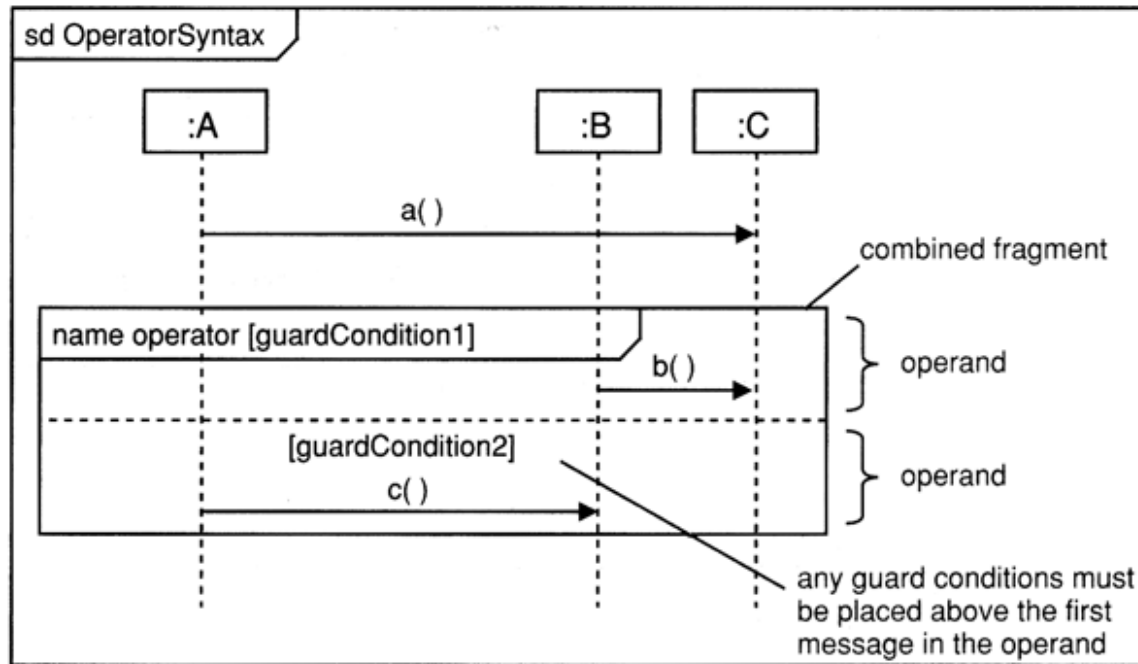
여러 개의 시나리오를 한 개 도면에 작성할 경우



Combined Fragments

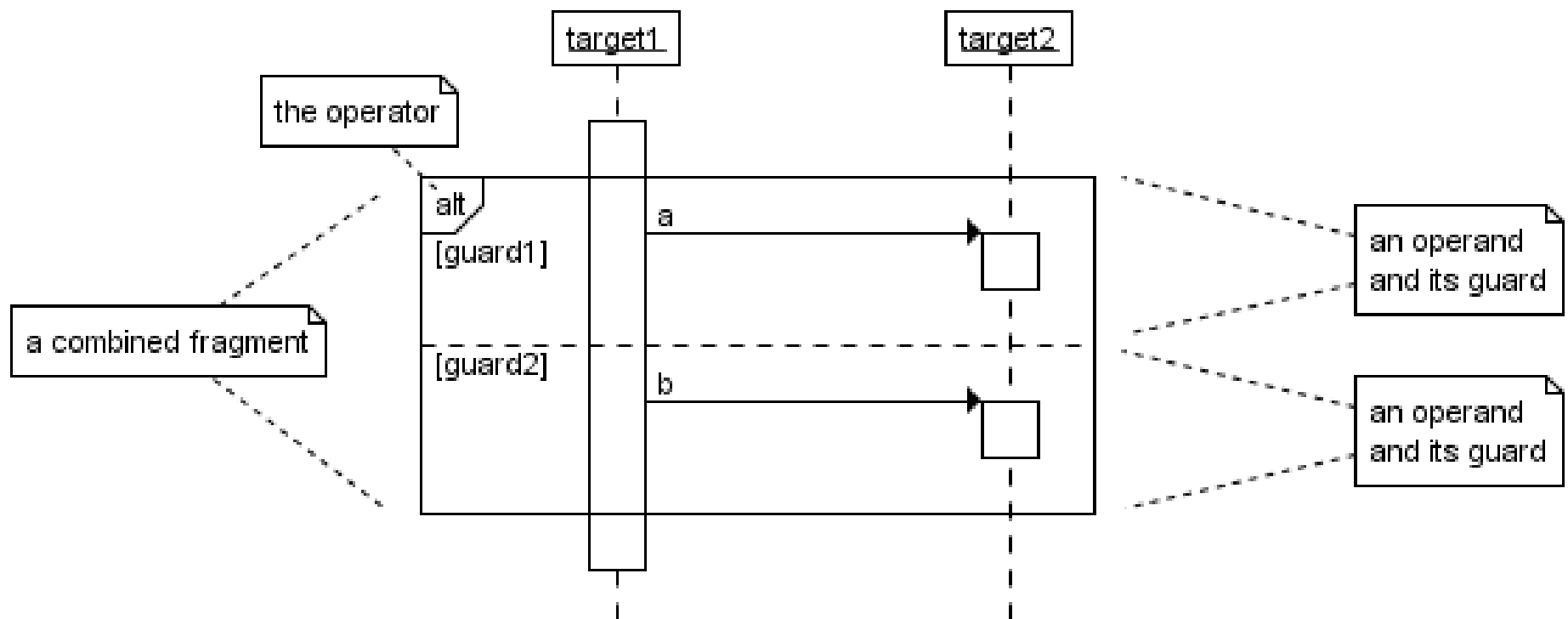
Combined Fragments and Operators





- Combined fragments divide a sequence diagram into different areas with different behavior.
- Each combined fragment has an **operator**, one or more **operands**, and zero or more **guard condition**.
- The operator determines **how** its operands execute.
