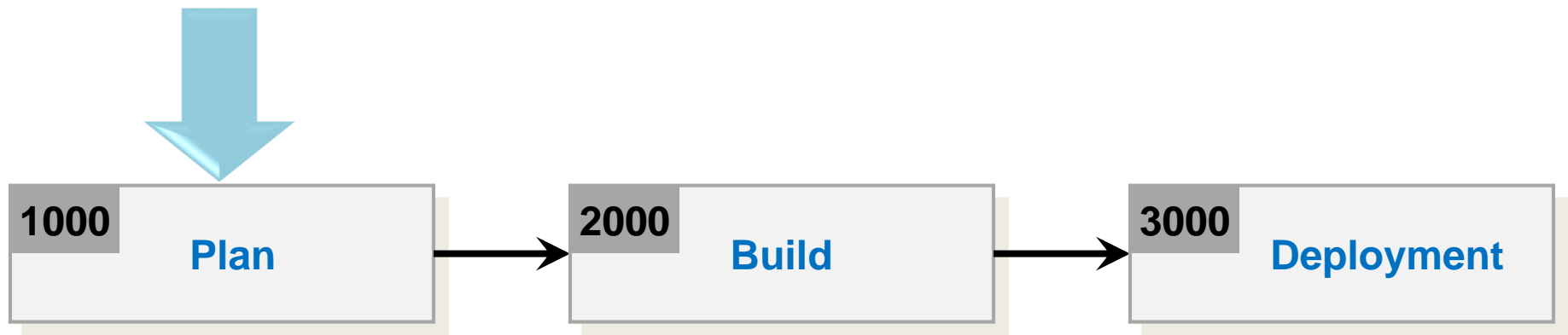


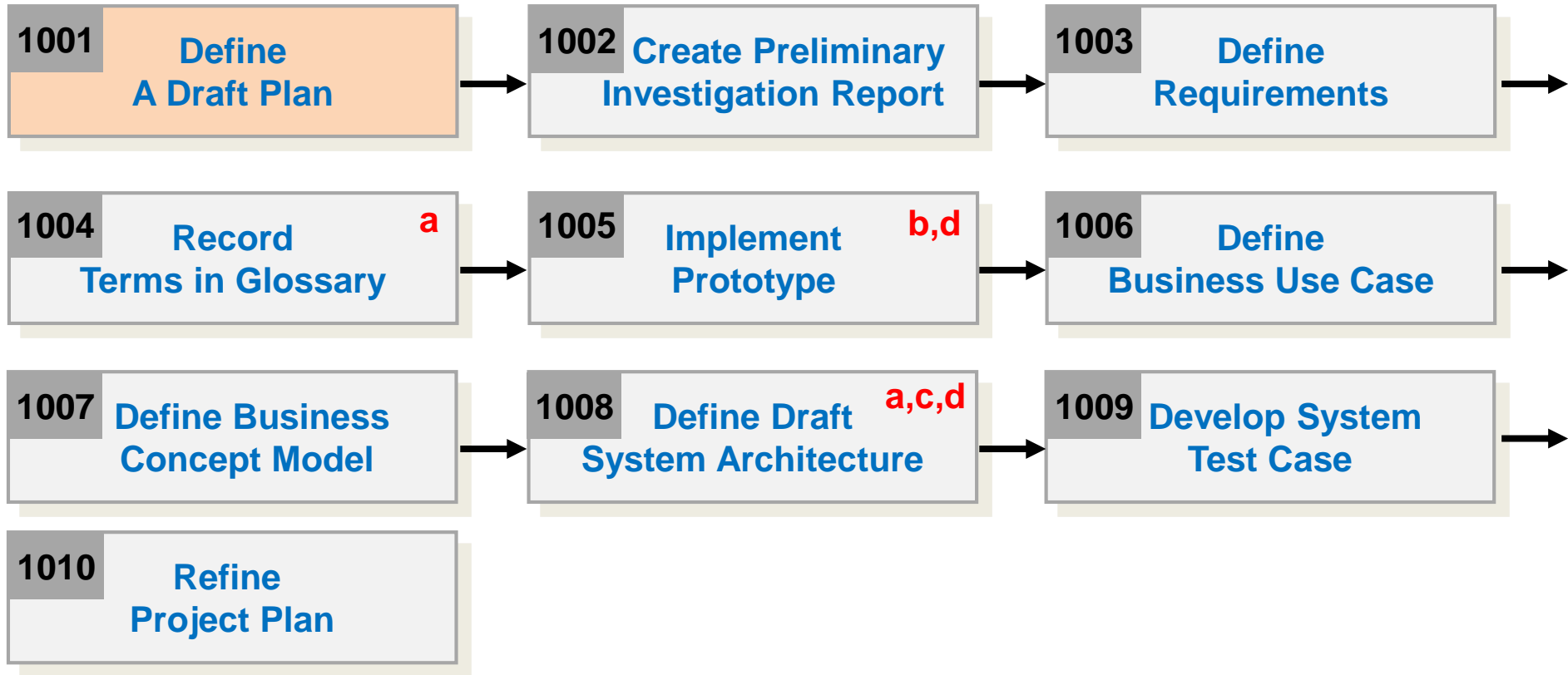
# **OOPT - Case Study**

## **Library Management System (LMS)**

# Stage 1000. Plan



# Activity 1001. Define a Draft Plan



# Activity 1001. Define a Draft Plan

- Motivation
  - The size of title volumes and the number of users for a city library are sharply increasing.
  - Hence, the city wants to develop a 'Library Management System' in order to automate most of the library operations.
  - Among the various library operations, they want to automate the most commonly used operations such as loan, reservation, purchase, discarding old books, and simple statistics.
- Project Objectives
  - To develop a computerized library management software, that provides typical library operations such as:
    - Lend and return books, Reserve books, Maintaining Borrow information, and Purchasing new books.
  - The new software should be easy to learn and use, and efficient.

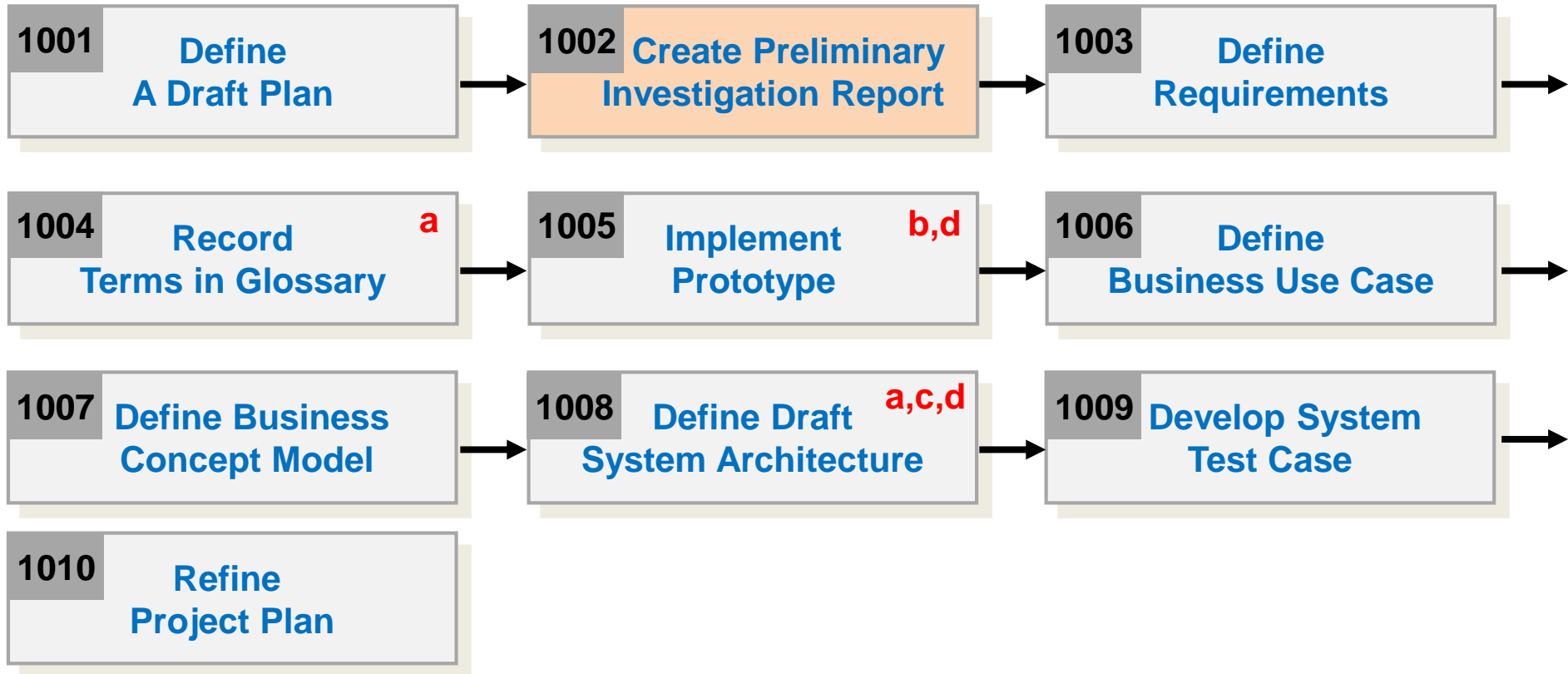
# Activity 1001. Define a Draft Plan

- Functional Requirements
  - Lend titles.
  - Return titles.
  - Reserve titles.
  - Purchase new titles.
  - Discard old titles.
  - Maintain borrower information.
  
- Non-Functional Requirements
  - The average response time for front desk operations should be less than 5 seconds.
  - The system should be designed to expandable and maintainable.

# Activity 1001. Define a Draft Plan

- Resource Estimation
  - Human Efforts(Man-Month): 6-10 M/M ?
  - Human Resource:
  - Project Duration:
  - Cost:
  
- Other Information
  - Future Version
    - Adopt 3-Tier Client/Server Architecture.
    - Add Web Interface.

# Activity 1002. Create Preliminary Investigation Report



# Activity 1002. Create Preliminary Investigation Report

- Alternative Solutions
  - Purchasing such a library managing software, if available.
  - Outsourcing
  - Other Options
  
- Project Justification (Business Demands)
  - Cost
  - Duration
  - Risk
  - Effect



# Activity 1002. Create Preliminary Investigation Report

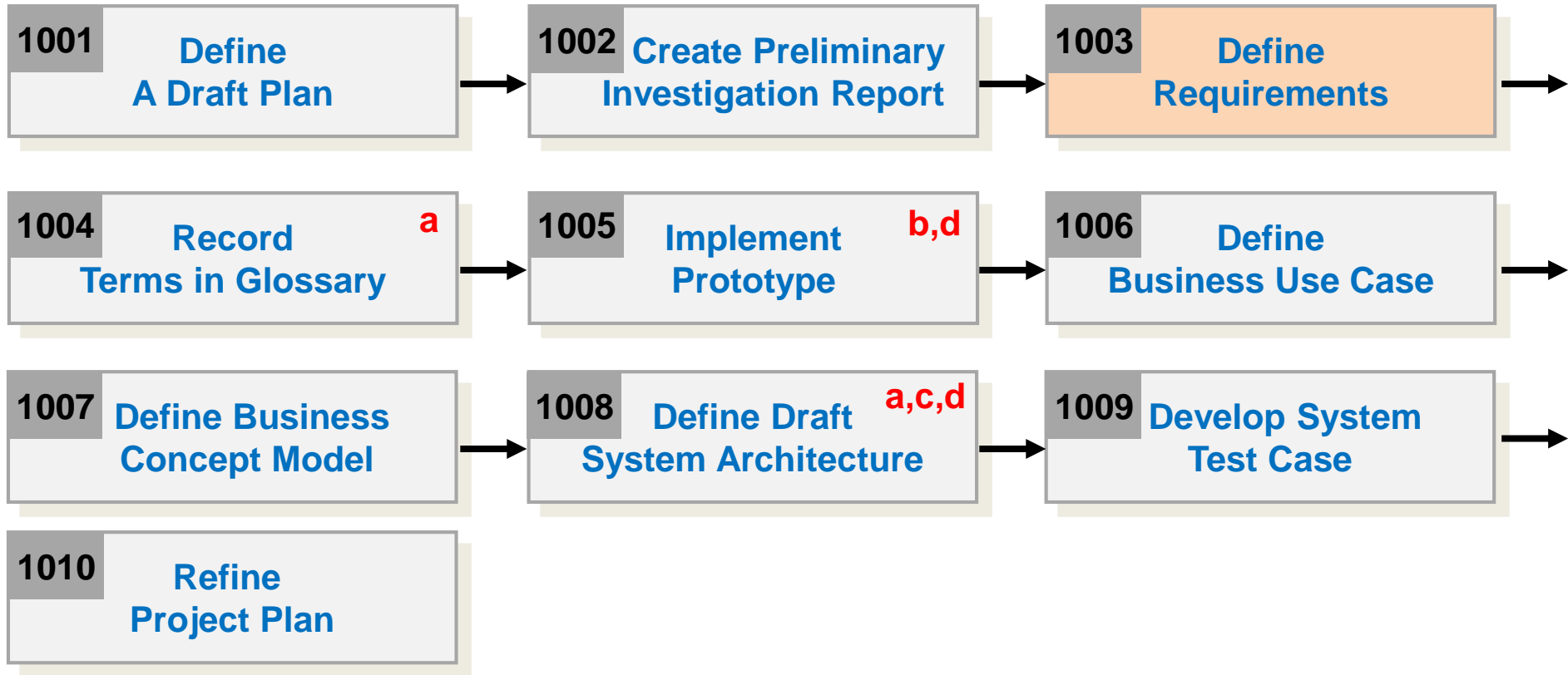
- Risk Management

Risk	Probability	Significance	Weight
Lack of OO experience	4	4	16
First adoption of OSP	4	5	20
Lack of domain knowledge	1	5	5
Team communication	3	3	9
Problem of requirements change	1	4	4
Lack of tool skill	2	2	4
Wandering	3	5	15

# Activity 1002. Create Preliminary Investigation Report

- Risk Reduction Plan
  - First adoption of OSP (20) : Try a pilot project using OSP
  - Lack of OO Project Experience (16) : Take part in a study group
  - Team Communication (9) : Have a team meeting on every Friday night
- Market Analysis
  - A few generic packages are available, however too expensive.
  - May be able to market the software to other similar-scaled libraries.
- Other Managerial Issues
  - The project should be completed by June, 2008.
    - Plan to participate in a SW exhibition.

# Activity 1003. Define Requirements



# Activity 1003. Define Requirements

- Functional Requirements (Version 0.9)
  - A library lends books and magazines to borrowers, who are registered in the system.
  - A library handles the purchase of new titles. Popular titles are bought in multiple copies.
  - Old books and magazines are removed when they are out of date or in poor condition.
  - The librarian is an employee of the library, who interacts with the customers and whose work is supported by the system.
  - A borrower can reserve a book or magazine that is not currently available in the library, so that when it's returned or purchased by the library, that person is notified.
  - The reservation is canceled
    - when the borrower checks out the book or magazine, or
    - through a explicit canceling procedure.
  - The library can easily create, update, and delete information about the titles, borrowers, loans, and reservations in the system.

# Activity 1003. Define Requirements

- User Interviews

Index	Question	Answer
1	Direct Interface with Borrower?	No, indirect
2	Can borrower search books on-line?	No, next version
3	Charge a fee for late return?	Yes, it just calculates the fee, and no direct interface with accounting software.
4	Charge a fee for lost books?	Yes, it just calculates the fee.
5	How to handle unregistered borrower?	First register and then lend items.
6	Is a notification available?	Yes, it can be printed on cards.
7	Calculate total number of titles checked out?	Yes
8	Specify max number of loans per borrower?	Yes
9	Specify max number of days for loans?	Yes
10	Send a kindly-reminder(SMS/Email) for return due?	No
11	Classify adult books?	Yes
12	Specify qualification for valid borrower?	No
13	Maintain reliable database?	Yes
14	Can control any system access?	Yes, through login and logout.

# Activity 1003. Define Requirements

- Functional Requirements (Version 1.0)
  - A library lends books and magazines to borrowers, who are registered in the system.
  - If the person has not been registered, the system first register the person. Then, lend titles.
  - A library handles the purchase of new titles. Popular titles are bought in multiple copies.
  - Old books and magazines are removed when they are out of date or in poor condition.
  - The librarian is an employee of the library who interacts with the customers(borrowers) and whose work is supported by the system.
  - A borrower can reserve a book or magazine that is not currently available in the library, so that when its returned or purchased by the library, that person is notified.
  - The system automatically prints 'post-cards' to notify the availability of the books. Then, the librarians mail them at the post office.

# Activity 1003. Define Requirements

- Functional Requirements (Version 1.0)
  - For unregistered person, the system first register the person. Then, make reservations
  - The reservation is canceled when the borrower checks out the book or magazine or through a explicit canceling procedure.
  - The library can easily create, update, and delete information about the titles, borrowers, loans, and reservations in the system.
  - Upon request, the system calculates the total # of items checked out.
  - For any over-due items, a late-return fee is calculated and charged.
  - For any items lost, a replacement-fee is computed and charged.
  - The system validates the system access through librarian IDs and passwords.
  - For each title, the librarians specify the maximum number of days that can be held by the borrowers.

# Activity 1003. Define Requirements

- Functional Requirements (Categorized Table)

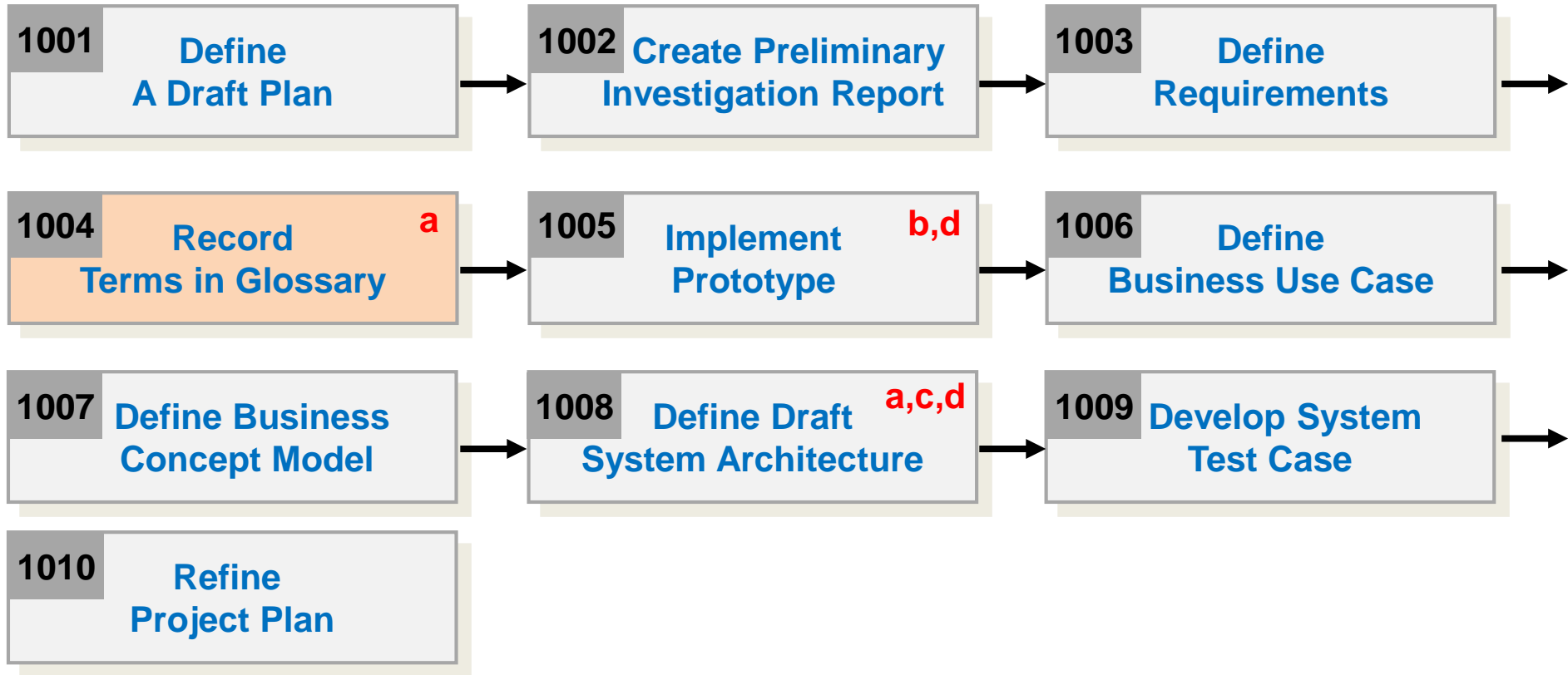
Ref. #	Function	Category
R1.1	Make reservation	Evident
R1.2	Remove reservation	Evident
R1.3	Lend Item	Evident
R1.4.1	Return title	Evident
R1.4.2	Calculate Late-Return-Fee	Hidden
R1.5	Calculate Replacement Fee	Evident
R1.6	Notify Availability	Hidden
R2.1	Add title	Evident
R2.2	Remove title	Evident
R2.3	Update title	Evident
R2.4	Add items	Evident
R2.5	Remove item	Evident
R2.6	Update item	Evident
R3.1	Add borrower	Evident
R3.2	Remove borrower	Evident
R3.3	Update borrower	Evident
R4.1	Validates system access	Evident
R5.1	Compute total # of items checked out	Evident



# Activity 1003. Define Requirements

- Performance Requirements
  - The average response time for front desk operations should be less than 5 seconds.
  - The post-card to notify availability must be printed out immediately after the reserved book becomes available.
- Operating Environment
  - Microsoft Windows 7 and 10
- Interface Requirements
  - The current version may incorporate a menu-driven approach.
  - Next version incorporates windows metaphor.
- Other Requirements
  - The system must control the system access.

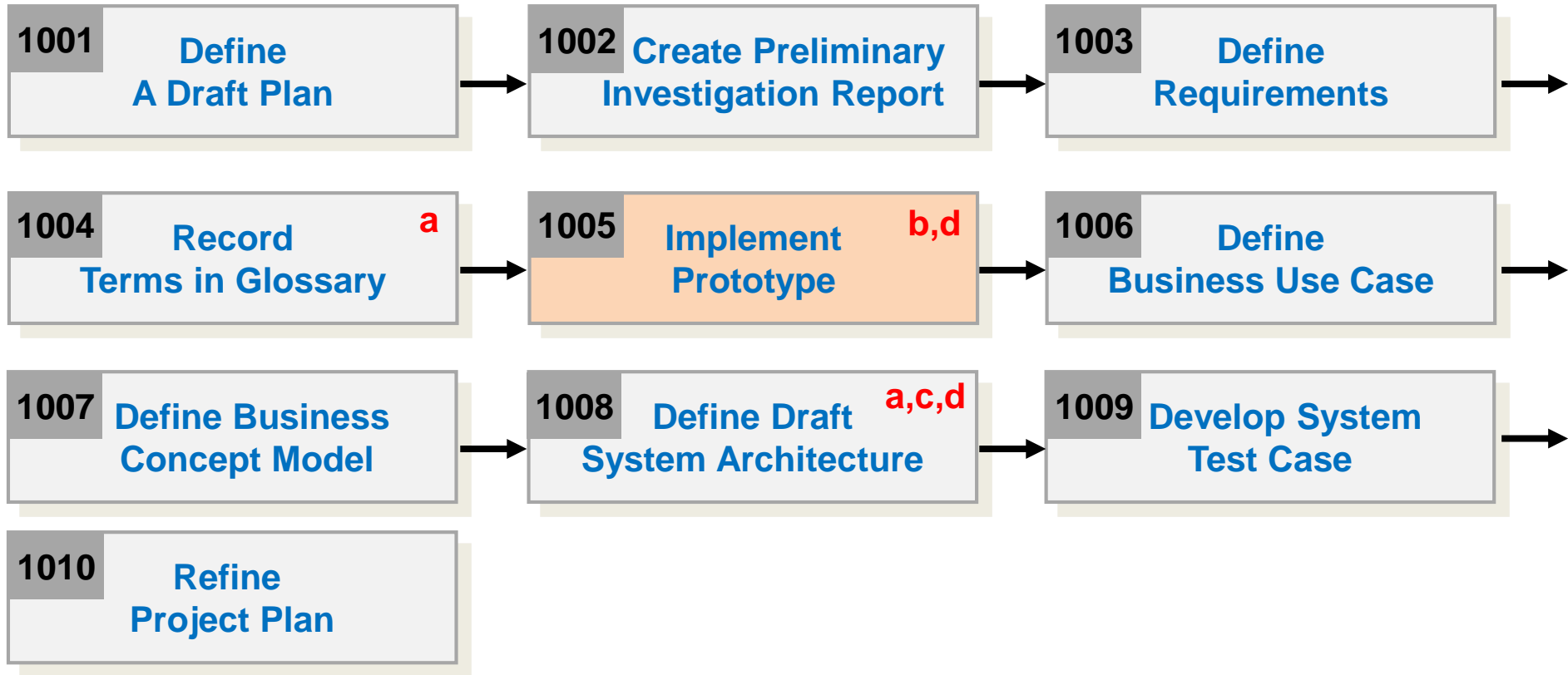
# Activity 1004. Record Terms in Glossary



# Activity 1004. Record Terms in Glossary

Term	Description	Remarks
Title	Books or Magazines, which are registered in the library system	
Item	Each copy of books or magazines	
Loan	An action of checking out an item from the library	
Librarian	An employee of the library who handles the requests of borrowers.	
...	...	

# Activity 1005. Implement Prototype

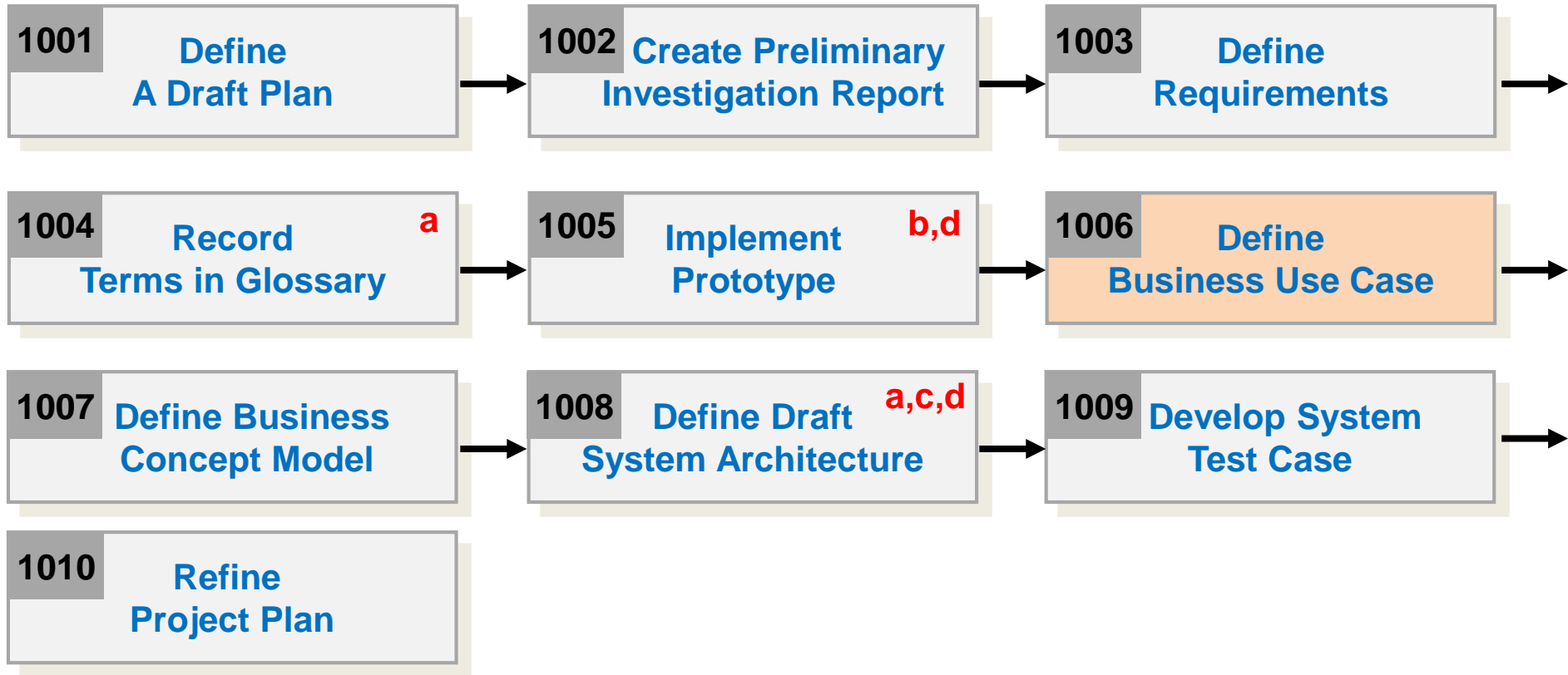


# Activity 1005. Implement Prototype

- User-Interface is sufficient for this LMS project

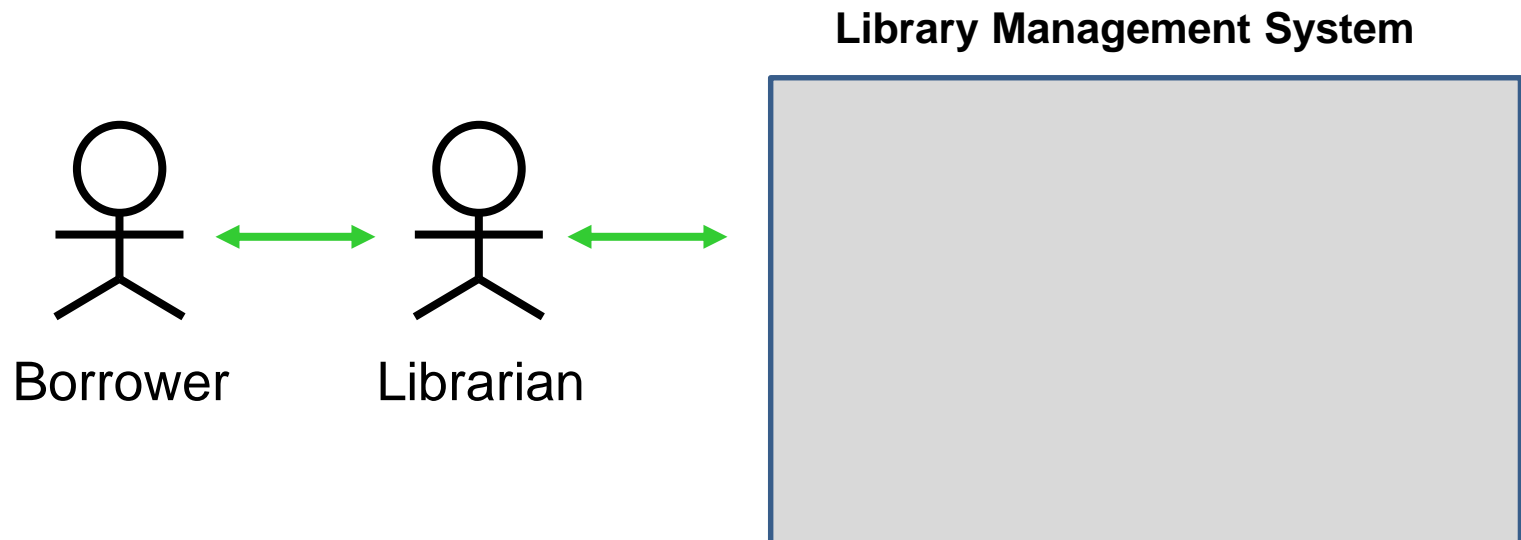
Authority	Loan	Maintenance	Statistics
Exit	Lend Item Return Item <hr/> Make Reservation Remove Reservation <hr/> Get Replacement Fee	Add Title Update Title Remove Title <hr/> Add Item Update Item Remove Item <hr/> Add Borrower Update Borrower Remove Borrower	Total # Loans

# Activity 1006. Define Business Use Case



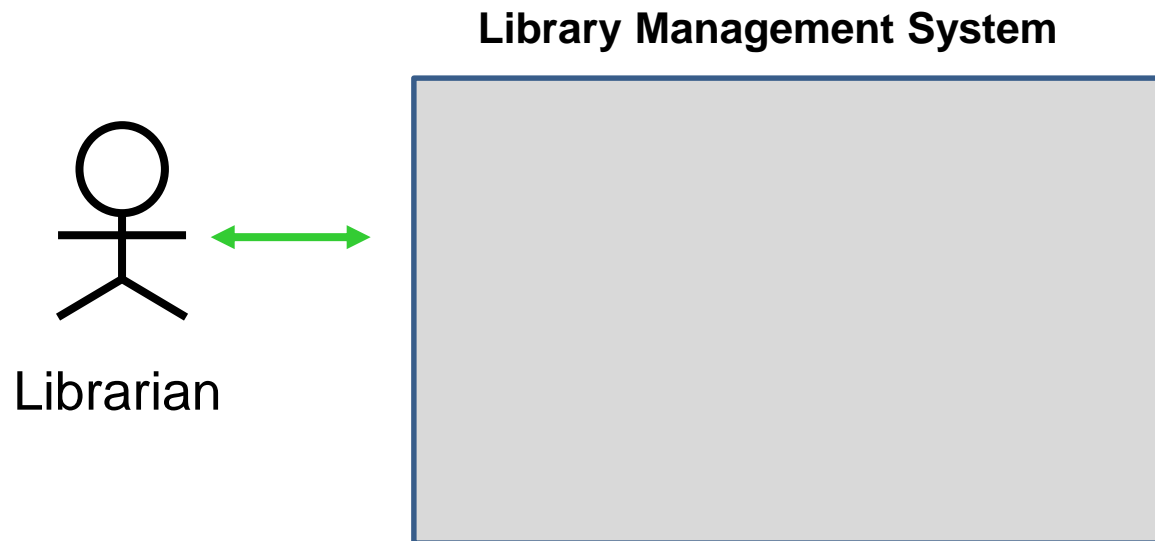
# Activity 1006. Define Business Use Case

- Step 1. Define system boundary
  - All the functions defined earlier are inside the system boundary.



# Activity 1006. Define Business Use Case

- Step 2. Identify the actors related to a system/organization
  - **Librarian** : an employee of the library who interacts with the customers(borrowers) and whose work is supported by the system.





