# Case Study : Analyze
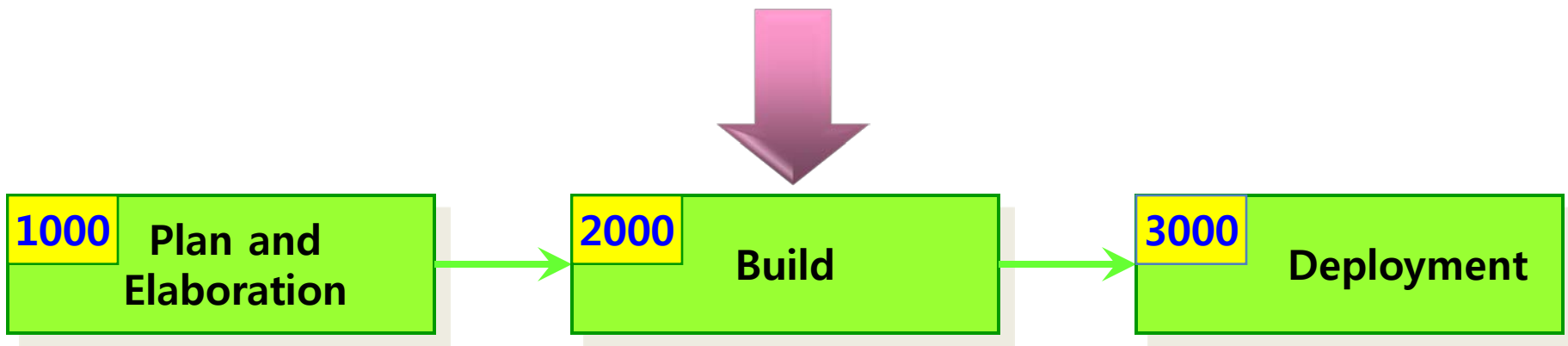# - Library Management System
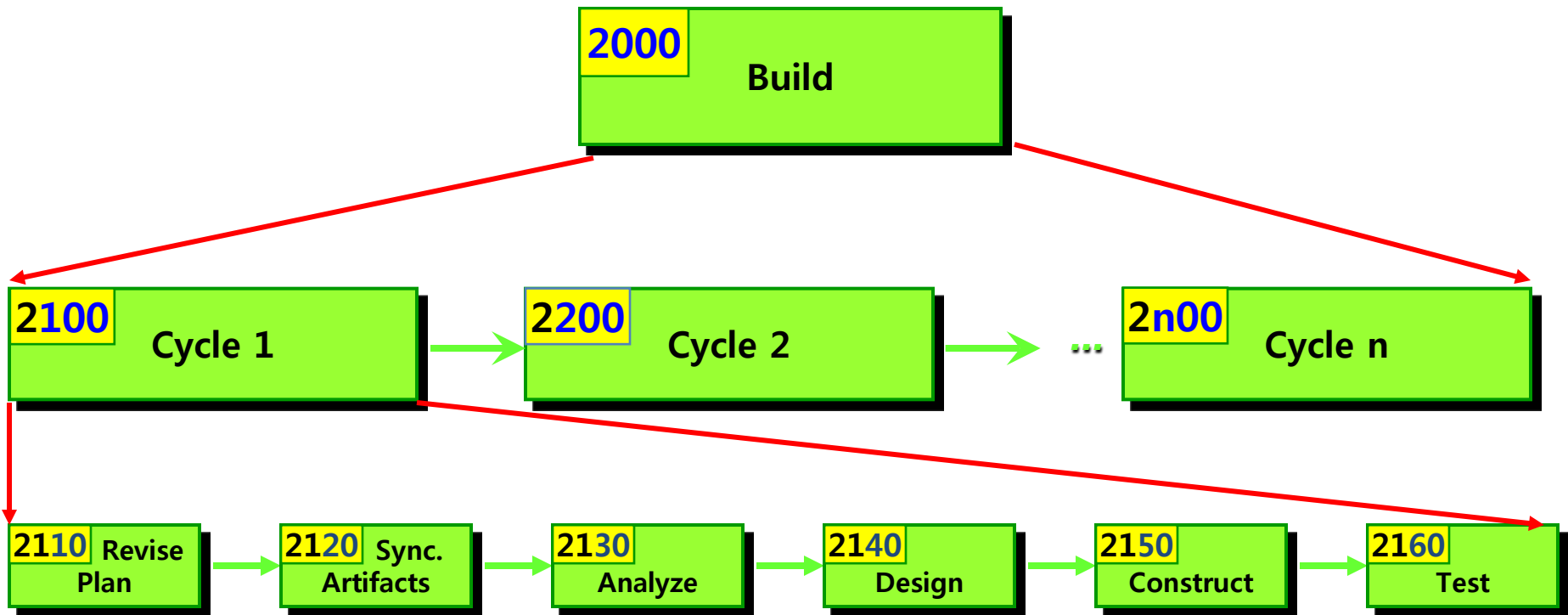
Lecturer: JUNBEOM YOO
jbyoo@konkuk.ac.kr

# Stage 2000. Build



| 1000 Plan and Elaboration | → | 2000 Build | → | 3000 Deployment |

# Stage 2000. Build

- 6 Phases of Stage 2000 Build

# Phase 2010.
# Revise Plan

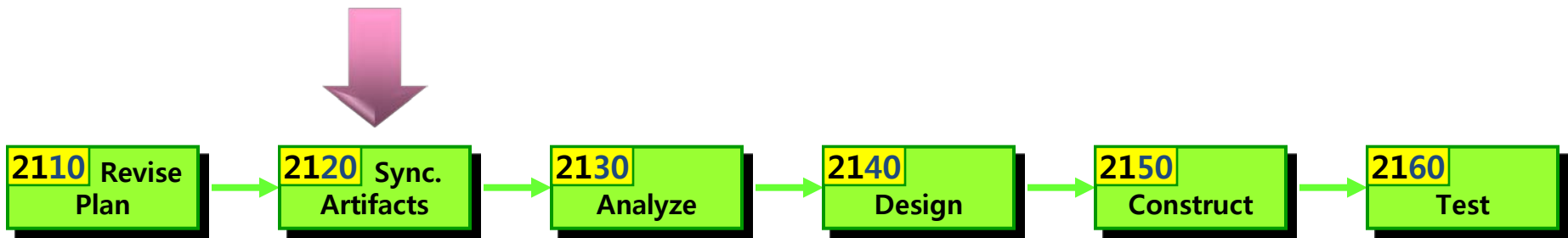| 2110 Revise Plan | 2120 Sync. Artifacts | 2130 Analyze | 2140 Design | 2150 Construct | 2160 Test |
|---|---|---|---|---|---|
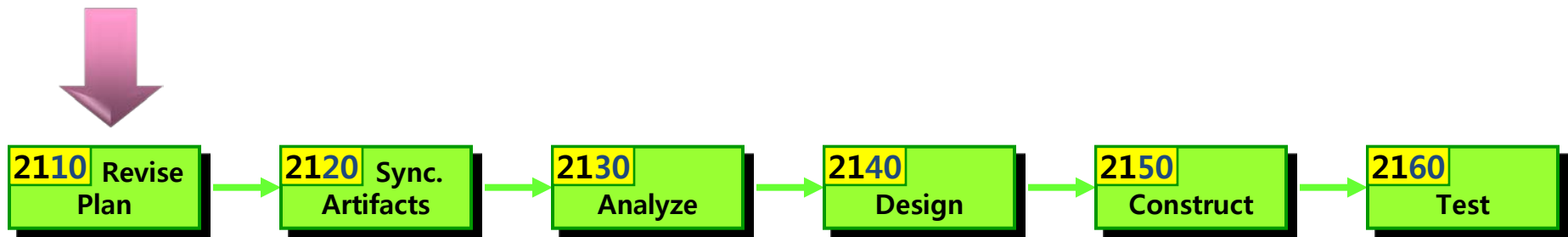
# Phase 2010. Revise Plan

- Not Applicable
  - Since this mini project has only 1 development cycle

# Phase 2020.
# Synchronize Artifacts

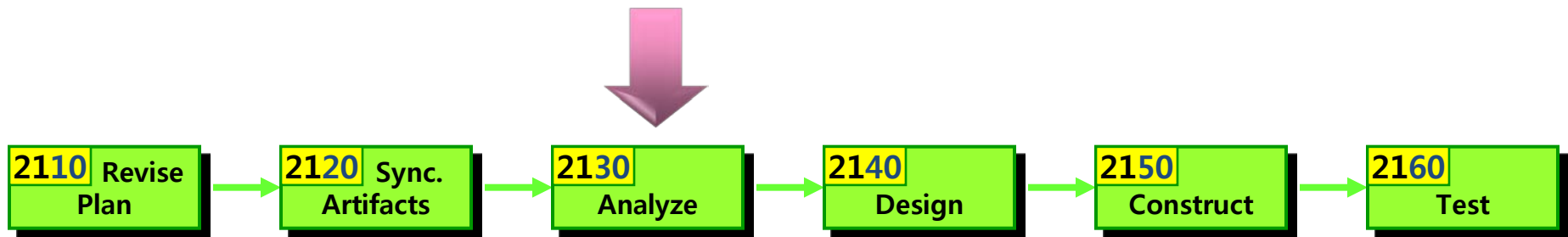| 2110 Revise Plan | 2120 Sync. Artifacts | 2130 Analyze | 2140 Design | 2150 Construct | 2160 Test |

# Phase 2020. Synchronize Artifacts

- Management of Configuration Items
  - Diagrams
  - UI Design
  - Database Design
  - Source Code
  - Object Code
  - etc

- Others are not Applicable
  - since this mini-project has only 1 development cycle.

# Phase 2030.
# Analyze



| 2110 Revise Plan | 2120 Sync. Artifacts | 2130 Analyze | 2140 Design | 2150 Construct | 2160 Test |

# Phase 2030. Analyze

- Phase 2030 Activities

a. **if not yet done**
b. **ongoing**
c. **optional**

**2130** Analyze

**2131** Define Essential Use Cases [a]

**2132** Refine Use Case Diagrams

**2133** Define Domain Model

**2134** Refine Glossary [b]

**2135** Define System Sequence Diagrams

**2136** Define Operation Contracts

**2137** Define State Diagrams [c]
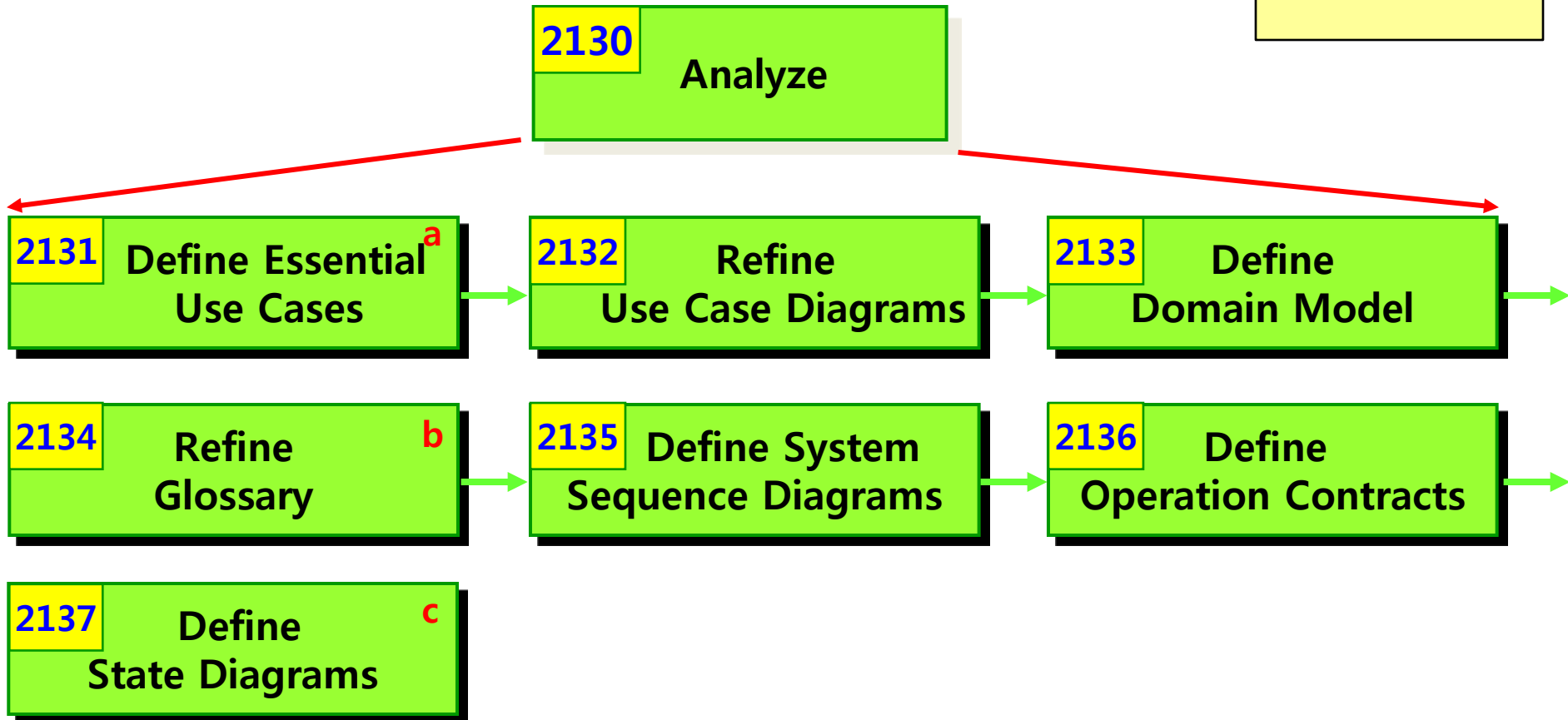
# Activity 2031. Define Essential Use Cases

a. **if not yet done**
b. **ongoing**
c. **optional**

| 2131 | **Define Essential Use Cases** a |
| 2132 | **Refine Use Case Diagrams** |
| 2133 | **Define Domain Model** |