- Guard conditions determine **whether** their operands execute.

Combined Fragment

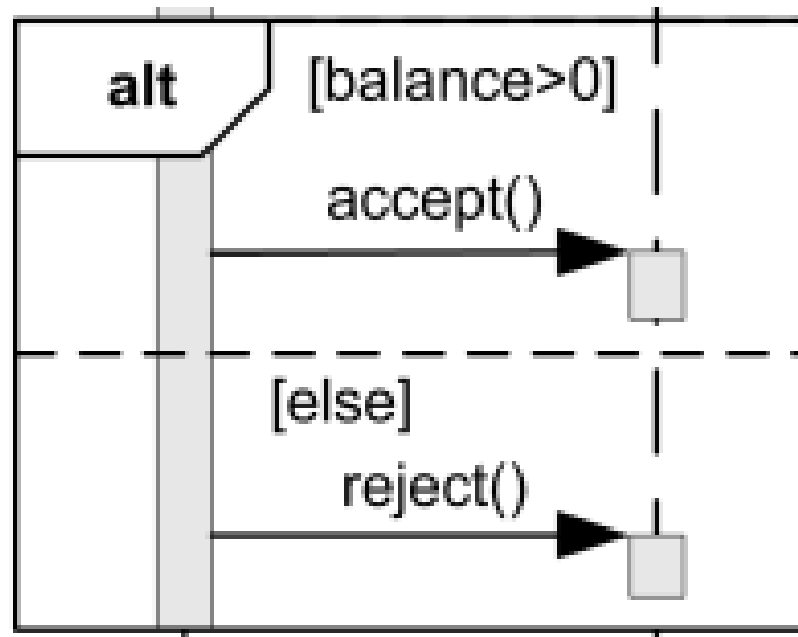




Interaction operators

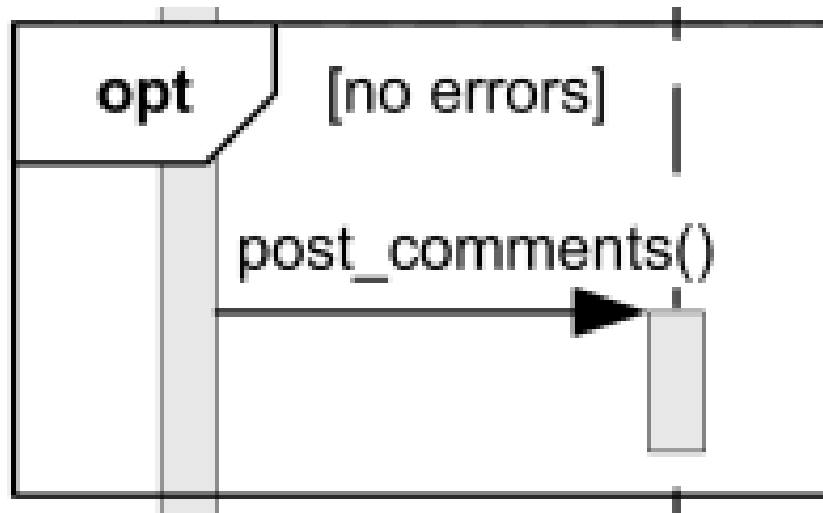
- alt - alternatives
- opt - option
- loop - iteration
- break - break
- par - parallel
- strict - strict sequencing
- seq - weak sequencing
- critical - critical region
- ignore - ignore
- consider - consider
- assert - assertion
- neg - negative