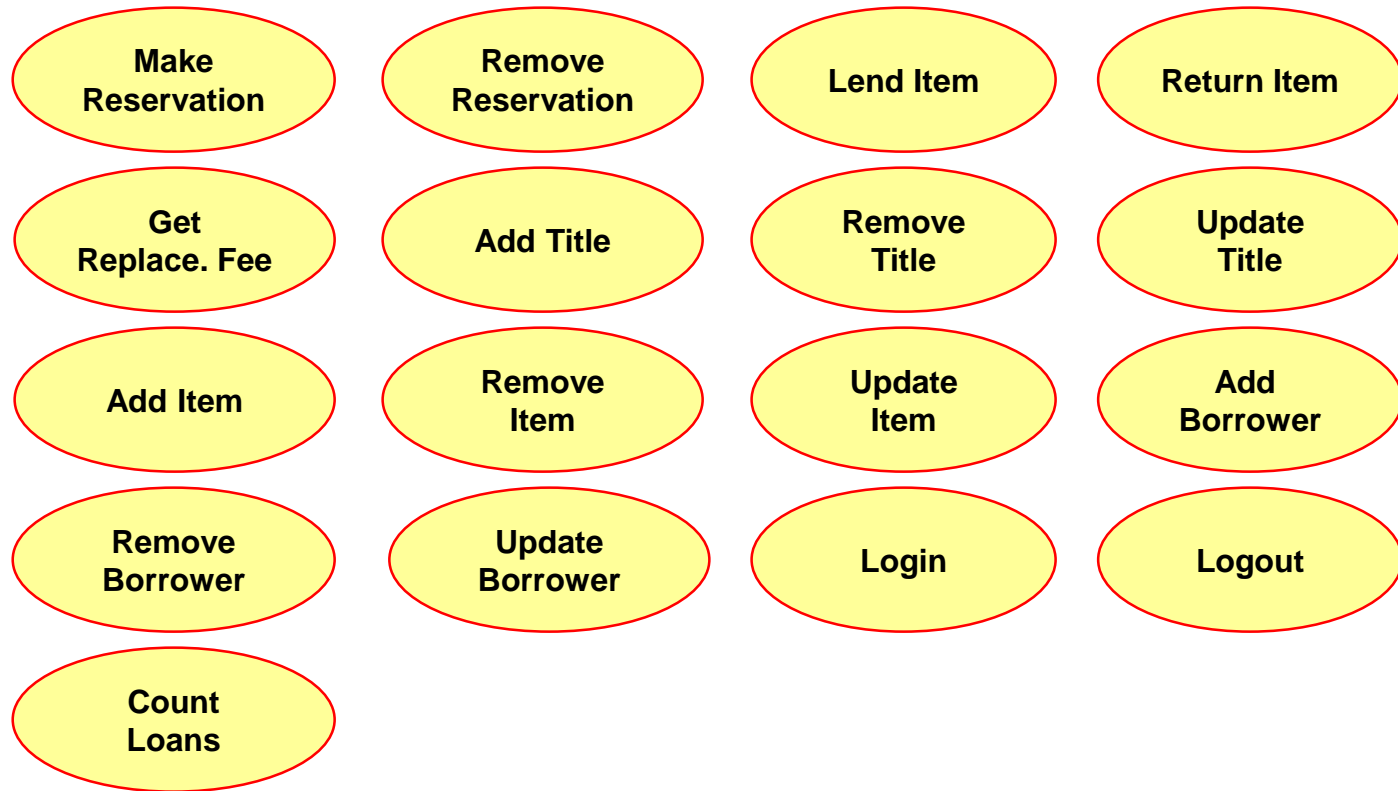
# Activity 1006. Define Business Use Case

- Step 3. Identify user goals for each actor
- Step 4. Record the primary actors and their goals in an actor-goal list

Actor	Goal
Librarian	Make reservation Remove reservation Lend Item Return title Calculate Late-Return-Fee Calculate Replacement Fee Notify Availability Add title Remove title Update title Add items Remove item Update item Add borrower Remove borrower Update borrower Validates system access Compute total # of items

# Activity 1006. Define Business Use Case

- Step. 5 Define use cases that satisfy user goals
  - Actor-based use cases



# Activity 1006. Define Business Use Case

- Step. 5 Define use cases that satisfy user goals
  - Event-based use cases

**Calculate  
Late-Return-Fee**

**Modify  
Availability**

# Activity 1006. Define Business Use Case

- Step 6. Allocate system functions into related use cases

Ref. #	Function	Use Case Number & Name
R1.1	Make reservation	1. Make Reservation
R1.2	Remove reservation	2. Remove Reservation
R1.3	Lend Item	3. Lend Item
R1.4.1	Return title	4. Return Title
R1.4.2	Calculate Late-Return-Fee	5. Calculate Late-Return-Fee
R1.5	Calculate Replacement Fee	6. Get Replacement Fee
R1.6	Notify Availability	7. Notify Availability
R2.1	Add title	8. Add Title
R2.2	Remove title	9. Remove Title
R2.3	Update title	10. Update Title
R2.4	Add items	11. Add Item
R2.5	Remove item	12. Remove Item
R2.6	Update item	13. Update Item
R3.1	Add borrower	14. Add Borrower
R3.2	Remove borrower	15. Remove Borrower
R3.3	Update borrower	16. Update Borrower
R4.1	Validates system access	17. Log-IN
R4.2	Validates system access	18. Log-Out
R5.1	Compute total # of items checked out	19. Count Loans

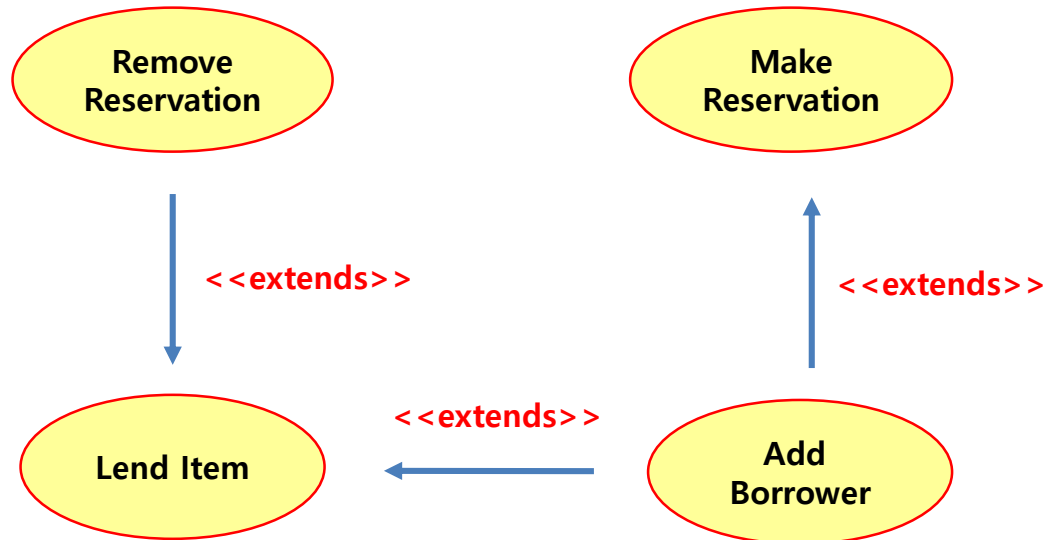
# Activity 1006. Define Business Use Case

- Step 7. Categorize use cases

Ref. #	Function	Use Case Number & Name	Category	Category
R1.1	Make reservation	1. Make Reservation	Primary	Evident
R1.2	Remove reservation	2. Remove Reservation	Primary	Evident
R1.3	Lend Item	3. Lend Item	Primary	Evident
R1.4.1	Return title	4. Return Title	Primary	Evident
R1.4.2	Calculate Late-Return-Fee	5. Calculate Late-Return-Fee	Primary	Hidden
R1.5	Calculate Replacement Fee	6. Get Replacement Fee	Primary	Evident
R1.6	Notify Availability	7. Notify Availability	Primary	Hidden
R2.1	Add title	8. Add Title	Primary	Evident
R2.2	Remove title	9. Remove Title	Primary	Evident
R2.3	Update title	10. Update Title	Primary	Evident
R2.4	Add items	11. Add Item	Primary	Evident
R2.5	Remove item	12. Remove Item	Primary	Evident
R2.6	Update item	13. Update Item	Primary	Evident
R3.1	Add borrower	14. Add Borrower	Primary	Evident
R3.2	Remove borrower	15. Remove Borrower	Primary	Evident
R3.3	Update borrower	16. Update Borrower	Primary	Evident
R4.1	Validates system access	17. Log-IN	Secondary	Evident
R4.2	Validates system access	18. Log-Out	Secondary	Evident
R5.1	Compute total # of items checked out	19. Count Loans	Secondary	Evident

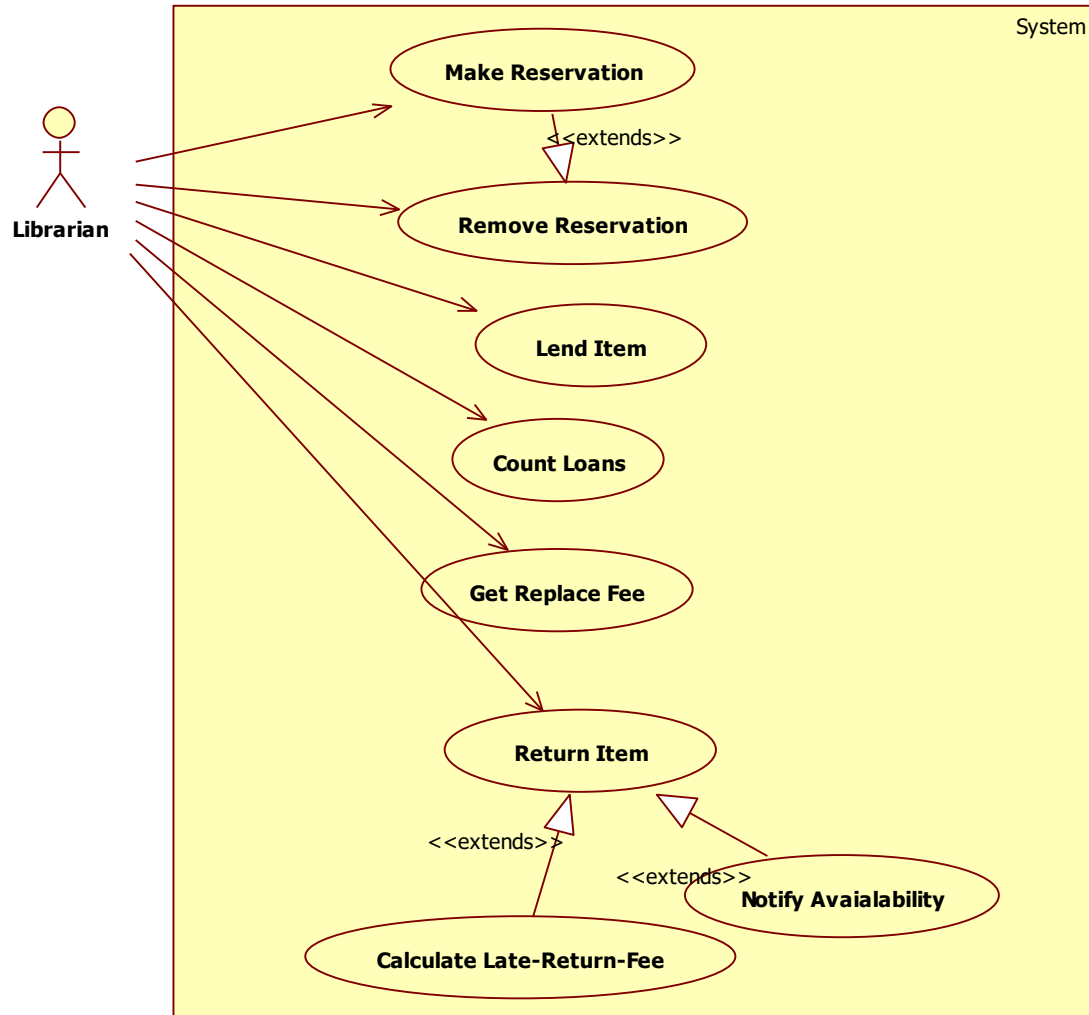
# Activity 1006. Define Business Use Case

- Step 8. Identify relationships between use cases (Optional)



- [illegible]

# Activity 1006. Define Business Use Case





# Activity 1006. Define Business Use Case

- Step 10. Describe use cases

<b>Use Case</b>	1. Make Reservation
<b>Actors</b>	Librarian
<b>Description</b>	<ul style="list-style-type: none"> <li>- This use case begins when a borrower arrives at the counter and then requests reservation.</li> <li>- For a registered borrower, it makes a reservation slip (software-wise).</li> <li>- For an unregistered borrower, the librarian registers the person and makes a reservation for the person.</li> </ul>

<b>Use Case</b>	2. Remove Reservation
<b>Actors</b>	Librarian
<b>Description</b>	<ul style="list-style-type: none"> <li>- A borrower who made a reservation can cancel his/her reservation. <ul style="list-style-type: none"> <li>• Explicitly cancels the reservation. (Evident)</li> </ul> </li> <li>- When a borrower checks out an item which he/she previously reserved, this use case is invoked automatically. <ul style="list-style-type: none"> <li>• Hidden system function</li> </ul> </li> </ul>

# Activity 1006. Define Business Use Case

<b>Use Case</b>	3. Lend Item
<b>Actors</b>	Librarian
<b>Description</b>	<ul style="list-style-type: none"> <li>- This use case begins when the borrower arrives at the front desk with items to lend.</li> <li>- If a borrower does not registered, register first his/her information in the system.</li> <li>- This use case records the date, borrower ID, item ID and other relevant information for this loan.</li> </ul>

<b>Use Case</b>	4. Return Item
<b>Actors</b>	Librarian
<b>Description</b>	<ul style="list-style-type: none"> <li>- This use case begins when a borrower returns items at the counter.</li> <li>- If the item is returned past due date, a late-return-fee is computed, so that the borrower should pay the penalty.</li> </ul>

# Activity 1006. Define Business Use Case

<b>Use Case</b>	5. Calculate Late-Return-Fee
<b>Actors</b>	None
<b>Description</b>	<ul style="list-style-type: none"> <li>- This use case computes the penalty amount for items returned late.</li> <li>- It first computes the number of extra days held by the borrower, then multiplies it by a pre-determined daily rate for late returns.</li> </ul>

<b>Use Case</b>	6. Get Replacement Fee
<b>Actors</b>	Librarian
<b>Description</b>	<ul style="list-style-type: none"> <li>- This use case computes the cost for replacing the lost book.</li> <li>- It first finds out the current price of the lost book, and add the handling cost to the book price.</li> </ul>

<b>Use Case</b>	7. Notify Availability
<b>Actors</b>	None
<b>Description</b>	<ul style="list-style-type: none"> <li>- This use case prints the book title that just became available, number of days held by the library, the name and address of the person who reserved on a post-card.</li> <li>- The actual mailing will be done manually by the librarian.</li> </ul>

# Activity 1006. Define Business Use Case

<b>Use Case</b>	8. Add Title
<b>Actors</b>	Librarian
<b>Description</b>	<ul style="list-style-type: none"> <li>- Whenever a new kind of book is purchased, the book information is recorded into the system.</li> <li>- Then, it invokes 'Add Item' use case to record the number of copies purchased.</li> </ul>

<b>Use Case</b>	9. Remove Title
<b>Actors</b>	Librarian
<b>Description</b>	<ul style="list-style-type: none"> <li>- Some old books are selected for removal by the librarians.</li> <li>- This use case deletes the information of the book to be removed.</li> <li>- And, it will be no longer available for loans.</li> </ul>

<b>Use Case</b>	10. Update Title
<b>Actors</b>	Librarian
<b>Description</b>	<ul style="list-style-type: none"> <li>- This use case will change the recorded information of the title.</li> <li>- What actual kinds of information?</li> </ul>

# Activity 1006. Define Business Use Case

<b>Use Case</b>	11. Add Item
<b>Actors</b>	Librarian
<b>Description</b>	<ul style="list-style-type: none"> <li>- When additional copies (of the currently available title) are purchases, this updates the total number of copies for each title. <ul style="list-style-type: none"> <li>• Date, Price, Bookstore, Available, etc.</li> </ul> </li> <li>- When a reservation has been made for this title, this use case invokes 'notify availability' use case.</li> </ul>

<b>Use Case</b>	12. Remove Item
<b>Actors</b>	Librarian
<b>Description</b>	<ul style="list-style-type: none"> <li>- This use case will update the number of items for each title.</li> <li>- If no more item is remaining after removal, this use case will invoke 'Remove Title' use case.</li> </ul>

<b>Use Case</b>	13. Update Item
<b>Actors</b>	Librarian
<b>Description</b>	<ul style="list-style-type: none"> <li>- This use case updates the information of the items.</li> <li>- What actual kinds of information will be updated ?</li> </ul>

# Activity 1006. Define Business Use Case

<b>Use Case</b>	14. Add Borrower
<b>Actors</b>	Librarian
<b>Description</b>	- This use case will record the information of the new borrower such as name, address, phone, loan priority, etc.

<b>Use Case</b>	15. Remove Borrower
<b>Actors</b>	Librarian
<b>Description</b>	<ul style="list-style-type: none"> <li>- This use case deletes the information of borrower from the system, so that the person can no longer check out titles.</li> <li>- This may happen if the borrower has a bad return history or has not been using the library longer than 2 years.</li> </ul>

<b>Use Case</b>	16. Update Borrower
<b>Actors</b>	Librarian
<b>Description</b>	- This use case updates the information of the borrower such as new address and phone.

# Activity 1006. Define Business Use Case

<b>Use Case</b>	17. Log-In
<b>Actors</b>	Librarian
<b>Description</b>	<ul style="list-style-type: none"> <li>- This use case reads the user ID and password of the librarian, and verifies.</li> <li>- If an invalid information is entered, it will re-prompt and read the ID and password.</li> <li>- After 3 successive failures of login, it records this 'attach' information and automatically returns to the initial menu.</li> </ul>

<b>Use Case</b>	18. Log-Out
<b>Actors</b>	Librarian
<b>Description</b>	<ul style="list-style-type: none"> <li>- This use case records the date and time of the current logout, and returns to the initial menu.</li> </ul>

<b>Use Case</b>	19. Count Loans
<b>Actors</b>	Librarian
<b>Description</b>	<ul style="list-style-type: none"> <li>- This use cases computes the total number of items checked out.</li> </ul>

# Activity 1006. Define Business Use Case

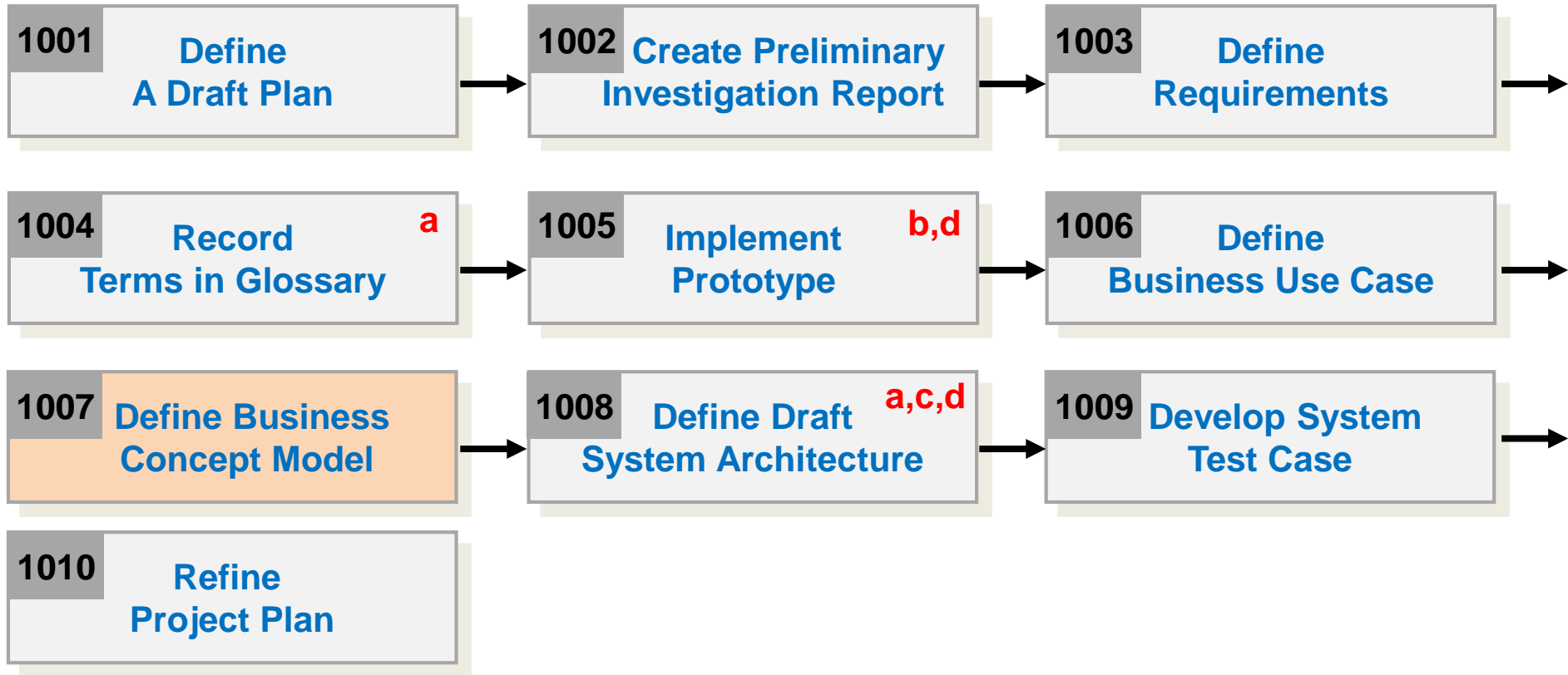
- Step 11. Rank use cases

Ref. #	Function	Use Case Number & Name	Category	Rank	Category
R1.1	Make reservation	1. Make Reservation	Primary	High	Evident
R1.2	Remove reservation	2. Remove Reservation	Primary	High	Evident
R1.3	Lend Item	3. Lend Item	Primary	High	Evident
R1.4.1	Return title	4. Return Title	Primary	High	Evident
R1.4.2	Calculate Late-Return-Fee	5. Calculate Late-Return-Fee	Primary	High	Hidden
R1.5	Calculate Replacement Fee	6. Get Replacement Fee	Primary	High	Evident
R1.6	Notify Availability	7. Notify Availability	Primary	High	Hidden
R2.1	Add title	8. Add Title	Primary	High	Evident
R2.2	Remove title	9. Remove Title	Primary	High	Evident
R2.3	Update title	10. Update Title	Primary	High	Evident
R2.4	Add items	11. Add Item	Primary	High	Evident
R2.5	Remove item	12. Remove Item	Primary	High	Evident
R2.6	Update item	13. Update Item	Primary	High	Evident
R3.1	Add borrower	14. Add Borrower	Primary	High	Evident
R3.2	Remove borrower	15. Remove Borrower	Primary	High	Evident
R3.3	Update borrower	16. Update Borrower	Primary	High	Evident
R4.1	Validates system access	17. Log-IN	Secondary	Medium	Evident
R4.2	Validates system access	18. Log-Out	Secondary	Medium	Evident
R5.1	Compute total # of items checked out	19. Count Loans	Secondary	Medium	Evident



# Activity 1007.

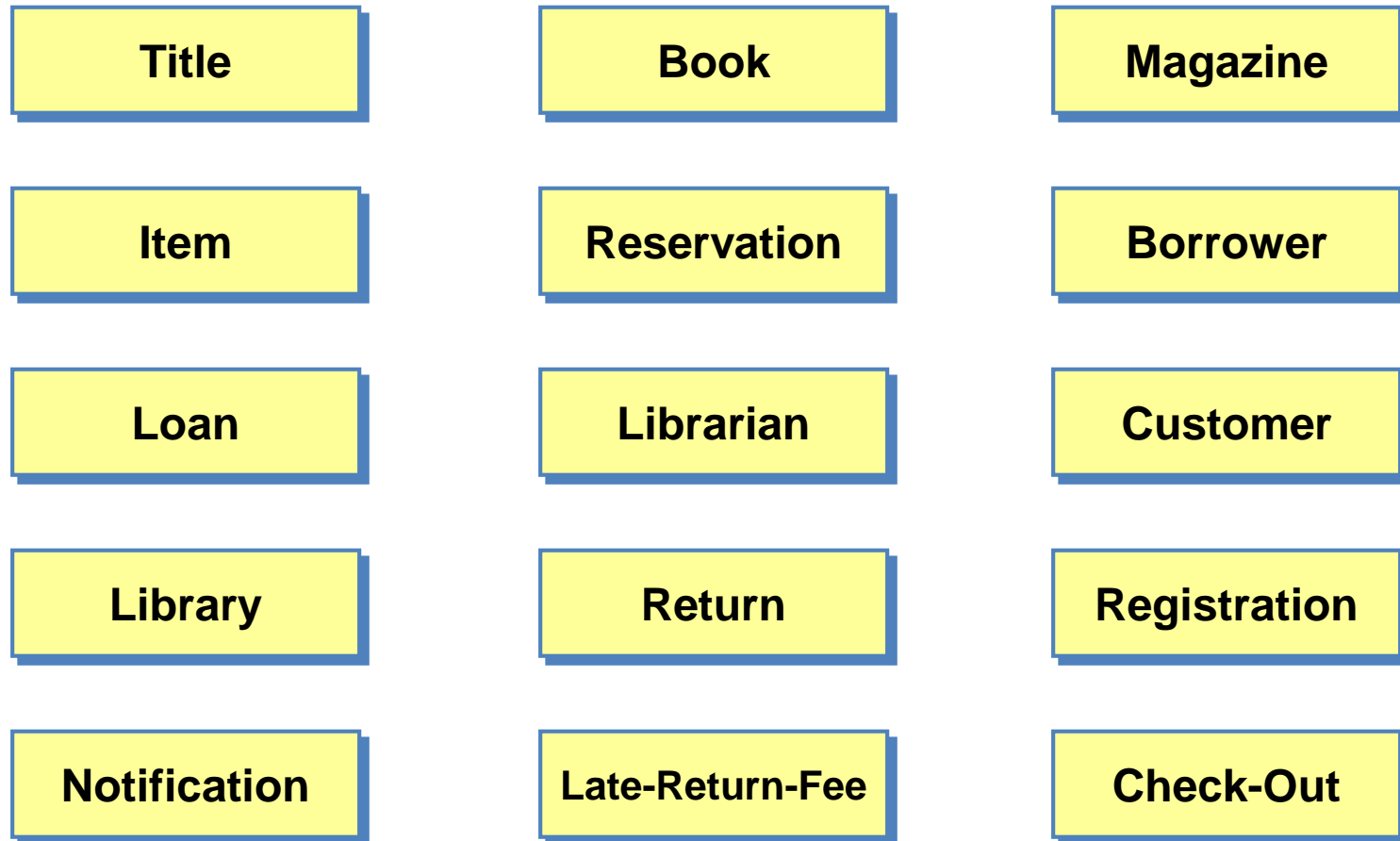
## Define Business Concept Model



# Activity 1007.

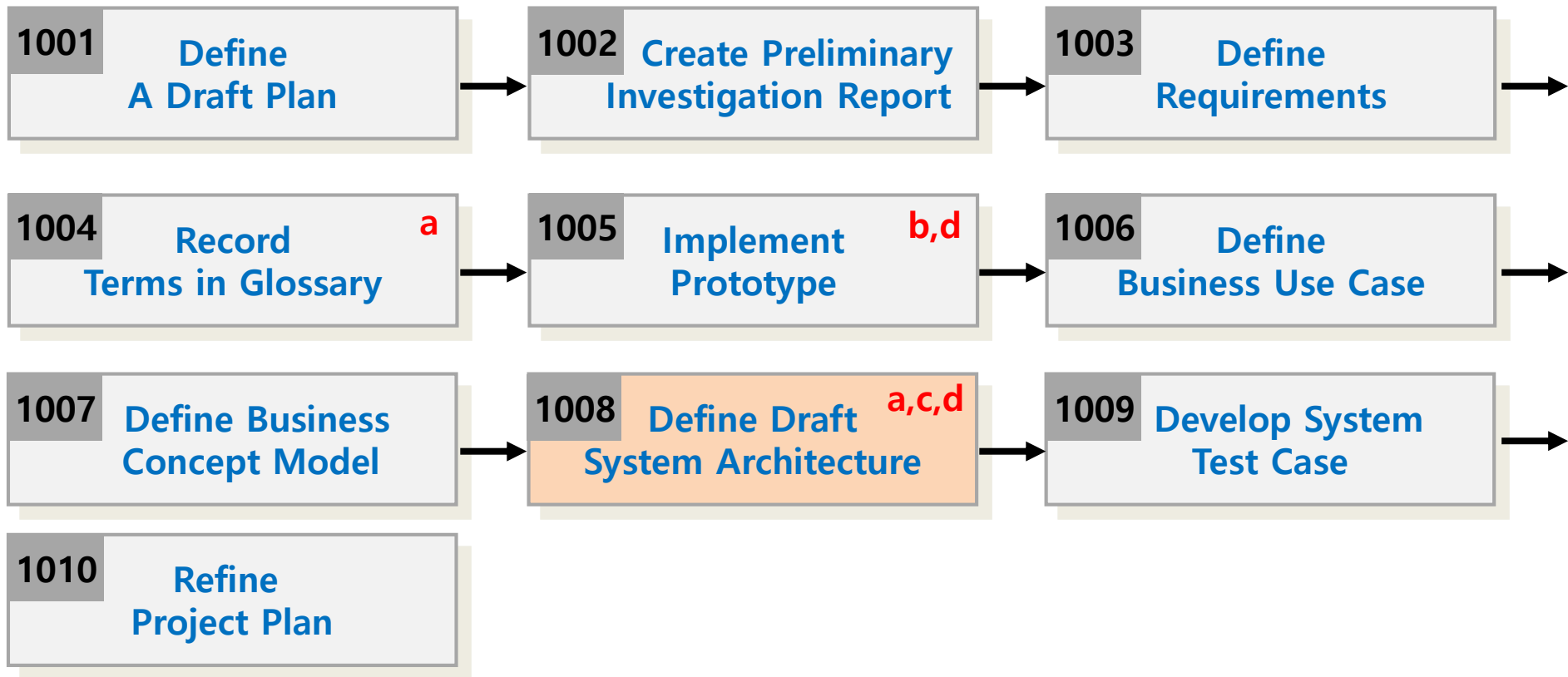
## Define Business Concept Model

- Identify 'Concepts' in the target domain.



# Activity 1008.

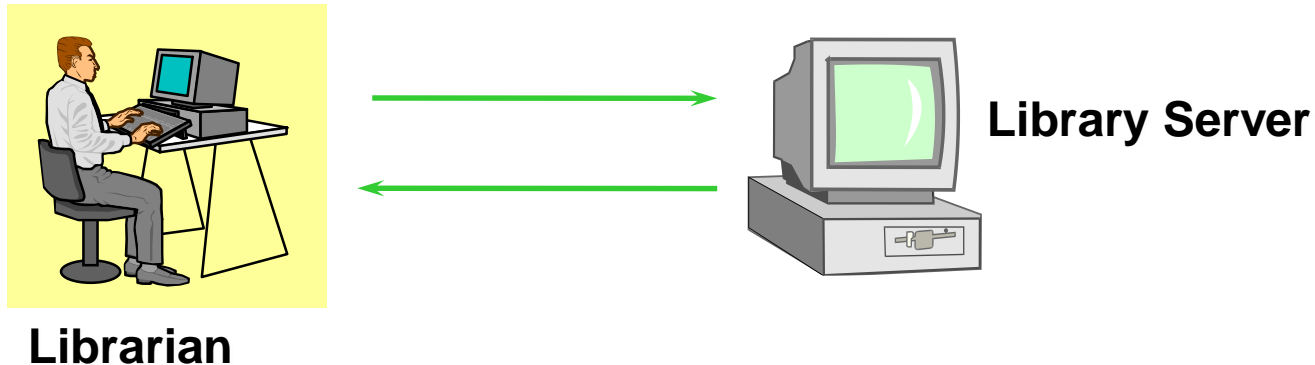
## Define Draft System Architecture



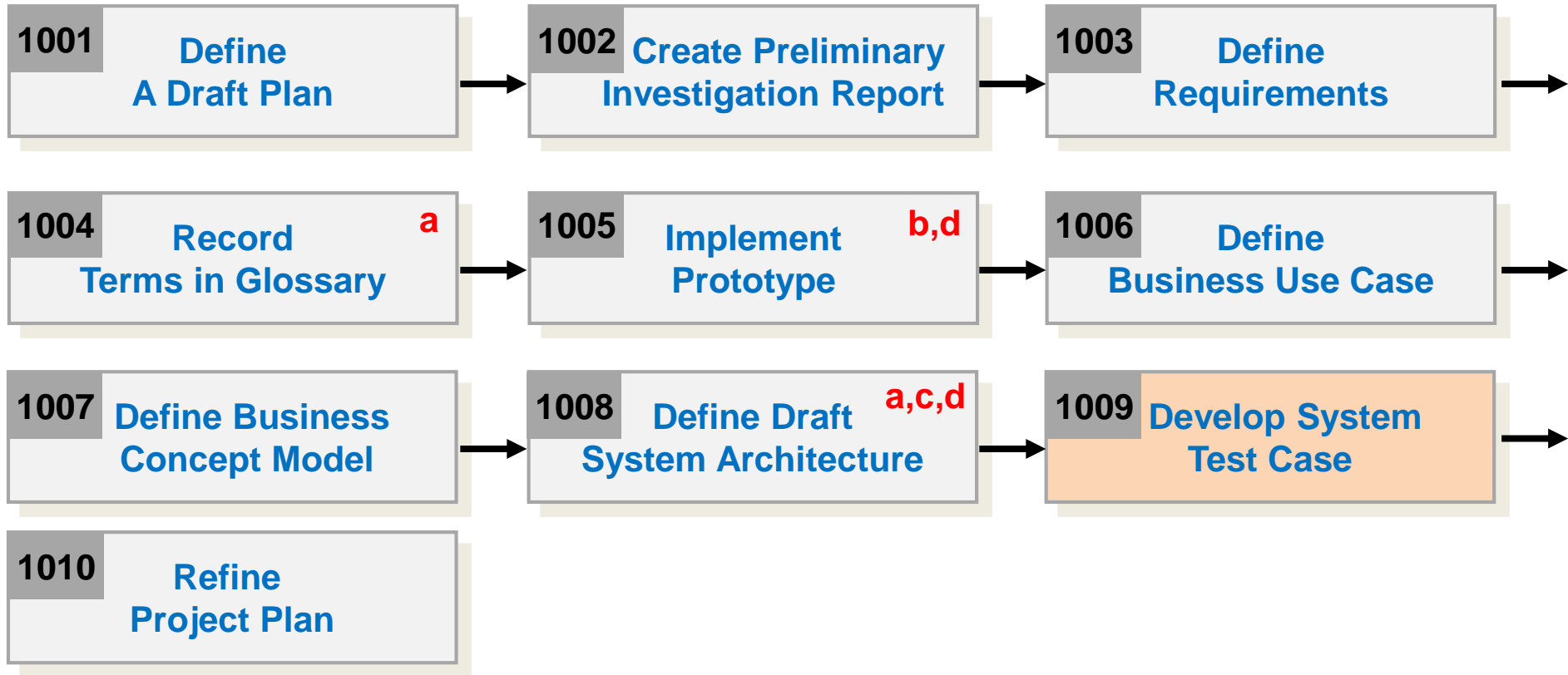
# Activity 1008.