| 2134 | **Refine Glossary** b |
| 2135 | **Define System Sequence Diagrams** |
| 2136 | **Define Operation Contracts** |

| 2137 | **Define State Diagrams** c |
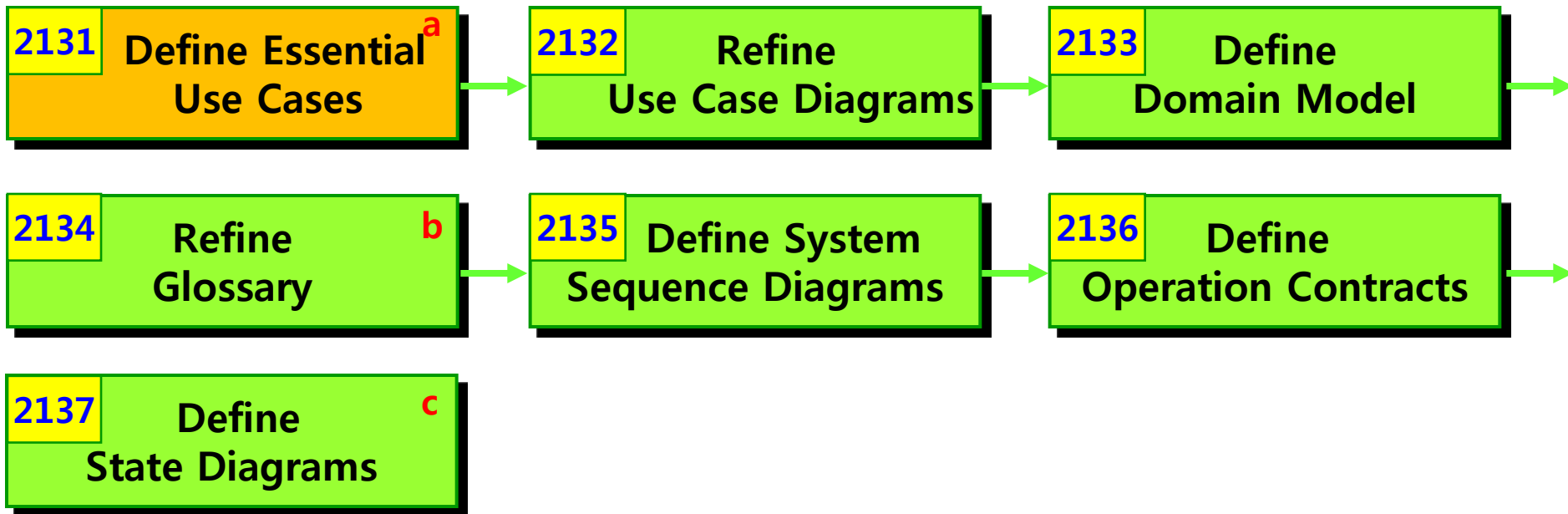
# Activity 2031. Define Essential Use Cases

- Step 1
  - Select each use case from high-level use cases.
- Step 2
  - Identify system functions related to selected use case from requirement specification. (Ex) R1.1, R1.2)
  - Identify related use cases to selected use case from high-level use cases.  (Ex) Use Cases: *Log In* use case)
- Step 3
  - Identify a Typical Courses of Events(Main Event Flow) for each use case from the requirement specification.
  - Identify an Alternative Courses of Events for each use case from the requirement specification.
- Step 4
  - Write essential use cases based on typical courses of events and alternative courses of events flows by applying expanded format.

# Activity 2031. Define Essential Use Cases

## 1. Make Reservation

| Use Case | 1. Make Reservation |
|---|---|
| Actor | Librarian |
| Purpose | (As in the business use case) |
| Overview | (As in the business use case) |
| Type | Primary and Essential |
| Cross Reference | System Functions: R1.1, R3.1<br>Use Case: "Add Borrower" |
| Pre-Requisites | Borrower should have an id_card. |
| Typical Courses of Events | (A) : Actor,  (S) : System<br>1.   (A) A librarian requests the reservation of title<br>2.   (S) Check if a corresponding title exists<br>3.   (S) Check if a corresponding borrower exists<br>4.   (S) If the borrower does not exist, invoke "Add Borrower"<br>5.   (S) Create reservation information |
| Alternative Courses of Events | N/A |
| Exceptional Courses of Events | Line 1: If invalid reservation information is entered, indicate an error. |

# Activity 2031. Define Essential Use Cases

## 2. Remove Reservation

| Use Case | 2. Remove Reservation |
|---|---|
| Actor | Librarian |
| Purpose | (As in the business use case) |
| Overview | (As in the business use case) |
| Type | Primary and Essential |
| Cross Reference | System Functions: R1.2, R1.3<br>Use Case: "Lend Item" |
| Pre-Requisites | N/A |
| Typical Courses of Events | (A) : Actor,  (S) : System<br>1.   (A) A librarian requests removing reservation of the title<br>2.   (S) Check if a corresponding title exists<br>3.   (S) Check if a corresponding borrower exists<br>4.   (S) Find the reservation<br>5.   (S) Remove the reservation |
| Alternative Courses of Events | N/A |
| Exceptional Courses of Events | Line 1: If invalid reservation information is entered, indicate an error. |

# Activity 2031. Define Essential Use Cases

## 3. Lend Item

| Use Case | 3. Lent Item |
|---|---|
| Actor | Librarian |
| Purpose | (As in the business use case) |
| Overview | (As in the business use case) |
| Type | Primary and Essential |
| Cross Reference | System Functions: R1.3, R1.2<br>Use Cases: "Remove Reservation", "Add Borrower" |
| Pre-Requisites | Borrower should have id_card. |
| Typical Courses of Events | (A) : Actor,  (S) : System<br>1.  (A) A librarian requests lending item<br>2.  (S) Check if a corresponding title exists<br>3.  (S) Check if a corresponding item is available<br>4.  (S) If the item was reserved, invoke "Remove Reservation"<br>5.  (S) Check if corresponding borrower exists<br>6.  (S) If the borrower does not exist, invoke "Add Borrower"<br>7.  (S) Create new loan |
| Alternative Courses of Events | N/A |
| Exceptional Courses of Events | Line 1: If invalid lending information is entered, indicate an error. |

# Activity 2031. Define Essential Use Cases

## 4. Return Item

| Use Case | 4. Return Item |
|---|---|
| **Actor** | Librarian |
| **Purpose** | (As in the business use case) |
| **Overview** | (As in the business use case) |
| **Type** | Primary and Essential |
| **Cross Reference** | System Functions: R1.4.1, R1.4.2, R1.6<br>Use Cases: "Calculate Late-Return-Fee", "Notify Availability" |
| **Pre-Requisites** | N/A |
| **Typical Courses of Events** | (A) : Actor,  (S) : System<br>1.  (A) A librarian requests returning item<br>2.  (S) Check if a corresponding title exists<br>3.  (S) Check if a corresponding borrower exists<br>4.  (S) Check if a corresponding item is loaned<br>5.  (S) Find the borrower of the item<br>6.  (S) Check whether the returning due-date is over or not<br>7.  (S) If the returning due-date is over, invoke "Calculate Late-Return-Fee"<br>8.  (S) Remove the loan<br>9.  (S) If the item is reserved, invoke "Notify Availability" |
| **Alternative Courses of Events** | N/A |
| **Exceptional Courses of Events** | Line 1: If invalid returning information is entered, indicate an error |

# Activity 2031. Define Essential Use Cases

## 5. Calculate Late-Return-Fee

| Use Case | 5. Calculate Late-Return-Fee |
|---|---|
| Actor | None |
| Purpose | (As in the business use case) |
| Overview | (As in the business use case) |
| Type | Primary and Essential |
| Cross Reference | System Functions: R1.4.1, R1.4.2<br>Use Case: "Return Item" |
| Pre-Requisites | Lending due-date should be over. |
| Typical Courses of Events | (A) : Actor,  (S) : System<br>1.   (S) Compute late-return time<br>2.   (S) Compute late-return fee<br>3.   (S) Print the late-return fee |
| Alternative Courses of Events | N/A |
| Exceptional Courses of Events | N/A |

# Activity 2031. Define Essential Use Cases

## 6. Get Replacement-Fee

| Use Case | 6. Get Replacement-Fee |
|---|---|
| Actor | Librarian |
| Purpose | (As in the business use case) |
| Overview | (As in the business use case) |
| Type | Primary and Essential |
| Cross Reference | System Functions: R1.5<br>Use Case: - |
| Pre-Requisites | Title should be lost. |
| Typical Courses of Events | (A) : Actor,  (S) : System<br>1.   (A) A librarian inputs a title's information<br>2.   (S) Check if a corresponding title exists<br>3.   (S) Find the price of the title<br>4.   (S) Compute replacement-fee |
| Alternative Courses of Events | N/A |
| Exceptional Courses of Events | N/A |

# Activity 2031. Define Essential Use Cases

## 7. Notify Availability

| | |
|---|---|
| **Use Case** | 7. Notify Availability |
| **Actor** | None |
| **Purpose** | (As in the business use case) |
| **Overview** | (As in the business use case) |
| **Type** | Primary and Essential |
| **Cross Reference** | System Functions: R1.4.1, R1.6, R2.4<br>Use Cases: "Return Item", "Add Item" |
| **Pre-Requisites** | The title should be returned or new title should be added. |
| **Typical Courses of Events** | (A) : Actor, (S) : System<br>1. (S) Notify the availability of the item<br>2. (S) Print a post-card |
| **Alternative Courses of Events** | N/A |
| **Exceptional Courses of Events** | N/A |

# Activity 2031. Define Essential Use Cases

## 8. Add Title

| Use Case | 8. Add Title |
|---|---|
| Actor | Librarian |
| Purpose | (As in the business use case) |
| Overview | (As in the business use case) |
| Type | Primary and Essential |
| Cross Reference | System Functions: R2.1, R2.4<br>Use Case: "Add Item" |
| Pre-Requisites | N/A |
| Typical Courses of Events | (A) : Actor,  (S) : System<br>1.  (A) A librarian inputs a title's information<br>2.  (S) Check if a corresponding title exists<br>3.  (S) Add a new title<br>4.  (S) Invoke "Add Item" |
| Alternative Courses of Events | N/A |
| Exceptional Courses of Events | Line 1: If invalid title information is entered, indicate an error. |

# Activity 2031. Define Essential Use Cases

## 9. Remove Title

| Use Case | 9. Remove Title |
|---|---|
| Actor | Librarian |
| Purpose | (As in the business use case) |
| Overview | (As in the business use case) |
| Type | Primary and Essential |
| Cross Reference | System Functions: R2.2<br>Use Case: - |
| Pre-Requisites | N/A |
| Typical Courses of Events | (A) : Actor,  (S) : System<br>1.  (A) A librarian inputs a title's information to deleted<br>2.  (S) Check if a corresponding title exists<br>3.  (S) Remove the items of the title<br>4.  (S) Remove the title |
| Alternative Courses of Events | N/A |
| Exceptional Courses of Events | Line 1: If invalid title information is entered, indicate an error. |

# Activity 2031. Define Essential Use Cases

## 10. Update Title

| Use Case | 10. Update Title |
|---|---|
| Actor | Librarian |
| Purpose | (As in the business use case) |
| Overview | (As in the business use case) |
| Type | Primary and Essential |
| Cross Reference | System Functions: R2.3<br>Use Case: - |
| Pre-Requisites | N/A |
| Typical Courses of Events | (A) : Actor,  (S) : System<br>1.  (A) A librarian inputs a title's information to change<br>2.  (S) Check if a corresponding title exists<br>3.  (S) Update the title's information |
| Alternative Courses of Events | N/A |
| Exceptional Courses of Events | Line 1: If invalid title information is entered, indicate an error. |