Alternatives(choice) 선택



Option

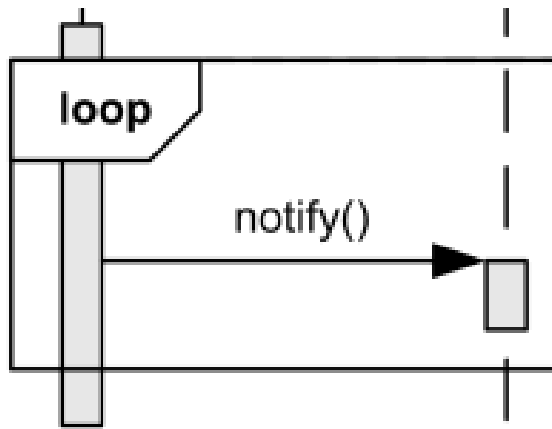
- either the (sole) operand happens or nothing happens.



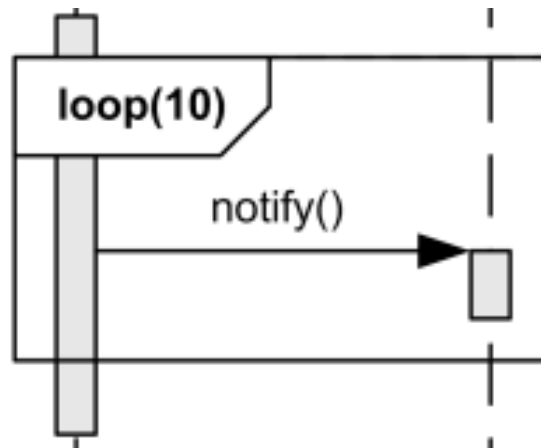
Post comments if there were no errors.

Loop

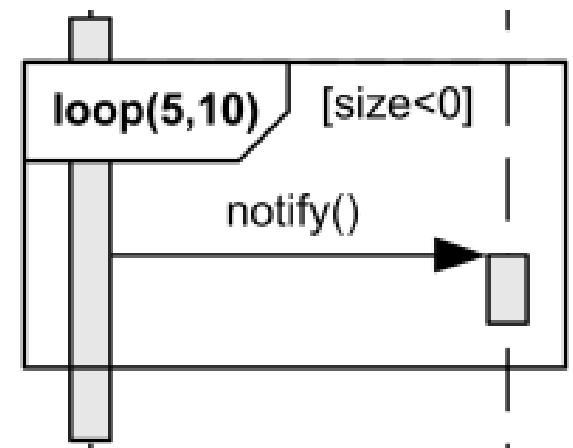
loop-operand ::= **loop** ['(' *min-int* [',' *max-int*] ')']
min-int ::= *non-negative-integer*
max-int ::= *positive-integer* | '*'