## Define Draft System Architecture

- Define system architecture



# Activity 1009. Develop System Test Case



# Activity 1009. Develop System Test Case

- Step 1. Identify important requirements

Ref. #	Function	Category
R1.1	<b>Make reservation</b>	Evident
R1.2	<b>Remove reservation</b>	Evident
R1.3	<b>Lend Item</b>	Evident
R1.4.1	<b>Return title</b>	Evident
R1.4.2	Calculate Late-Return-Fee	Hidden
R1.5	<b>Calculate Replacement Fee</b>	Evident
R1.6	Notify Availability	Hidden
R2.1	<b>Add title</b>	Evident
R2.2	<b>Remove title</b>	Evident
R2.3	<b>Update title</b>	Evident
R2.4	<b>Add items</b>	Evident
R2.5	<b>Remove item</b>	Evident
R2.6	<b>Update item</b>	Evident
R3.1	<b>Add borrower</b>	Evident
R3.2	<b>Remove borrower</b>	Evident
R3.3	<b>Update borrower</b>	Evident
R4.1	<b>Validates system access</b>	Evident
R5.1	<b>Compute total # of items checked out</b>	Evident

# Activity 1009. Develop System Test Case

- Step 2. Develop system test cases with various system testing techniques
  - First, brute force testing

No.	Tests	Description
1	<b>Make reservation</b>	Correct한 borrower가 correct한 title 예약
2	<b>Make reservation</b>	Correct한 borrower가 incorrect한 title 예약
3	<b>Make reservation</b>	Correct한 borrower가 대여중인 title 예약
4	<b>Make reservation</b>	Incorrect한 borrower가 예약
5	<b>Remove reservation</b>	Correct한 borrower가 예약 취소
6	<b>Remove reservation</b>	Incorrect한 borrower가 예약 취소
7	<b>Lend Item</b>	Correct한 borrower가 대여 가능한 title 대여
8	<b>Lend Item</b>	Correct한 borrower가 incorrect한 title 대여
9	<b>Lend Item</b>	Correct한 borrower가 모두 대여중인 title 대여
10	<b>Lend Item</b>	Incorrect한 borrower가 대여
11	<b>Return title</b>	Borrower가 title 반납
12	<b>Return title</b>	Borrower가 연체된 title 반납
13	<b>Add title</b>	새 title 추가
14	<b>Remove title</b>	기존의 title 제거
15	<b>Remove title</b>	존재하지 않는 title 제거
16	<b>Update title</b>	Title 정보 update
17	<b>Add item</b>	Title item 추가
18	<b>Add item</b>	존재하지 않는 title의 item추가

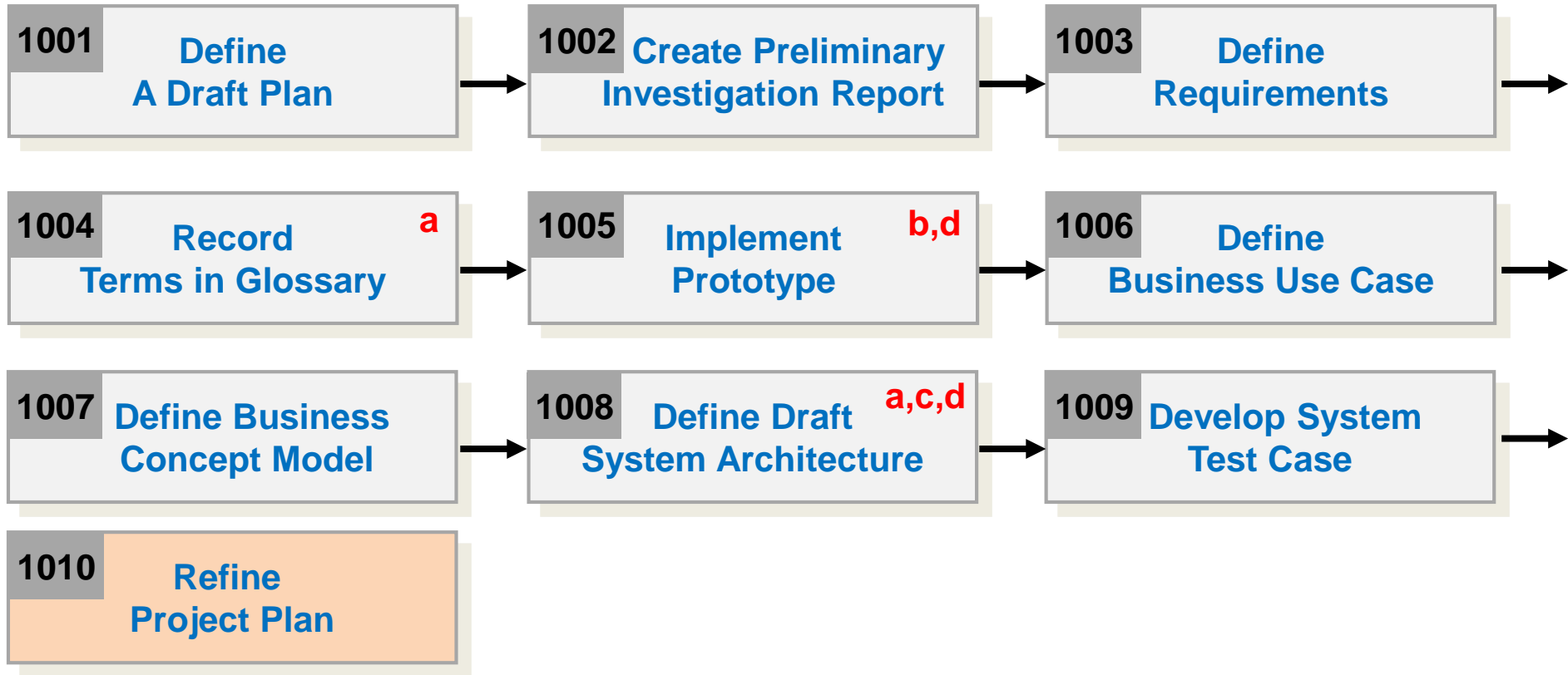
# Activity 1009. Develop System Test Case

- Step 2. Develop system test cases with various system testing techniques
  - First, brute force testing

No.	Tests	Description
19	Remove item	Title의 item제거
20	Remove item	존재하지 않는 title의 item제거
21	Update item	올바른 item의 정보 update
22	Update item	Title에 존재하지 않는 item update
23	Add borrower	Borrower 추가
24	Remove borrower	Borrower 삭제
25	Update borrower	기존의 borrower update
26	Update borrower	삭제된 borrower update
27	Validates system access	Correct id/pw로 로그인
28	Validates system access	Incorrect id/pw로 로그인
29	Validates system access	로그아웃
30	Compute total # of items checked out	계산 시도



# Activity 1010. Refine Project Plan



# Activity 1010. Refine Project Plan

- Project Scope
  - The library management software automates typical library operations; reservation, lending item, adding, removing, and updating the information of title, item, and borrower.
  
- Project Objectives
  - To develop a computerized library management software, that provides typical library operations such as:
    - Lend and return books, Reserve books, Maintaining Borrow information, and Purchasing new books.
  - The new software should be easy to learn and use, and efficient.

# Activity 1010. Refine Project Plan

- Functional Requirements

Ref. #	Function	Category
R1.1	Make reservation	Evident
R1.2	Remove reservation	Evident
R1.3	Lend Item	Evident
R1.4.1	Return title	Evident
R1.4.2	Calculate Late-Return-Fee	Hidden
R1.5	Calculate Replacement Fee	Evident
R1.6	Notify Availability	Hidden
R2.1	Add title	Evident
R2.2	Remove title	Evident
R2.3	Update title	Evident
R2.4	Add items	Evident
R2.5	Remove item	Evident
R2.6	Update item	Evident
R3.1	Add borrower	Evident
R3.2	Remove borrower	Evident
R3.3	Update borrower	Evident
R4.1	Validates system access	Evident
R5.1	Compute total # of items checked out	Evident

# Activity 1010. Refine Project Plan

- Performance Requirements
  - When making reservations, the information of reservation will appear within 5 seconds.
  - When lending items, the content of lending item will appear within 5 seconds.
  - When returning items, the content of returning item will appear within 5 seconds.
- Operating Environment
  - Microsoft Windows 7 and 10
- User Interface Requirements
  - Menu-driven approach
  - Should be designed for upgrading to 'Window-based' version.

# Activity 1010. Refine Project Plan

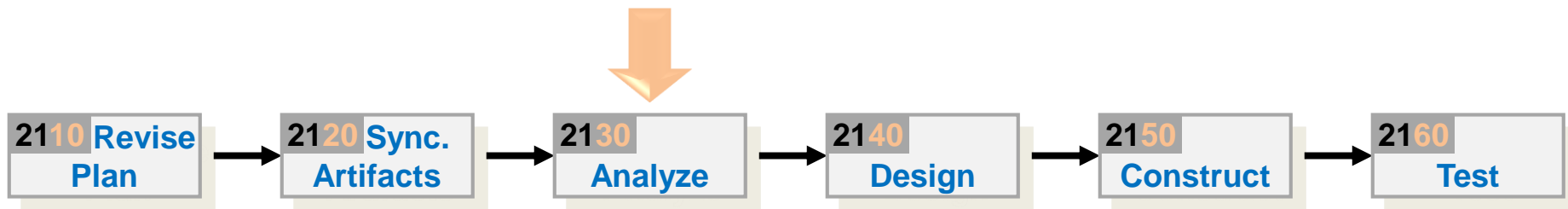
- Other Requirements
  - The content of database should be maintained reliably.
  - System should control the system access.
  
- Resources
  - Man Month : 6 Persons
    - A Team Leader
    - A Document Manager
    - 3-4 Engineers
  - Period : 5 Days (Around 40 Hours)
  - Hardware : skylake processor
  - Software
    - OS : Windows 7/10
    - Programming Language : Java
    - Case Tools : Rational Rose, Paradigm Plus

# Activity 1010. Refine Project Plan

- Scheduling

Stage	Phase(ooxo)/Activity(ooox)	Schedule(Day)				
		1	2	3	4	5
1000. Plan & Elaborate	1001. Define Draft Plan	=====				
	1002. Create Preliminary Investigation Report	=====				
	1003. Define Requirements	=====				
	1004. Record Terms in Glossary	=====				
	1005. Implement Prototype	=====				
	1006. Define Use Cases	=====				
	1007. Define Draft Conceptual Model	=====				
	1008. Define Draft System Architecture	=====				
	1009. Refine Plan	=====				
	2010. Revise Plan	=====				
2000. Build	2020. Synchronize Artifacts	=====				
	2030. Analyze	=====				
	2031. Define Essential Use Case	=====				
	2032. Refine Use Case Diagrams	=====				
	2033. Refine Conceptual Model	=====				
	2034. Refine Glossary	=====				
	2035. Define System Sequence Diagrams	=====				
	2036. Define Operation Contracts	=====				
	2037. Define State Diagrams	=====				
	2040. Design	=====				
	2041. Define Real Use Cases	=====				
	2042. Define Reports, UI and Storyboards	=====				
	2043. Refine System Architecture	=====				
	2044. Define Interaction Diagrams	=====				
	2045. Define Design Class Diagrams	=====				
	2046. Define Database Schema	=====				
	2050. Construct	=====				
	2051. Implement Class & Interface Definition	=====				
	2052. Implement Methods.	=====				
	2053. Implement Windows	=====				
	2054. Implement Reports	=====				
	2055. Implement DB Schema	=====				
	2056. Write Test Code	=====				
	2060. Test	=====				
	2061. Unit Testing	=====				
	2062. Integration Testing	=====				
	2063. System Testing	=====				
	2064. Performance Testing	=====				
	2065. Acceptance Testing	=====				
	2066. Documentation Testing	=====				
3000. Deploy-ment	3001. Complete Technical Documents	=====				
	3002. Complete User Documents	=====				
	3003. System Testing	=====				
	3004. Acceptance Testing	=====				
	3005. Documentation Testing	=====				
	3006. Train	=====				
	3007. Establish Parallel Runs and Crossover	=====				
	3008. Establish Support	=====				
	3009. Install	=====				

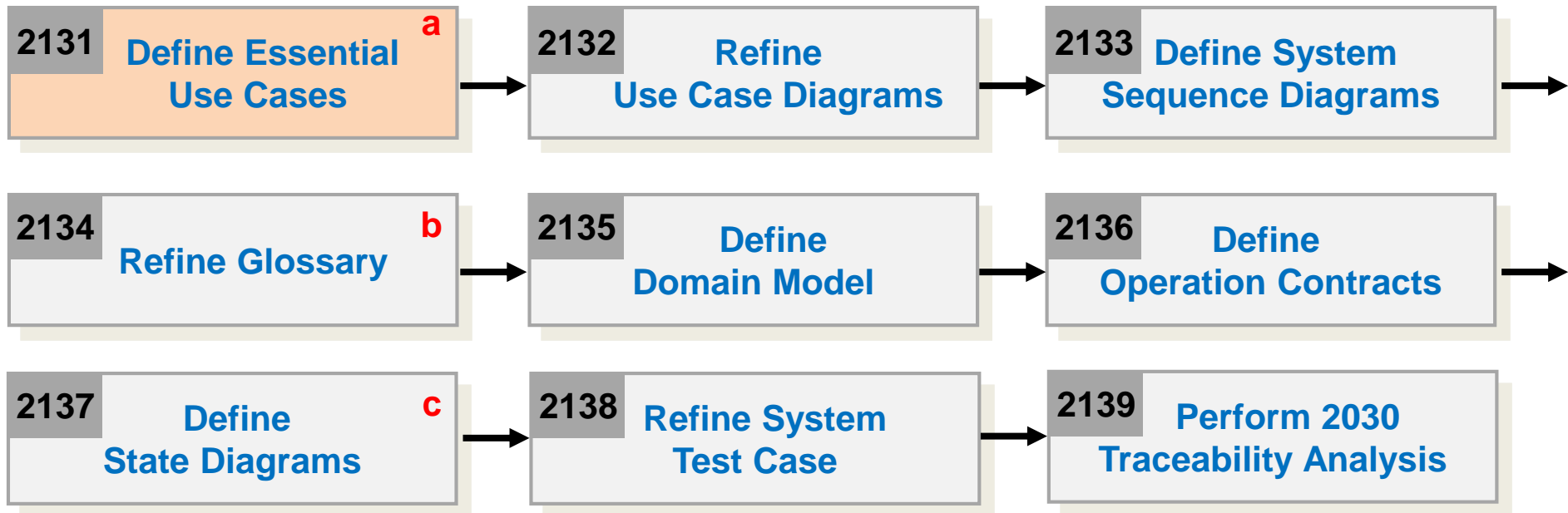
## Phase 2030. Analyze



# Activity 2031. Define Essential Use Cases

- Phase 2030 Activities

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





# Activity 2031. Define Essential Use Cases

- 1. Make Reservation

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

# Activity 2031. Define Essential Use Cases

- 2. Remove Reservation

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

# Activity 2031. Define Essential Use Cases

- 3. Lend Item

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

# Activity 2031. Define Essential Use Cases

- 4. Return Item

<b>Use Case</b>	4. Return Item
<b>Actor</b>	Librarian
<b>Purpose</b>	(As in the business use case)
<b>Overview</b>	(As in the business use case)
<b>Type</b>	Primary and Essential
<b>Cross Reference</b>	System Functions: R1.4.1, R1.4.2, R1.6 Use Cases: "Calculate Late-Return-Fee", "Notify Availability"
<b>Pre-Requisites</b>	N/A
<b>Typical Courses of Events</b>	<p>(A) : Actor, (S) : System</p> <ol style="list-style-type: none"> <li>1. (A) A librarian requests returning item</li> <li>2. (S) Check if a corresponding title exists</li> <li>3. (S) Check if a corresponding borrower exists</li> <li>4. (S) Check if a corresponding item is loaned</li> <li>5. (S) Find the borrower of the item</li> <li>6. (S) Check whether the returning due-date is over or not</li> <li>7. (S) If the returning due-date is over, invoke "Calculate Late-Return-Fee"</li> <li>8. (S) Remove the loan</li> <li>9. (S) If the item is reserved, invoke "Notify Availability"</li> </ol>
<b>Alternative Courses of Events</b>	N/A
<b>Exceptional Courses of Events</b>	Line 1: If invalid returning information is entered, indicate an error

# Activity 2031. Define Essential Use Cases

- 5. Calculate Late-Return-Fee

<b>Use Case</b>	5. Calculate Late-Return-Fee
<b>Actor</b>	b
<b>Purpose</b>	(As in the business use case)
<b>Overview</b>	(As in the business use case)
<b>Type</b>	Primary and Essential
<b>Cross Reference</b>	System Functions: R1.4.1, R1.4.2 Use Case: "Return Item"
<b>Pre-Requisites</b>	Lending due-date should be over.
<b>Typical Courses of Events</b>	(A) : Actor, (S) : System 1. (S) Compute late-return time 2. (S) Compute late-return fee 3. (S) Print the late-return fee
<b>Alternative Courses of Events</b>	N/A
<b>Exceptional Courses of Events</b>	N/A

# Activity 2031. Define Essential Use Cases

- 6. Get Replacement-Fee

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

# Activity 2031. Define Essential Use Cases

- 7. Notify Availability

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

# Activity 2031. Define Essential Use Cases

- 8. Add Title

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



# Activity 2031. Define Essential Use Cases

- 9. Remove Title

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

# Activity 2031. Define Essential Use Cases

- 10. Update Title

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

# Activity 2031. Define Essential Use Cases

- 11. Add Item

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

# Activity 2031. Define Essential Use Cases

- 12. Remove Item

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

# Activity 2031. Define Essential Use Cases

- 13. Update Item

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

# Activity 2031. Define Essential Use Cases

- 14. Add Borrower

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

# Activity 2031. Define Essential Use Cases

- 15. Remove Borrower

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

# Activity 2031. Define Essential Use Cases

- 16. Update Borrower

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



# Activity 2031. Define Essential Use Cases

- 17. Log-In

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

# Activity 2031. Define Essential Use Cases

- 18. Log-Out

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

# Activity 2031. Define Essential Use Cases

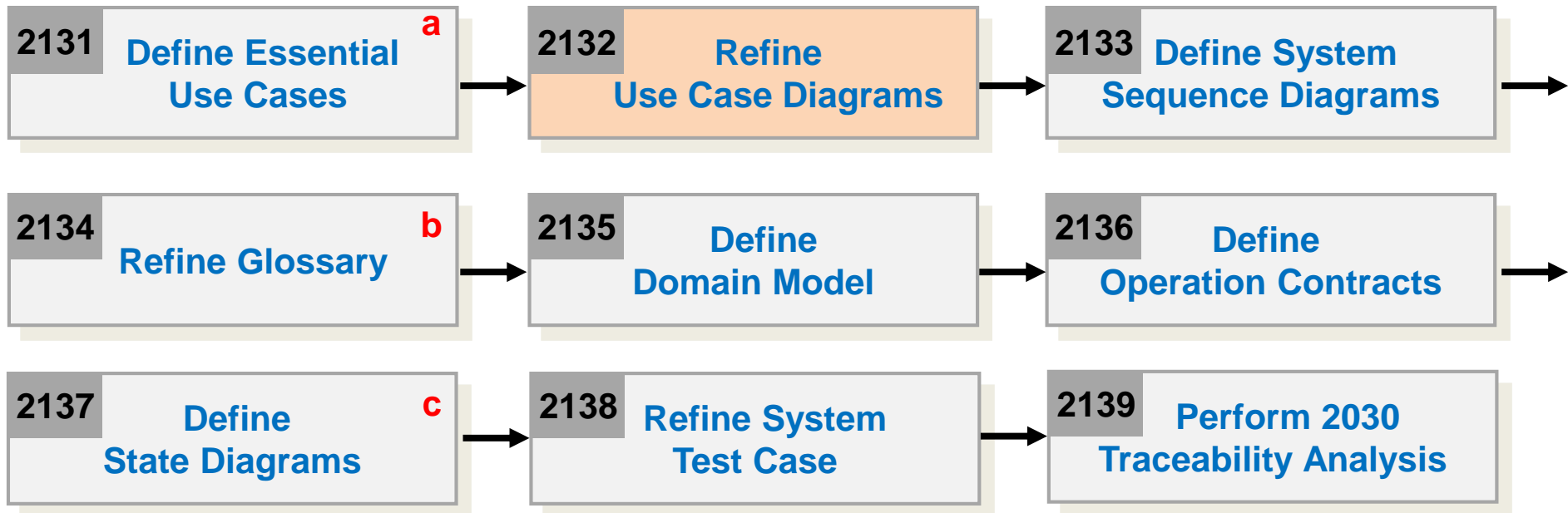
- 19. Count Loans

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

# Activity 2032. Refine Use Case Diagrams

- Phase 2030 Activities

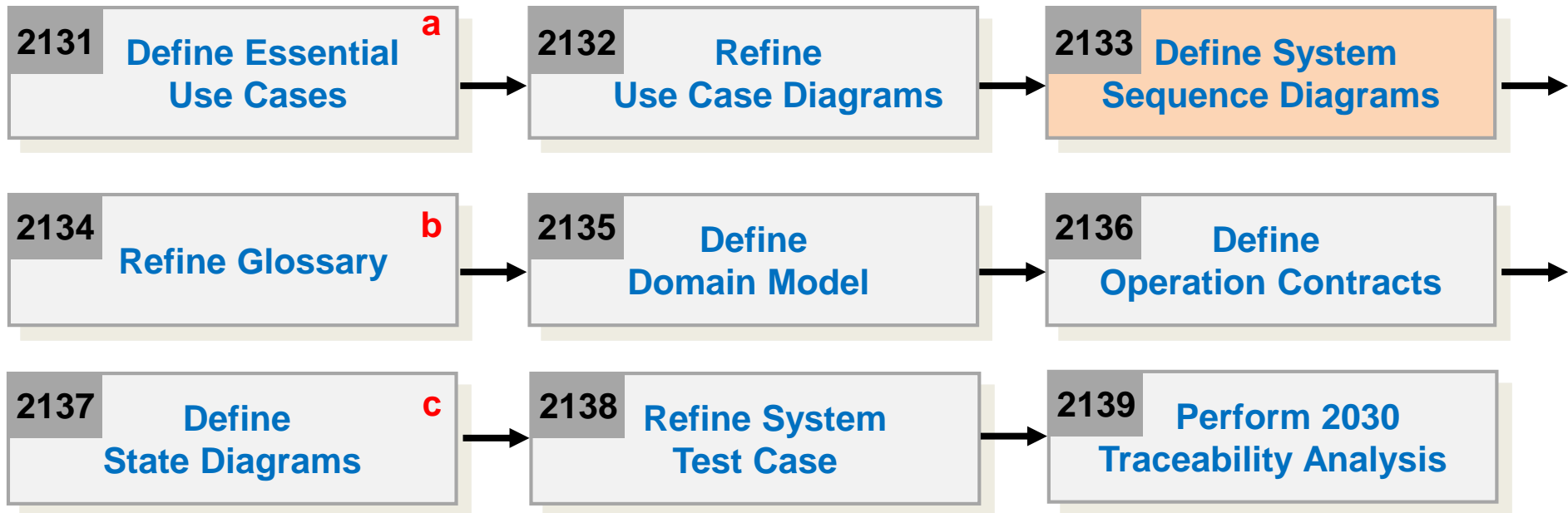
a. if not yet done  
b. ongoing  
c. optional



# Activity 2033. Define System Sequence Diagrams

- Phase 2030 Activities

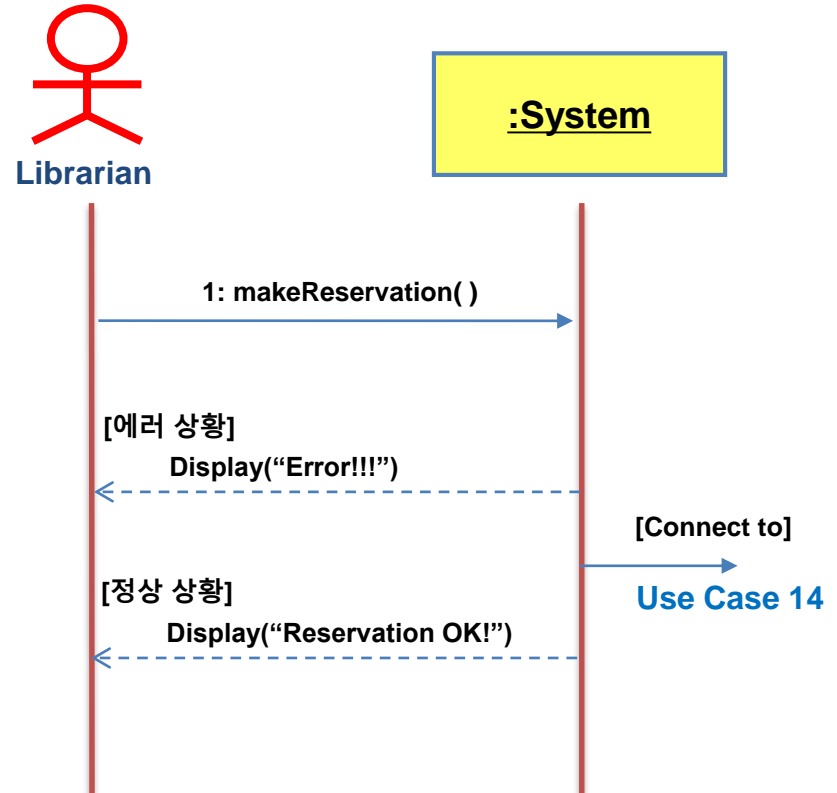
a. if not yet done  
b. ongoing  
c. optional



# Activity 2033. Define System Sequence Diagrams

## USE CASE: 1. Make Reservation

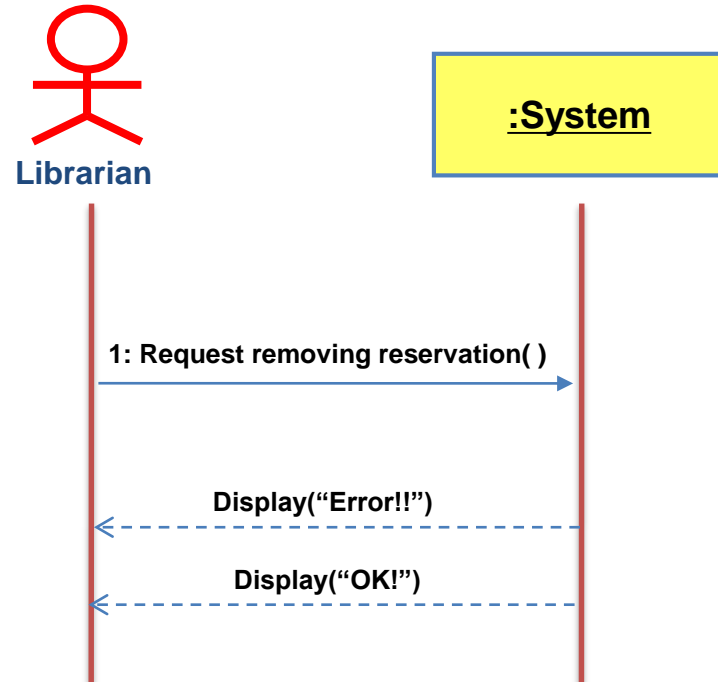
1. (A) A librarian requests the reservation of title.
2. (S) Check if corresponding title exist.
3. (S) Check if corresponding borrower exist.
4. (S) If the borrower does not exist, invoke "Add Borrower".  
(→ connect to the other Use Case)
5. (S) Create reservation information.



# Activity 2033. Define System Sequence Diagrams

## USE CASE: 2. Remove Reservation

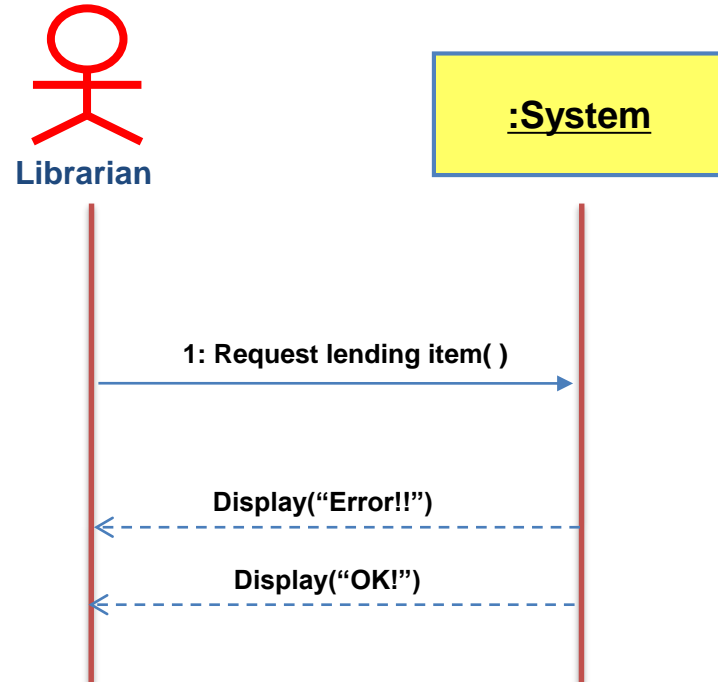
1. (A) A librarian requests removing reservation.
2. (S) Check if corresponding title exist.
3. (S) Check if corresponding borrower exist.
4. (S) Find the reservation.
5. (S) Remove the reservation.



# Activity 2033. Define System Sequence Diagrams

## USE CASE: 3. Lend Item

1. (A) A librarian requests lending item.
2. (S) Check if corresponding title exist.
3. (S) Check if corresponding item is available.
4. (S) If the item was reserved, invoke "Remove Reservation".
5. (S) Check if corresponding borrower exist.
6. (S) If the borrower does not exist, invoke "Add Borrower".
7. (A) Create new loan.

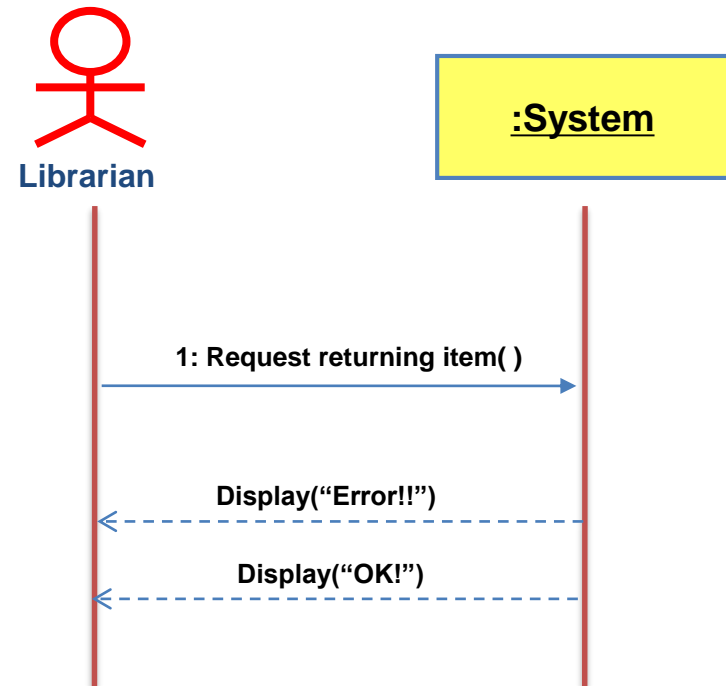




# Activity 2033. Define System Sequence Diagrams

## USE CASE: 4. Return Item

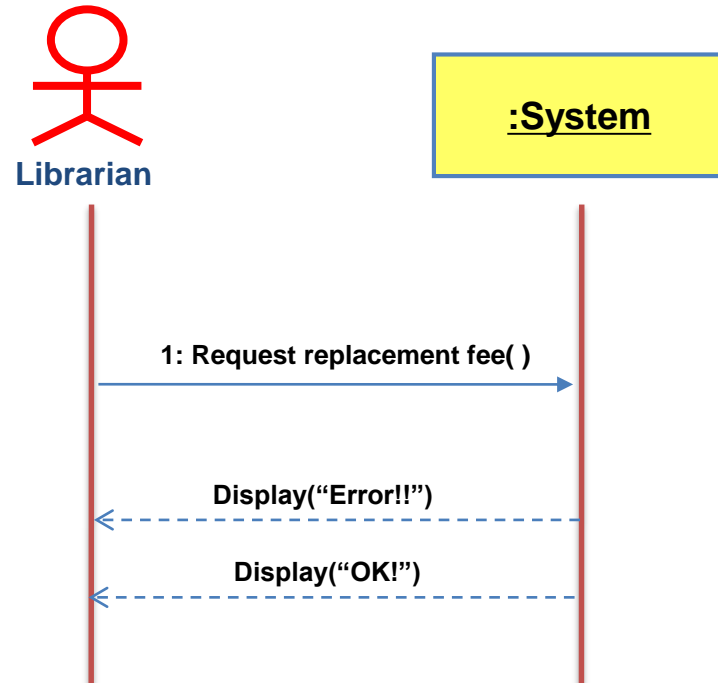
1. (A) A librarian requests returning item
2. (S) Check if a corresponding title exists
3. (S) Check if a corresponding borrower exists
4. (S) Check if a corresponding item is loaned
5. (S) Find the borrower of the item
6. (S) Check whether the returning due-date is over or not
7. (S) If the returning due-date is over, invoke "Calculate Late-Return-Fee"
8. (S) Remove the loan
9. (S) If the item is reserved, invoke "Notify Availability"



# Activity 2033. Define System Sequence Diagrams

## USE CASE: 6. Get Replacement Fee

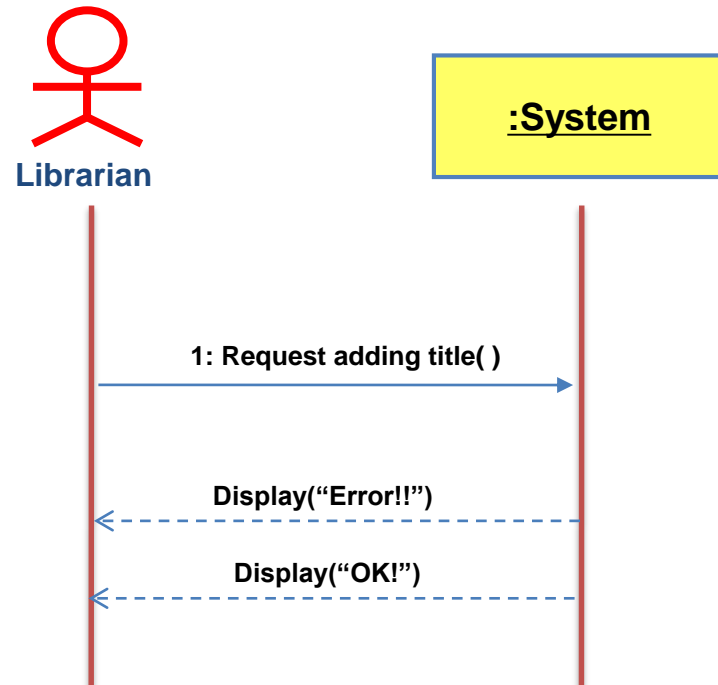
1. (A) A librarian inputs a title's information
2. (S) Check if a corresponding title exists
3. (S) Find the price of the title
4. (S) Compute replacement-fee