# Activity 2031. Define Essential Use Cases

## 11. Add Item

| Use Case | 11. Add Item |
|---|---|
| **Actor** | Librarian |
| **Purpose** | (As in the business use case) |
| **Overview** | (As in the business use case) |
| **Type** | Primary and Essential |
| **Cross Reference** | System Functions: R1.6, R2.1, R2.4<br>Use Cases: "Notify Availability", "Add Title" |
| **Pre-Requisites** | N/A |
| **Typical Courses of Events** | (A) : Actor,  (S) : System<br>1.   (A) A librarian inputs a item to add<br>2.   (S) Check if a corresponding title exists<br>3.   (S) Add the item<br>4.   (S) Invoke "Notify Availability" |
| **Alternative Courses of Events** | N/A |
| **Exceptional Courses of Events** | Line 1: If invalid title information is entered, indicate an error. |

# Activity 2031. Define Essential Use Cases

## 12. Remove Item

| Use Case | 12. Remove Item |
|---|---|
| **Actor** | Librarian |
| **Purpose** | (As in the business use case) |
| **Overview** | (As in the business use case) |
| **Type** | Primary and Essential |
| **Cross Reference** | System Functions: R2.1, R2.5<br>Use Case: "Remove Title" |
| **Pre-Requisites** | N/A |
| **Typical Courses of Events** | (A) : Actor,  (S) : System<br>1.   (A) A librarian inputs an item's information to remove<br>2.   (S) Check if a corresponding title exists<br>3.   (S) Check if a corresponding item exists<br>4.   (S) Remove the item<br>5.   (S) If there is no remaining item, invoke "Remove Title" |
| **Alternative Courses of Events** | N/A |
| **Exceptional Courses of Events** | Line 1: If invalid title information is entered, indicate an error. |

# Activity 2031. Define Essential Use Cases

## 13. Update Item

| | |
|---|---|
| **Use Case** | 13. Update Item |
| **Actor** | Librarian |
| **Purpose** | (As in the business use case) |
| **Overview** | (As in the business use case) |
| **Type** | Primary and Essential |
| **Cross Reference** | System Functions: R2.6<br>Use Case: - |
| **Pre-Requisites** | N/A |
| **Typical Courses of Events** | (A) : Actor,  (S) : System<br>1.   (A) A librarian inputs an item's information to update<br>2.   (S) Check if a corresponding title exists<br>3.   (S) Check if a corresponding item exists<br>4.   (S) Update the item's information |
| **Alternative Courses of Events** | N/A |
| **Exceptional Courses of Events** | Line 1: If invalid title information is entered, indicate an error. |

# Activity 2031. Define Essential Use Cases

## 14. Add Borrower

| Use Case | 14. Add Borrower |
| --- | --- |
| **Actor** | Librarian |
| **Purpose** | (As in the business use case) |
| **Overview** | (As in the business use case) |
| **Type** | Primary and Essential |
| **Cross Reference** | System Functions: R1.1, R1.3, R3.1<br>Use Cases: "Make Reservation", "Lend Item" |
| **Pre-Requisites** | N/A |
| **Typical Courses of Events** | (A) : Actor,  (S) : System<br>1.  (A) A librarian inputs borrower's information such as SSN, name, address, zip code, phone number, and age.<br>2.  (S) Check if the corresponding borrower exists<br>3.  (S) Add New borrower |
| **Alternative Courses of Events** | N/A |
| **Exceptional Courses of Events** | Line 1: If invalid borrower information is entered, indicate an error. |

# Activity 2031. Define Essential Use Cases

## 15. Remove Borrower

| Use Case | 15. Remove Borrower |
|---|---|
| Actor | Librarian |
| Purpose | (As in the business use case) |
| Overview | (As in the business use case) |
| Type | Primary and Essential |
| Cross Reference | System Functions: R3.2<br>Use Case: - |
| Pre-Requisites | N/A |
| Typical Courses of Events | (A) : Actor,  (S) : System<br>1.  (A) A librarian inputs a borrower's information to remove<br>2.  (S) Check if a corresponding borrower exists<br>3.  (S) If there is a loan of the borrower, remove the loan.<br>4.  (S) Remove the borrower's information |
| Alternative Courses of Events | N/A |
| Exceptional Courses of Events | Line 1: If invalid borrower information is entered, indicate an error. |

# Activity 2031. Define Essential Use Cases

## 16. Update Borrower

| Use Case | 16. Update Borrower |
|---|---|
| Actor | Librarian |
| Purpose | (As in the business use case) |
| Overview | (As in the business use case) |
| Type | Primary and Essential |
| Cross Reference | System Functions: R3.2<br>Use Case: - |
| Pre-Requisites | N/A |
| Typical Courses of Events | (A) : Actor,  (S) : System<br>1.   (A) A librarian inputs a borrower's information to change<br>2.   (S) Check if a corresponding borrower exists<br>3.   (S) Update the borrower's information |
| Alternative Courses of Events | N/A |
| Exceptional Courses of Events | Line 1: If invalid borrower information is entered, indicate an error. |

# Activity 2031. Define Essential Use Cases

## 17. Log-In

| Use Case | 17. Log-In |
|---|---|
| **Actor** | Librarian |
| **Purpose** | (As in the business use case) |
| **Overview** | (As in the business use case) |
| **Type** | Primary and Essential |
| **Cross Reference** | System Functions: R4.1<br>Use Case: - |
| **Pre-Requisites** | A librarian should have user name and password. |
| **Typical Courses of Events** | (A) : Actor,  (S) : System<br>1.  (A) A librarian enters his(her) user name and password into the system<br>2.  (S) Check if the user name and password are correct |
| **Alternative Courses of Events** | N/A |
| **Exceptional Courses of Events** | Line 1: If invalid user name and password entered, indicate an error. |

# Activity 2031. Define Essential Use Cases

## 18. Log-Out

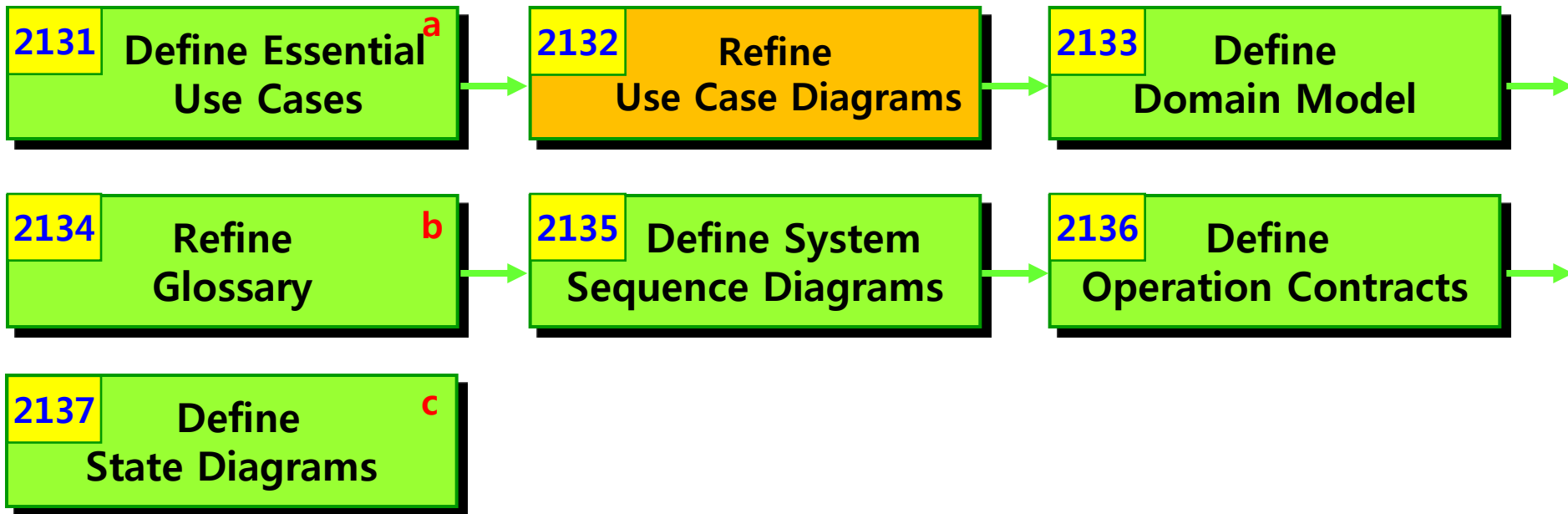| Use Case | 18. Log-Out |
|---|---|
| Actor | Librarian |
| Purpose | (As in the business use case) |
| Overview | (As in the business use case) |
| Type | Primary and Essential |
| Cross Reference | System Functions: R4.1<br>Use Case: - |
| Pre-Requisites | A librarian should have user name and password. |
| Typical Courses of Events | (A) : Actor,  (S) : System<br>1.  (A) A librarian exits the system<br>2.  (S) Log the librarian's information |
| Alternative Courses of Events | N/A |
| Exceptional Courses of Events | Line 1: If invalid user name and password entered, indicate an error. |

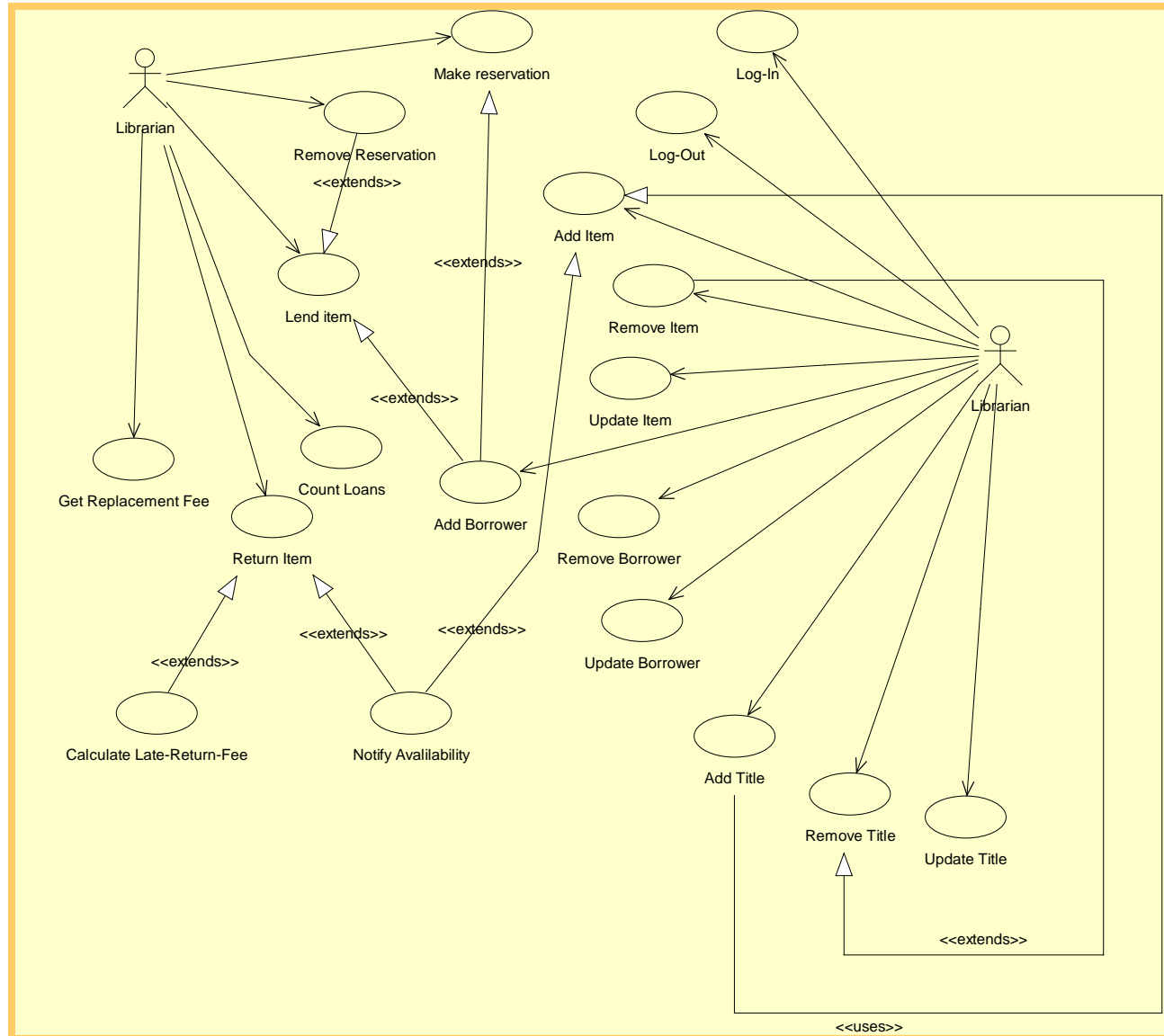# Activity 2031. Define Essential Use Cases

## 19. Count Loans

| Use Case | 19. Count Loans |
|---|---|
| Actor | Librarian |
| Purpose | (As in the business use case) |
| Overview | (As in the business use case) |
| Type | Primary and Essential |
| Cross Reference | System Functions: R5.1<br>Use Case: - |
| Pre-Requisites | A librarian should have user name and password. |
| Typical Courses of Events | (A) : Actor, (S) : System<br>1. (A) A librarian requests total counts of titles checked out<br>2. (S) Find loan information<br>3. (S) Calculate total counts of titles checked out<br>4. (S) Print total counts. |
| Alternative Courses of Events | N/A |
| Exceptional Courses of Events | Line 1: If invalid user name and password entered, indicate an error. |