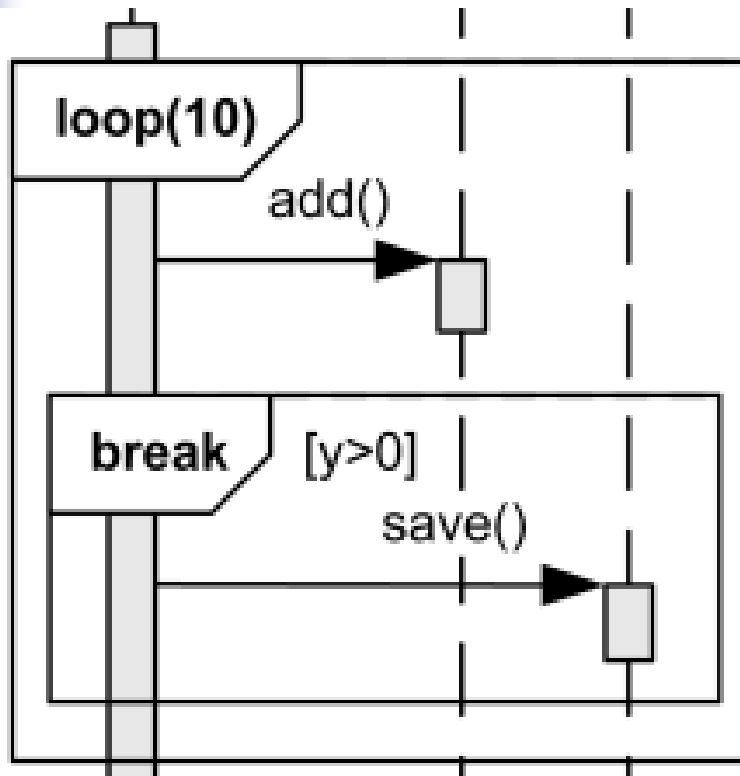
Infinite loop



Loop to execute
exactly 10 times.

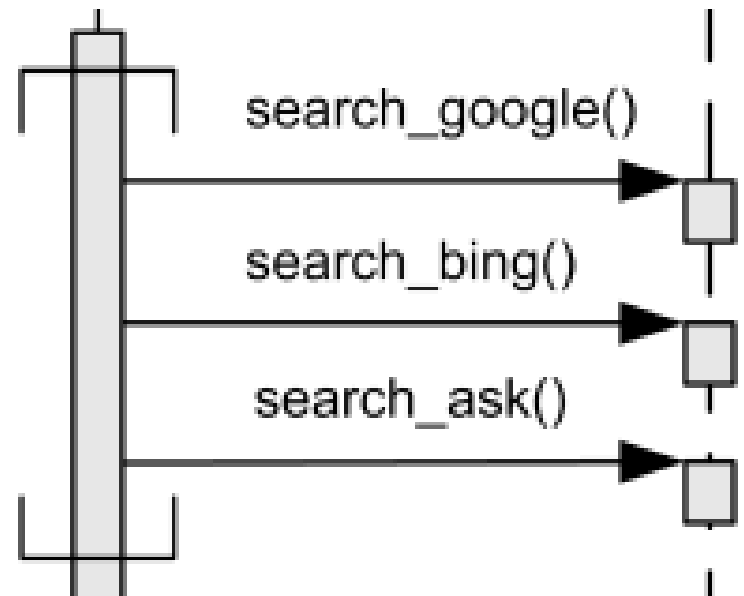
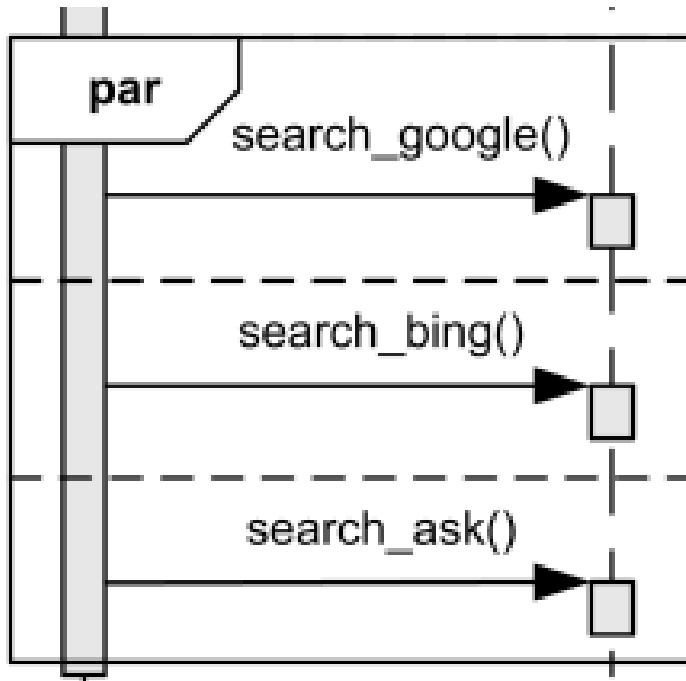