# Activity 2033. Define System Sequence Diagrams

## USE CASE: 8. Add Title

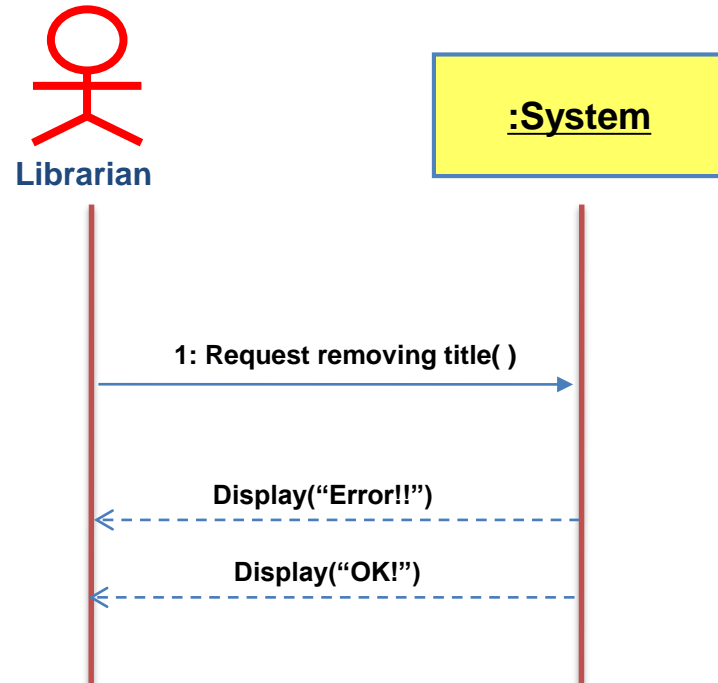
1. (A) A librarian inputs a title's information
2. (S) Check if a corresponding title exists
3. (S) Add a new title
4. (S) Invoke "Add Item"



# Activity 2033. Define System Sequence Diagrams

## USE CASE: 9. Remove Title

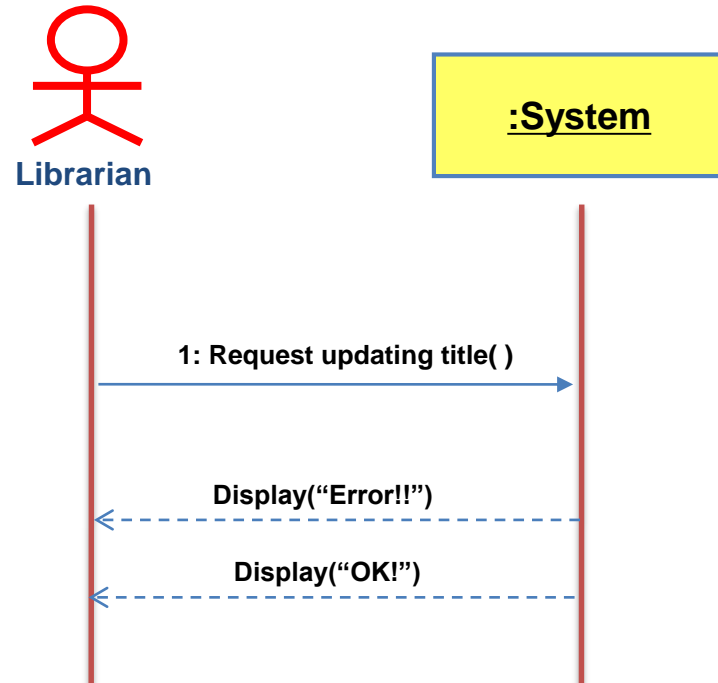
1. (A) A librarian inputs a title's information to deleted
2. (S) Check if a corresponding title exists
3. (S) Remove the items of the title
4. (A) Remove the title



# Activity 2033. Define System Sequence Diagrams

## USE CASE: 10. Update Title

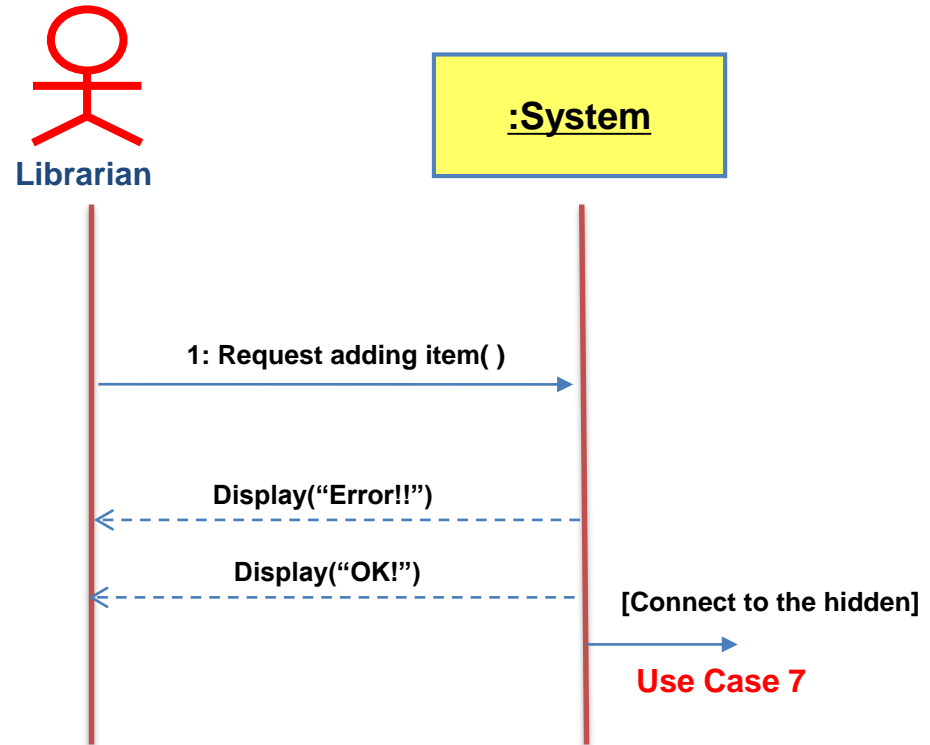
1. (A) A librarian inputs a title's information to change
2. (S) Check if a corresponding title exists
3. (S) Update the title's information



# Activity 2033. Define System Sequence Diagrams

## USE CASE: 11. Add Item

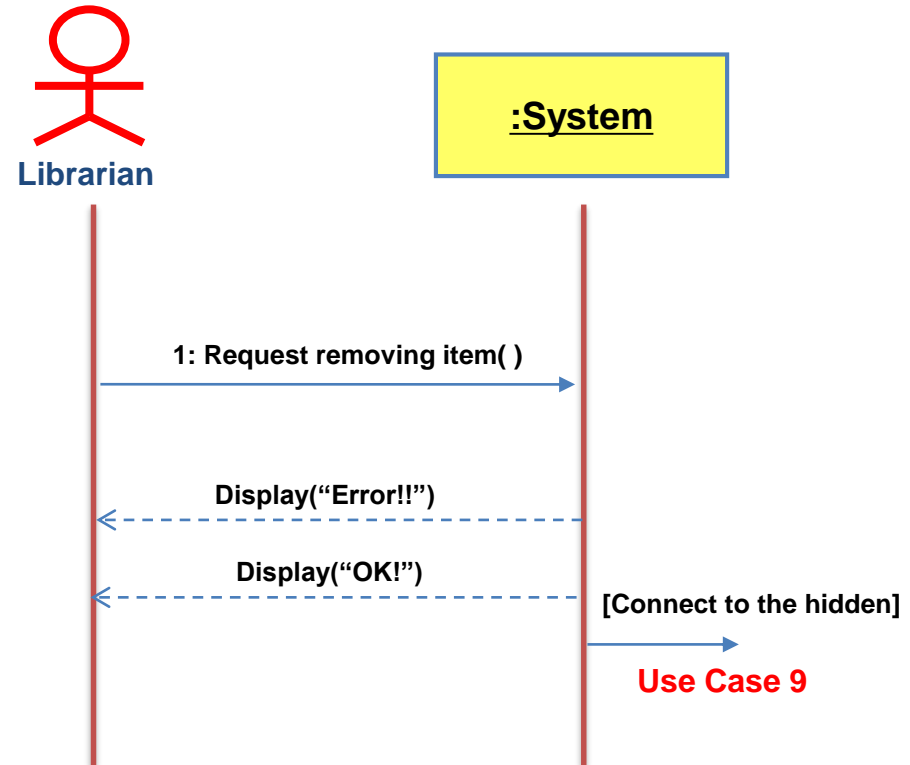
1. (A) A librarian inputs a item to add
2. (S) Check if a corresponding title exists
3. (S) Add the item
4. (S) Invoke “Notify Availability”



# Activity 2033. Define System Sequence Diagrams

## USE CASE: 12. Remove Item

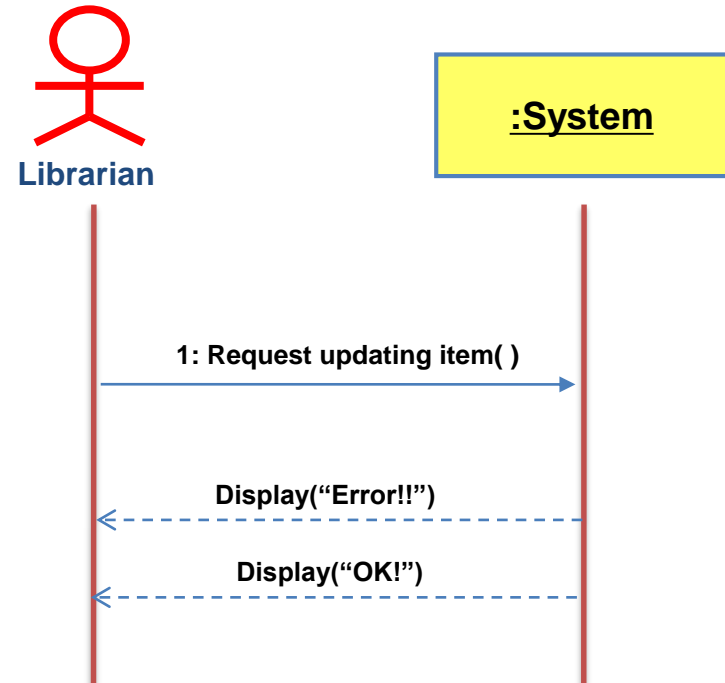
1. (A) A librarian inputs an item's information to remove
2. (S) Check if a corresponding title exists
3. (S) Check if a corresponding item exists
4. (S) Remove the item
5. (S) If there is no remaining item, invoke "Remove Title"



# Activity 2033. Define System Sequence Diagrams

## USE CASE: 13. Update Item

1. (A) A librarian inputs an item's information to update
2. (S) Check if a corresponding title exists
3. (S) Check if a corresponding item exists
4. (S) Update the item's information

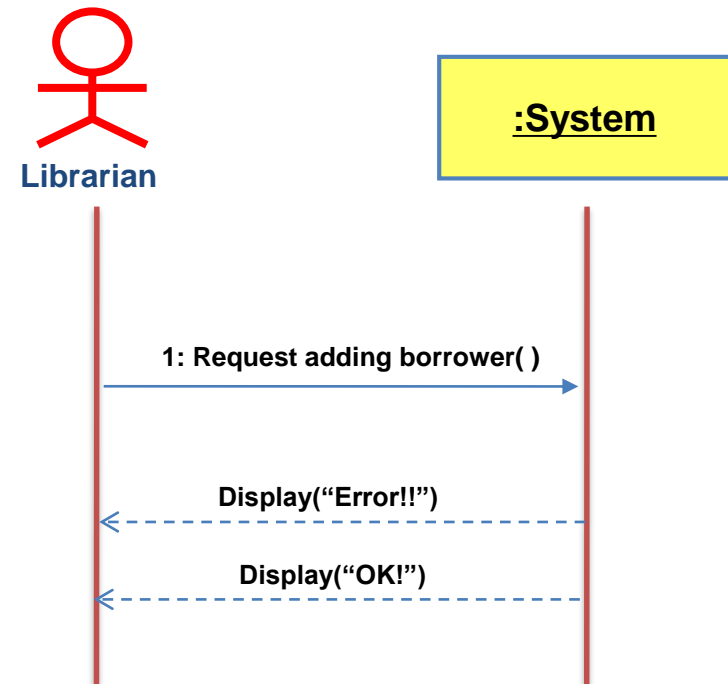




# Activity 2033. Define System Sequence Diagrams

## USE CASE: 14. Add Borrower

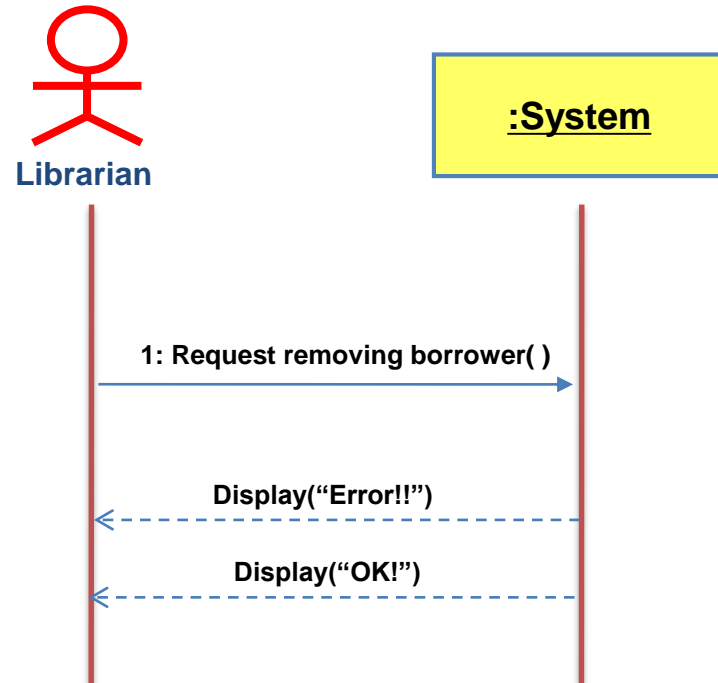
1. (A) A librarian inputs borrower's information such as SSN, name, address, zip code, phone number, and age.
2. (S) Check if the corresponding borrower exists
3. (S) Add New borrower



# Activity 2033. Define System Sequence Diagrams

## USE CASE: 15. Remove Borrower

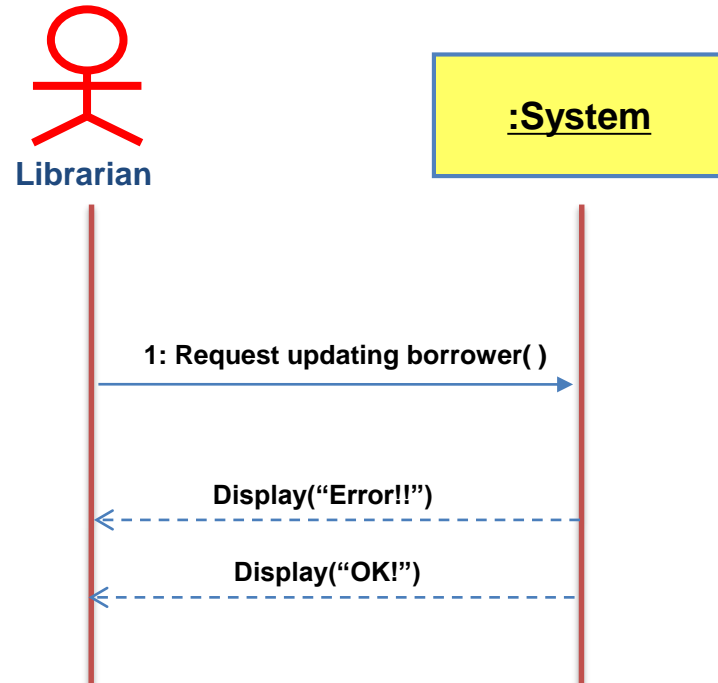
1. (A) A librarian inputs a borrower's information to remove
2. (S) Check if a corresponding borrower exists
3. (S) If there is a loan of the borrower, remove the loan.
4. (S) Remove the borrower's information



# Activity 2033. Define System Sequence Diagrams

## USE CASE: 16. Update Borrower

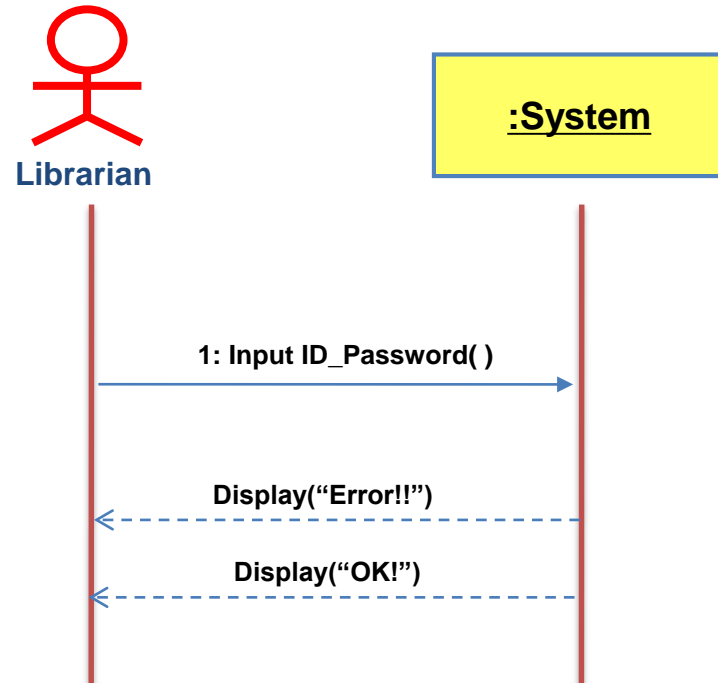
1. (A) A librarian inputs a borrower's information to change
2. (S) Check if a corresponding borrower exists
3. (S) Update the borrower's information



# Activity 2033. Define System Sequence Diagrams

## USE CASE: 17. Log-In

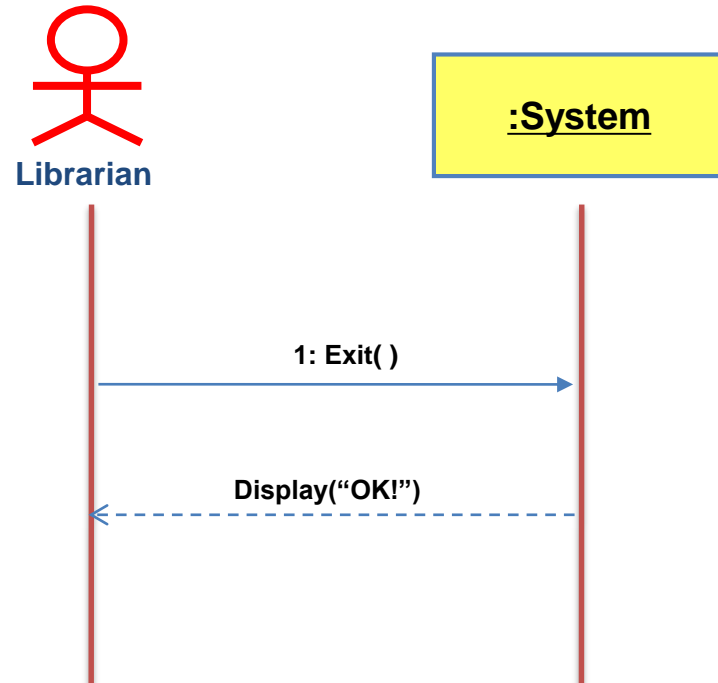
1. (A) A librarian enters his(her) user name and password into the system
2. (S) Check if the user name and password are correct



# Activity 2033. Define System Sequence Diagrams

## USE CASE: 17. Log-Out

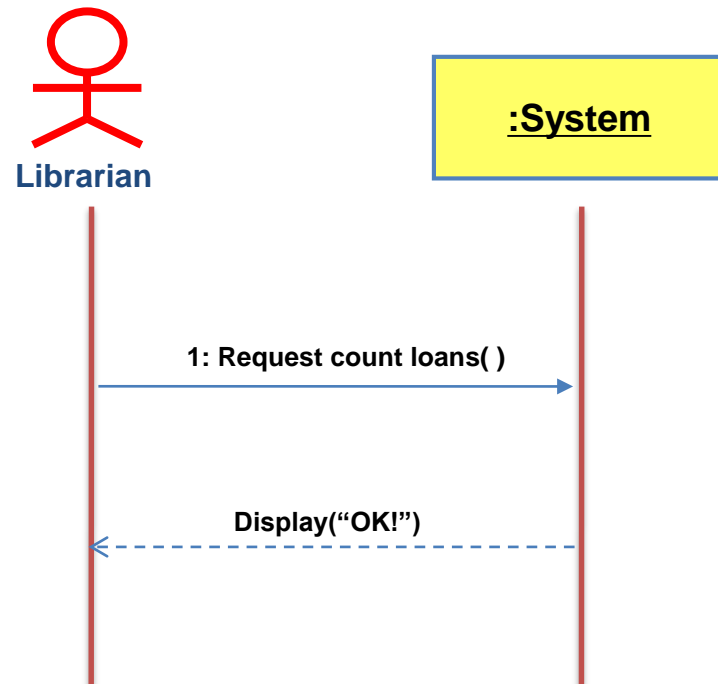
1. (A) A librarian exits the system
2. (S) Log-off the librarian's information



# Activity 2033. Define System Sequence Diagrams

## USE CASE: 18. Count Loans

1. (A) A librarian requests total counts of titles checked out
2. (S) Find loan information
3. (S) Calculate total counts of titles checked out
4. (S) Print total counts.

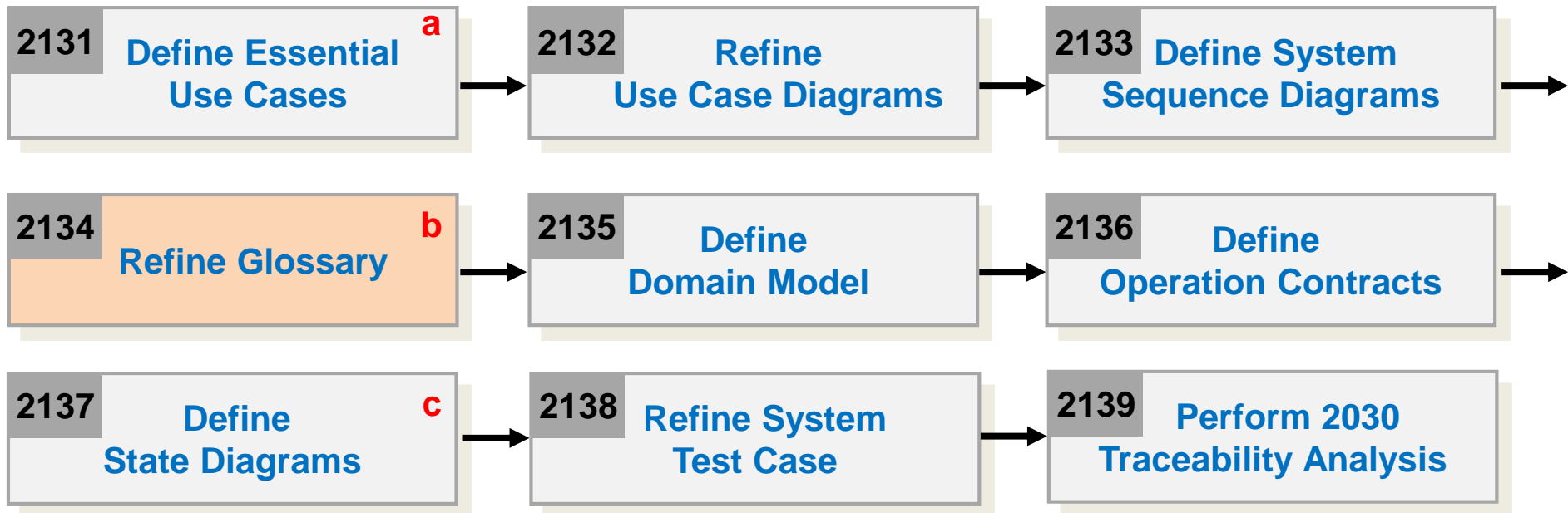


Use Case	Name of Actor-Activated Event	System Operations
1. Make Reservation	1: Request making reservation( )	1. makeReservation( )
2. Remove Reservation	1: Request removing reservation( )	2. removeReservation( )
3. Lend Item	1: Request lending item( )	3. LendItem( )
4. Return Item	1: Request returning 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 2034. Refine Glossary

- Phase 2030 Activities

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





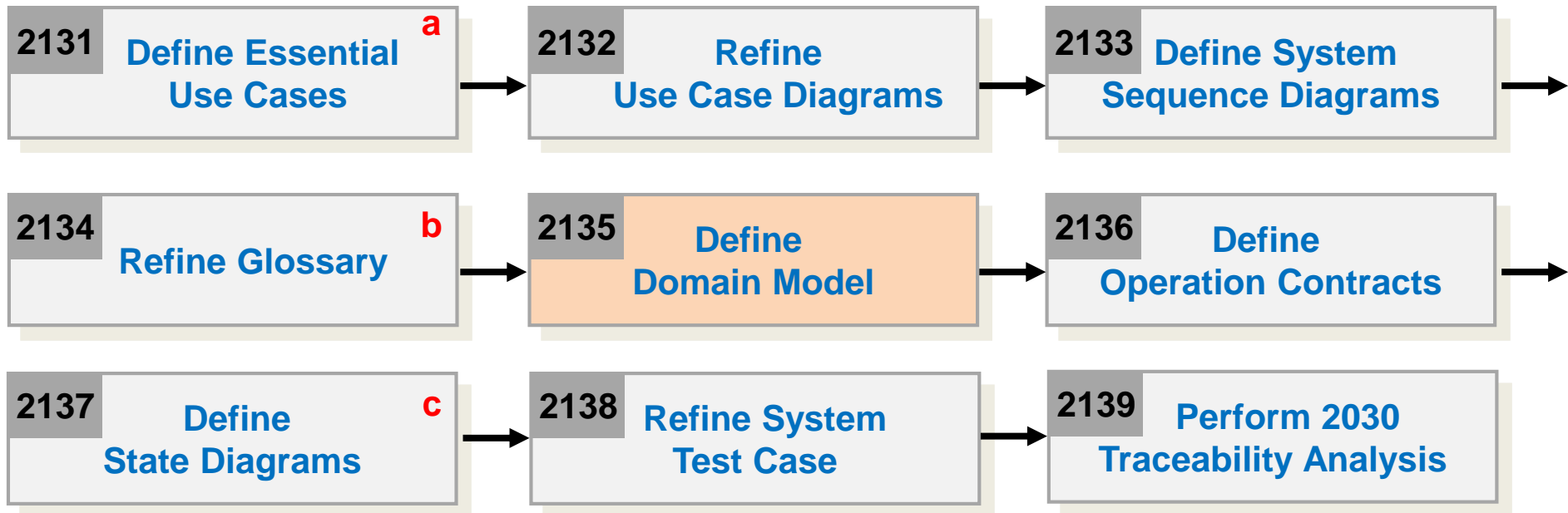
# Activity 2034. Refine Glossary

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 Domain Model

- Phase 2030 Activities

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

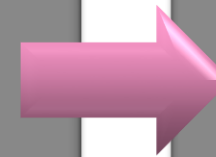


# Activity 2035. Define Domain Model

- Step 1. List concepts
  - Guideline 2

## Main Concerns

1. Borrower requests the reservation of the title
2. Librarian receives the request and reserve the item of the title
3. Borrower can requests loan of the title
4. Librarian can manage the title such as add, remove, update
5. Item of the tile is also managed by librarian
6. Title consists of book and magazine
7. Librarian can manage the borrower information
8. Identifying librarian in system is supplied by login, logout function
9. Loan fee is calculated in system



Borrower  
Reservation  
Title  
Item  
Book  
Magazine  
Manage  
Librarian  
Certification  
Fee

# Activity 2035. Define Domain Model

- Step 2. Assign class names into concepts
  - Title
  - Librarian
  - Book
  - Magazine
  - Loan
  - Reservation
  - Borrower
  - Item

# Activity 2035. Define Domain Model

- Step 3. Identify associations according to association categories

Association Category	Associations
A is known/logged/recorded/reported/captured in B	Item – Loan Item – Title Loan – Borrower 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 Borrower – Reservation
A is an organization submit of B	Book – Title Magazine – Title

# Activity 2035. Define Domain Model

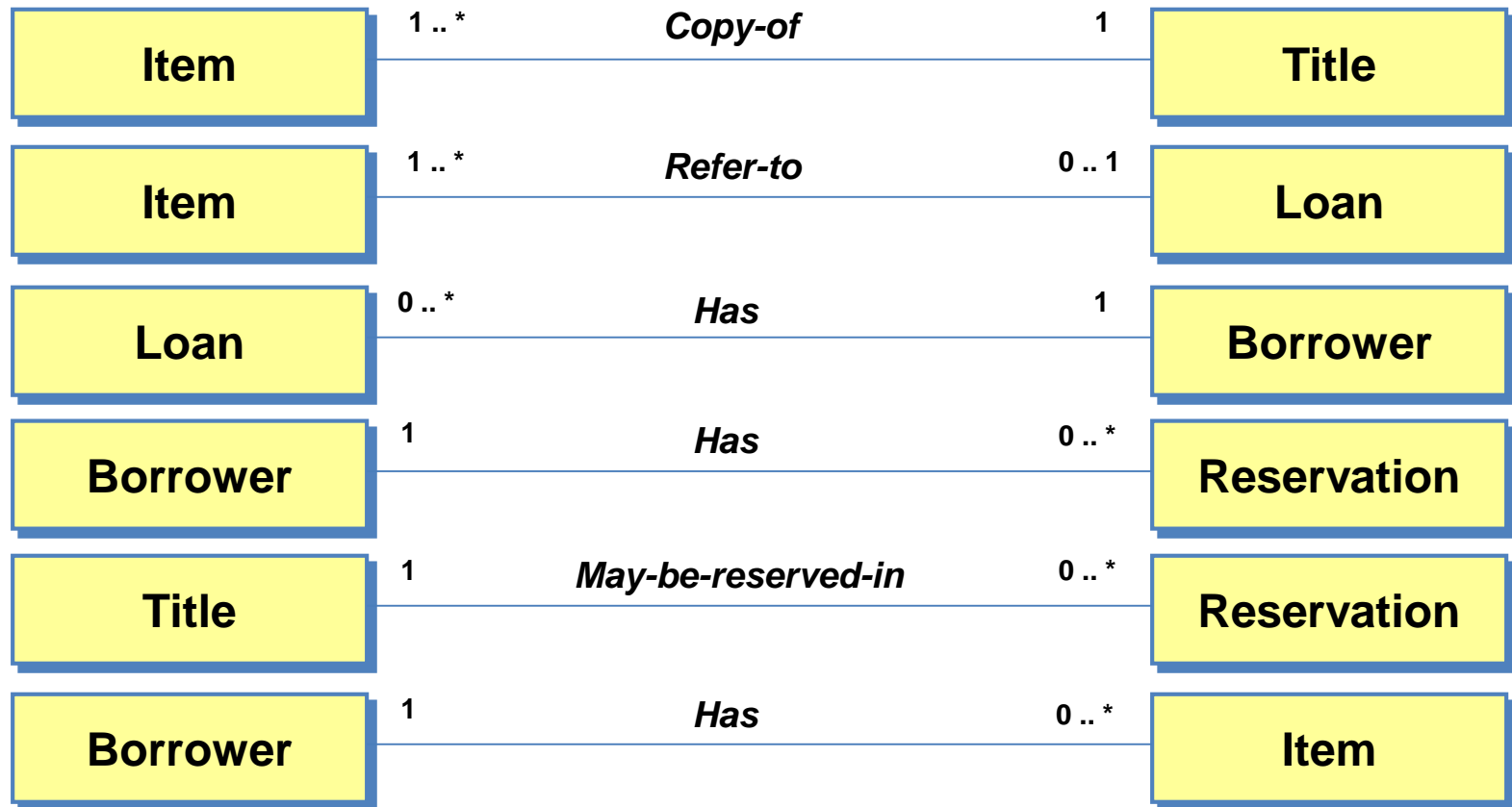
- Step 4. Assign priorities into identified associations

Association Name	Priority
Item – Title	High

- Step 5. 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 2035. Define Domain Model

- Step 6. Add multiplicity into the ends of an associations



# Activity 2035. Define Domain Model

- Step 7. Identify attributes by reading

<b>&lt;&lt;Business Object&gt;&gt; Item</b>
<b>ID : Integer</b> <b>available : Boolean</b>

<b>&lt;&lt;Business Object&gt;&gt; Title</b>
<b>name : String</b> <b>isbn : String</b> <b>count : Integer</b> <b>price : Float</b> <b>publisher : String</b> <b>lending time : Integer</b>

<b>&lt;&lt;Business Object&gt;&gt; Reservation</b>
<b>date : Date</b>

<b>&lt;&lt;Business Object&gt;&gt; Book</b>
<b>author: String</b>

<b>&lt;&lt;Business Object&gt;&gt; Loan</b>
<b>date: Date</b> <b>late-return-fee : Integer</b>

<b>&lt;&lt;Business Object&gt;&gt; Librarian</b>
<b>name : String</b> <b>userID : String</b> <b>password : string</b>

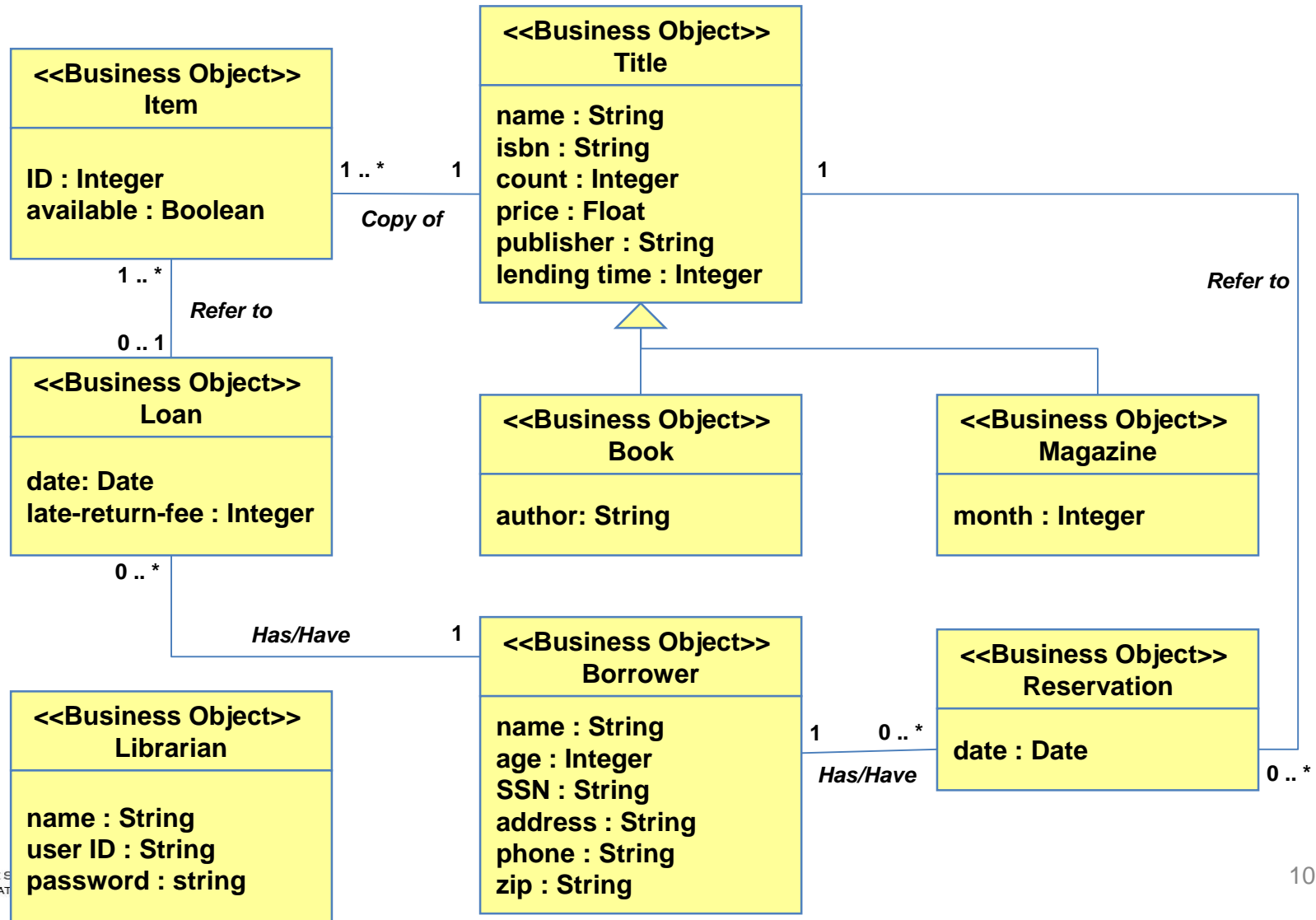
<b>&lt;&lt;Business Object&gt;&gt; Magazine</b>
<b>month : Integer</b>