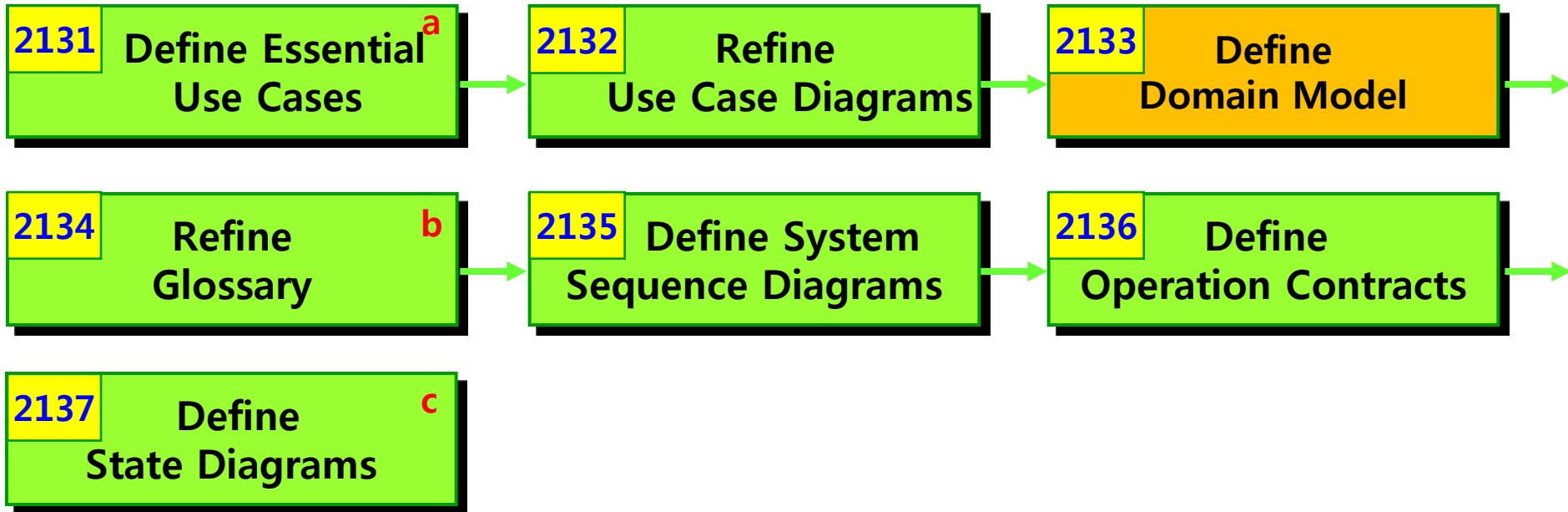# Activity 2032. Refine Use Case Diagrams

a. **if not yet done**
b. **ongoing**
c. **optional**

| 2131 **Define Essential Use Cases** <sup>a</sup> | 2132 **Refine Use Case Diagrams** | 2133 **Define Domain Model** |

**2131** Define Essential Use Cases ᵃ → **2132** Refine Use Case Diagrams → **2133** Define Domain Model →

**2134** Refine Glossary ᵇ → **2135** Define System Sequence Diagrams → **2136** Define Operation Contracts →

**2137** Define State Diagrams ᶜ

# Activity 2032. Refine Use Case Diagrams

# Activity 2033. Define Domain Model

**a. if not yet done**
**b. ongoing**
**c. optional**

| 2131 | Define Essential Use Cases <sup>a</sup> |
| 2132 | Refine Use Case Diagrams |
| 2133 | Define Domain Model |

| 2134 | Refine Glossary <sup>b</sup> |
| 2135 | Define System Sequence Diagrams |
| 2136 | Define Operation Contracts |

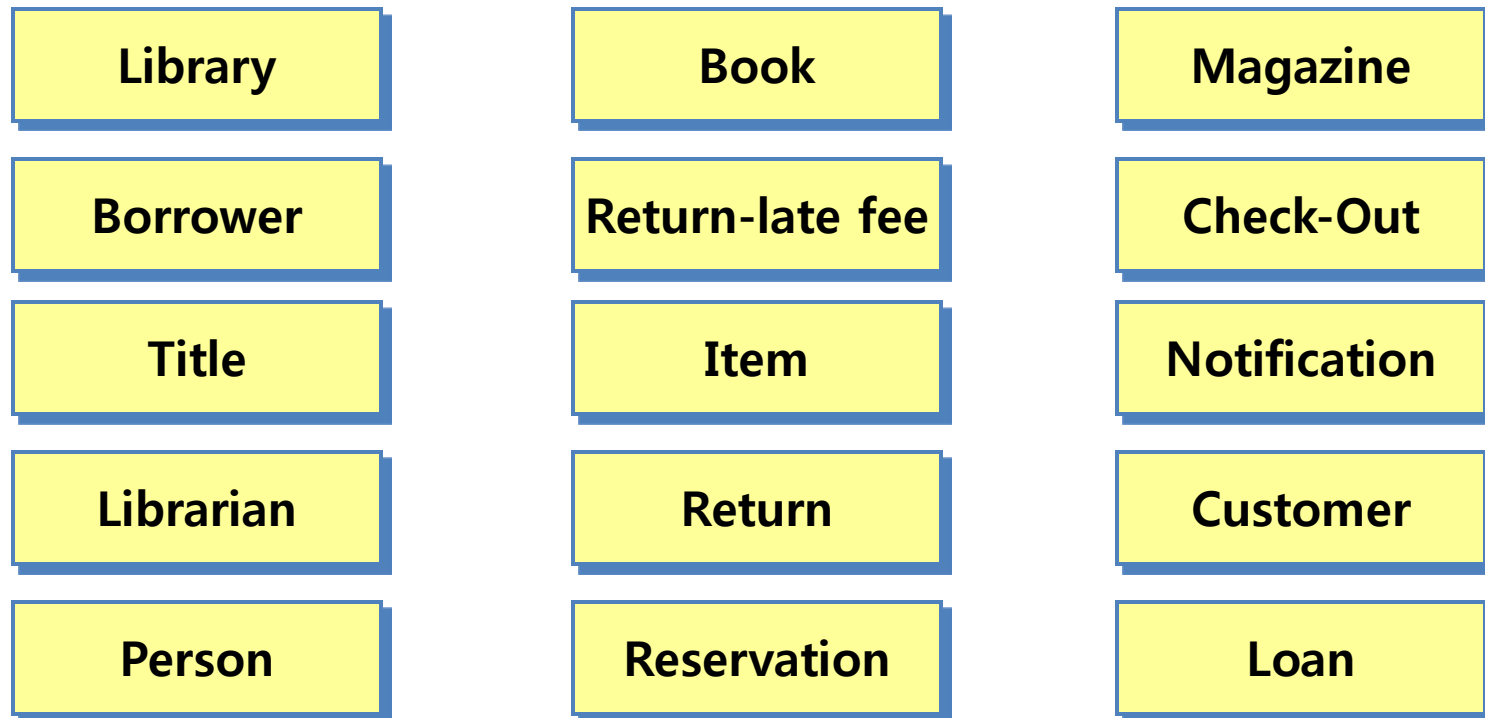| 2137 | Define State Diagrams <sup>c</sup> |

# Activity 2033. Define Domain Model

- Step 1. List concepts(domain class) from use-cases
  - Guideline 1
    - Identify concepts by making a list of candidate concepts from the 'Concept Category List'.

| Concept Category | Domain Class |
| --- | --- |
| Physical or tangible objects | Title, Book, Magazine, Item |
| Specifications, designs, or descriptions of things | Borrower, Librarian, Customer, Late-Return-Fee |
| Places | Library |
| Transactions | Reservation, Loan, Return, Registration, Notification, Check-Out |
| Roles of people | Librarian, Borrower, Customer |
| Containers of other things | Library |
| Things in a container | Item, Title, Book, Magazine, Librarian |
| organizations | Library |

# Activity 2033. Define Domain Model

– Guideline 2
- Identity the noun and noun phrases in expanded use cases description and consider them as candidate concepts or attributes.

| Library | Book | Magazine |
|---------|------|----------|
| Borrower | Return-late fee | Check-Out |
| Title | Item | Notification |
| Librarian | Return | Customer |
| Person | Reservation | Loan |

# Activity 2033. Define Domain Model

- Step 2. Assign class name into a concept
- Step 3. Draw a conceptual class diagram

| Title | Librarian | Book |
|-------|-----------|------|

| Magazine | Loan | Reservation |
|----------|------|-------------|

| Borrower | Item |
|----------|------|

# Activity 2033. Define Domain Model

- Step 4. Identify and add associations
  - Use 'Association Category List'

| Association Category | Associations |
|---|---|
| A is known/logged/recorded/reported/captured in B | Item – Loan<br>Item – Title<br>Loan – Borrower<br>Title – Reservation |
| A is a line item of B | Item – Title |
| A is recorded in B | Item – Title |
| A is related to a transaction of B | Borrower – Loan<br>Borrower – Reservation |
| A is an organization submit of B | Book – Title<br>Magazine – Title |

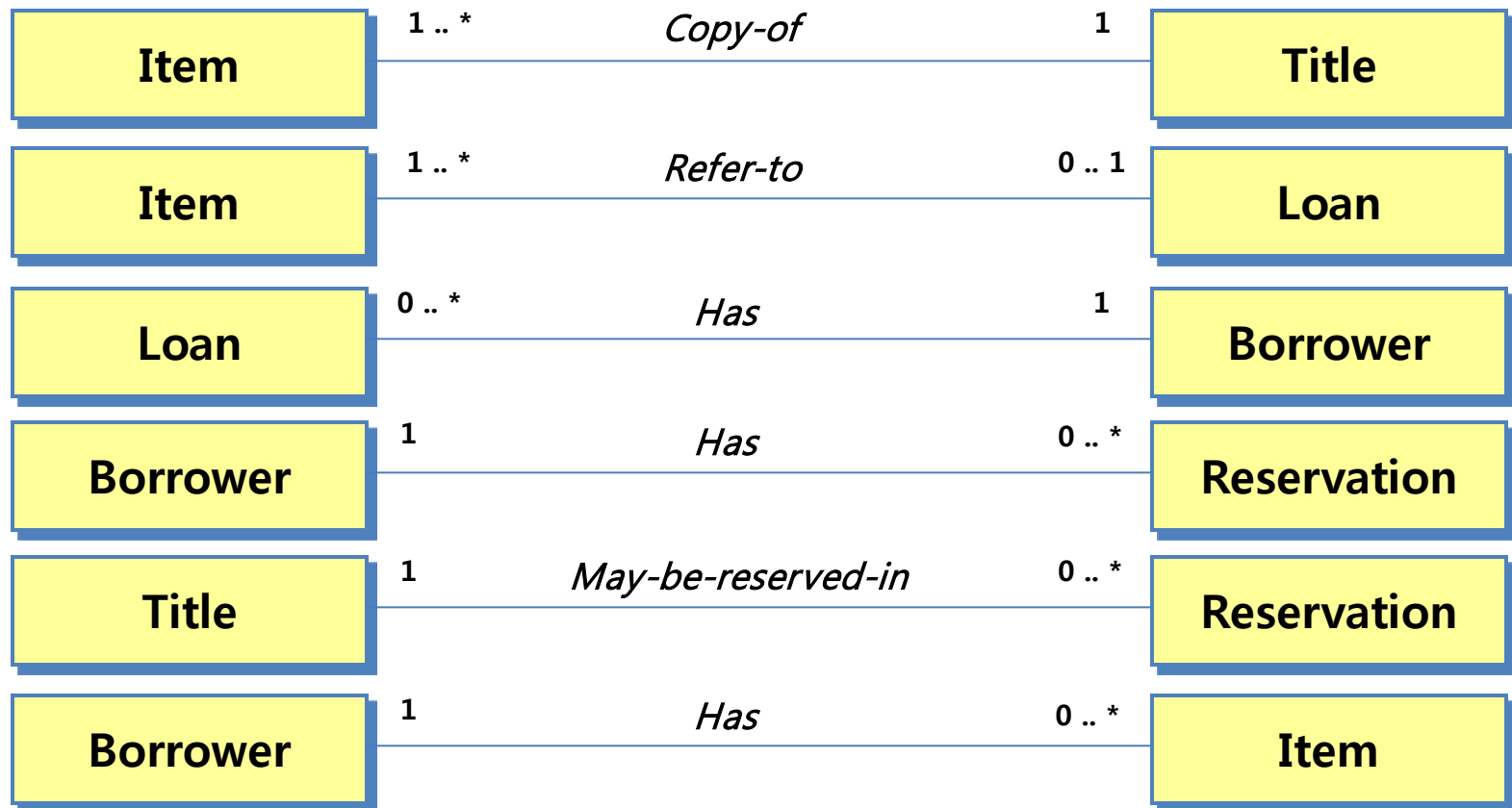- Step 5. Assign priority into associations

