Question 2: From Requirements to Design: Library System

Informal Description of the Library System (same as seen in class)

The system to build is an automated library book borrowing system that is to be used in the departmental libraries of a university. The goal is to relieve the librarians from processing book loans. The system does not require users to identify themselves to search for books according to certain criteria and to check the availability of a particular book. However, to check-out books and to check their respective book loan status, users must first identify themselves to the system.

A single receipt is printed for each user check-out session; it details for each book: the title of the book, the unique identifier of the book, the date the book was borrowed, and the date the book is to be returned by. At the start of each week, the system sends warning emails to all borrowers that have overdue books.

Books have physically attached barcodes, which are used for the identification of books that are checked out (a barcode scanner is to be used). If the book does not scan, it should be also possible to enter the barcode manually. Books that are on reserve are not available for loan.

Some Use Cases of the Library System

A partial use case model has been established for the library system. So far it only includes two use cases: CheckAvailability and RegisterNewBook.

CheckAvailability Use Case

Use Case: CheckAvailability
Scope: Library System
Level: User Goal
Intention in Context: The intention of the User is to query the system in order to determine if it is possible to take out a copy of a given book.
Multiplicity: Many Users can check availability of books at a given time. A User can not check for availability of several books simultaneously.
Primary Actor: User
Main Success Scenario:

1. *User* asks *System* if a given book is available for loan.
2. *System* informs *User* of availability of book.

Register New Book Use Case

Use Case: RegisterNewBook
Scope: Library System
Level: User Goal
Intention in Context: The intention of the *Librarian* is to enter a new book into the system.
Multiplicity: There can be only one *Librarian* registering new books at a given time.
Primary Actor: Librarian
Facilitator Actor: AdminTerminal
Secondary Actor: BarcodePrinter
Main Success Scenario:

Librarian steps in front of the AdminTerminal.
1. *AdminTerminal* informs *System* about title of the new book and the number of copies that were bought.
Steps 2 and 3 are repeated for each book copy.
2. *System* instructs *BarcodePrinter* to print a barcode for the copy.
3. *Librarian glues printed barcode to the book copy.*

Partial Environment Model of the Library System

Figure 1 shows a partial environment model of the library system based on the two use cases above. The messages and parameters are:

- isAvailableForLoan(b: Book)

- registerBook(title: String, nbOfCopies: Integer)

The output messages and parameters are:

- printBarcode(code: Integer)
- availabilityAck(yesNo: Boolean)

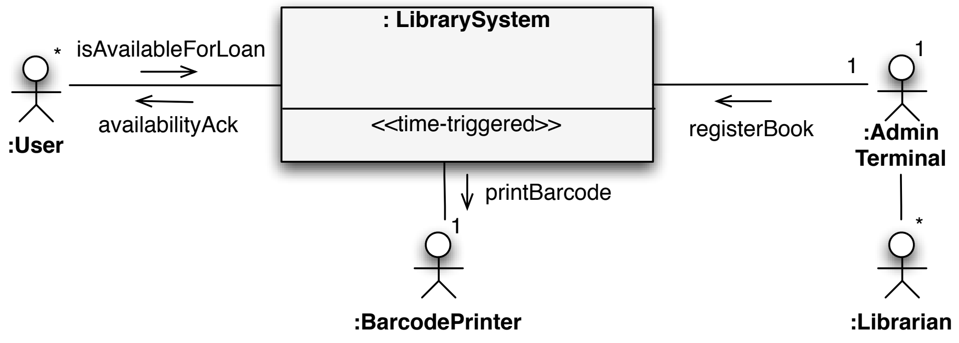

Figure 1: Partial Environment Model of Library System

Partial Concept Model of the Library System

Figure 2 shows a partial concept model of the system.