<b>&lt;&lt;Business Object&gt;&gt; Borrower</b>
<b>name : String</b> <b>age : Integer</b> <b>SSN : String</b> <b>address : String</b> <b>phone : String</b> <b>zip : String</b>



# Activity 2035. Define Domain Model

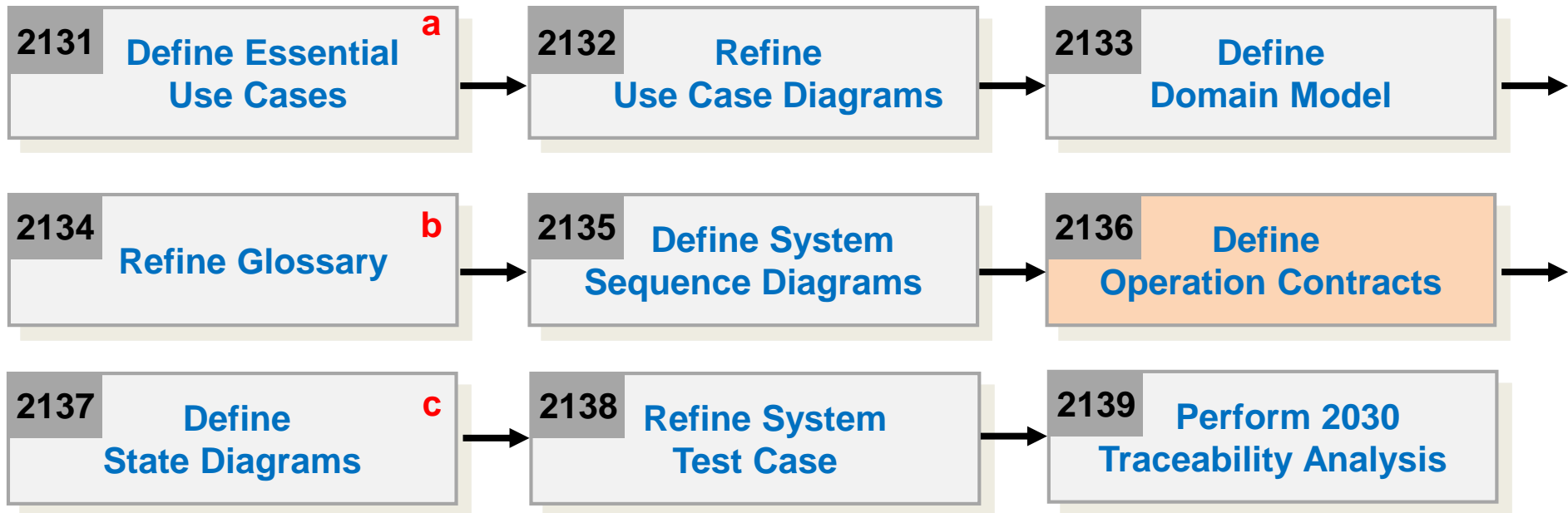
- Step 8. Draw them in a conceptual class diagram



# Activity 2036. Define Operation Contracts

- Phase 2030 Activities

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



# Activity 2036. Define Operation Contracts

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

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

# Activity 2036. Define Operation Contracts

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

# Activity 2036. Define Operation Contracts

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

# Activity 2036. Define Operation Contracts

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

# Activity 2036. Define Operation Contracts

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



# Activity 2036. Define Operation Contracts

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

# Activity 2036. Define Operation Contracts

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

# Activity 2036. Define Operation Contracts

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

# Activity 2036. Define Operation Contracts

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

# Activity 2036. Define Operation Contracts

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

# Activity 2036. Define Operation Contracts

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

# Activity 2036. Define Operation Contracts

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

# Activity 2036. Define Operation Contracts

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



# Activity 2036. Define Operation Contracts

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

# Activity 2036. Define Operation Contracts

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

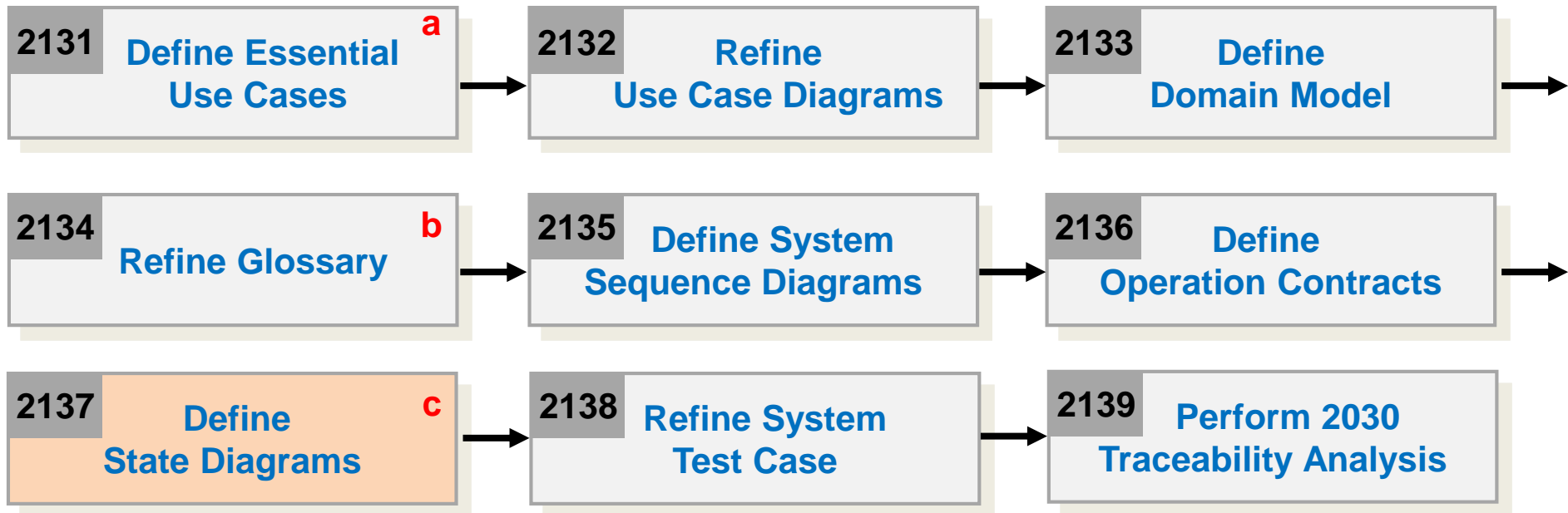
# Activity 2036. Define Operation Contracts

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

# Activity 2037. Define State Diagrams

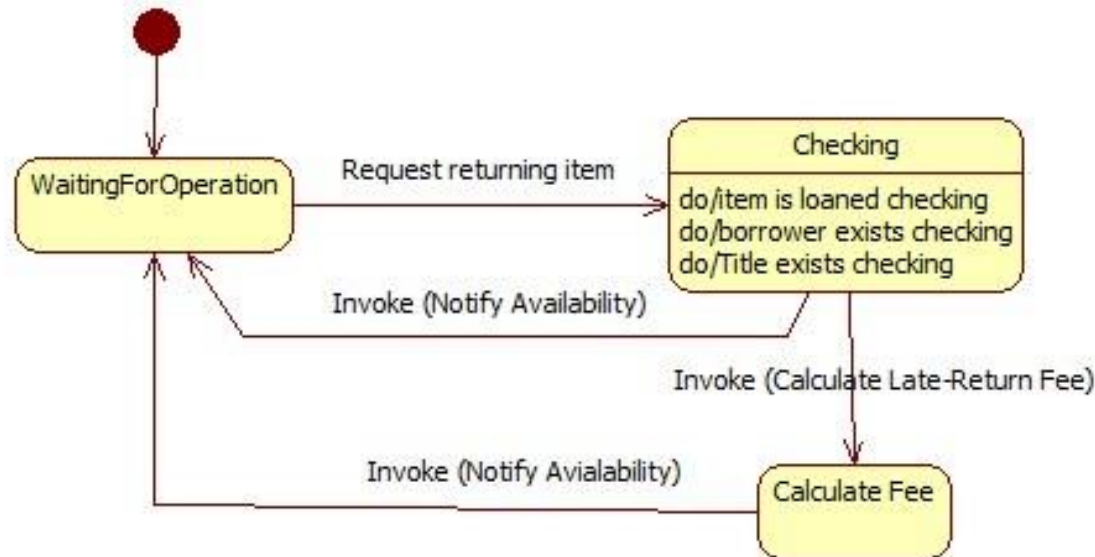
- Phase 2030 Activities

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



# Activity 2037. Define State Diagrams

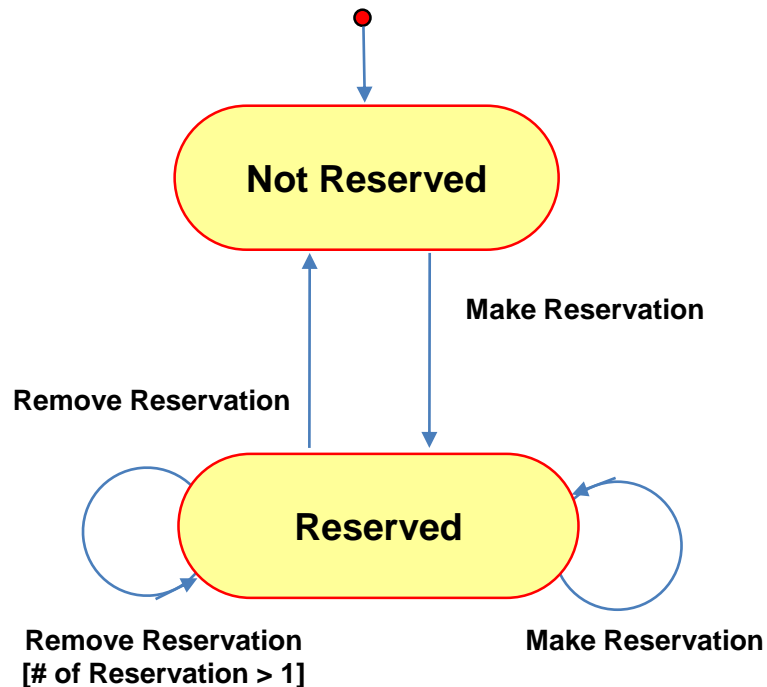
- State Diagram for Use case



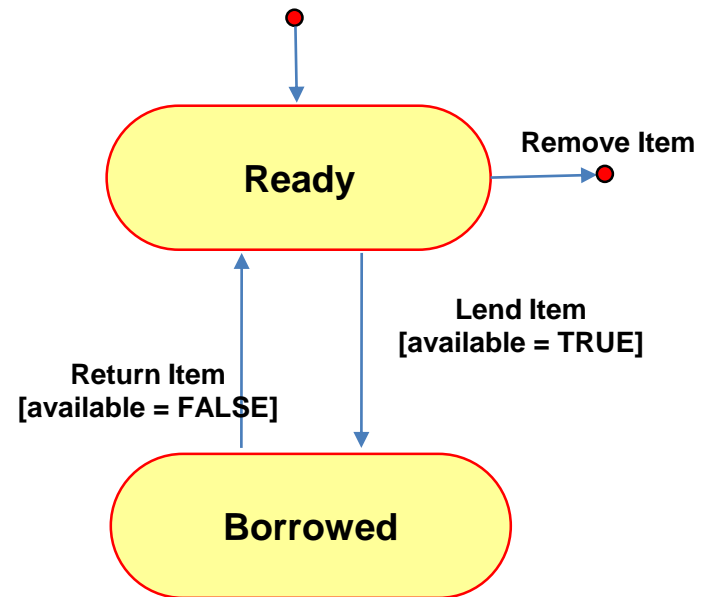
# Activity 2037. Define State Diagrams

- State Diagram for Domain Model

< State Diagram for “Title” >



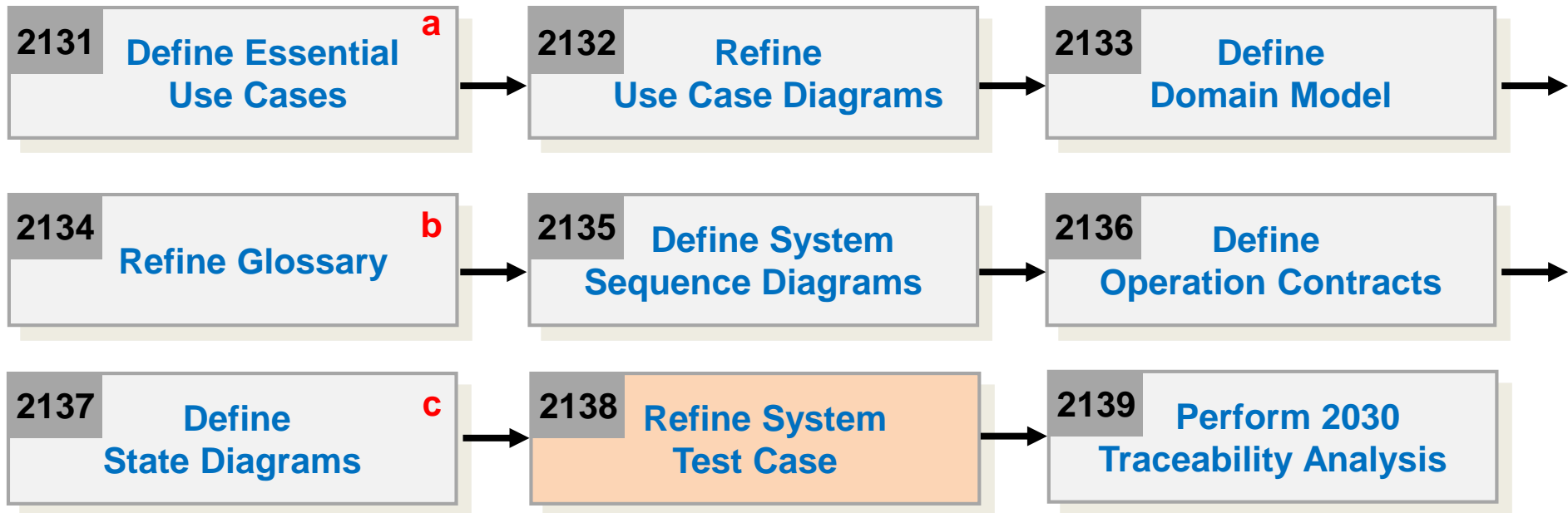
< State Diagram for “Item” >



# Activity 2038. Refine System Test Case

- Phase 2030 Activities

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



# Phase 2038. Refine System Test Case

- Step 1. Identify important requirements

Ref. #	Function	Category
R1.1	<b>Make reservation</b>	Evident
R1.2	<b>Remove reservation</b>	Evident
R1.3	<b>Lend Item</b>	Evident
R1.4.1	<b>Return title</b>	Evident
R1.4.2	Calculate Late-Return-Fee	Hidden
R1.5	<b>Calculate Replacement Fee</b>	Evident
R1.6	Notify Availability	Hidden
R2.1	<b>Add title</b>	Evident
R2.2	<b>Remove title</b>	Evident
R2.3	<b>Update title</b>	Evident
R2.4	<b>Add items</b>	Evident
R2.5	<b>Remove item</b>	Evident
R2.6	<b>Update item</b>	Evident
R3.1	<b>Add borrower</b>	Evident
R3.2	<b>Remove borrower</b>	Evident
R3.3	<b>Update borrower</b>	Evident
R4.1	<b>Validates system access</b>	Evident
R5.1	<b>Compute total # of items checked out</b>	Evident



# Activity 2038. Refine System Test Case

- Step 2. Develop system test cases with various system testing techniques
  - First, brute force testing

No.	Tests	Description
1	<b>Make reservation</b>	Correct한 borrower가 correct한 title 예약
2	<b>Make reservation</b>	Correct한 borrower가 incorrect한 title 예약
3	<b>Make reservation</b>	Correct한 borrower가 대여중인 title 예약
4	<b>Make reservation</b>	Incorrect한 borrower가 예약
5	<b>Remove reservation</b>	Correct한 borrower가 예약 취소
6	<b>Remove reservation</b>	Incorrect한 borrower가 예약 취소
7	<b>Lend Item</b>	Correct한 borrower가 대여 가능한 title 대여
8	<b>Lend Item</b>	Correct한 borrower가 incorrect한 title 대여
9	<b>Lend Item</b>	Correct한 borrower가 모두 대여중인 title 대여
10	<b>Lend Item</b>	Incorrect한 borrower가 대여
11	<b>Return title</b>	Borrower가 title 반납
12	<b>Return title</b>	Borrower가 연체된 title 반납
13	<b>Add title</b>	새 title 추가
14	<b>Remove title</b>	기존의 title 제거
15	<b>Remove title</b>	존재하지 않는 title 제거
16	<b>Update title</b>	Title 정보 update
17	<b>Add item</b>	Title item 추가
18	<b>Add item</b>	존재하지 않는 title의 item추가

# Activity 2038. Refine System Test Case

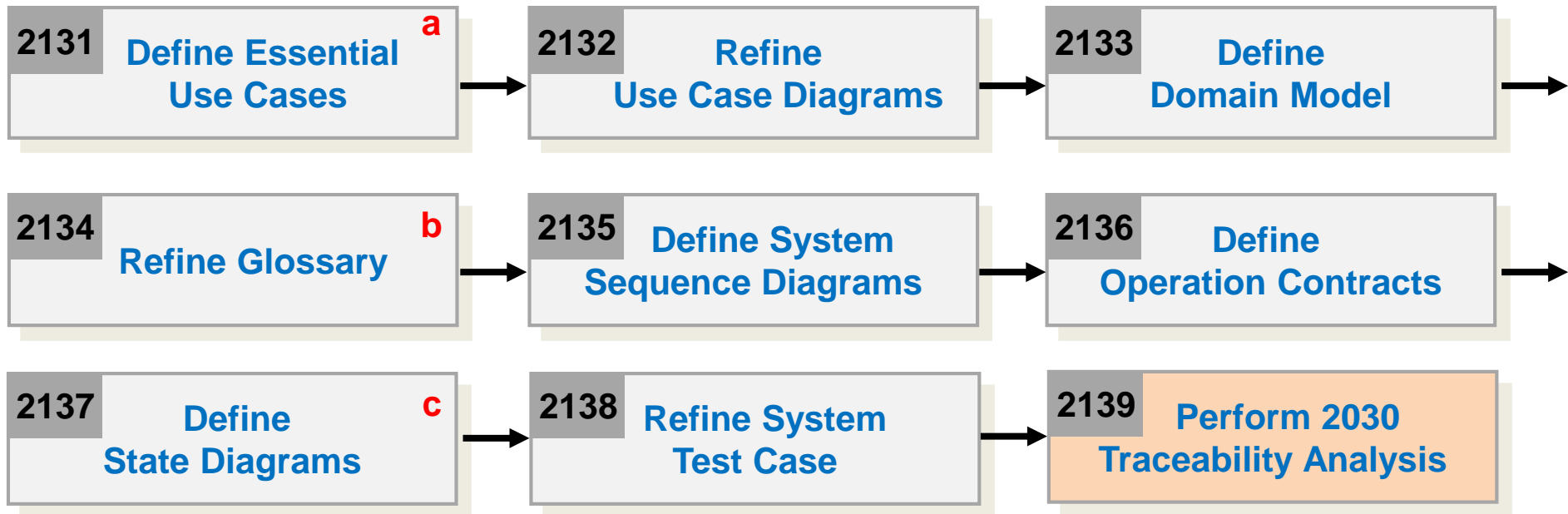
- Step 2. Develop system test cases with various system testing techniques
  - First, brute force testing

No.	Tests	Description
19	Remove item	Title의 item제거
20	Remove item	존재하지 않는 title의 item제거
21	Update item	올바른 item의 정보 update
22	Update item	Title에 존재하지 않는 item update
23	Add borrower	Borrower 추가
24	Remove borrower	Borrower 삭제
25	Update borrower	기존의 borrower update
26	Update borrower	삭제된 borrower update
27	Validates system access	Correct id/pw로 로그인
28	Validates system access	Incorrect id/pw로 로그인
29	Validates system access	로그아웃
30	Compute total # of items checked out	계산 시도

# Activity 2039. Perform 2030 Traceability Analysis

- Phase 2030 Activities

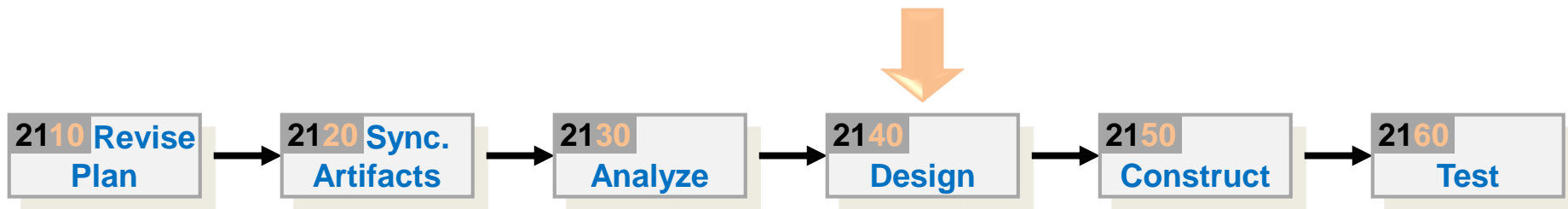
a. if not yet done  
b. ongoing  
c. optional



# Activity 2039. Perform 2030 Traceability Analysis

System Function		Essential Use Case		Operation in sequence diagram
Make reservation	→	Make Reservation	→	makeReservation( )
Remove reservation	→	Remove Reservation	→	removeReservation( )
Lend Item	→	Lend Item	→	LendItem( )
Return title	→	Return Title	→	returnItem( )
Calculate Late-Return-Fee	→	Calculate Late-Return-Fee	→	getReplacementFee( )
Calculate Replacement Fee	→	Get Replacement Fee	→	addTitle( )
Notify Availability	→	Notify Availability	→	removeTitle( )
Add title	→	Add Title	→	updateTitle( )
Remove title	→	Remove Title	→	addItem( )
Update title	→	Update Title	→	removeItem( )
Add items	→	Add Item	→	updateItem( )
Remove item	→	Remove Item	→	addBorrower( )
Update item	→	Update Item	→	removeBorrower( )
Add borrower	→	Add Borrower	→	updateBorrower( )
Remove borrower	→	Remove Borrower	→	log-In( )
Update borrower	→	Update Borrower	→	log-Out( )
Validates system access	→	Log-IN	→	countLoans( )
Compute total # of items checked out	→	Log-Out	→	
	→	Count Loans	→	

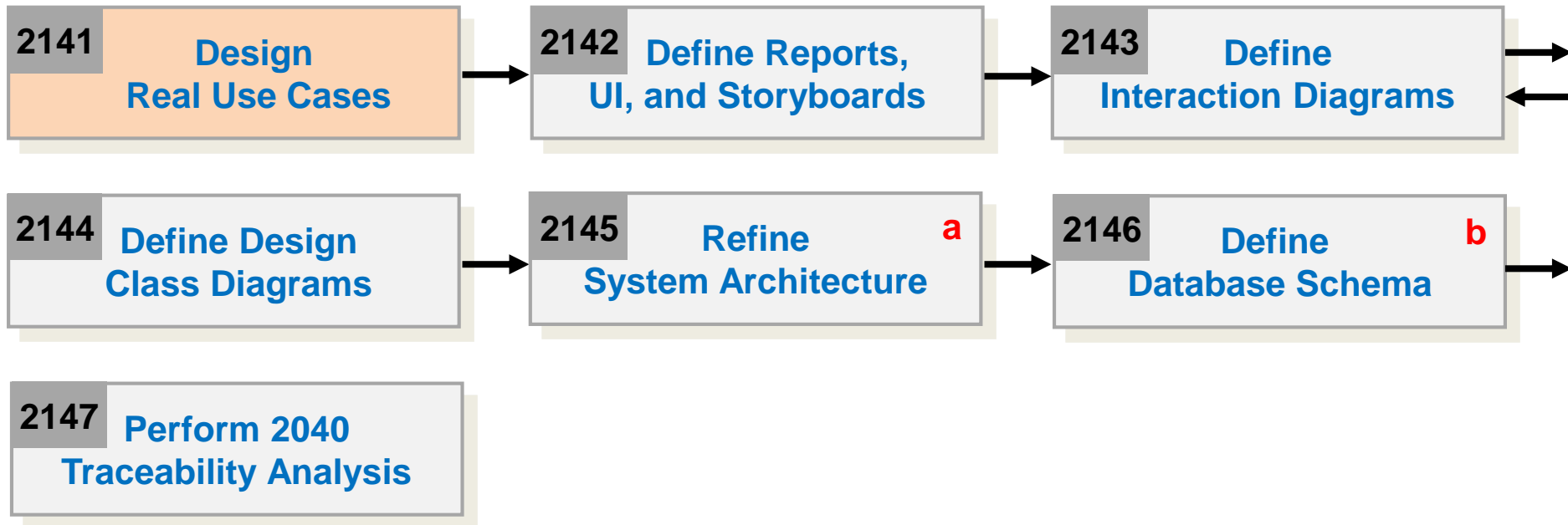
## Phase 2040. Design



# Phase 2041. Design Real Use Cases

- 7 Activities

a. Varied order  
b. optional



# Phase 2041. Design Real Use Cases

- Make Reservation

<b>Use Case</b>	<b>1. Make Reservation</b>
<b>Actor</b>	Librarian
<b>Purpose</b>	Create a new reservation
<b>Overview</b>	(As in the business use case)
<b>Type</b>	Primary and Real
<b>Cross Reference</b>	System Functions: R1.1, R3.1 Use Case: "Add Borrower"
<b>Pre-Requisites</b>	A borrower should be registered.
<b>Typical Courses of Events</b>	<p>(A) : Actor, (S) : System</p> <ol style="list-style-type: none"> <li>1. (A) A librarian inputs an isbn and ssn of the title</li> <li>2. (S) Find a corresponding title</li> <li>3. (S) Find a corresponding borrower</li> <li>4. (S) Create a new reservation</li> <li>5. (S) Store the new reservation</li> <li>6. (S) Increase reservationCount in the borrower</li> <li>7. (S) Increase reservationCount in the title</li> </ol>
<b>Alternative Courses of Events</b>	N/A
<b>Exceptional Courses of Events</b>	<p>Line 2: If the title does not exist, display an error message.</p> <p>Line 3: If the borrower does not exist, display an error message.</p>

# Phase 2041. Design Real Use Cases

- Remove Reservation

<b>Use Case</b>	<b>2. Remove Reservation</b>
<b>Actor</b>	Librarian
<b>Purpose</b>	Remove a reservation information
<b>Overview</b>	(As in the business use case)
<b>Type</b>	Primary and Real
<b>Cross Reference</b>	System Functions: R1.2, R1.3 Use Case: "Lend Item"
<b>Pre-Requisites</b>	A borrower should be registered. A title should have been reserved.
<b>Typical Courses of Events</b>	(A) : Actor, (S) : System 1. (A) A librarian inputs an isbn of the title 2. (S) Find a corresponding reservation 3. (S) Remove the reservation 4. (S) Decrease reservationCount of the borrower 5. (S) Decrease reservationCount of the title
<b>Alternative Courses of Events</b>	N/A
<b>Exceptional Courses of Events</b>	Line 2: If the reservation doe not exist, display an error message.



# Phase 2041. Design Real Use Cases

- Lend Item

<b>Use Case</b>	<b>3. Lent Item</b>
<b>Actor</b>	Librarian
<b>Purpose</b>	Lend items to a borrower
<b>Overview</b>	(As in the business use case)
<b>Type</b>	Primary and Real
<b>Cross Reference</b>	System Functions: R1.3, R1.2, R3.1 Use Cases: "Remove Reservation", "Add Borrower"
<b>Pre-Requisites</b>	An item should exist.
<b>Typical Courses of Events</b>	<p>(A) : Actor, (S) : System</p> <ol style="list-style-type: none"> <li>1. (A) A librarian inputs an item's ID and ssn of the borrower</li> <li>2. (S) Find a corresponding borrower</li> <li>3. (S) Find a corresponding item</li> <li>4. (S) Create a new loan</li> <li>5. (S) Store the new loan</li> <li>6. (S) Set validLoan to true</li> <li>7. (S) Increase loanCount of borrower</li> <li>8. (S) Set available to false</li> <li>9. (S) Decrease AvailableCount of the title</li> </ol>
<b>Alternative Courses of Events</b>	N/A
<b>Exceptional Courses of Events</b>	Line 2: If the borrower does not exist, invoke "Add Borrower" use case.

# Phase 2041. Design Real Use Cases

- Return Item

<b>Use Case</b>	<b>4. Return Item</b>	
<b>Actor</b>	Librarian	
<b>Purpose</b>	Return items loaned	
<b>Overview</b>	(As in the business use case)	
<b>Type</b>	Primary and Real	
<b>Cross Reference</b>	System Functions: R1.4.1, R1.4.2, R1.6 Use Cases: "Calculate Late-Return-Fee", "Notify Availability"	
<b>Pre-Requisites</b>	An item should have been loaned.	
<b>Typical Courses of Events</b>	(A) : Actor, (S) : System 1. (A) A librarian inputs an item's ID 2. (S) Find a corresponding loan 3. (S) Get item information from the loan 4. (S) Get title information from the item 5. (S) Get loanPeriod from the title 6. (S) Compute calculateLateReturnFee 7. (S) Check reservationCount of the title.	8. (S) If the title is reserved, find the corresponding reservation 9. (S) Decrease loanCount of the loan. 10. (S) Decrease loanCount of the Borrower. 11. (S) Set validLoan of the borrower to false. 12. (S) Set available of the item to true. 13. (S) Increase AvailbaleCount of the title.
<b>Alternative Courses of Events</b>	N/A	
<b>Exceptional Courses of Events</b>	Line 2: If the loan does not exist, display an error message.	

# Phase 2041. Design Real Use Cases

- Calculate Late-Return-Fee

<b>Use Case</b>	<b>5. Calculate Late-Return-Fee</b>
<b>Actor</b>	<b>None</b>
<b>Purpose</b>	Compute late-return fee for an item returned late
<b>Overview</b>	(As in the business use case)
<b>Type</b>	Primary and Real
<b>Cross Reference</b>	System Functions: R1.4.1, R1.4.2 Use Cases: "Return Item"
<b>Pre-Requisites</b>	Lending time of an item should have expired
<b>Typical Courses of Events</b>	(A) : Actor, (S) : System 1. (S) Calculate Late-Return-Fee of the item 2. (S) Display the Late-Return-Fee
<b>Alternative Courses of Events</b>	N/A
<b>Exceptional Courses of Events</b>	N/A

# Phase 2041. Design Real Use Cases

- Get Replacement-Fee

<b>Use Case</b>	<b>6. Get Replacement-Fee</b>
<b>Actor</b>	Librarian
<b>Purpose</b>	Compute replacement-fee for a lost title
<b>Overview</b>	(As in the business use case)
<b>Type</b>	Primary and Real
<b>Cross Reference</b>	System Functions: R1.5 Use Cases: -
<b>Pre-Requisites</b>	A title should be lost.
<b>Typical Courses of Events</b>	<p>(A) : Actor, (S) : System</p> <ol style="list-style-type: none"> <li>1. (A) A librarian inputs an item's ID</li> <li>1. (S) Find a corresponding loan</li> <li>2. (S) Get an item from the loan</li> <li>3. (S) Get a title from the item</li> <li>4. (S) Get price of the title</li> <li>5. (S) Compute replacementFee</li> <li>6. (S) Set validLoan to false</li> <li>7. (S) Update the loan</li> <li>8. (S) Decrease loanCount of the borrower.</li> <li>9. (S) Set the isborrowed of the item to false.</li> <li>10. (S) Decrease numOfItem of the title.</li> </ol>
<b>Alternative Courses of Events</b>	N/A
<b>Exceptional Courses of Events</b>	Line 2: If the loan does not exist, display an error message.

# Phase 2041. Design Real Use Cases

- Notify Availability

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

# Phase 2041. Design Real Use Cases

- Add Title

<b>Use Case</b>	<b>8. Add Title</b>
<b>Actor</b>	Librarian
<b>Purpose</b>	Register a new title
<b>Overview</b>	(As in the business use case)
<b>Type</b>	Primary and Real
<b>Cross Reference</b>	System Functions: R2.1, R2.4 Use Case: "Add Item"
<b>Pre-Requisites</b>	N/A
<b>Typical Courses of Events</b>	<p>(A) : Actor, (S) : System</p> <ol style="list-style-type: none"> <li>1. (A) A librarian inputs title information such as name, isbn, price, publisher, loanPeriod (Book: author, Magazine:month, publishCycle)</li> <li>2. (S) Find a corresponding title</li> <li>3. (S) Create a new title</li> <li>4. (S) Store the new title</li> <li>5. (S) Invoke "Add Item"</li> </ol>
<b>Alternative Courses of Events</b>	N/A
<b>Exceptional Courses of Events</b>	Line 1: If the title already exists, display an error message.

# Phase 2041. Design Real Use Cases

- Remove Title

<b>Use Case</b>	<b>9. Remove Title</b>
<b>Actor</b>	Librarian
<b>Purpose</b>	Delete information of a title
<b>Overview</b>	(As in the business use case)
<b>Type</b>	Primary and Real
<b>Cross Reference</b>	System Functions: R2.2 Use Case: -
<b>Pre-Requisites</b>	N/A
<b>Typical Courses of Events</b>	(A) : Actor, (S) : System 1. (A) A librarian inputs a title's isbn to remove 2. (S) Find a corresponding title 3. (S) Check if the corresponding title is reserved. 4. (S) If the title is reserved, Remove the reservation 5. (S) Check the item of the title is loaned. 6. (S) Remove the title
<b>Alternative Courses of Events</b>	N/A
<b>Exceptional Courses of Events</b>	Line 2: If the title does not exist, display an error message. Line 5: If the item of the title is loaned, display an error mesasge.