| Association Name | Priority |
|---|---|
| Item – Title | High |

# Activity 2033. Define Domain Model

- Step 6. Assign names into associations
  - Item *Copy-of* Title
  - Item *Refer-to* Loan
  - Loan *Has* Borrower
  - Borrower *Has* Reservation
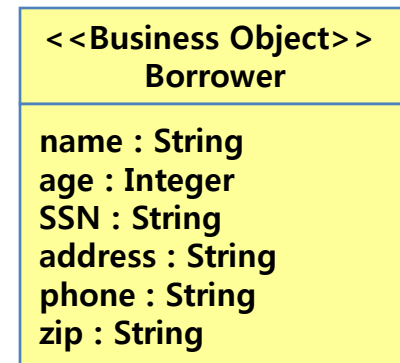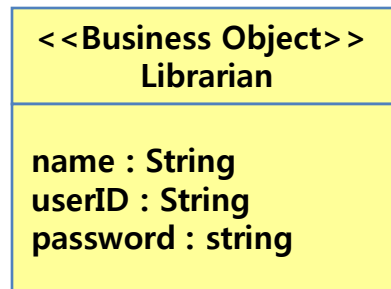  - Title *May-be-reserved-in* Reservation
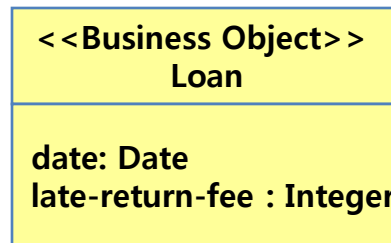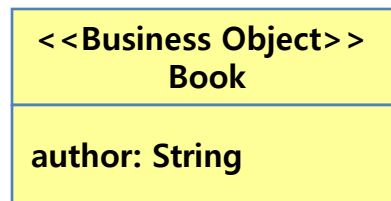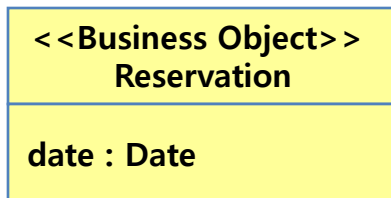  - Borrower *Has* Item

# Activity 2033. Define Domain Model

- Step 7. Add Roles and Multiplicity

| Item | 1 .. * | Copy-of | 1 | Title |
|------|--------|---------|---|-------|

| Item | 1 .. * | Refer-to | 0 .. 1 | Loan |
|------|--------|----------|--------|------|

| Loan | 0 .. * | Has | 1 | Borrower |
|------|--------|-----|---|----------|

| Borrower | 1 | Has | 0 .. * | Reservation |
|----------|---|-----|--------|-------------|

| Title | 1 | May-be-reserved-in | 0 .. * | Reservation |
|-------|---|--------------------|--------|-------------|

| Borrower | 1 | Has | 0 .. * | Item |
|----------|---|-----|--------|------|

# Activity 2033. Define Domain Model

- Step 8. Add Attributes

<<Business Object>>
**Item**

ID : Integer
available : Boolean

<<Business Object>>
**Title**

name : String
isbn : String
count : Integer
price : Float
publisher : String
lending time : Integer

<<Business Object>>
**Reservation**

date : Date

<<Business Object>>
**Book**

author: String

<<Business Object>>
**Loan**

date: Date
late-return-fee : Integer

<<Business Object>>
**Librarian**

name : String
userID : String
password : string

<<Business Object>>
**Magazine**

month : Integer

<<Business Object>>
**Borrower**

name : String
age : Integer
SSN : String
address : String
phone : String
zip : String
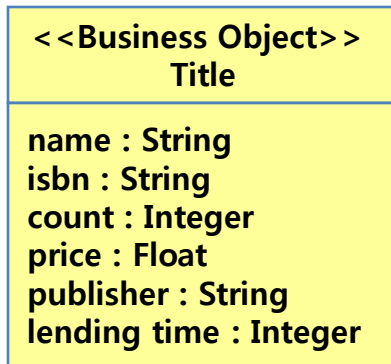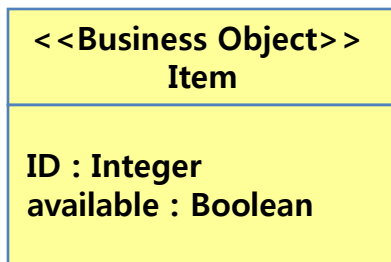
# Activity 2033. Define Domain Model
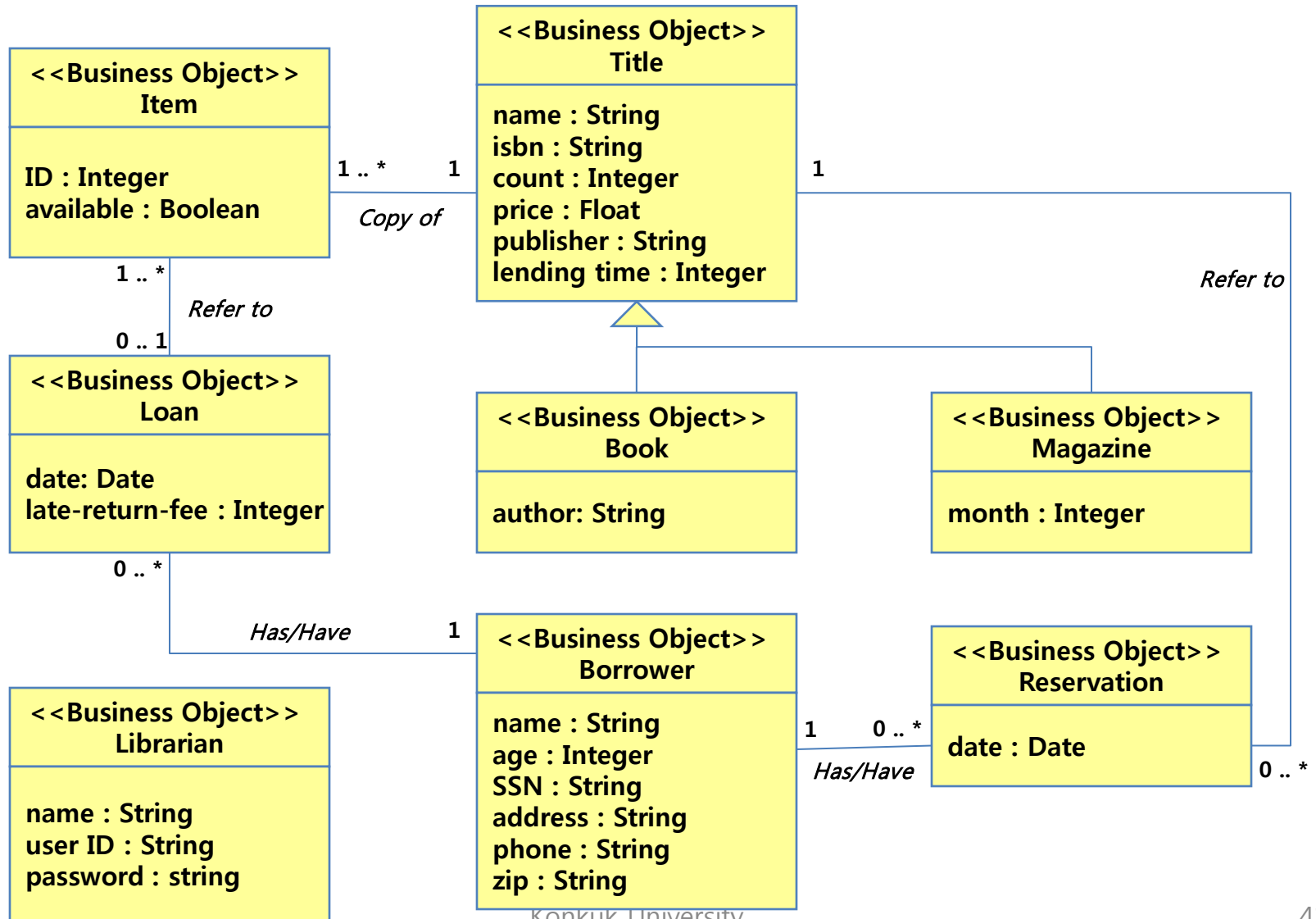
# Activity 2034. Refine Glossary

a. **if not yet done**
b. **ongoing**
c. **optional**

| 2131 | Define Essential Use Cases [a] |
| 2132 | Refine Use Case Diagrams |
| 2133 | Define Domain Model |

| 2134 | Refine Glossary [b] |
| 2135 | Define System Sequence Diagrams |
| 2136 | Define Operation Contracts |

| 2137 | Define State Diagrams [c] |
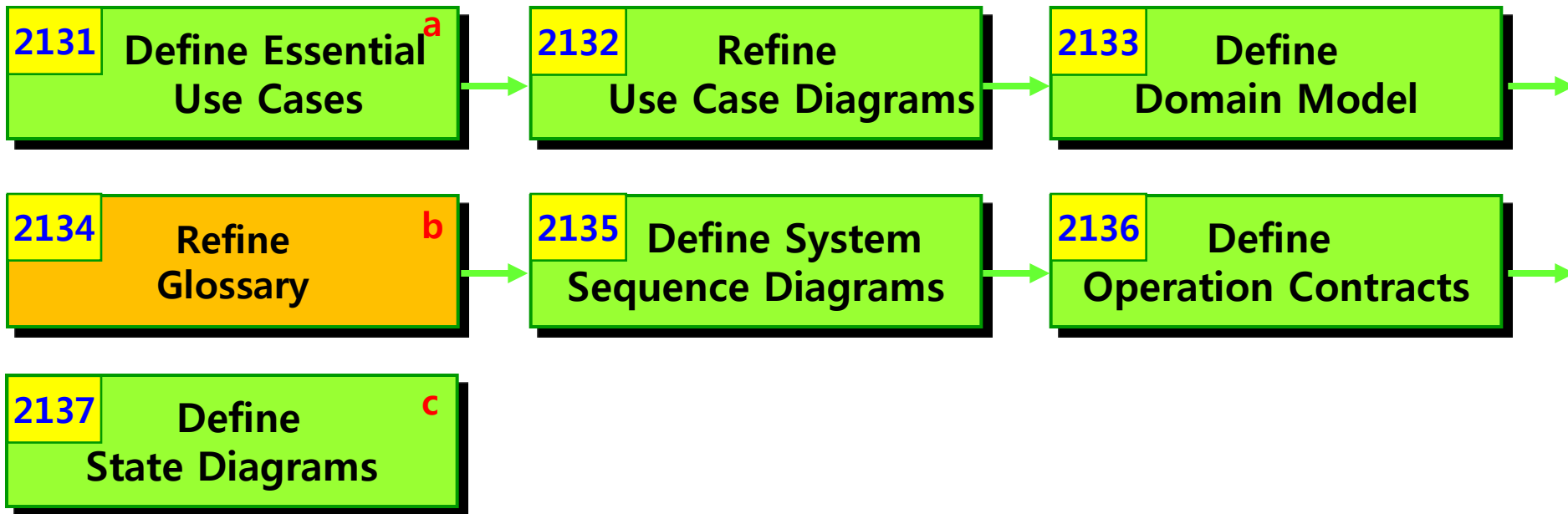
# Activity 2034. Refine Glossary

- Refine terms

| Term | Category | Remarks |
|------|----------|---------|
| Title | Class | A type of books or magazines which are registered in the library system. |
| Item | Class | Each copy of book or magazine. |
| Reservation | Class | An action of lending title that is available for use when it is needed. |
| Borrower | Class | A person that lends, returns item. |
| Loan | Class | An action of lending a book/magazine from the library. |
| Librarian | Class | An employee of the library who interacts with the borrower. |
| Librarian.name | Attribute | The name of librarian. |
| Librarian.userId | Attribute | The user name of librarian. |
| Librarian.password | Attribute | The password of librarian. |
| Title.name | Attribute | The title of a book or a magazine. |
| Title.publisher | Attribute | The publishing company of the title. |
| Title.isbn | Attribute | The International Standard Book Number of title. |
| Title.price | Attribute | The price of title. |
| Title.count | Attribute | The number of item contained in a title. |
| Title.lendingtime | Attribute | The lending time of a title. |
| Book.author | Attribute | The author name of a book. |
| Magazine.month | Attribute | The publication cycle of a magazine. |
| Reservation.date : Date | Attribute | The date of reservation. |
| Item.id : Integer | Attribute | Item number. |
| Loan.date : Date | Attribute | Lending date of an item. |
| Loan.late-return-fee | Attribute | Over lending time of an item. |
| Borrower.SSN | Attribute | The resident registration number |
| Borrower.name | Attribute | A borrower name. |
| Borrower.address | Attribute | A borrower address. |
| Borrower.zip | Attribute | A zip code of borrower. |
| Borrower.age | Attribute | A borrower age. |