Break



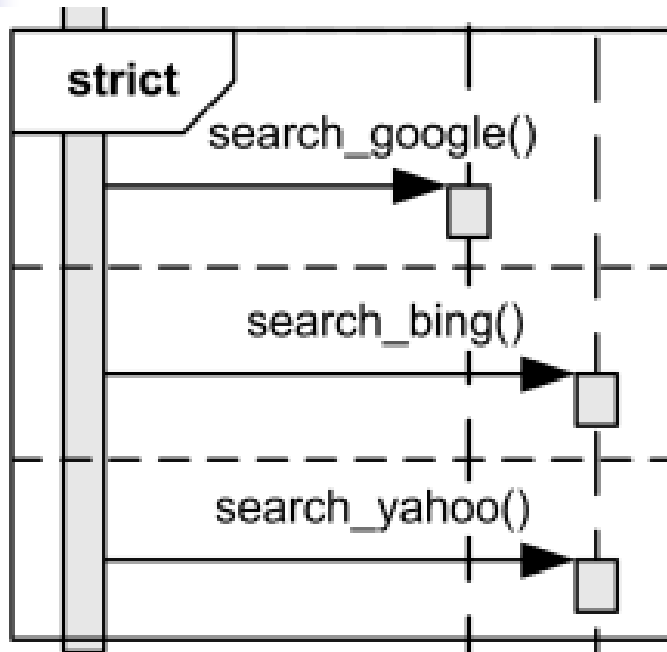
Break enclosing loop if $y > 0$.

Parallel



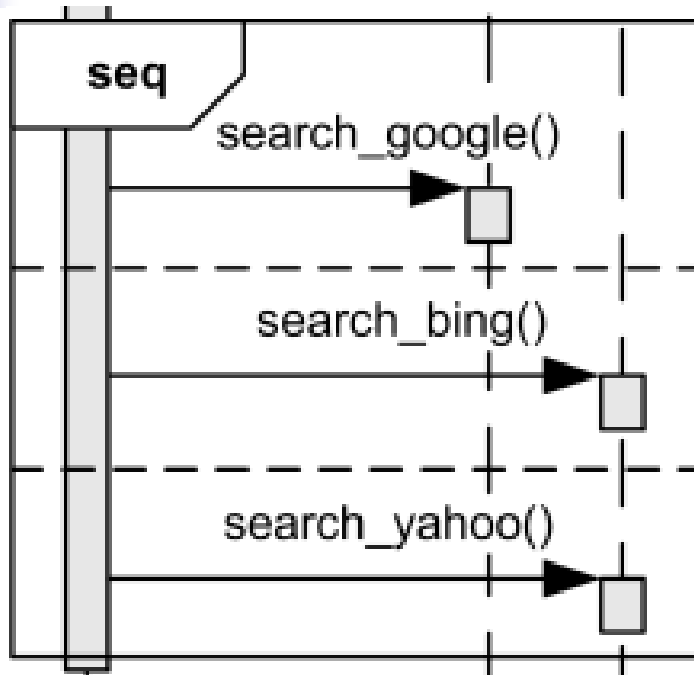
Search Google, Bing and Ask in any order, possibly parallel.

Strict Sequencing



Search Google, Bing and Yahoo in the strict sequential order.