# Phase 2041. Design Real Use Cases

- Update Title

<b>Use Case</b>	<b>10. Update Title</b>
<b>Actor</b>	Librarian
<b>Purpose</b>	Update information of a title
<b>Overview</b>	(As in the business use case)
<b>Type</b>	Primary and Real
<b>Cross Reference</b>	System Functions: R2.3 Use Case: -
<b>Pre-Requisites</b>	N/A
<b>Typical Courses of Events</b>	(A) : Actor, (S) : System 1. (A) A librarian inputs a title's isbn and information of the title to change 2. (S) Find a corresponding title 3. (S) Update the title
<b>Alternative Courses of Events</b>	N/A
<b>Exceptional Courses of Events</b>	Line 2: If the item does not exist, display "Not Existing Title". Error message. Line 3: If the isbn is changed, then update items too,



# Phase 2041. Design Real Use Cases

- Add Item

<b>Use Case</b>	<b>11. Add Item</b>
<b>Actor</b>	Librarian
<b>Purpose</b>	Add a new item
<b>Overview</b>	(As in the business use case)
<b>Type</b>	Primary and Real
<b>Cross Reference</b>	System Functions: R2.4 Use Cases: "Add Title"
<b>Pre-Requisites</b>	N/A
<b>Typical Courses of Events</b>	(A) : Actor, (S) : System 1. (A) A librarian inputs an item's id 2. (S) Find a corresponding title 3. (S) Get an item's ID from the title 4. (S) Create a new item 5. (S) Store the new item 6. (S) Increase numOfItem of the title 7. (S) Increase availablecount of the item
<b>Alternative Courses of Events</b>	N/A
<b>Exceptional Courses of Events</b>	Line 2: Line 2: If the title does not exist, display an error message.

# Phase 2041. Design Real Use Cases

- Remove Item

<b>Use Case</b>	<b>12. Remove Item</b>
<b>Actor</b>	Librarian
<b>Purpose</b>	Remove information of an item
<b>Overview</b>	(As in the business use case)
<b>Type</b>	Primary and Real
<b>Cross Reference</b>	System Functions: R2.1, R2.5 Use Case: "Remove Title"
<b>Pre-Requisites</b>	N/A
<b>Typical Courses of Events</b>	(A) : Actor, (S) : System 1. (A) A librarian inputs an item's ID 2. (S) Find a corresponding item 3. (S) Check if the item is borrowed 4. (S) If the item is borrowed, decrease numOfItem of the title 5. (S) Decrease availableCount of the title 6. (S) Remove the item
<b>Alternative Courses of Events</b>	N/A
<b>Exceptional Courses of Events</b>	Line 2: If the item does not exist, display an error message. Line 3: If the item was already borrowed, display an error message

# Phase 2041. Design Real Use Cases

- Update Item

<b>Use Case</b>	<b>13. Update Item</b>
<b>Actor</b>	Librarian
<b>Purpose</b>	Update information of an item
<b>Overview</b>	(As in the business use case)
<b>Type</b>	Primary and Real
<b>Cross Reference</b>	System Functions: R2.6 Use Case: -
<b>Pre-Requisites</b>	N/A
<b>Typical Courses of Events</b>	<p>(A) : Actor, (S) : System</p> <ol style="list-style-type: none"> <li>1. (A) A librarian inputs the item's id and information to change</li> <li>2. (S) Find A corresponding item</li> <li>3. (S) Update the item</li> <li>4. (S) If a lost of the item is true, decrease numOfItem of the title.</li> <li>5. (S) Decrease the availableCount of the title.</li> <li>6. (S) If a lost of the item is false, increase numOfItem of the title. (What? Only for these cases "Update Item" are used?)</li> <li>7. (S) Increase availableCount of the title</li> </ol>
<b>Alternative Courses of Events</b>	N/A
<b>Exceptional Courses of Events</b>	Line 2: If the item does not exist, display an error message.

# Phase 2041. Design Real Use Cases

- Add Borrower

<b>Use Case</b>	<b>14. Add Borrower</b>
<b>Actor</b>	Librarian
<b>Purpose</b>	Register a new borrower
<b>Overview</b>	(As in the business use case)
<b>Type</b>	Primary and Real
<b>Cross Reference</b>	System Functions: R1.1, R1.3, R3.1 Use Cases: "Make Reservation", "Lend Item"
<b>Pre-Requisites</b>	N/A
<b>Typical Courses of Events</b>	(A) : Actor, (S) : System 1. (A) A librarian inputs a borrower's name, ssn, and address. 2. (S) Find a corresponding borrower 3. (S) Create a new borrower 4. (S) Store the new borrower
<b>Alternative Courses of Events</b>	N/A
<b>Exceptional Courses of Events</b>	Line 2: If the borrower exists already, display an error message.

# Phase 2041. Design Real Use Cases

- Remove Borrower

<b>Use Case</b>	<b>15. Remove Borrower</b>
<b>Actor</b>	Librarian
<b>Purpose</b>	Remove information of a borrower
<b>Overview</b>	(As in the business use case)
<b>Type</b>	Primary and Real
<b>Cross Reference</b>	System Functions: R3.2 Use Case: -
<b>Pre-Requisites</b>	N/A
<b>Typical Courses of Events</b>	(A) : Actor, (S) : System 1. (A) A librarian inputs the borrower's ssn 2. (S) Find a corresponding borrower 3. (S) Find a loan of the borrower 4. (S) If the loan is invalid, find a reservation 5. (S) Get the title of the reservation 6. (S) Decrease reservationCount of the title 7. (S) Remove borrower
<b>Alternative Courses of Events</b>	N/A
<b>Exceptional Courses of Events</b>	Line 2: If the borrower does not exist, display an error message. Line 3: If the loan is still valid, display an error message.

# Phase 2041. Design Real Use Cases

- Update Borrower

<b>Use Case</b>	<b>16. Update Borrower</b>
<b>Actor</b>	Librarian
<b>Purpose</b>	Update information of a borrower
<b>Overview</b>	(As in the business use case)
<b>Type</b>	Primary and Real
<b>Cross Reference</b>	System Functions: R3.3 Use Case: -
<b>Pre-Requisites</b>	N/A
<b>Typical Courses of Events</b>	(A) : Actor, (S) : System 1. (A) A librarian inputs a borrower's ssn and information to change 2. (S) Find a corresponding borrower 3. (S) Update the borrower
<b>Alternative Courses of Events</b>	N/A
<b>Exceptional Courses of Events</b>	Line 2: If the borrower does not exist, display an error message.

# Phase 2041. Design Real Use Cases

- Log-In

<b>Use Case</b>	<b>17. Log-In</b>
<b>Actor</b>	Librarian
<b>Purpose</b>	Check access authority of a librarian
<b>Overview</b>	(As in the business use case)
<b>Type</b>	Secondary and Real
<b>Cross Reference</b>	System Functions: R4.1 Use Case: -
<b>Pre-Requisites</b>	A librarian should have user name and password.
<b>Typical Courses of Events</b>	(A) : Actor, (S) : System 1. (A) A librarian inputs an userID and password 2. (S) Check if the userID and password are correct
<b>Alternative Courses of Events</b>	N/A
<b>Exceptional Courses of Events</b>	Line 2: If the userID and password are not correct, display an error message.

# Phase 2041. Design Real Use Cases

- Log-Out

<b>Use Case</b>	<b>18. Log-Out</b>
<b>Actor</b>	Librarian
<b>Purpose</b>	Exit the library management system
<b>Overview</b>	(As in the business use case)
<b>Type</b>	Secondary and Essential
<b>Cross Reference</b>	System Functions: R4.1 Use Case: -
<b>Pre-Requisites</b>	A librarian should have user name and password.
<b>Typical Courses of Events</b>	(A) : Actor, (S) : System 1. (A) A librarian selects "LogOut" 2. (S) Check if the userID is correct and then exit the system
<b>Alternative Courses of Events</b>	N/A
<b>Exceptional Courses of Events</b>	Line 2: If the userID is incorrect, display an error message.



# Phase 2041. Design Real Use Cases

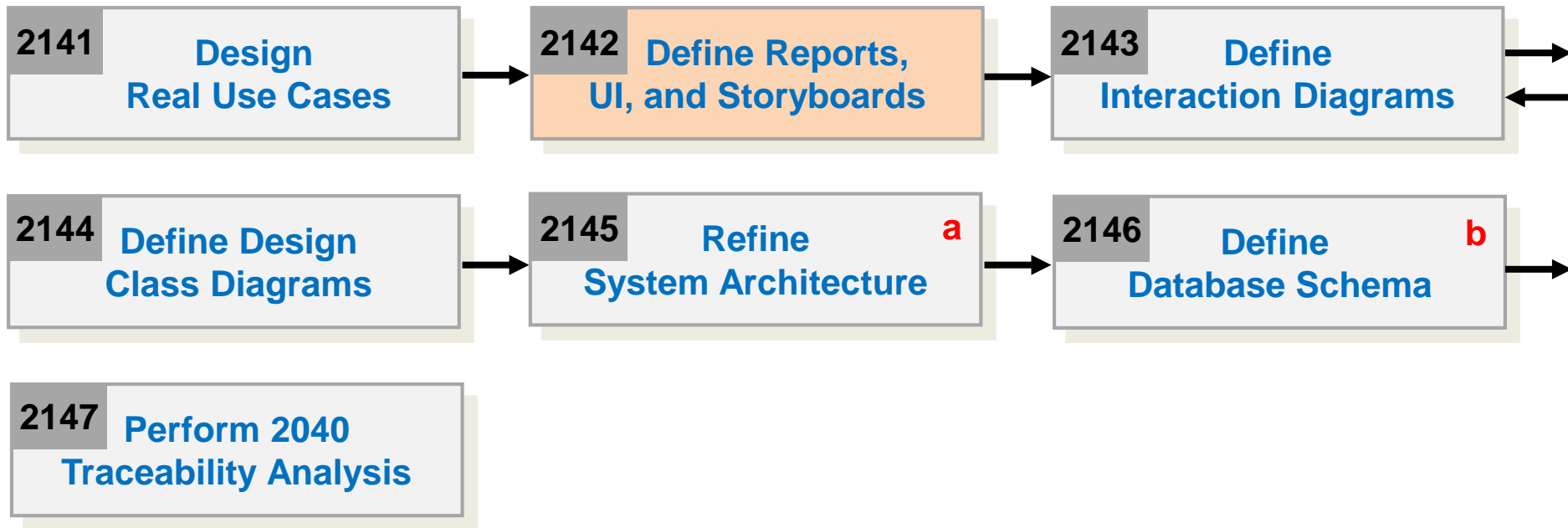
- Count Loans

<b>Use Case</b>	<b>19. Count Loans</b>
<b>Actor</b>	Librarian
<b>Purpose</b>	Compute total count of the titles checked out
<b>Overview</b>	(As in the business use case)
<b>Type</b>	Secondary and Essential
<b>Cross Reference</b>	System Functions: R5.1 Use Case: -
<b>Pre-Requisites</b>	A librarian should have user name and password.
<b>Typical Courses of Events</b>	(A) : Actor, (S) : System 1. (A) A librarian requests loan count 2. (S) Get numOfLoan of the loan
<b>Alternative Courses of Events</b>	N/A
<b>Exceptional Courses of Events</b>	N/A (Really?)

# Phase 2042. Define Reports, UI, and Storyboards

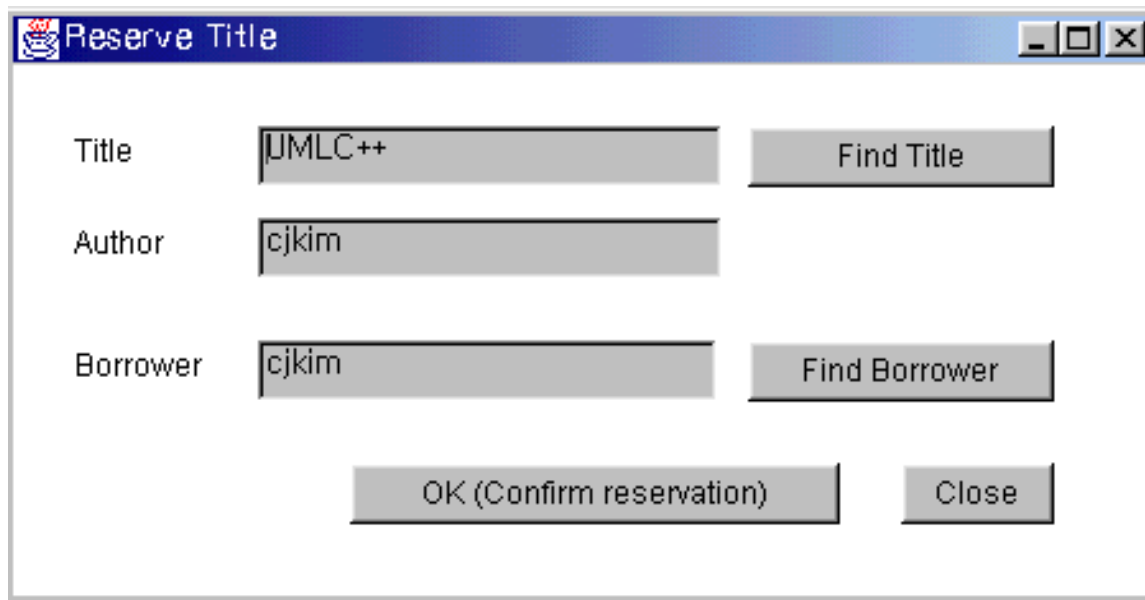
- 7 Activities

a. Varied order  
b. optional



# Phase 2042. Define Reports, UI, and Storyboards

- Make Reservation



**Reserve Title**

Title

Author

Borrower

# Phase 2042. Define Reports, UI, and Storyboards

- Lent Item

**Lend Item**

Title

Author

Items ☐ B = Borrowed  
F = Free

Borrower

# Phase 2042. Define Reports, UI, and Storyboards

- Count Loans

The screenshot shows a 'Title Information' dialog box with the following fields and buttons:

- Title Name:** UMLC++
- Author:** cjkim
- ISBN / Nr:** 700630-1031410
- Type:** Book
- Buttons:** Find, OK

Below the input fields, there are two sections:

- Items Available:**
  - (Item:1 ): Borrowed
  - (Item:2 ): Borrowed
  - (Item:3 ): Free
- Reservations:** 1111

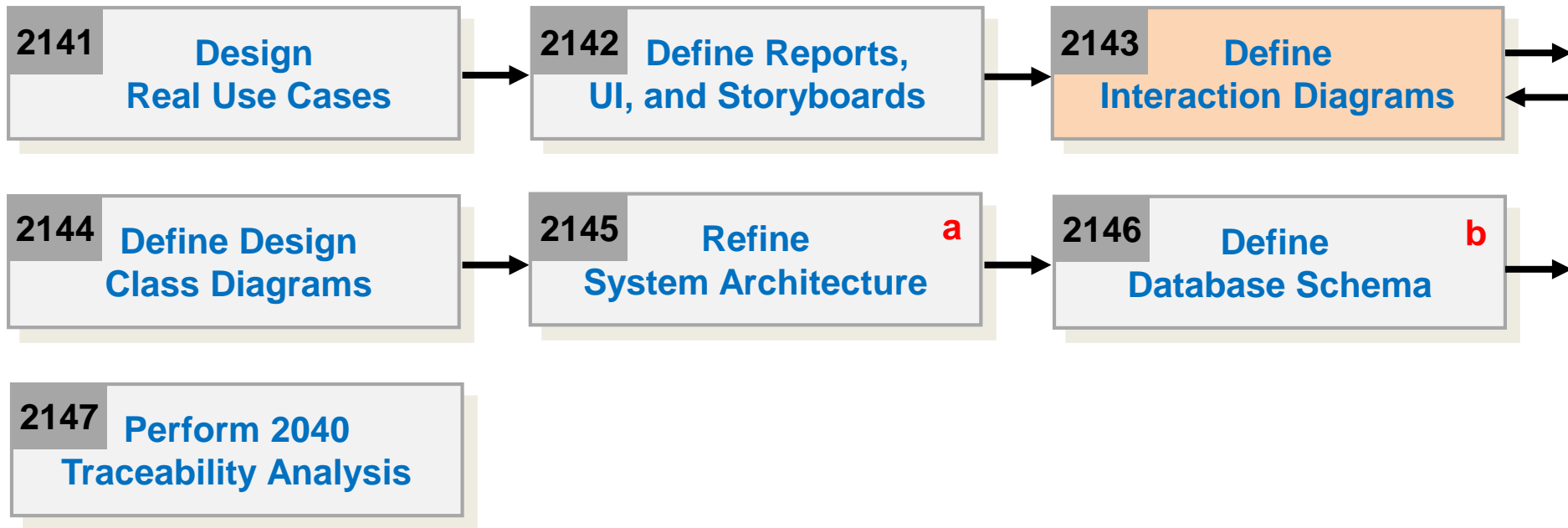
At the bottom, there are two summary lines:

- Total : 3, Borrowed : 2, Free : 1**
- Total : 1**

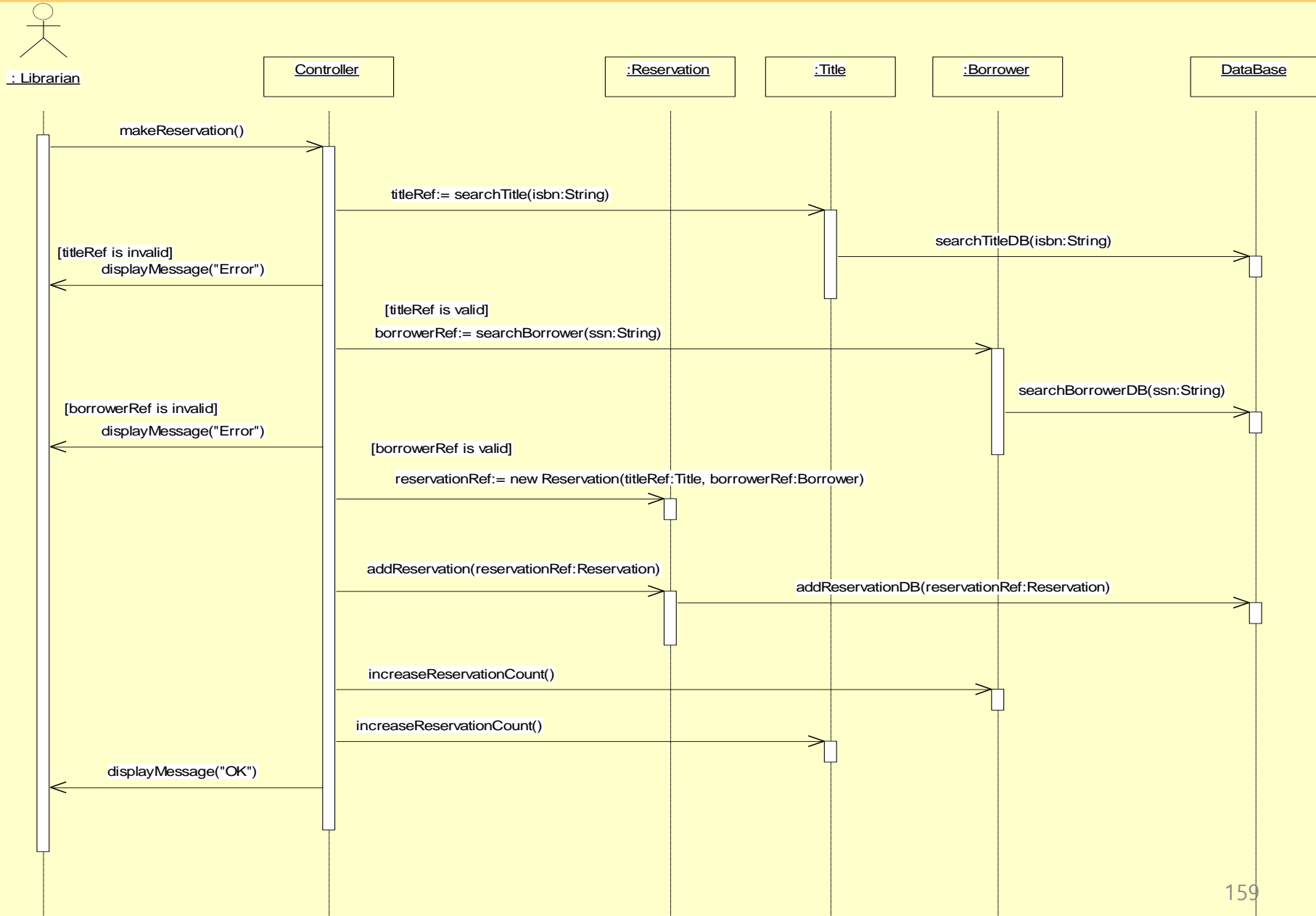
# Phase 2043. Define Interaction Diagrams