# Activity 2035.
# Define System Sequence Diagrams

a. if not yet done
b. ongoing
c. optional

| 2131 | **Define Essential Use Cases** [a] |
|---|---|

| 2132 | **Refine Use Case Diagrams** |
|---|---|

| 2133 | **Define Domain Model** |
|---|---|

| 2134 | **Refine Glossary** [b] |
|---|---|

| 2135 | **Define System Sequence Diagrams** |
|---|---|

| 2136 | **Define Operation Contracts** |
|---|---|

| 2137 | **Define State Diagrams** [c] |
|---|---|
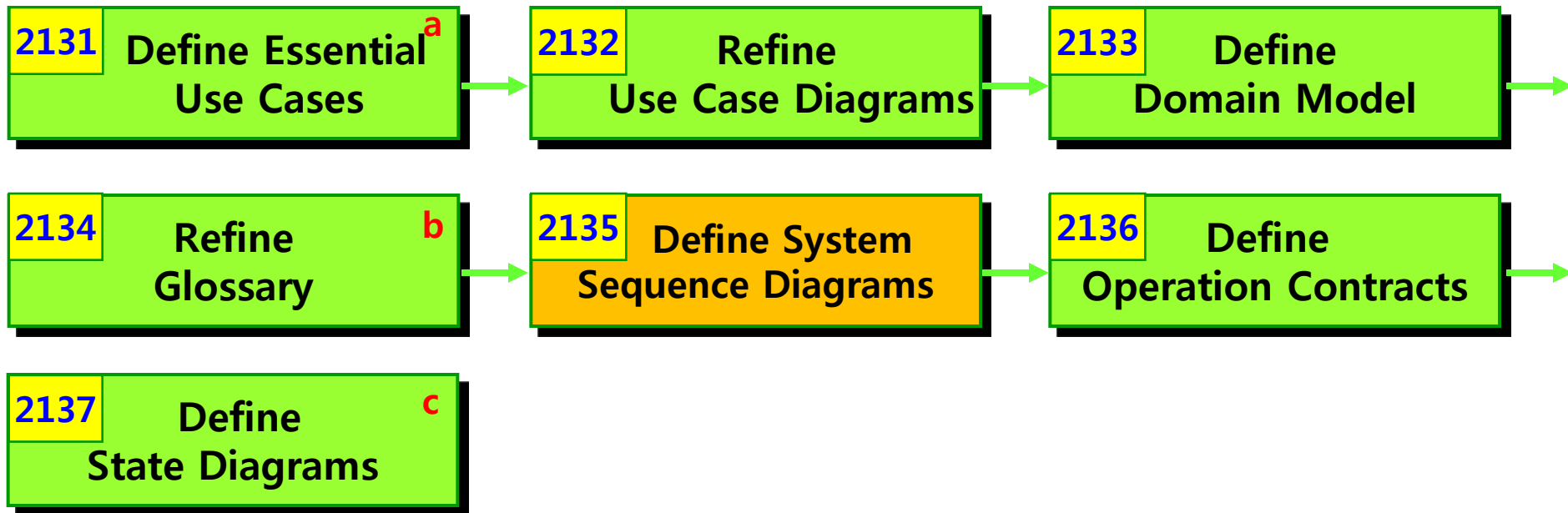
# Activity 2035.
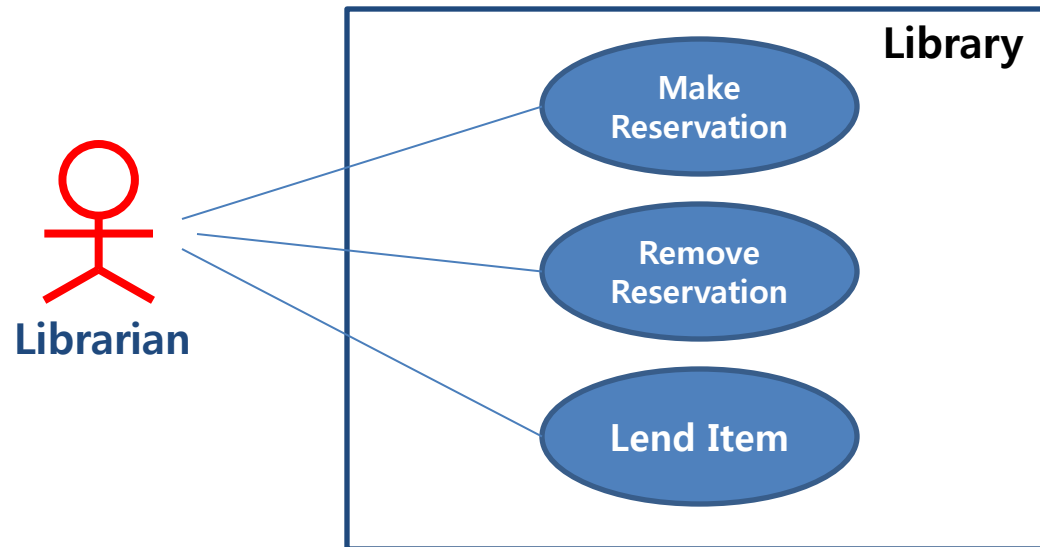# Define System Sequence Diagrams

- Step 1.
  - Draw a black box representing the system based on a use case
- Step 2.
  - Identify each actor that directly operate on the system from the typical(normal) course of events in a use case
    - Draw a line for each actor

**Librarian**
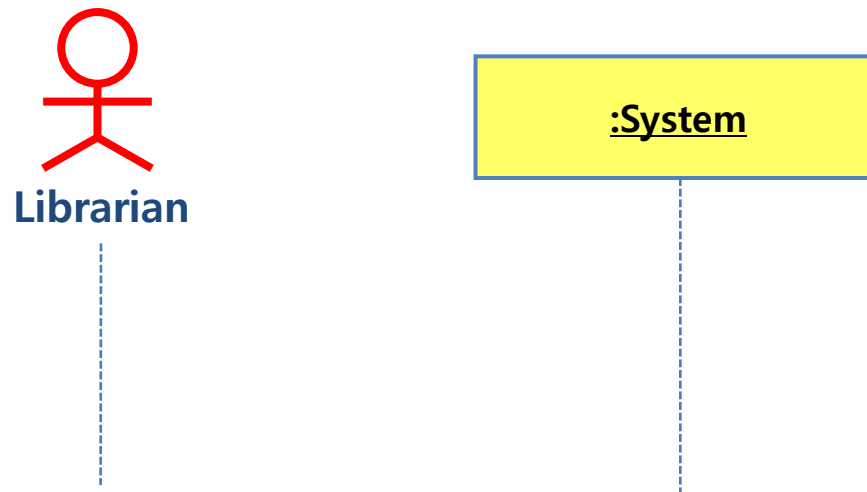
**:System**
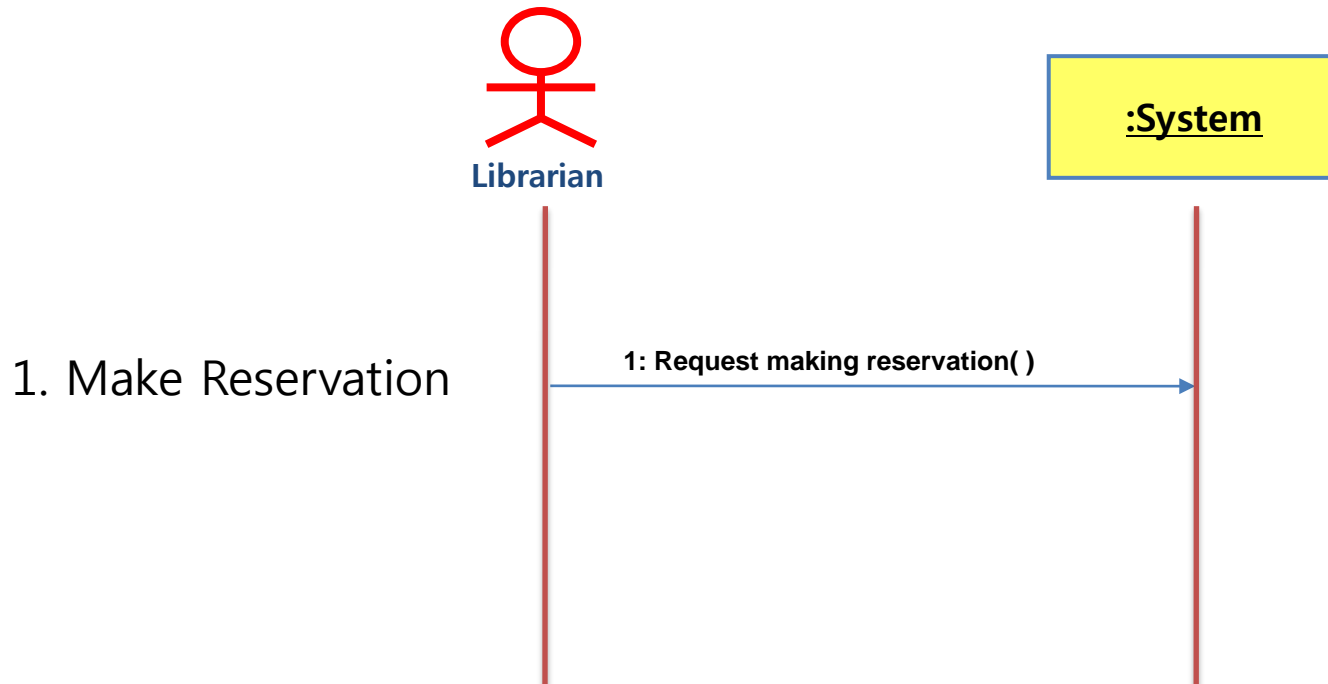
# Activity 2035.
# Define System Sequence Diagrams

- Step 3. Determine system boundary
  - Identify and name the system events that each actor generates.

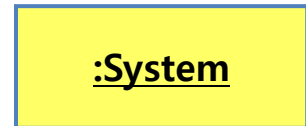# Activity 2035.
# Define System Sequence Diagrams

- System Sequence Diagrams
  - System sequence diagrams for all 19 uses cases (excepted for No. 5 & 7 use cases) are the same, because they have only one actor-activated event.

**Librarian**

**:System**

1. Make Reservation

**1: Request making reservation( )**

# Activity 2035.
# Define System Sequence Diagrams

| Use Case | Name of Actor-Activated Event |
|---|---|
| 1. Make Reservation | 1: Request making reservation( ) |
| 2. Remove Reservation | 1: Request removing reservation( ) |
| 3. Lend Item | 1: Request lending item( ) |
| 4. Return Item | 1: Request returning item( ) |
| 5. Calculate Late-Return-Fee | N/A |
| 6. Get Replacement Fee | 1: Request replacement fee( ) |
| 7. Notify Availability | N/A |
| 8. Add Title | 1: Request adding title( ) |
| 9. Remove Title | 1: Request removing title( ) |
| 10. Update Title | 1: Request updating title( ) |
| 11. Add Item | 1: Request adding item( ) |
| 12. Remove Item | 1: Request removing item( ) |
| 13. Update Item | 1: Request updating item( ) |
| 14. Add Borrower | 1: Request adding borrower( ) |
| 15. Remove Borrower | 1: Request removing borrower( ) |
| 16. Update Borrower | 1: Request updating borrower( ) |
| 17. Log-In | 1: Input ID_Password( ) |
| 18. Log-Out | 1: Exit( ) |
| 19. Count Loans | 1: Request count loans( ) |