Weak Sequencing



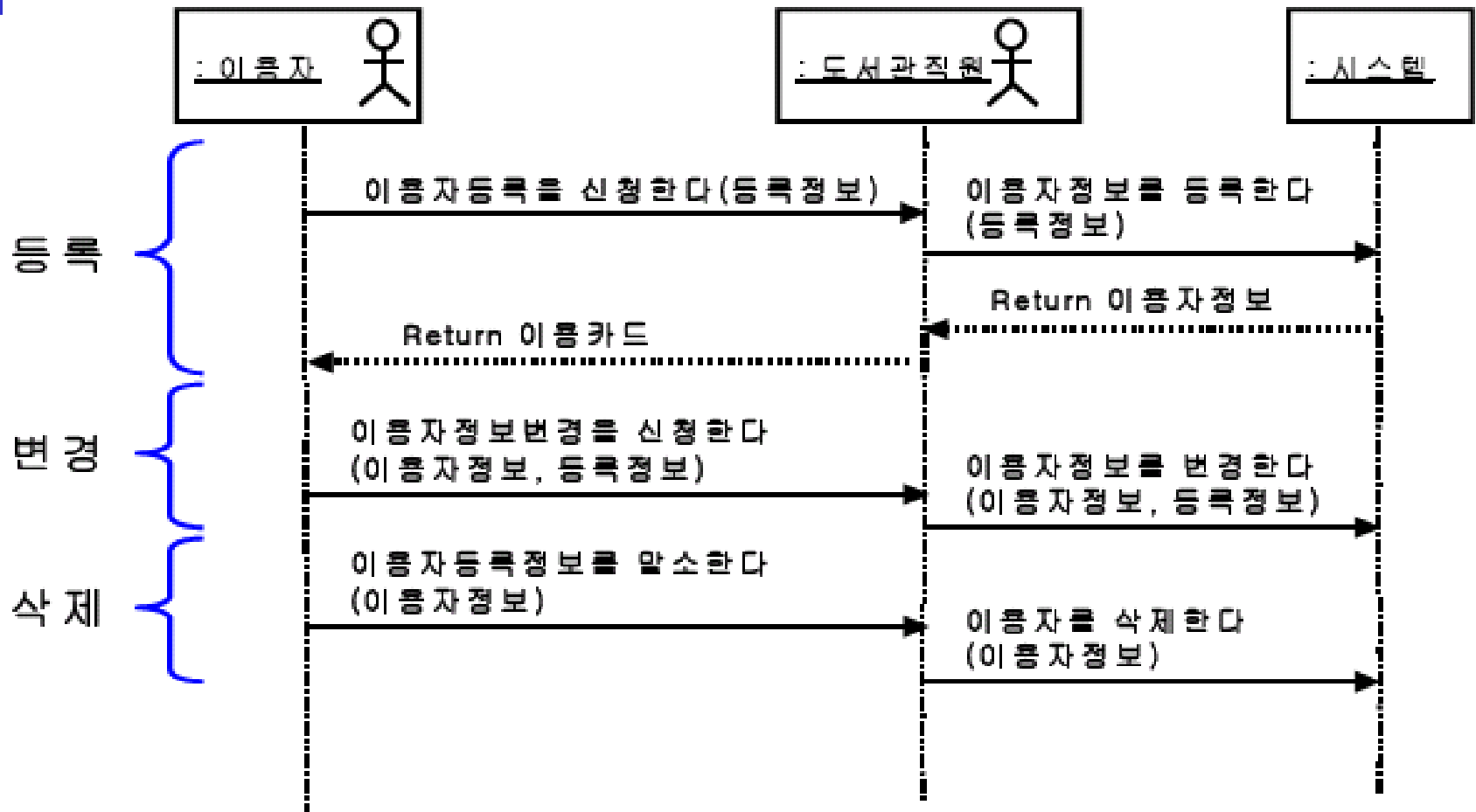
Search Google possibly parallel with Bing and Yahoo, but search Bing before Yahoo