- 7 Activities

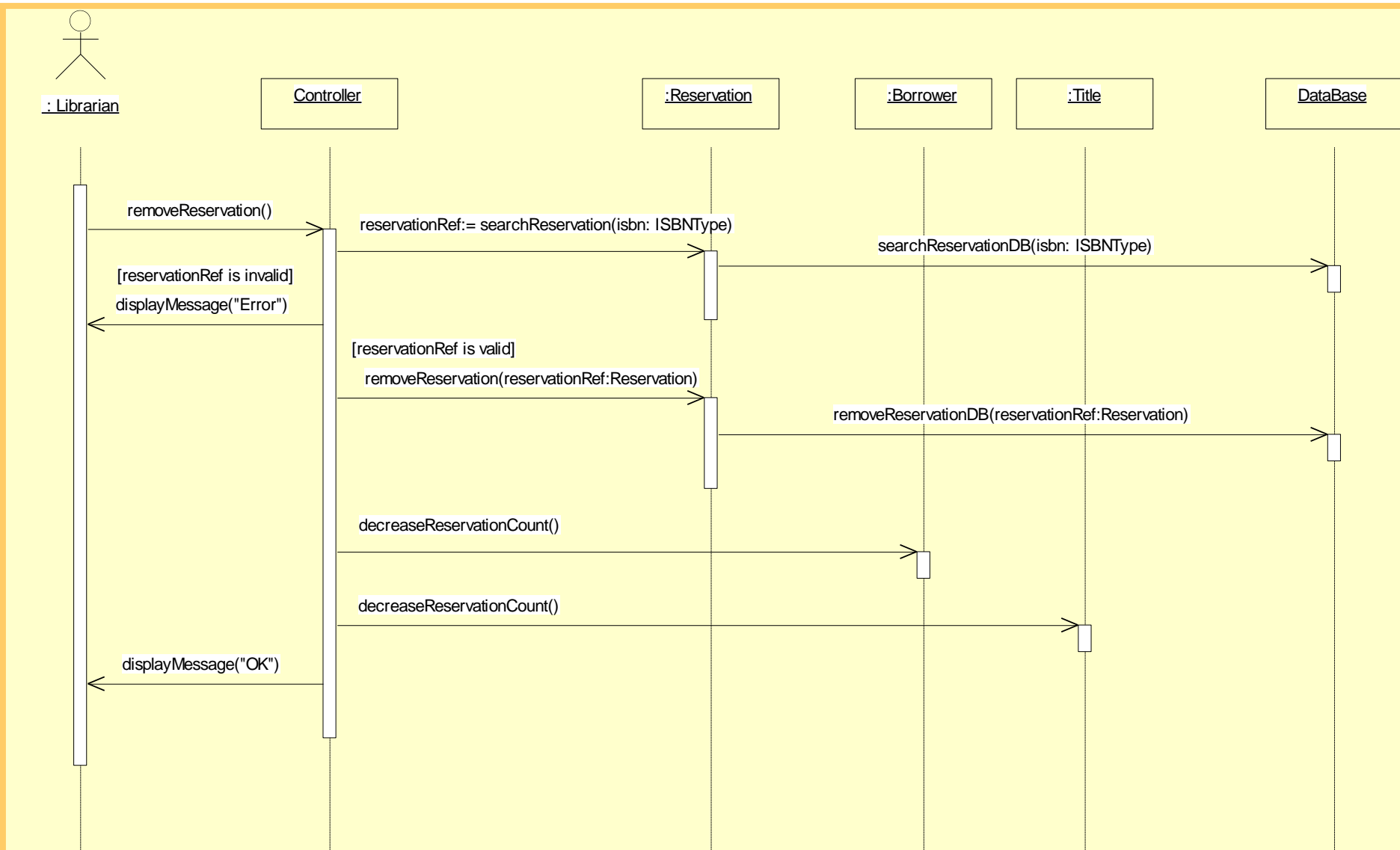
a. Varied order  
b. optional



# 1. Make Reservation

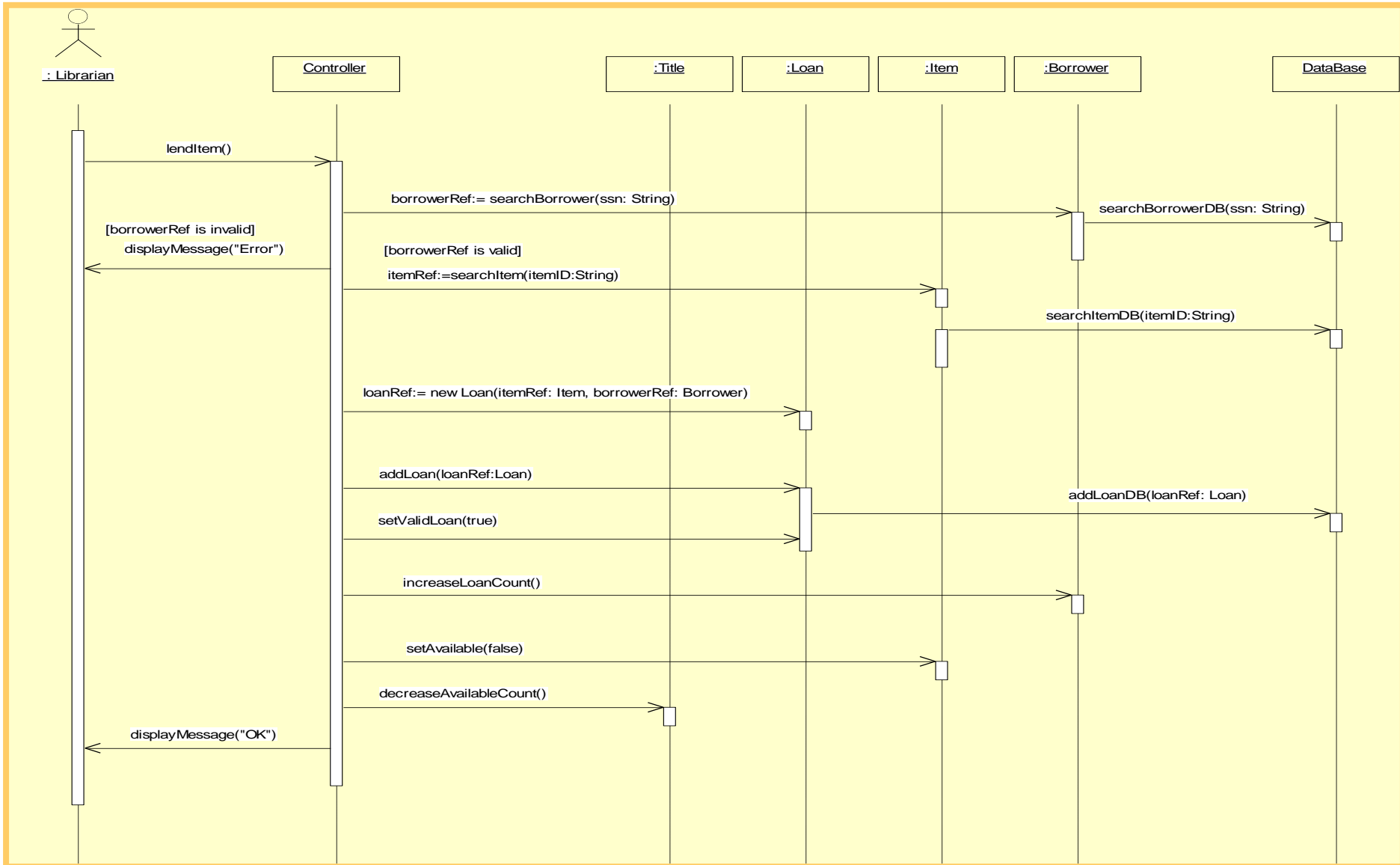


## 2. Remove Reservation

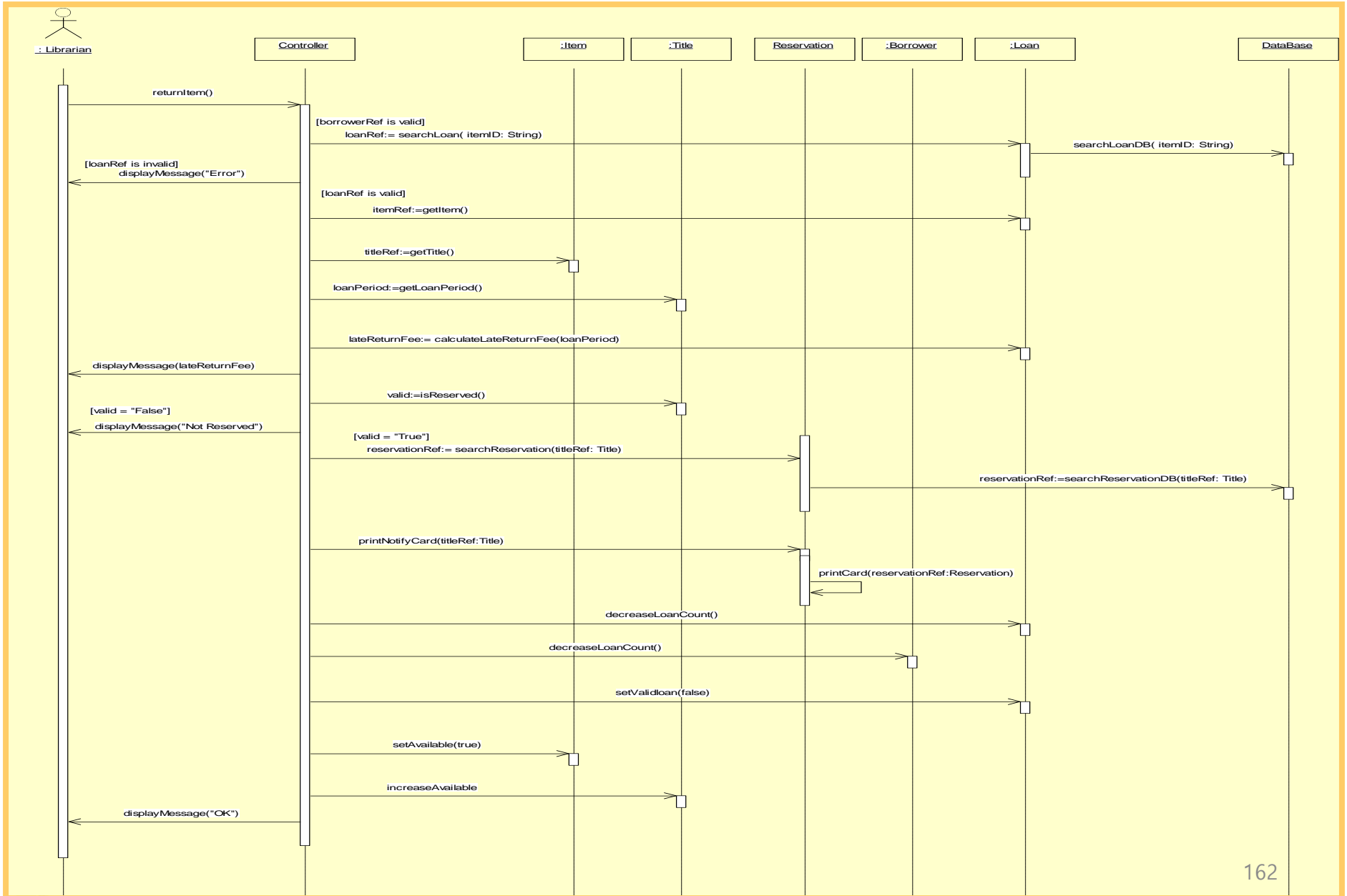




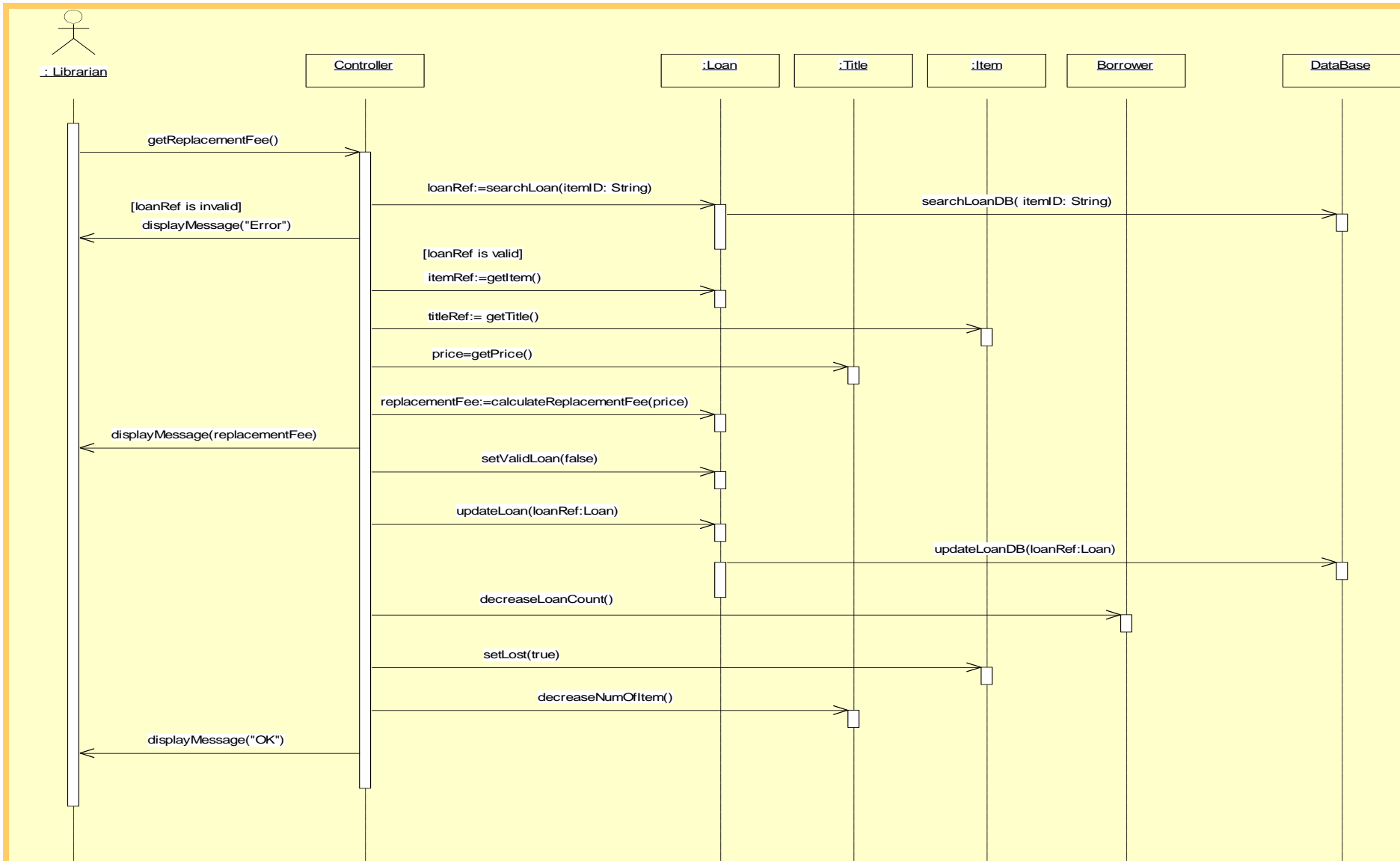
# 3. Lend Item



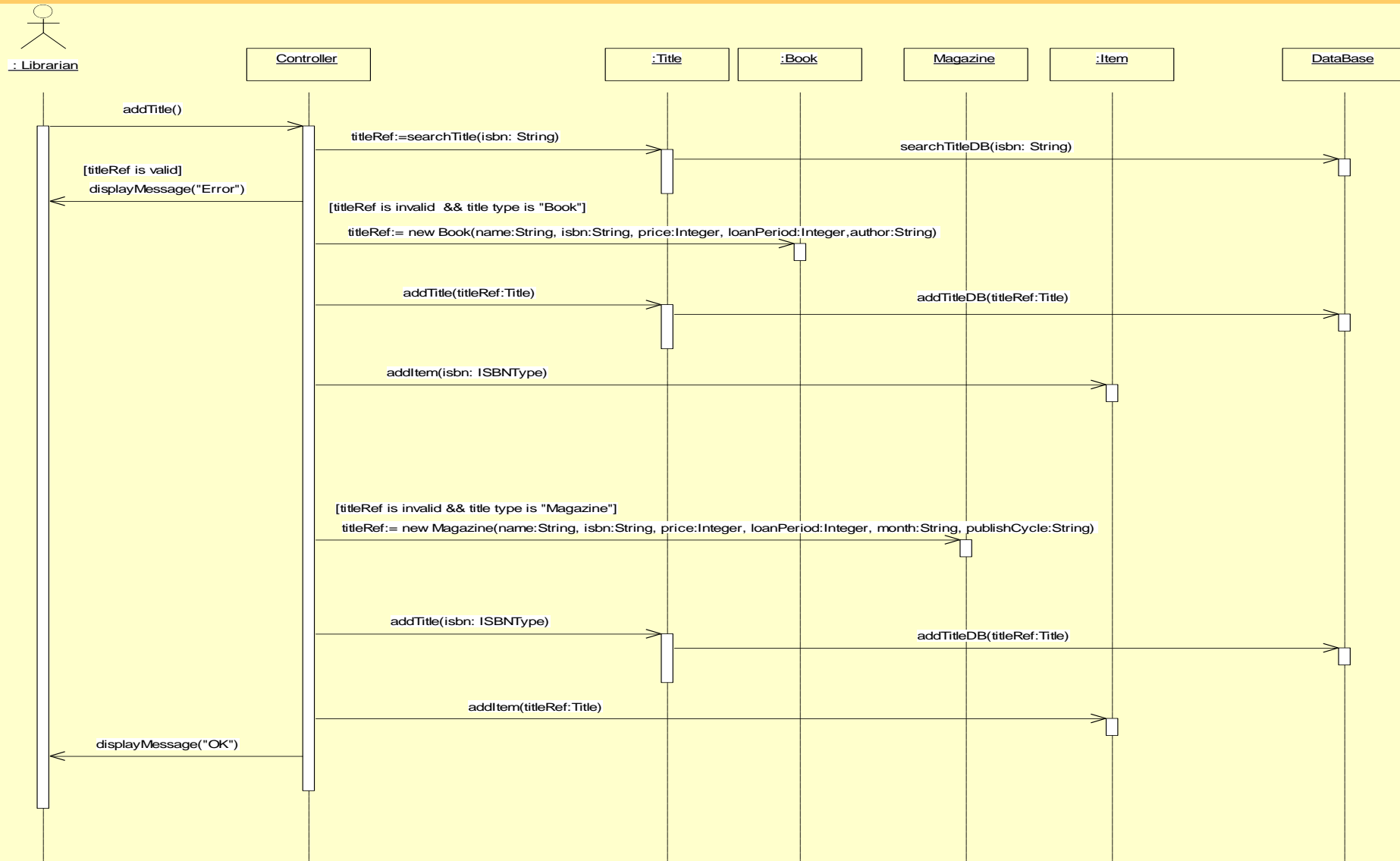
# 4. Return Item



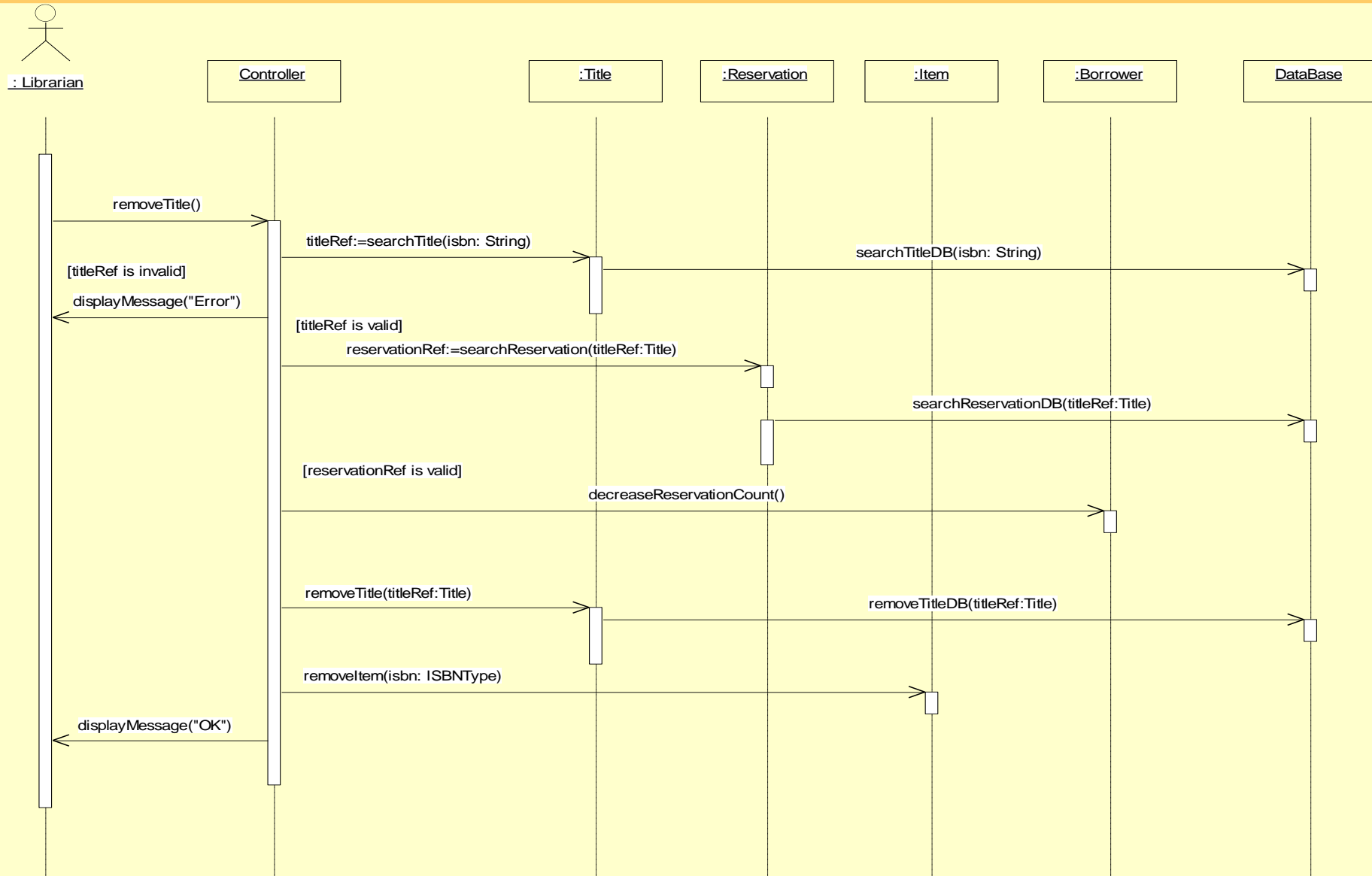
## 6. Get Replacement-Fee



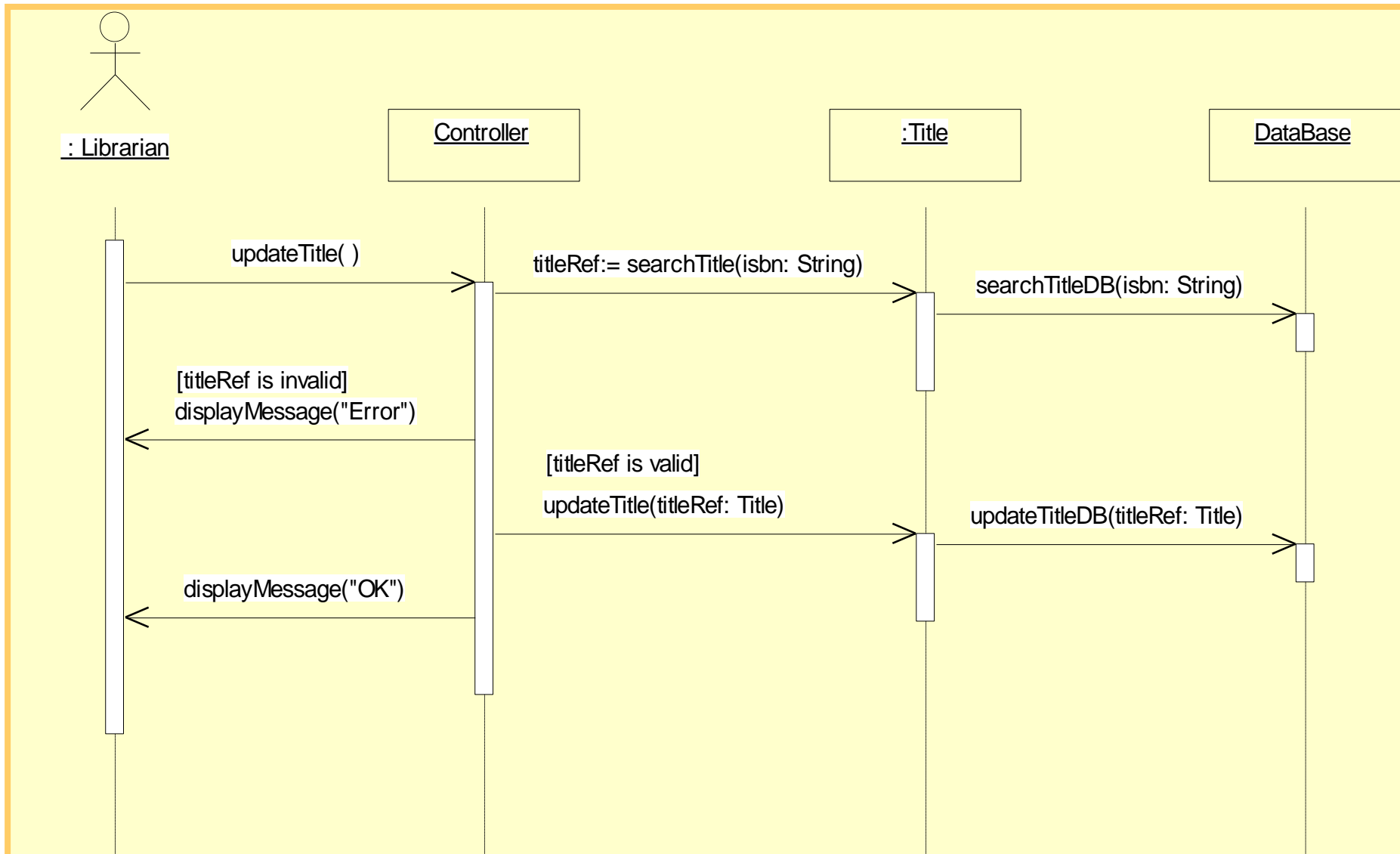
## 8. Add Title



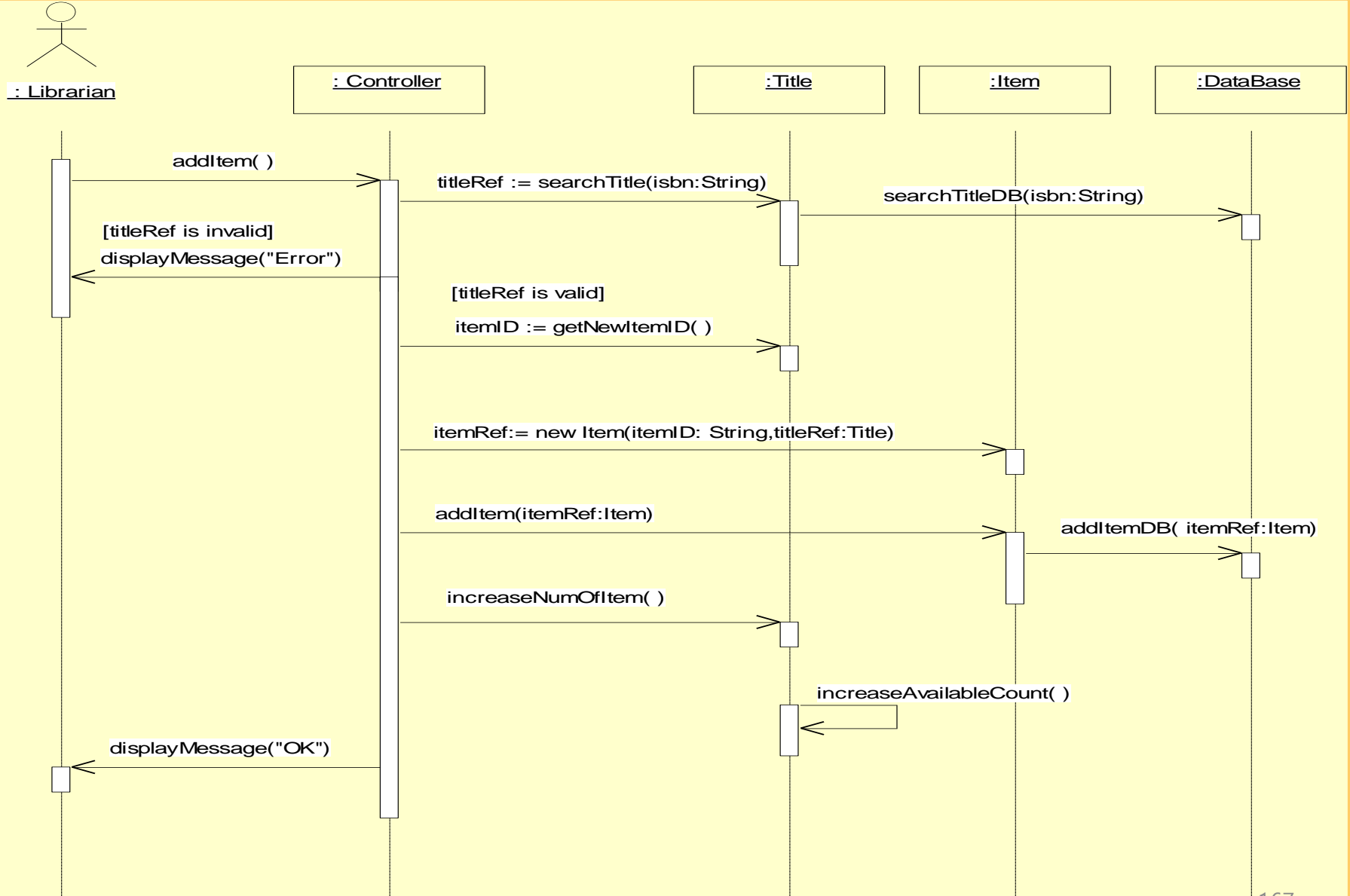
# 9. Remove Title



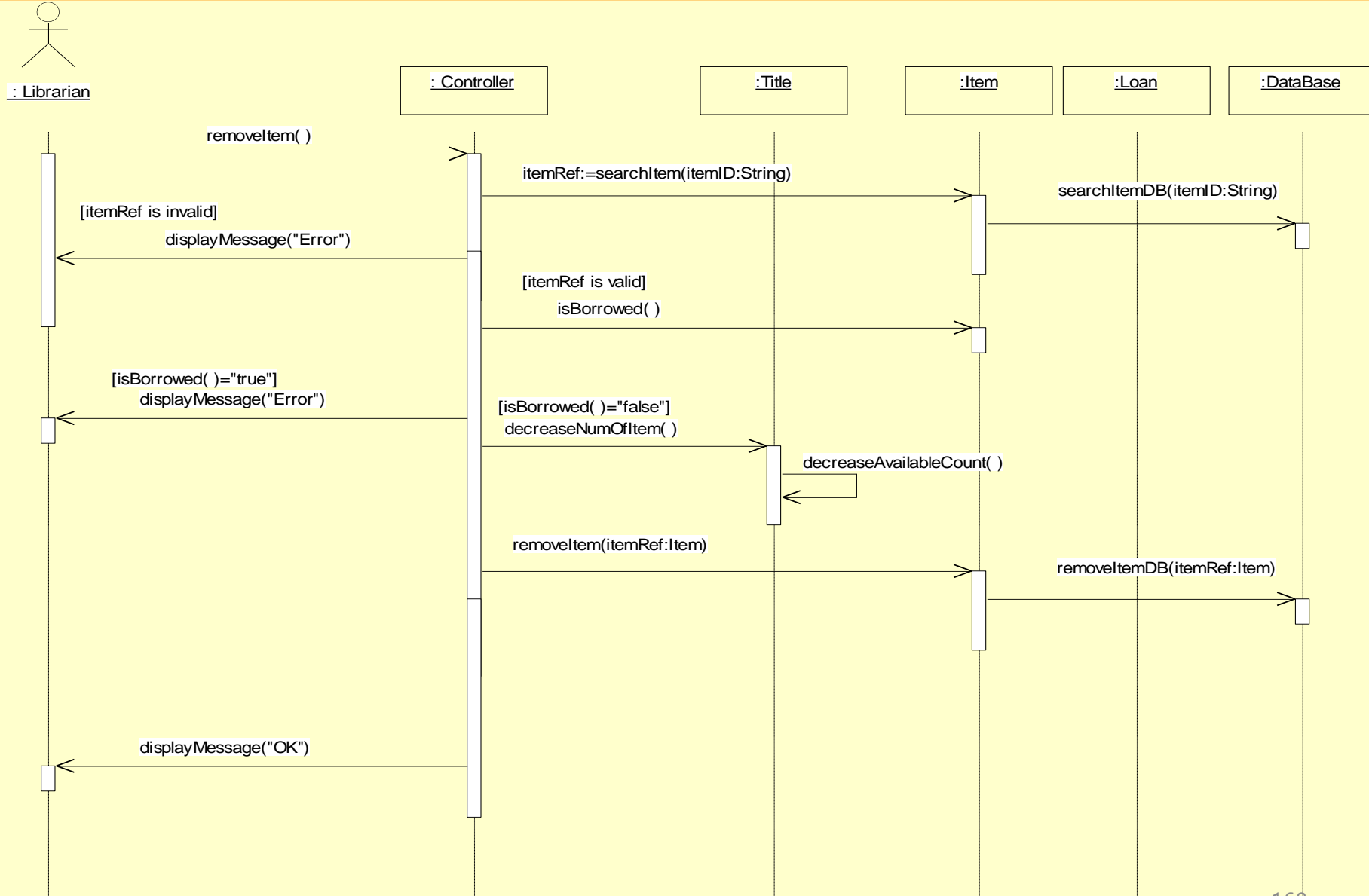
# 10. Update Title



# 11. Add Item

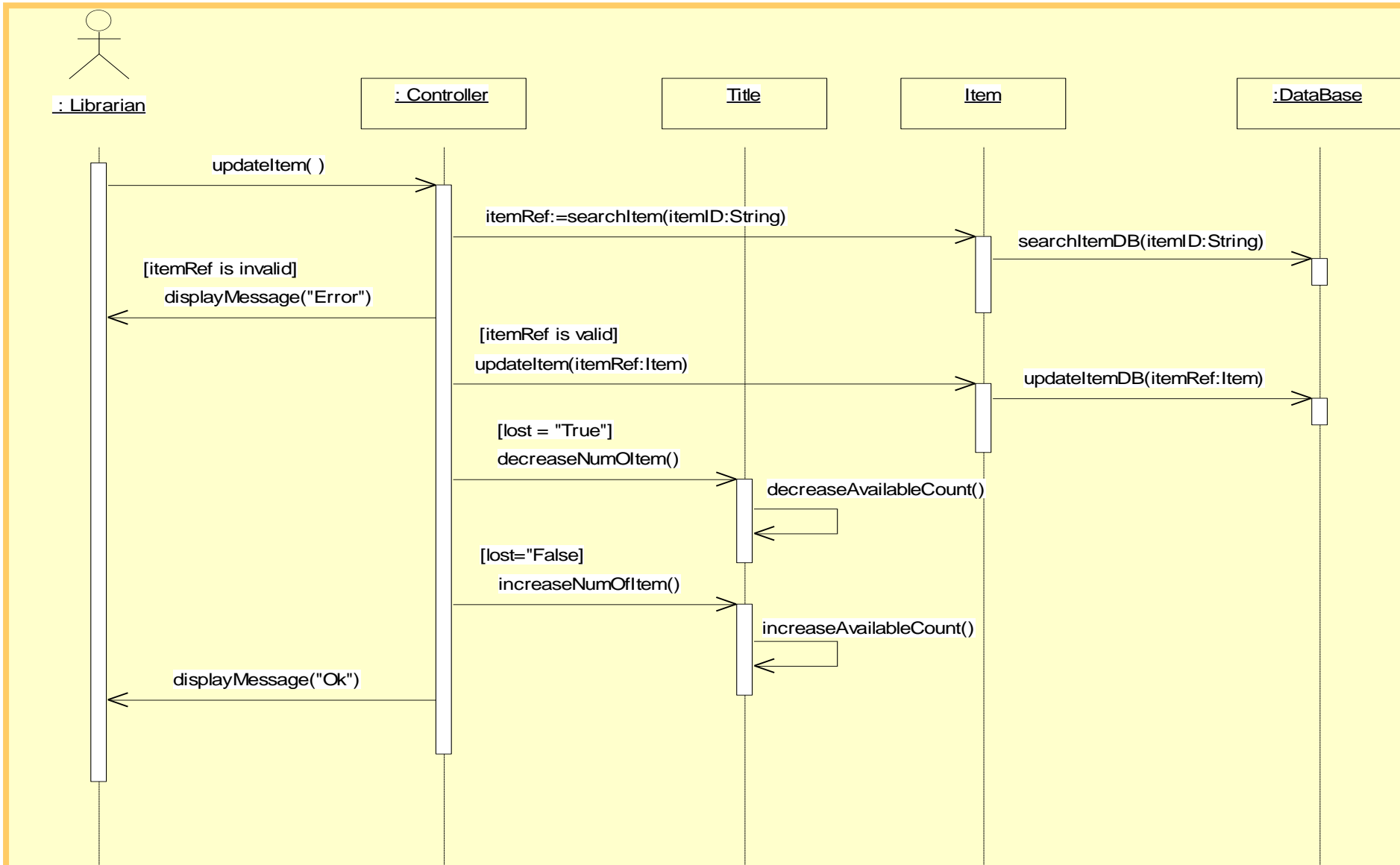


# 12. Remove Item

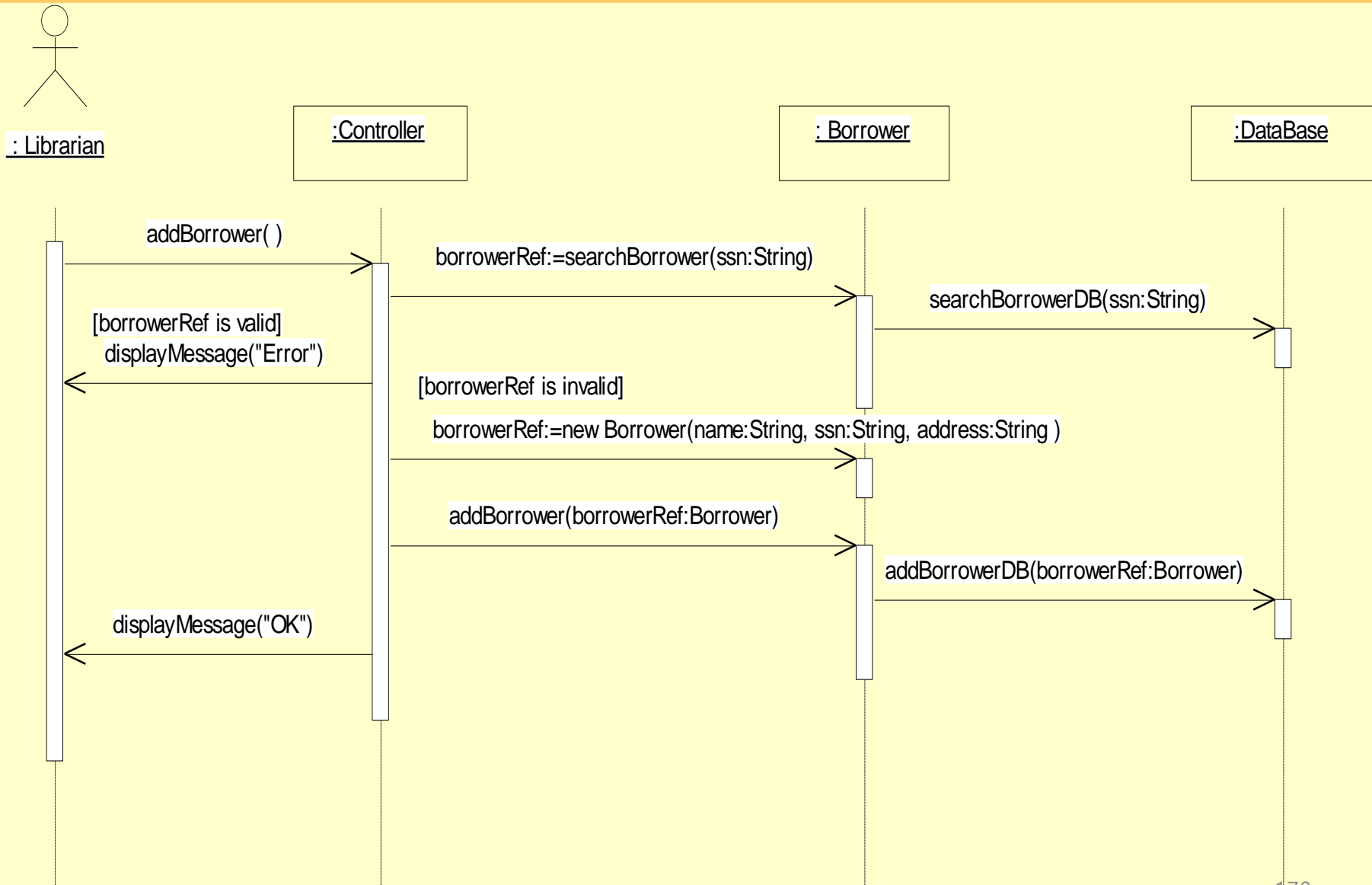




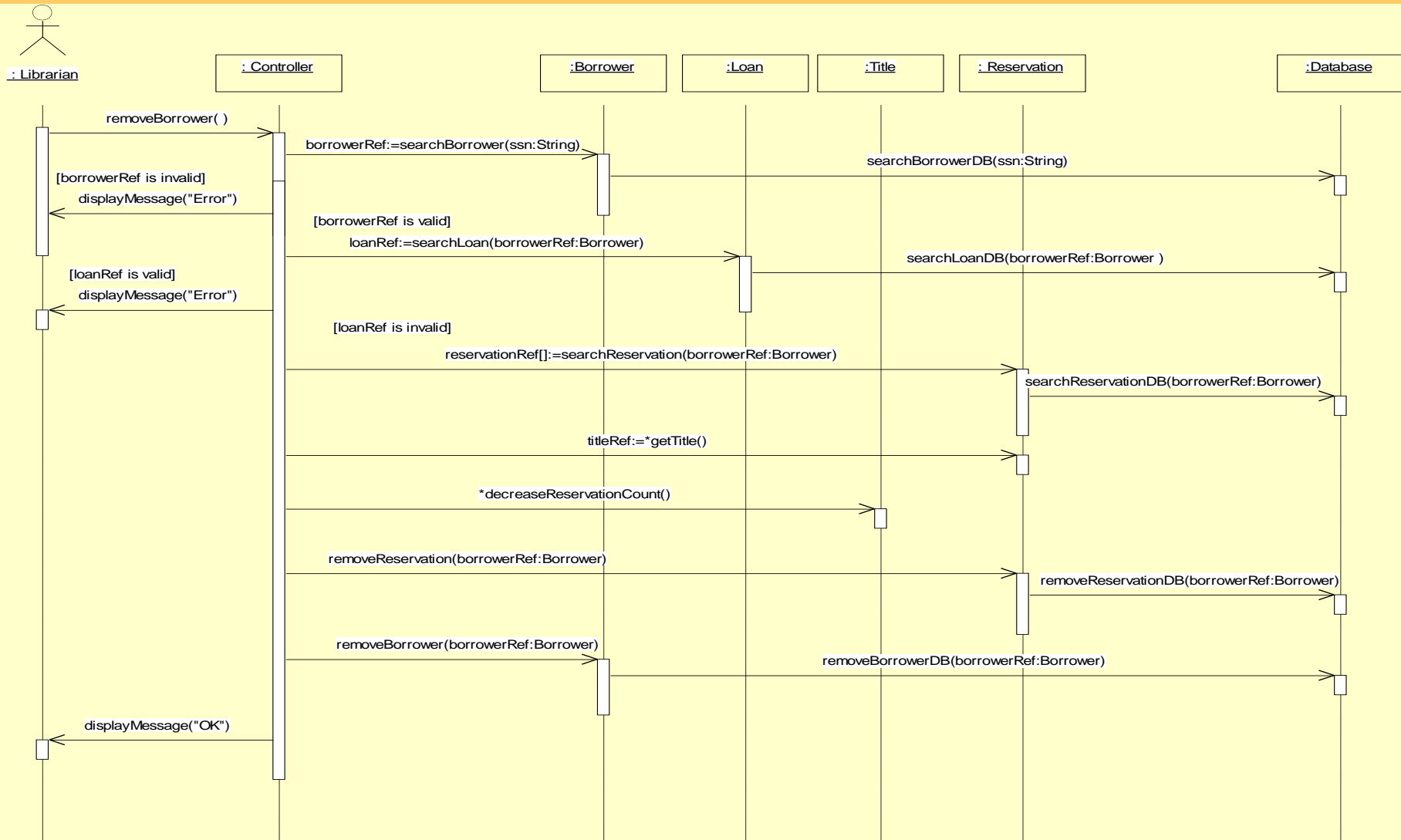
# 12. Update Item



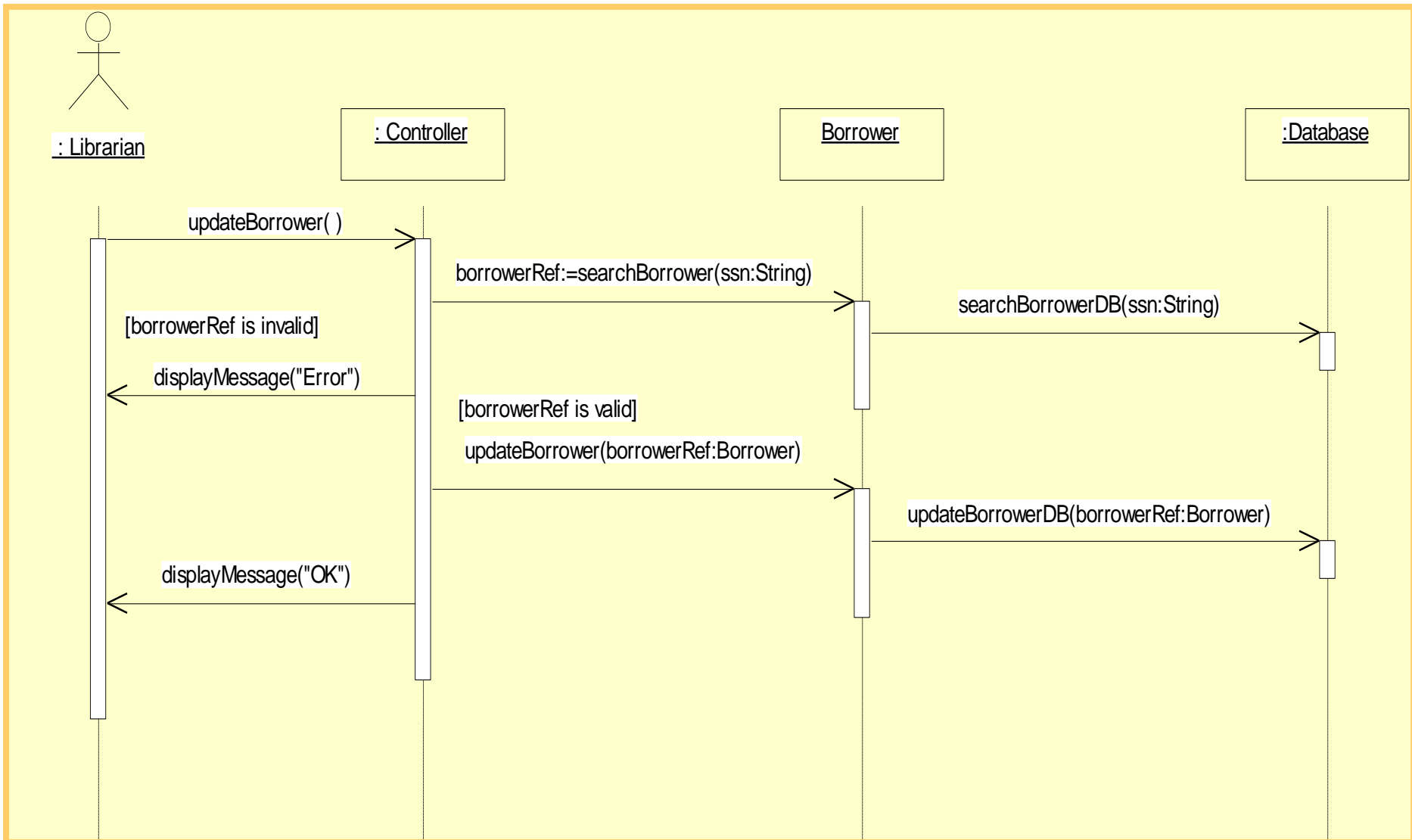
# 14. Add Borrower



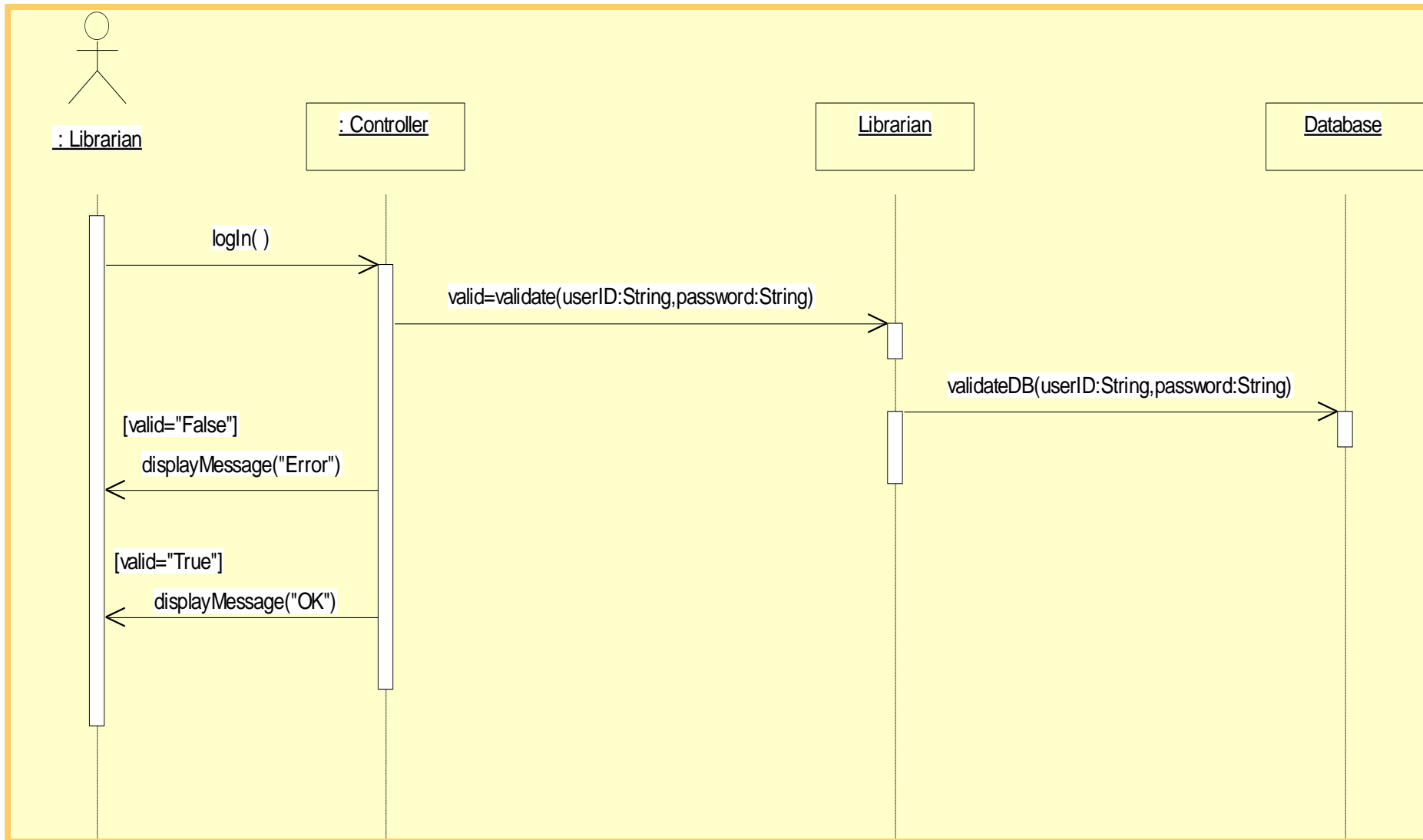
# 15. Remove Borrower



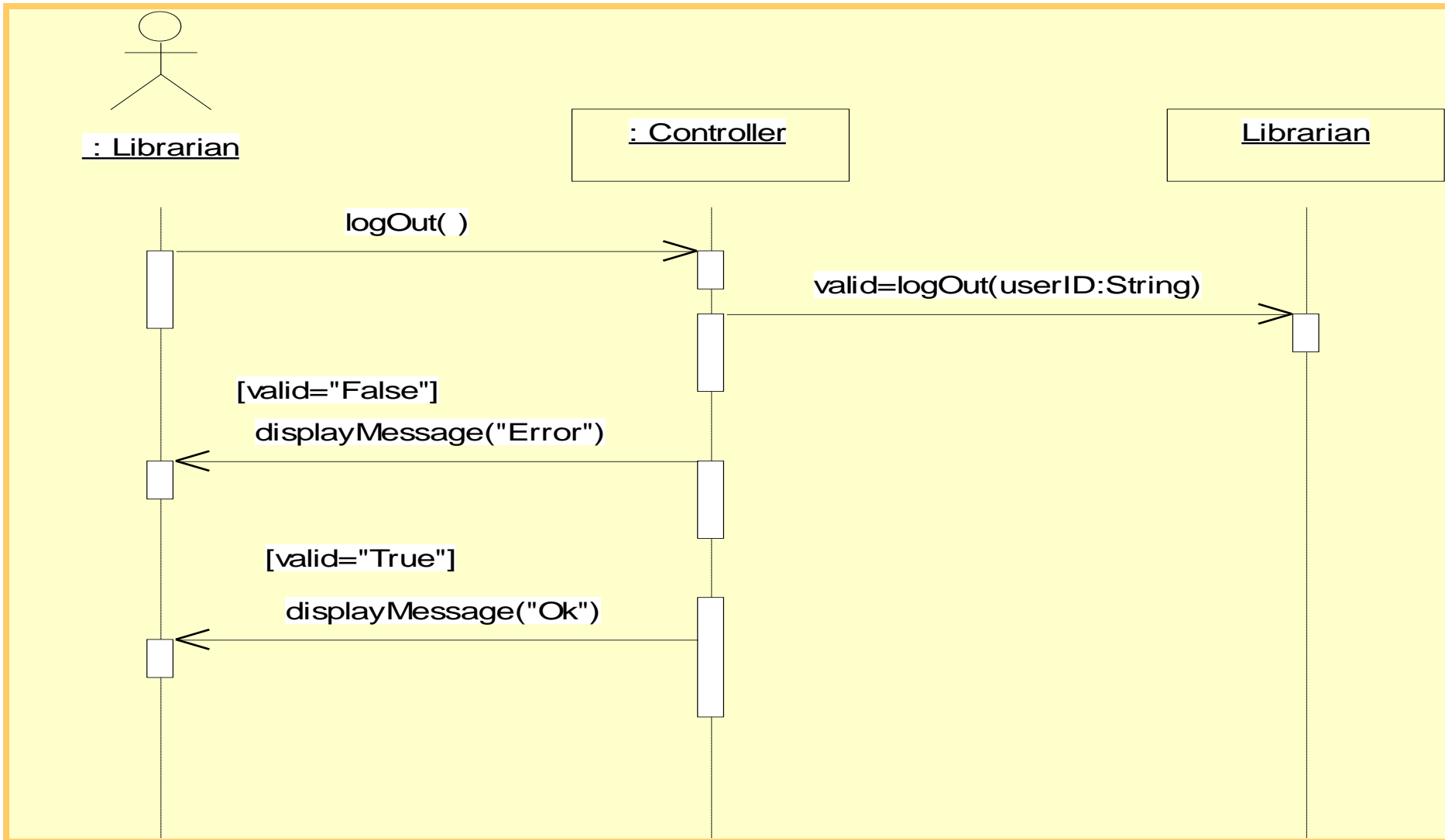