**Librarian**

**:System**

**1: Request making reservation( )**

# Activity 2035.
# Define System Sequence Diagrams

- Step 4. Include use case text for all system sequence diagrams

**USE CASE: 1. Make Reservation**

1. A librarian requests the reservation of title.

2. Check if corresponding title exist.

3. Check if corresponding borrower exist.

4. If the borrower does not exist, invoke "Add Borrower".

5. Create reservation information.

**Librarian**

**:System**
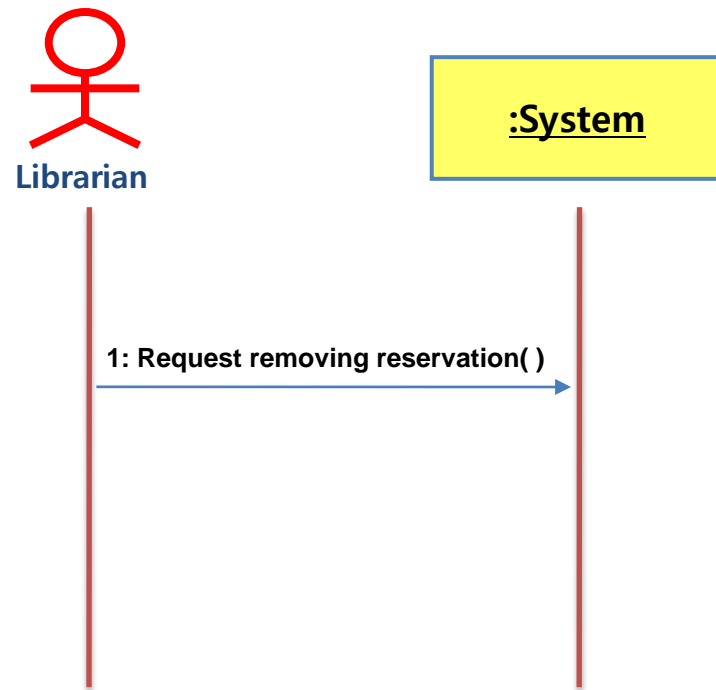
1: Request making reservation( )
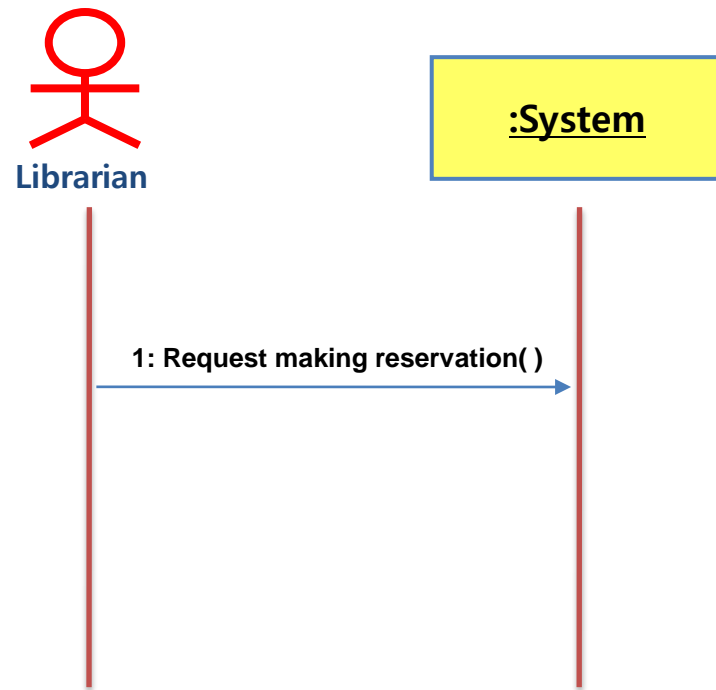
# Activity 2035.
# Define System Sequence Diagrams

**USE CASE: 2. Remove Reservation**

1. **A librarian requests removing reservation.**

2. **Check if corresponding title exist.**

3. **Check if corresponding borrower exist.**

4. **Find the reservation.**

5. **Remove the reservation.**

**Librarian**

**:System**

**1: Request removing reservation( )**

# Activity 2035.
# Define System Sequence Diagrams

**USE CASE: 3. Lend Item**

1. **A librarian requests lending item.**

2. **Check if corresponding title exist.**

3. **Check if corresponding item is available.**

4. **If the item was reserved, invoke "Remove Reservation".**

5. **Check if corresponding borrower exist.**

6. **If the borrower does not exist, invoke "Add Borrower".**

7. **Create new loan.**



**Librarian**

**:System**

**1: Request lending item( )**
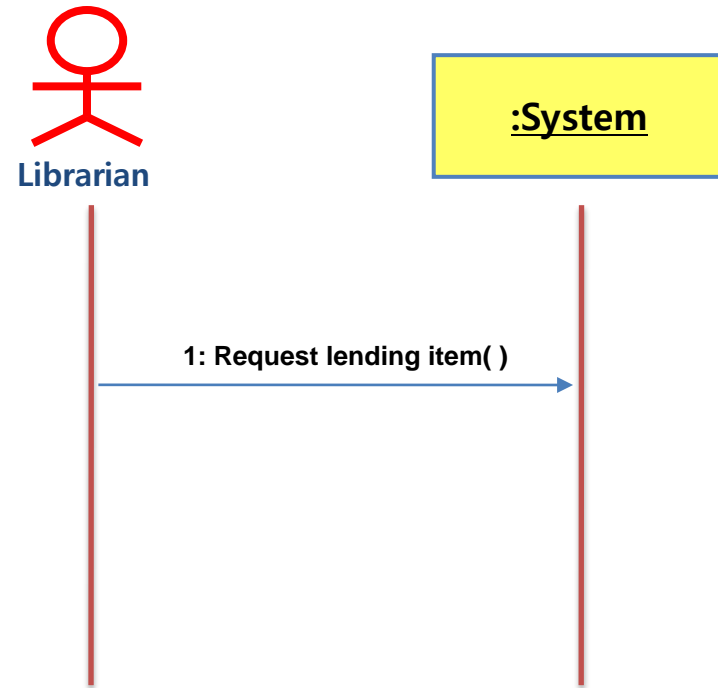
# Activity 2036. Define Operation Contracts

a. if not yet done
b. ongoing
c. optional

**2131** Define Essential Use Cases [a]

**2132** Refine Use Case Diagrams

**2133** Define Domain Model

**2134** Refine Glossary [b]

**2135** Define System Sequence Diagrams

**2136** Define Operation Contracts

**2137** Define State Diagrams [c]
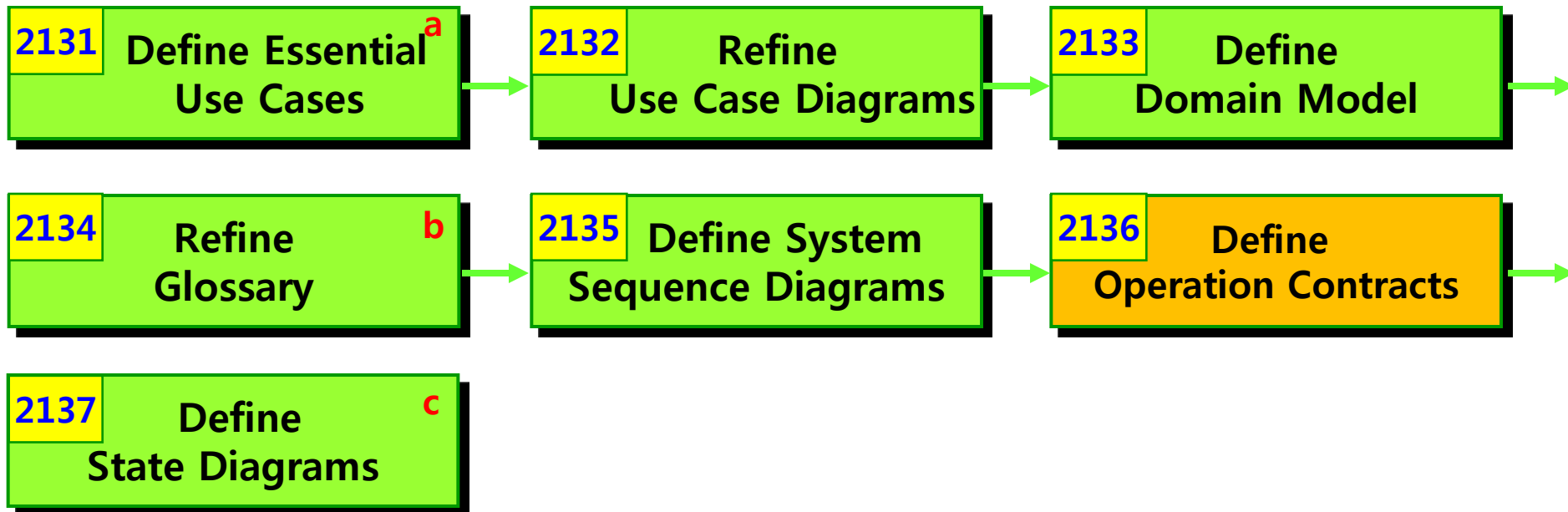
# Activity 2036. Define Operation Contracts

- Step 1. Identify system operations from system sequence diagrams

| Use Case | Name of Actor-Activated Event | System Operations |
|---|---|---|
| 1. Make Reservation | 1: Request making reservation( ) | 1. makeReservation( ) |
| 2. Remove Reservation | 1: Request removing reservation( ) | 2. removeReservation( ) |
| 3. Lend Item | 1: Request lending item( ) | 3. LendItem( ) |
| 4. Return Item | 1: Request returning item( ) | 4. returnItem( ) |
| 5. Calculate Late-Return-Fee | N/A | N/A |
| 6. Get Replacement Fee | 1: Request replacement fee( ) | 5. getReplacementFee( ) |
| 7. Notify Availability | N/A | N/A |
| 8. Add Title | 1: Request adding title( ) | 6. addTitle( ) |
| 9. Remove Title | 1: Request removing title( ) | 7. removeTitle( ) |
| 10. Update Title | 1: Request updating title( ) | 8. updateTitle( ) |
| 11. Add Item | 1: Request adding item( ) | 9. addItem( ) |
| 12. Remove Item | 1: Request removing item( ) | 10. removeItem( ) |
| 13. Update Item | 1: Request updating item( ) | 11. updateItem( ) |
| 14. Add Borrower | 1: Request adding borrower( ) | 12. addBorrower( ) |
| 15. Remove Borrower | 1: Request removing borrower( ) | 13. removeBorrower( ) |
| 16. Update Borrower | 1: Request updating borrower( ) | 14. updateBorrower( ) |
| 17. Log-In | 1: Input ID_Password( ) | 15. log-In( ) |
| 18. Log-Out | 1: Exit( ) | 16. log-Out( ) |
| 19. Count Loans | 1: Request count loans( ) | 17. countLoans( ) |

# Activity 2036. Define Operation Contracts

- Step 2. Fill contracts according to the format below

| Name | Name of operation, and parameters |
|---|---|
| **Responsibilities** | An informal description of the responsibilities that the operation must fill |
| **Type** | Name of type(concept, software class, interface) |
| **Cross References** | System function reference numbers, use cases, etc. |
| **Notes** | Design notes, algorithms, and so on. |
| **Exceptions** | Exceptional cases |
| **Output** | Non-UI outputs, such as messages or records that are sent outside of the system |
| **Pre-conditions** | Assumptions that the state of the system before execution of the operation |
| **Post-conditions** | The state of the system after completion of the operation |
| **...** | |

# Activity 2036. Define Operation Contracts

| Name | makeReservation( ) |
|---|---|
| Responsibilities | Checks if title and borrower information exist, and creates a new reservation |
| Type | System |
| Cross References | System functions: R1.1, R2.1 |
| Notes | |
| Exceptions | N/A |
| Output | Results from making the reservation |
| Pre-conditions | Title and Borrower information should be entered. |
| Post-conditions | A new reservation has created.<br>Reservation.title has set to the title.<br>Reservation.borrower has set to the borrower.<br>The Reservation is associated with the Title.<br>The Reservation is associated with the Borrower. |

# Activity 2036. Define Operation Contracts

| Name | **removeReservation( )** |
|---|---|
| **Responsibilities** | Receive reservation information from a librarian and removes the reservation information |
| **Type** | System |
| **Cross References** | System functions: R1.2<br>Use case: "Remove Reservation" |
| **Notes** | |
| **Exceptions** | N/A |
| **Output** | Results from removing the reservation |
| **Pre-conditions** | The title should be reserved. |
| **Post-conditions** | The Reservation has deleted.<br>The Reservation is associated with Title. (Why?)<br>The Reservation is associated with Borrower. (Why?) |

# Activity 2036. Define Operation Contracts

| Name | **lendItem( )** |
|---|---|
| **Responsibilities** | Checks whether the item to lend exists or not and lends the item |
| **Type** | System |
| **Cross References** | System functions: R1.2, R1.3<br>Use case: "Lent Item", "Make Reservation", "Remove Reservation" |
| **Notes** | |
| **Exceptions** | N/A |
| **Output** | Results from lending the item |
| **Pre-conditions** | The title of the item should exist. |
| **Post-conditions** | A new loan has created.<br>The Loan is associated with the Item.<br>The Loan is associated with the Borrower. |

# Activity 2036. Define Operation Contracts

| Name | **returnItem( )** |
|---|---|
| **Responsibilities** | Receives an item's information and returns the item |
| **Type** | System |
| **Cross References** | System functions: R1.4.1, R1.4.2<br>Use case: "Return Item", "Calculate Late-Return-Fee" |
| **Notes** | |
| **Exceptions** | N/A |
| **Output** | Results from returning the item |
| **Pre-conditions** | Information of the item to return should be entered. |
| **Post-conditions** | Item.loan was set to the  loan.<br>The Item is associated with the Loan.<br>The Loan has deleted.<br>The Loan is associated with the Borrower. |

# Activity 2036. Define Operation Contracts

| Name | getReplacementFee( ) |
|---|---|
| Responsibilities | Requests to calculate for lost items or items in a poor condition |
| Type | System |
| Cross References | System functions: R1.5<br>Use case: "Get Replacement Fee" |
| Notes | |
| Exceptions | N/A |
| Output | Data on the calculated replacement fee |
| Pre-conditions | ISBM of the lost item should be entered. |
| Post-conditions | Item.lost has set to a true value.<br>A count of the title has decremented.<br>An available count of the title has decremented<br>A Loan has deleted. (Why?) |