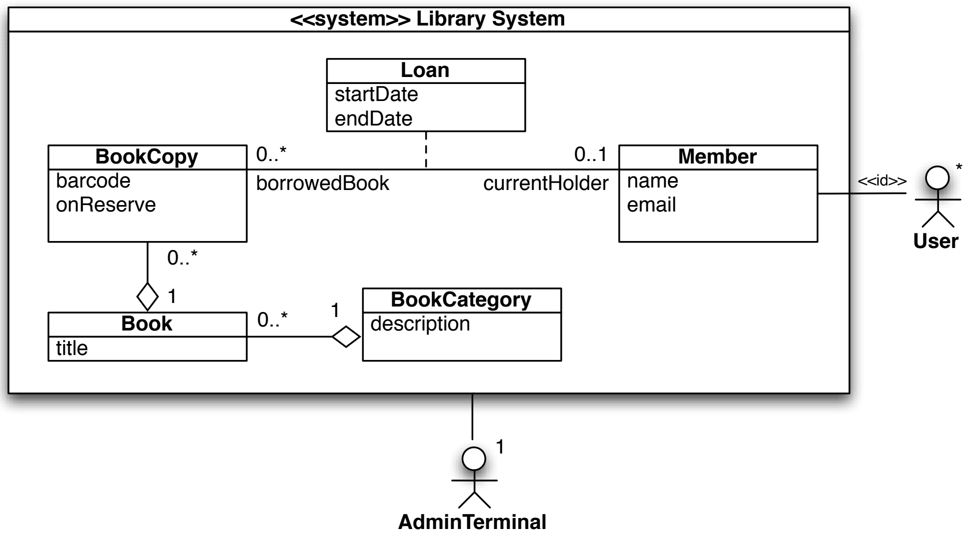

Figure 2: Partial Environment Model of Library System

Partial Operation Model of the Library System

A partial operation model describing the effects of the two system operations checkAvailability and registerBook is given below.

Operation: LibrarySystem::isAvailableForLoan(b: Book)
Scope: Book, BookCopy, Loan

Messages: User::{availabilityAck}
Description: If the book b is not on reserve, and there are currently copies in the library, i.e. not all copies are taken by other members, then the system sends a positive acknowledgment to the user that triggered the system operation. Otherwise it sends a negative acknowledgment.

Operation: LibrarySystem::registerBook(title: String, nbOfCopies: Integer, c: BookCategory)
Scope: Book, BookCopy
Messages: BarcodePrinter::{printBarcode}
New: newBook: Book, newCopies: Set(BookCopy)
Description: The effect of this operation is to create a new instance of Book and initialize it's attributes and associations. Also, the operation creates nbOfCopies instances of BookCopy and associates them with the new book. Each copy is assigned a unique barcode, and the BarcodePrinter is instructed to print the assigned barcodes.

Question 2.1

Elaborate a design for the system operation isAvailableForLoan. The Graphical User Interface (GUI) that allows a user to trigger the operation was designed by other developers. They have already made certain design decisions:

- They decided to implement the conceptual parameter b: Book of the system operation with a string containing the title of the book. You can safely assume that the GUI is not going to pass a string value that does not match a title of an existing book in your system, and that book titles are unique.
- They designed a class UserDisplay that can display messages in forms of strings to the user (operation display(message: String)). When the GUI invokes the isAvailableForLoan operation that you are designing, it will pass a reference to an instance of the UserDisplay class as a parameter. This will allow your design to display the acknowledgment on the correct user screen.

You can mode your design using a communication diagram or a sequence diagram, whichever you prefer. Please justify the choice of controller.

Question 2.2

Elaborate a design for the system operation registerBook. You can assume that all the book category objects representing categories that the system can handle already exist in memory. You therefore only have to decide how to implement the book category parameter (that the GUI is supposed to pass to you to identify the corresponding book category object).

You can mode your design using a communication diagram or a sequence diagram, whichever you prefer. Please justify the choice of controller. Make sure that this design is "compatible" with the design you did for question 1.1.

Question 2.3

Elaborate a design class model for your design, i.e. for the classes you used in question 2.1 and question 2.2. Remember that the design class model must show attributes and methods for each class you used in 2.1 and 2.2, as well as navigable associations with role names between them, if any. You can assume that template collection classes like the ones that Java provides are available in order to implement any multi-objects you used. Also, remember that the design class model does not need to have design-equivalents for all the classes shown in the concept model. You only need to show the ones that you are actually using in 2.1 and 2.2.

Question 2.4

Answer the following question for the design you presented in 2.3: Do any of your business classes (i.e. the classes that represent classes in the concept model) depend on classes that implement communication with the outside world? If yes, list them, together with the kind of dependency (permanent association, parameter dependency, instantiation dependency, call dependency).