## 16. Update Borrower



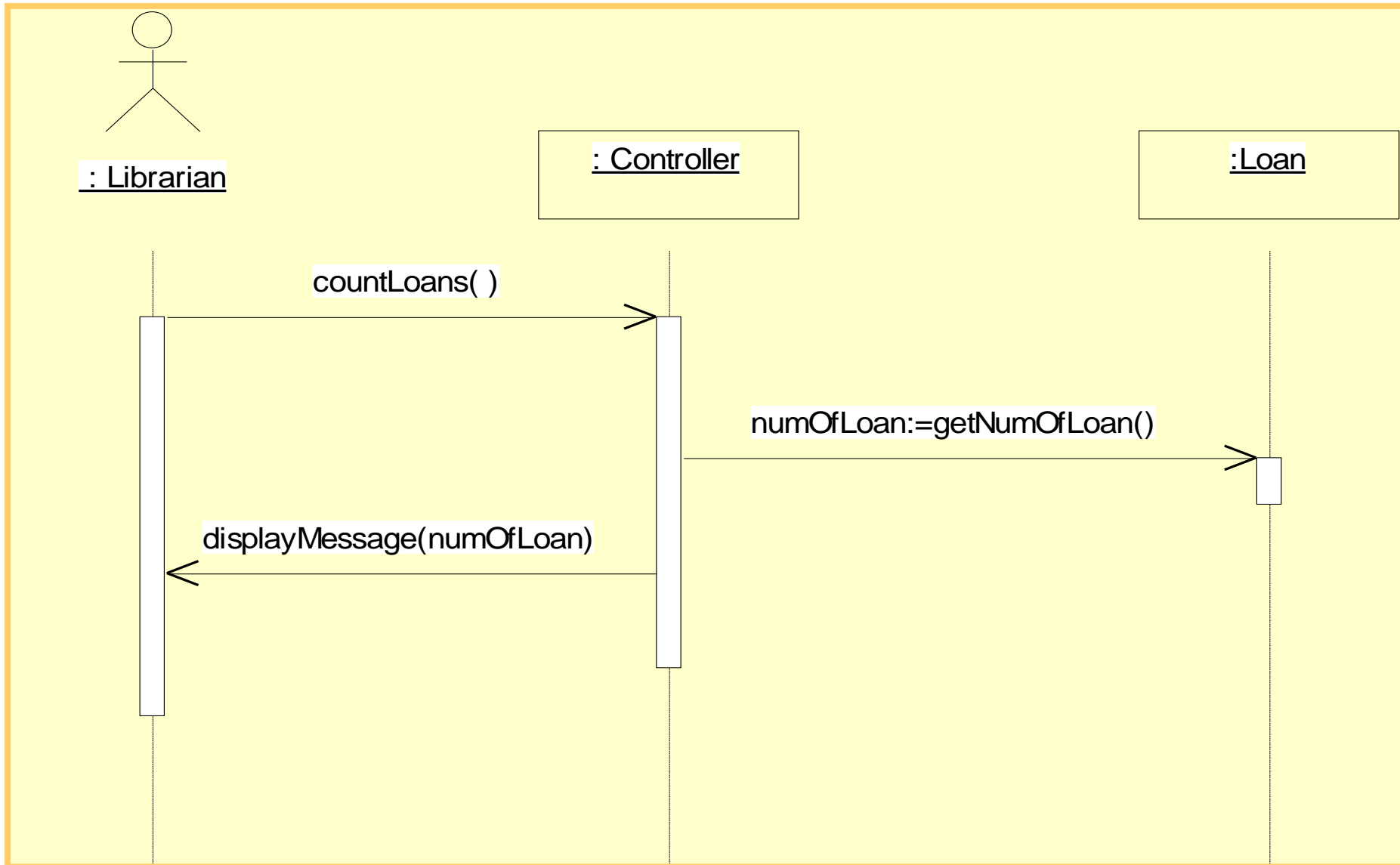
# 17. Log-In



# 18. Log-Out



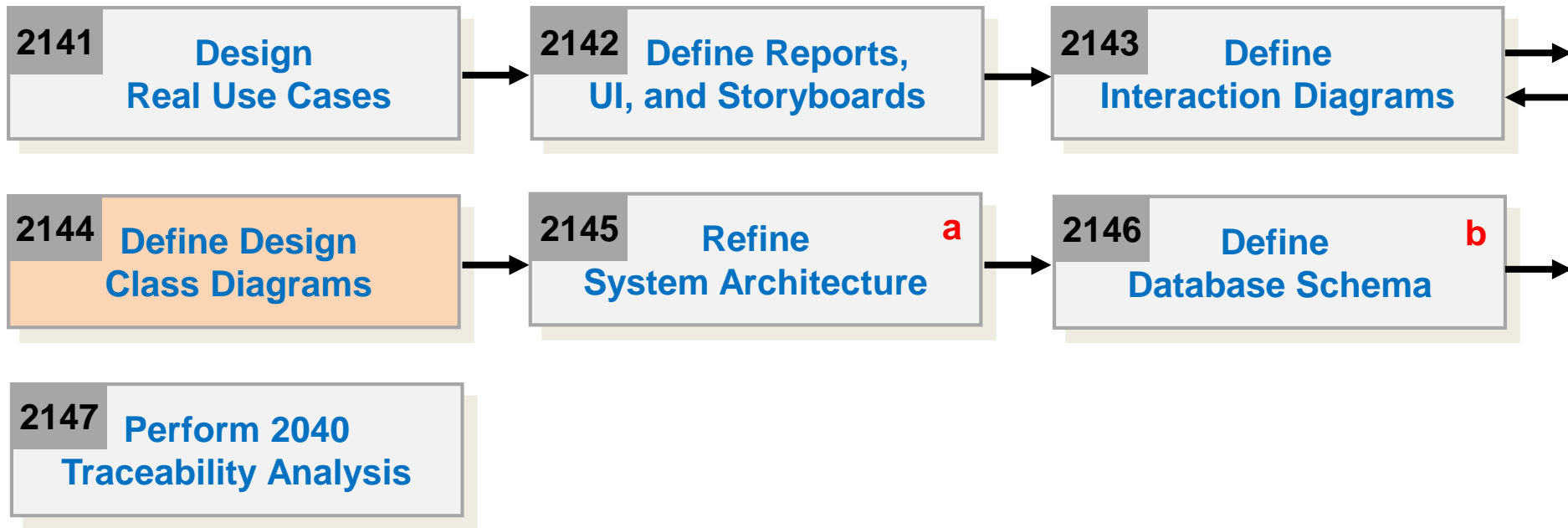
# 19. Count Loans



# Phase 2044. Define Design Class Diagram

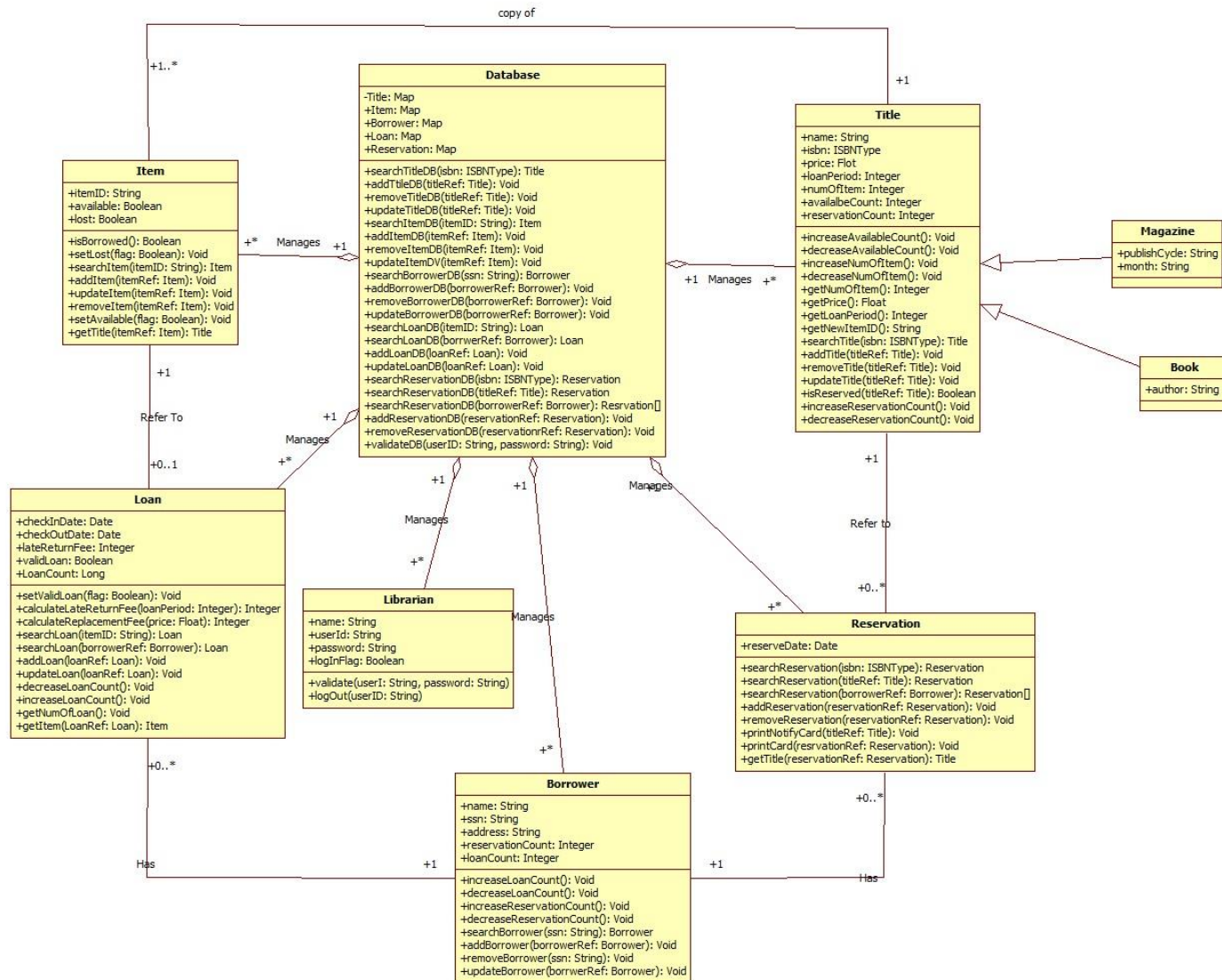
- 7 Activities

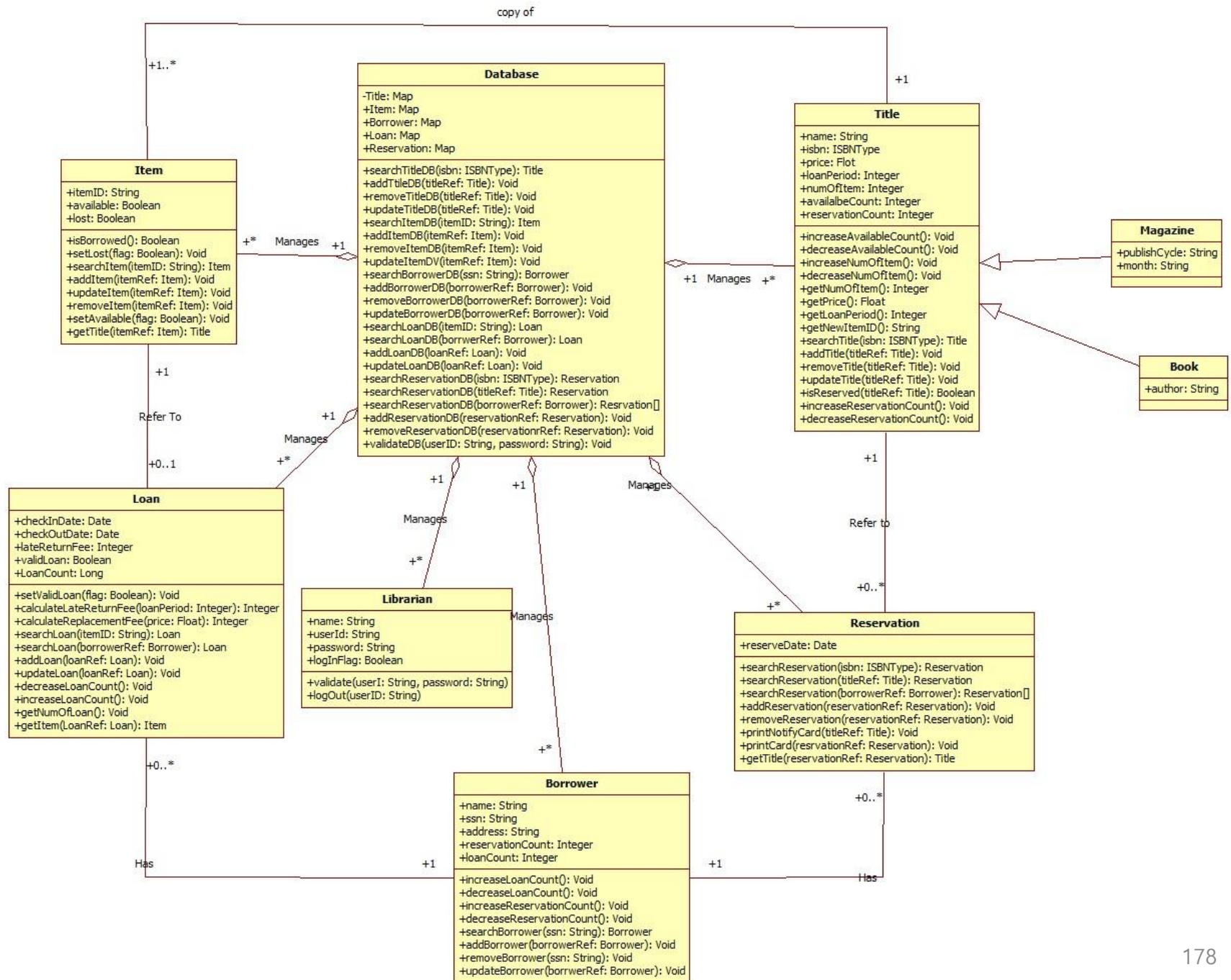
a. Varied order  
b. optional





## Phase 2044. Define Design Class Diagram

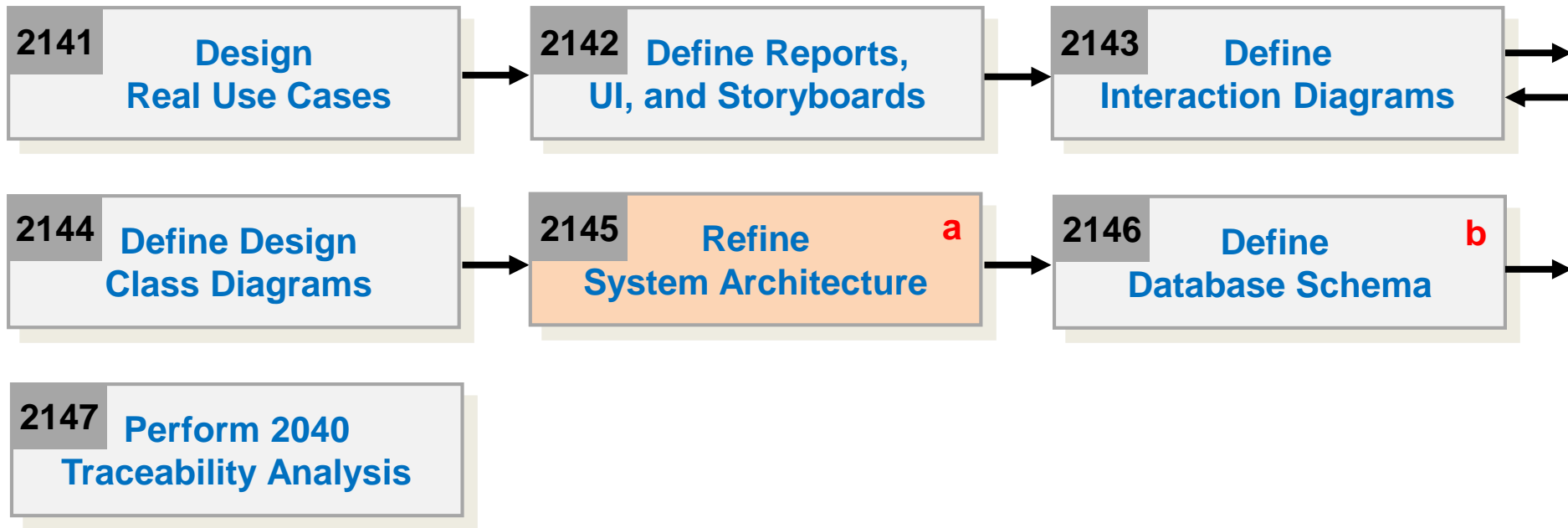




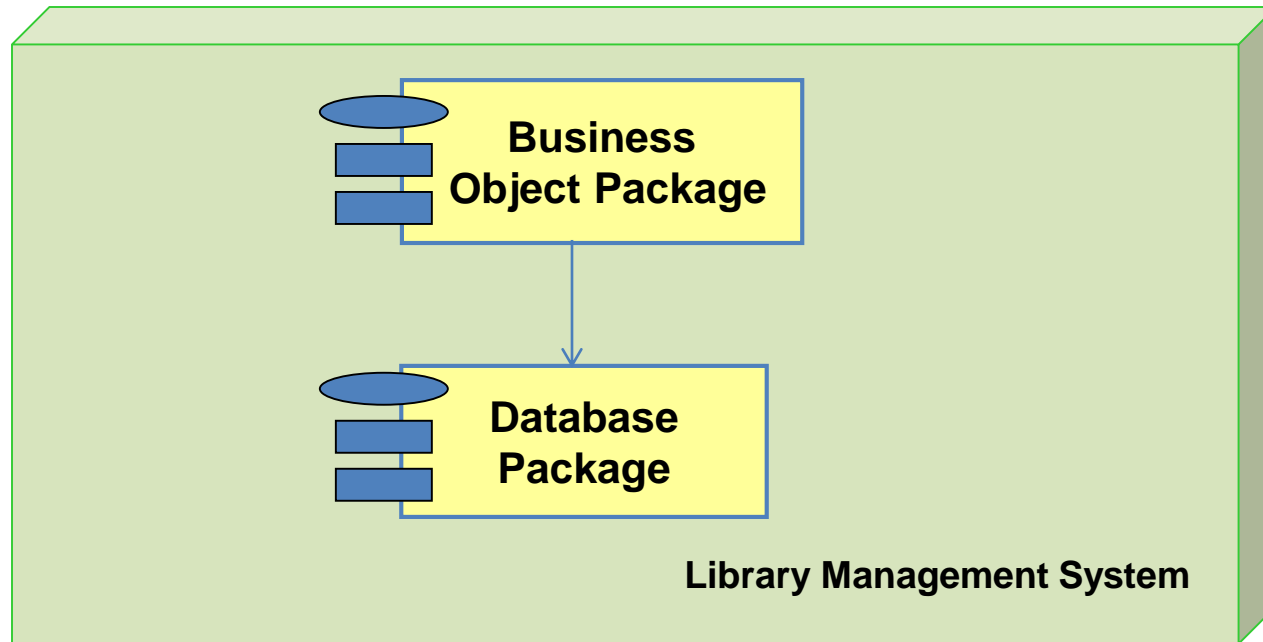
# Phase 2045. Refine System Architecture

- 7 Activities

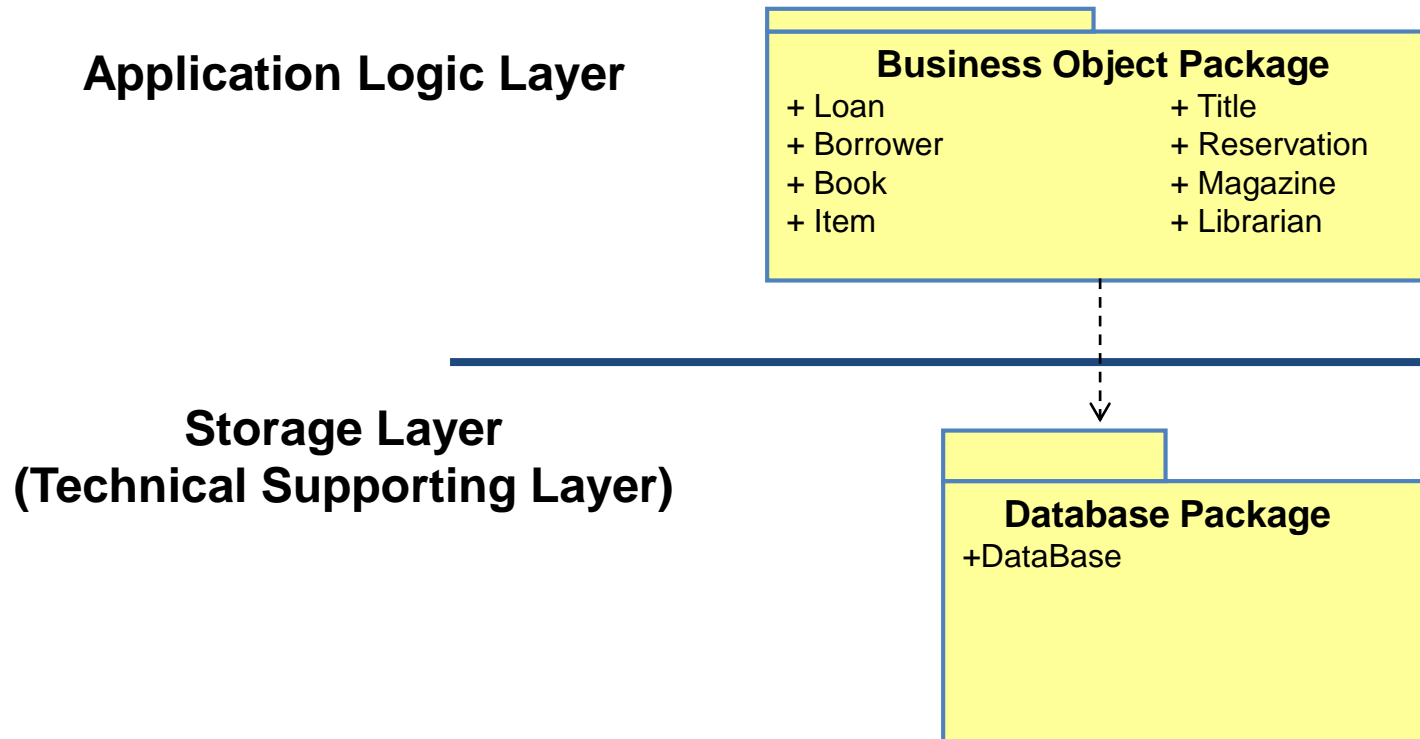
a. Varied order  
b. optional



# Phase 2045. Refine System Architecture



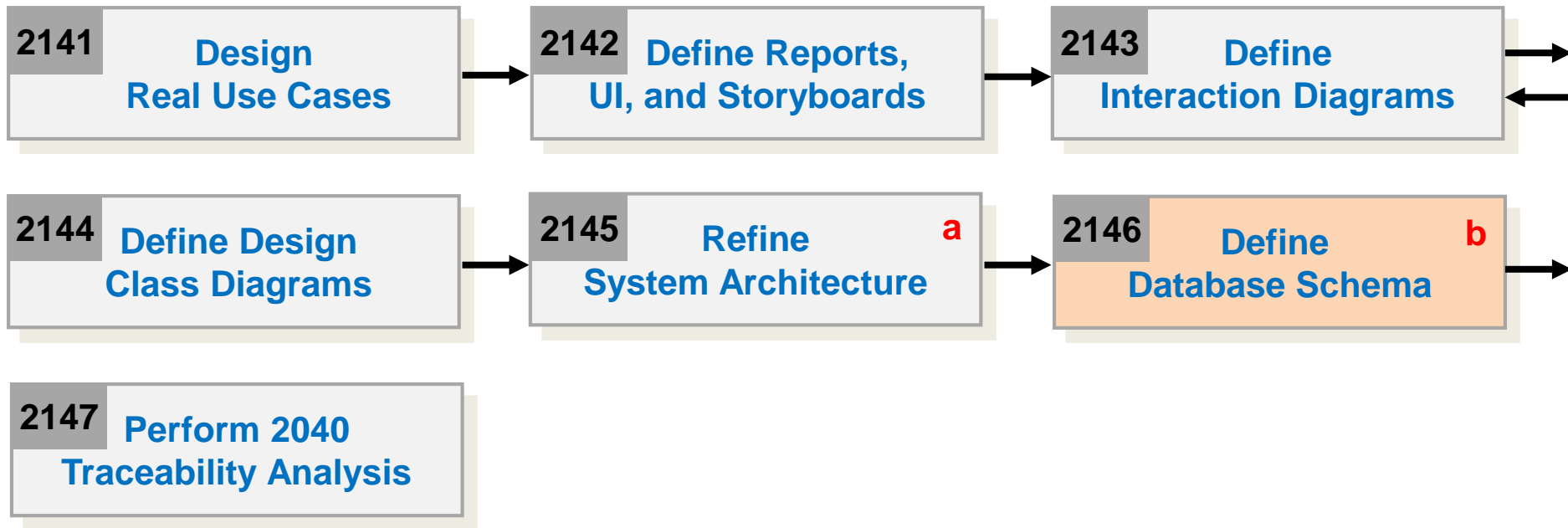
# Phase 2045. Refine System Architecture



# Phase 2046 Define Database Schema

- 7 Activities

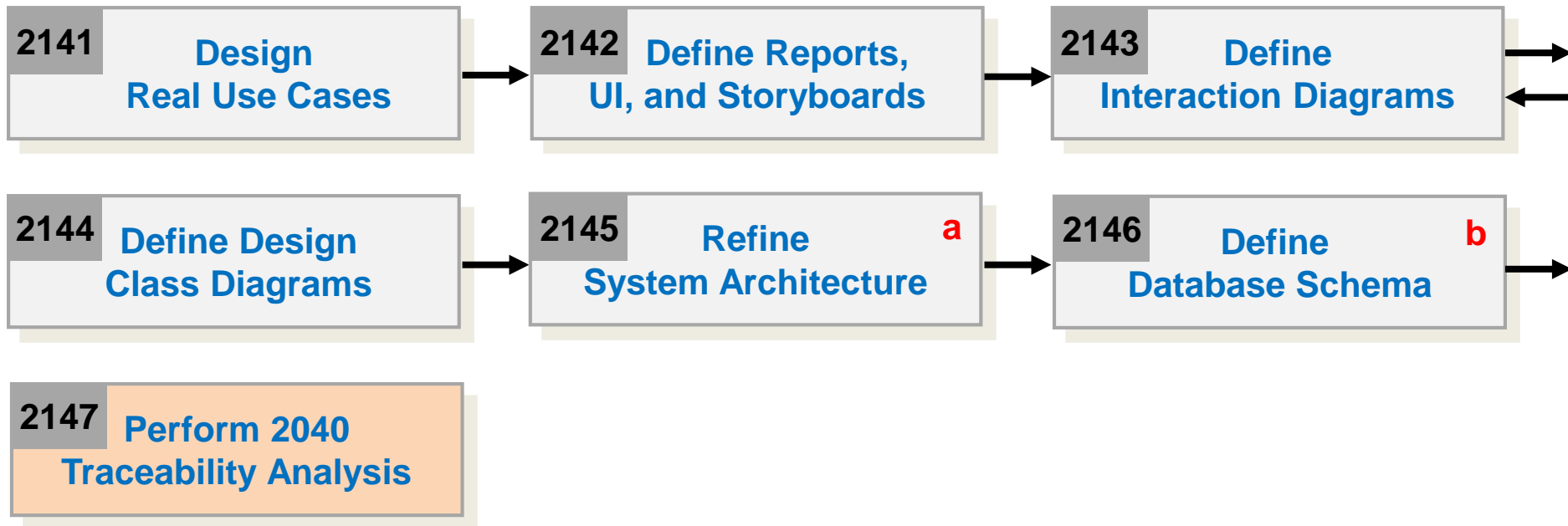
**a.** Varied order  
**b.** optional



# Phase 2047. Perform 2040 Traceability Analysis

- 7 Activities

a. Varied order  
b. optional





# Phase 2047. Perform 2040 Traceability Analysis