# Activity 2036. Define Operation Contracts

| Name | **addTitle( )** |
|---|---|
| **Responsibilities** | Adds a new title |
| **Type** | System |
| **Cross References** | System functions: R2.1, R2.4<br>Use case: "Add Title", "Add Item" |
| **Notes** | |
| **Exceptions** | If the title already exists, indicate an error. |
| **Output** | Results from returning the item |
| **Pre-conditions** | ISBM of the lost item should be entered. |
| **Post-conditions** | A new Title has created.<br>Title.name has set to the name.<br>Title.isbn has set to the isbn.<br>Title.price has set to the price.<br>Title.numOfCount has set to the numOfCount.<br>Title.availableCount has set to the availableCount.<br>Title.publisher has set to the publisher.<br>Title.loanPeriod has set to the loanPeriod.<br>Title.reservationCount has set to the reservationCount.<br>Title is associated with Item. |

0

# Activity 2036. Define Operation Contracts

| Name | **removeTitle( )** |
|---|---|
| **Responsibilities** | Removes an old book or magazine |
| **Type** | System |
| **Cross References** | System functions: R2.2<br>Use case: "Remove Title" |
| **Notes** | |
| **Exceptions** | If the title does not exist, indicate an error. |
| **Output** | Results from removing the title |
| **Pre-conditions** | Information of the title should be entered. |
| **Post-conditions** | The Title has deleted.<br>The Title is associated with an Item. (Why?)<br>The Title is associated with an Reservation. (Why?)<br>The Item is associated with an Loan. (Why?) |

# Activity 2036. Define Operation Contracts

| Name | **updateTitle( )** |
|---|---|
| **Responsibilities** | Updates an old book or magazine |
| **Type** | System |
| **Cross References** | System functions: R2.3<br>Use case: "Update Title" |
| **Notes** | |
| **Exceptions** | If the title does not exist, indicate an error. |
| **Output** | Results from updating the title |
| **Pre-conditions** | Information of the title should be entered. |
| **Post-conditions** | The Title has updated.<br>The Title is associated with an Item.<br>The Title is associated with an Reservation.<br>The Item is associated with an Loan. |

# Activity 2036. Define Operation Contracts

| Name | **removeItem( )** |
|---|---|
| **Responsibilities** | Removes an item |
| **Type** | System |
| **Cross References** | System functions: R2.5<br>Use case: "Remove Item" |
| **Notes** | |
| **Exceptions** | If the item's title does not exist, indicate an error. |
| **Output** | Information of the removed item |
| **Pre-conditions** | Information of the title and item should be entered. |
| **Post-conditions** | The Item has removed.<br>The Item is associated with Title. (Why?)<br>The Item is associated with Loan. (Why?) |

# Activity 2036. Define Operation Contracts

| Name | **updateItem( )** |
|---|---|
| **Responsibilities** | Updates an item |
| **Type** | System |
| **Cross References** | System functions: R2.6<br>Use case: "Update Item" |
| **Notes** | |
| **Exceptions** | If the item's title does not exist, indicate an error. |
| **Output** | Information of the updated item |
| **Pre-conditions** | Information of the title and item should be entered. |
| **Post-conditions** | The Item has updated.<br>The Item is associated with Title.<br>The Item is associated with Loan. |

# Activity 2036. Define Operation Contracts

| Name | **addBorrower( )** |
|---|---|
| **Responsibilities** | Adds a new borrower's information |
| **Type** | System |
| **Cross References** | System functions: R3.1<br>Use case: "Add Borrower" |
| **Notes** | |
| **Exceptions** | If the borrower exists, indicate an error. |
| **Output** | Results from adding the new borrower |
| **Pre-conditions** | Information of the borrower should be entered. |
| **Post-conditions** | A new Borrower has created.<br>Borrower.SSN has set to the SSN.<br>Borrower.name has set to the name.<br>Borrower.address has set to the address.<br>Borrower.reservationCount has set to reservationCount.<br>Borrower.loanCount has set to loanCount.<br>Borrower is associated with Loan.<br>Borrower is associated with Reservation. |

# Activity 2036. Define Operation Contracts

| Name | **removeBorrower( )** |
|---|---|
| **Responsibilities** | Removes a borrower's information |
| **Type** | System |
| **Cross References** | System functions: R3.2<br>Use case: "Remove Borrower" |
| **Notes** | |
| **Exceptions** | If the borrower does not exist, indicate an error. |
| **Output** | Results from removing the borrower |
| **Pre-conditions** | Information of the borrower should be entered. |
| **Post-conditions** | A Borrower has deleted.<br>Borrower is associated with Loan. (Why?)<br>Borrower is associated with Reservation. (Why?) |

# Activity 2036. Define Operation Contracts

| Name | **updateBorrower( )** |
|---|---|
| **Responsibilities** | Updates a borrower's information |
| **Type** | System |
| **Cross References** | System functions: R3.3<br>Use case: "Update Borrower" |
| **Notes** | |
| **Exceptions** | If the borrower does not exist, indicate an error. |
| **Output** | Results from updating the borrower |
| **Pre-conditions** | Information of the borrower should be entered. |
| **Post-conditions** | A Borrower has updated.<br>Borrower is associated with Loan.<br>Borrower is associated with Reservation. |

# Activity 2036. Define Operation Contracts

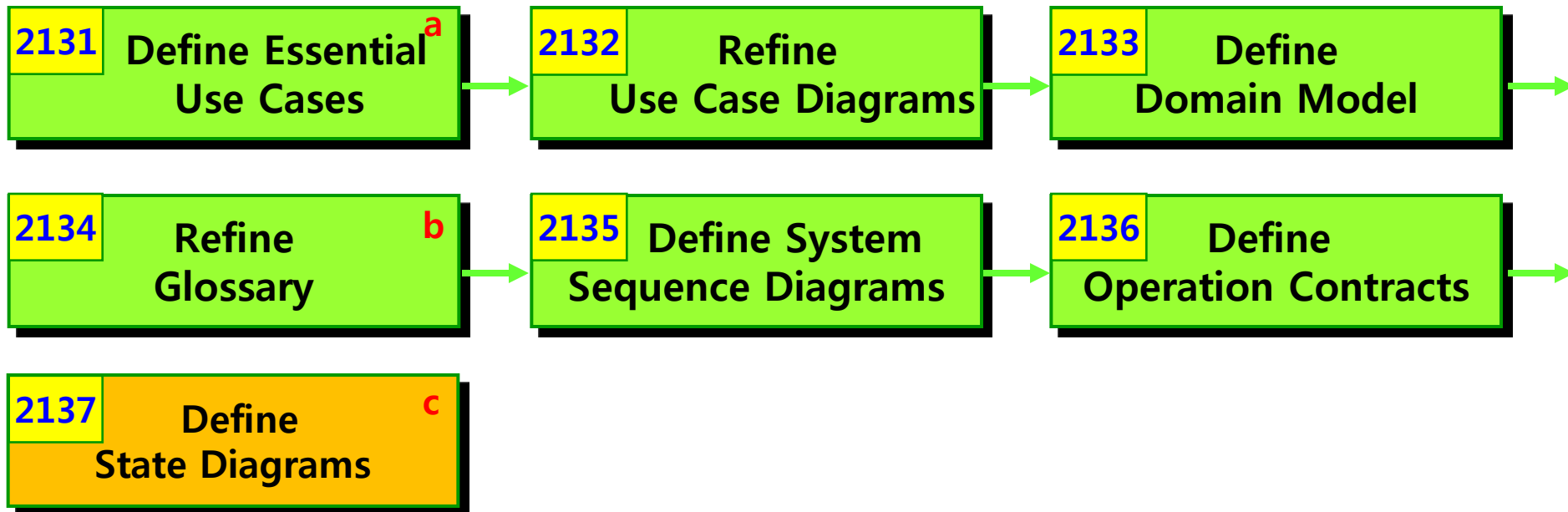| Name | **Log-In( )** |
|---|---|
| **Responsibilities** | Inputs an ID and Password of a librarian |
| **Type** | System |
| **Cross References** | System functions: R4.1<br>Use case: "Log-In" |
| **Notes** | |
| **Exceptions** | If the librarian does not exist, indicate an error. |
| **Output** | Approval information |
| **Pre-conditions** | Authentication information should be entered. |
| **Post-conditions** | The authentication information is associated with the librarian. (What information is it?) |

# Activity 2036. Define Operation Contracts

| Name | Log-Out( ) |
|---|---|
| Responsibilities | Logouts from the system |
| Type | System |
| Cross References | System functions: R4.1<br>Use case: "Log-Out" |
| Notes | |
| Exceptions | N/A |
| Output | Exits from the system |
| Pre-conditions | - |
| Post-conditions | - |

# Activity 2036. Define Operation Contracts

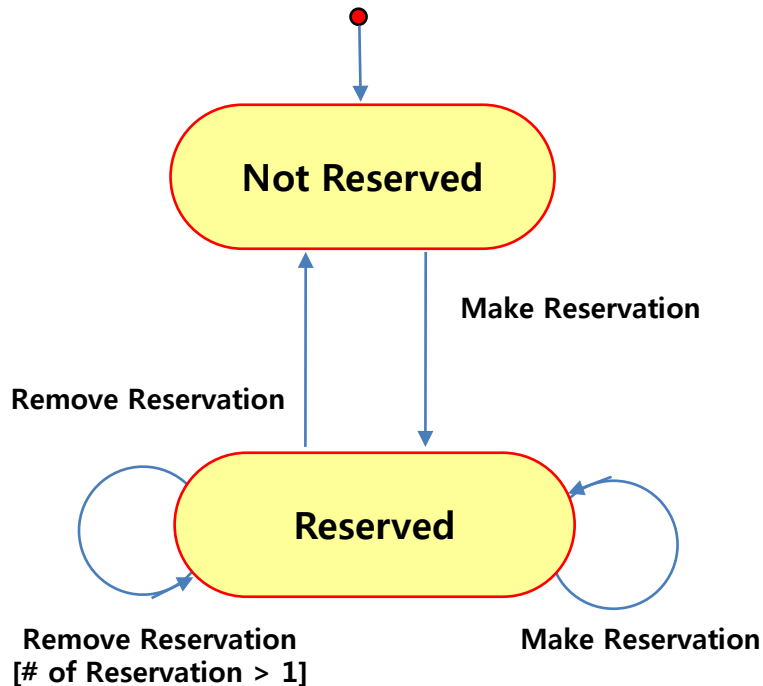| Name | **countLoans( )** |
|---|---|
| **Responsibilities** | Requests for calculating a total counts of all titles checked |
| **Type** | System |
| **Cross References** | System functions: R5.1<br>Use case: "Count Loans" |
| **Notes** | |
| **Exceptions** | N/A |
| **Output** | Calculated data on the loans |
| **Pre-conditions** | It should calculate only the number of titles checked out. |
| **Post-conditions** | Number of titles checked out has calculated. |

# Activity 2037. Define State Diagrams

a. if not yet done
b. ongoing
c. optional

| 2131 Define Essential Use Cases [a] | 2132 Refine Use Case Diagrams | 2133 Define Domain Model |
|---|---|---|
| 2134 Refine Glossary [b] | 2135 Define System Sequence Diagrams | 2136 Define Operation Contracts |
| 2137 Define State Diagrams [c] | | |

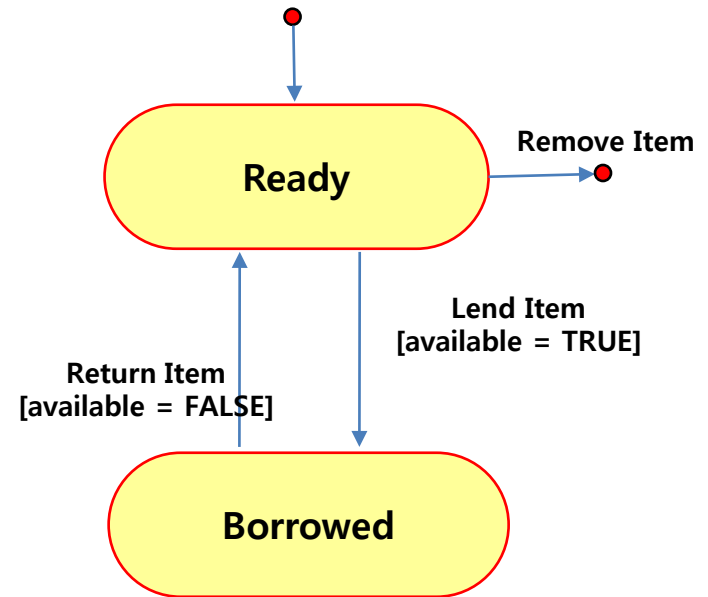# Activity 2036. Define State Diagrams

**< State Diagram for "Title" >**

**< State Diagram for "Item" >**

# Phase 2030. Analyze

- Phase 2030 Activities

a. **if not yet done**
b. **ongoing**
c. **optional**

**2130** **Analyze**

**2131** **Define Essential Use Cases** a

**2132** **Refine Use Case Diagrams**

**2133** **Define Domain Model**

**2134** **Refine Glossary** b

**2135** **Define System Sequence Diagrams**

**2136** **Define Operation Contracts**

**2137** **Define State Diagrams** c