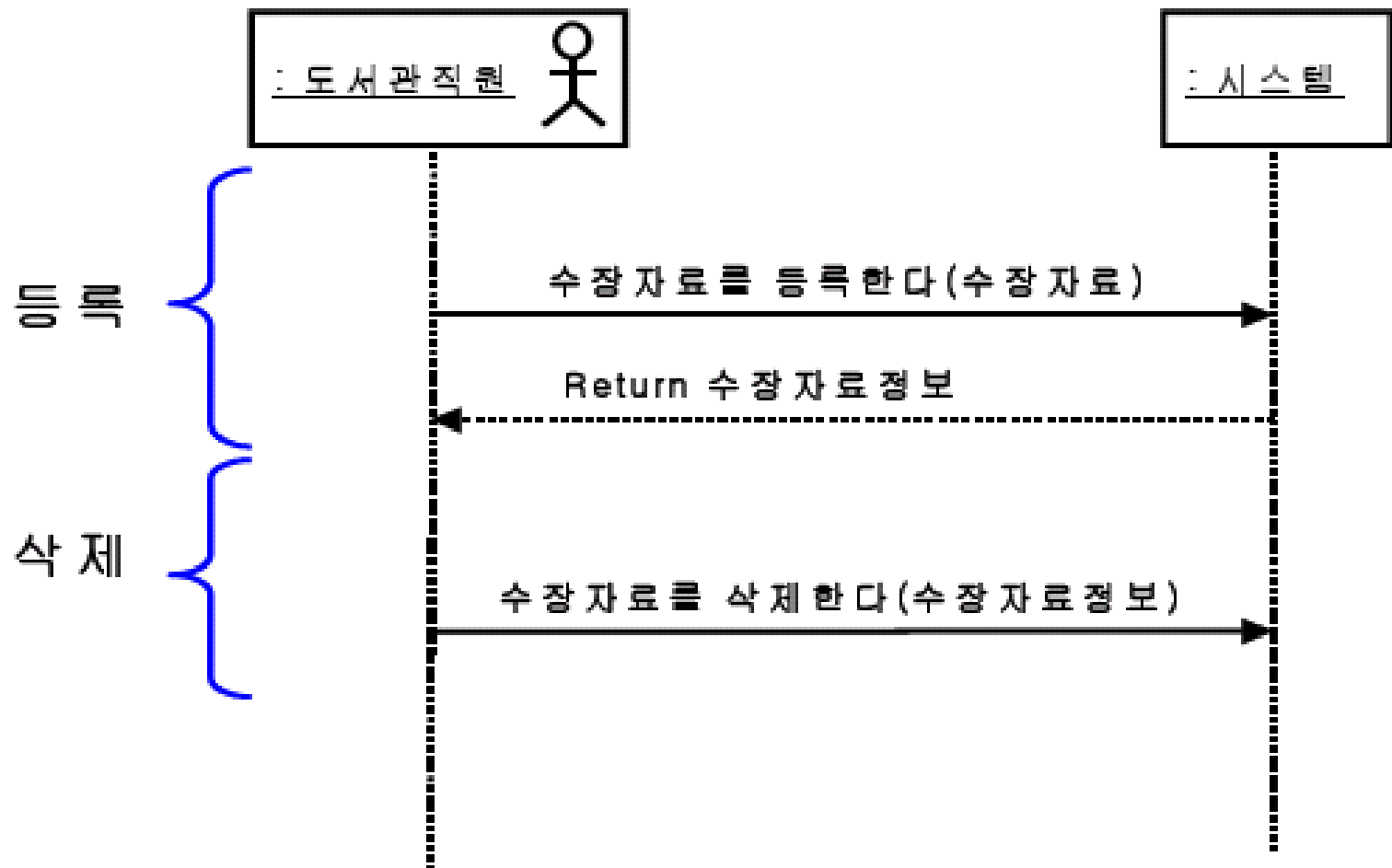
UC Scenario작성시 주의사항

- 유저에게 가치있는 업무흐름을 맨처음부터 끝까지 추적하듯 작성해야...
 - 예 : 책 검색 → 대출 → 반납
- 유저 및 고객이 살펴보고 이해할 수 있도록...
 - 시스템 내부의 구현사항은 기입하지 않도록...
 - 유저/고객의 입장에서 작성
 - 가능하다면 유저/고객과 함께 review
- 정보의 흐름에 일관성유지, 모순/단절 없도록
- 경우에 따라서는 확대, 축소, 분할, 합병

“이용자 등록/변경/삭제”에 관한 UC Scenario



"수장자료의 등록/삭제"





3.11 요구사항의 추가

- 지금까지 작성한 UCD, AD, SD들을 A시립 도서관 담당자와 review해 보면, 이전에 언급되지 않던 사항들이 나타나게 됨.
 - 반납기한 초과된 경우의 대응책은?
 - “도서관이용규칙”도 반영되어야 함.
 - 대출에 관한 규칙
 -에 관한 규칙
 - ...

- ▶ 한사람이 한번만 등록할 수 있다. 즉, 중복해서 등록할 수 없다(그러나, 사용자 등록정보는 신청자 스스로 작성해서 신고하도록 되어 있다.)
- ▶ 등록할 수 있는 사람은, A시 거주자 또는 근무지가 A시로 되어 있는 사람들에게 한정된다.
- ▶ 등록에 필요한 사항으로는, 성명, 주소, 성별, 생년월일, 연락처(전화번호 또는 e-mail 주소), 주민이 아닌 사람의 경우에는, 근무지명과 근무지 주소, 연락처가 추가됨.
- ▶ 등록한 사람에게는 이용자 카드를 발급한다. 분실했을 경우에는 재발급한다.
- ▶ 이용자격이 상실되면 신고하여 변경한다.
- ▶ 대출가능한 소장자료는, 책 및 잡지류의 경우 10권 이내, 음악 및 영상물은 5개 이내로 제한되며, 최장 2주일까지 대출가능. 반납기한이 휴관일인 경우에는 익일까지 반납한다.
- ▶ 반납기한을 넘기고도 반납하지 않은 대출물이 있을 경우에는, 반납완료시까지 새로운 대출은 불가능하다.
- ▶ 예약할 수 있는 자료의 개수는 제한없음. 도서관에 소장되어 있지 않은 서적이라도 예약가능(즉, 구입희망신청 가능).
- ▶ 예약한 자료가 완비되었을 경우, 예약등록순서에 의해 연락처로 연락한다. 연락한 후 2주일 경과해도 대출하지 않았을 경우에는 다음 예약자에게 연락함.



3.12 UC Scenario ➔ 테스트 시나리오

- 현재까지 작성된 UCS
 - 주요 사항들만 파악
- 각종 규칙까지 철저히 파악하여 작성된 UCS
 - 개발완료후의 “인수테스트”에 활용가능

“어떤 UCS를 사용할 것인가?” 선택시, 고려사항

- 각종 규칙들은 앞으로 얼마나 추가/변경 될 것인가?
- 개발자들(설계자 포함)은 얼마나 모델을 중요시하는가?
- UML Tool에서는, UCS로부터 자동적으로 테스트 시나리오를 추출가능한가?



3.13 UC의 사전/사후조건

- 각 UC마다 “사전조건”과 “사후조건”을 기입
 - 事前條件(Preconditions)
 - What must be true *before* the use case can start.
 - 해당UC가 실행되기 위해 필요한 조건
 - 즉, 사전조건들이 모두 만족되어야만 UC실행가능
 - 事後條件(Postconditions)
 - What will be true *after* the use case has executed.
 - 해당UC실행후에 만족되어야하는 조건



Hoare Triple

`{ PreCondition } Operation() { PostCondition }`

`{ t >= 0 } getSqaare(t) { result = t*t }`



Example

Use Case Description

- **Use Case name:** Select Destination
- **Summary:** The user in the elevator presses an up or down elevator button to select a destination floor to which move
- **Dependency**
- **Actor:** Elevator User
- **Precondition:** User in the elevator
- **Description:**
 - 1. User presses an elevator button to go up. The elevator button sensor sends the elevator button request to the system, identifying the destination floor the user wishes to visit.
 - 2. The new request is added to the list of floors to visit. If the elevator is stationary,
Include Dispatch Elevator abstract use.
 - **3. Include Stop Elevator at Floor abstract use case.**
 - If there are other outstanding requests, the elevator visits these floors on the way to the floor requested by the user, following the above sequence of dispatching and stopping. Eventually, the elevator arrives at the destination floor selected by the user.
- **Alternatives:**
 - 1. User presses down elevator button to move down. System response is the same as for the main sequence.
 - 2. If the elevator is at a floor and there is no new floor to move to, the elevator stays at the current floor, with the door open.
- **Postcondition:** Elevator has arrived at the destination floor selected by the user.



사전/사후조건의 예

“수장자료를 대출한다”

■ 사전조건

1. 대출하려는 사람이 이용자로서 등록되어 있을 것.
2. 대출하려는 사람에게 반납기한이 초과된 미반납자료가 없을 것.
3. 대출가능한 수량 이내이어야 함.
4. 수장자료가 대출가능해야 함.

■ 사후조건

1. 대출자의 대출완료점수가 증가한다.
2. 수장자료는 대출자에게 대출된 상태가 된다.



사전/사후조건의 기입

- 자연언어
- OCL(Object Constraint Language)
- Java언어와 비슷한 "유사코드"
- 주의사항
 - "조건"을 기입
 - "해야할 일"을 기입하는 것이 아님!!!



“이용자를 등록한다”

- 사전조건

1. 이용자는 아직 등록되어 있지 않다.
2. 이용자는 등록할 자격을 갖추고 있다.

- 사후조건

1. 이용자는 등록되어 있다.



“예약한다”

- 사전조건

1. 대상자료가 미수장 또는 대출중
2. 이용자가 등록되어 있다.

- 사후조건

1. 대상자료의 예약리스트에 이용자가 추가되었다.
2. 이용자는 대상자료를 예약하였다.



“자료를 검색한다”

- 사전조건 : 없음.
- 사후조건 : 없음.



“수장자료를 반납한다”

- 사전조건

1. 이용자가 해당수장자료를 대출하고 있다.
2. 해당수장자료는 대출중이다.

- 사후조건

1. 이용자는 해당수장자료를 대출하고 있지 않다.
2. 해당수장자료는 대출중이 아니다.
3. “해당수장자료는 예약되어 있지 않다.” 또는 “해당수장자료를 맨처음 예약한 사람에게 연락하고 있다.”



이용자정보를 변경한다

- 사전조건

1. 이용자가 등록되어 있다.
2. 정보변경하더라도 이용자의 자격을 만족하고 있다.

- 사후조건

1. 이용자정보가 신청된 정보와 동일하다.



이용자를 삭제한다

- 사전조건
 1. 이용자가 등록되어 있다
- 사후조건
 1. 이용자가 등록되어 있지 않다.

문제점

이용자가 책을 대출하고 있는 중에도 “이용자를 삭제한다”가 가능.

해결책

삭제가능한 이용자 = 대출중인 책이 없는 이용자



이용자를 삭제한다

- 사전조건
 1. 이용자가 등록되어 있다
 2. **이용자는 대출하고 있는 자료가 없다.**
- 사후조건
 1. 이용자가 등록되어 있지 않다.