Kevin Tang <kevin26428@gmail.com>
Library Transaction System

1. Algorithm:
    a. Display name, and email address
    b. Define variables for patrons and library items and transactions.
    c. Print out user menu and an admin submenu and take in input from user in a while loop
    d. Execute the correct function that user wants
        i. Quit
        ii. Add new patron
        iii. Edit patron
        iv. Remove patron
        v. Print list of patrons by id number
        vi. Print list of patrons with outstanding fines
        vii. Print list of library items a patron has
        viii. Add a new library item to the catalog
        ix. Edit a library item's information
        x. Remove item from catalog
        xi. Checkout a library item
        xii. Recheck a library item
        xiii. Return or check in a library item
        xiv. Compute the due date of a checked out item
        xv. Record that a patron has paid some money toward fines
        xvi. Print overdue items
        xvii. Print address mailing labels
        xviii. Load
        xix. Save
        xx. Admin menu- only for Librarians!
    e. After user exits application deconstructors are called for variables
2. Description of all data elements (classes, member variables):

Item Name	**Type**	**Used by**	**Description**
Patron	Class		Entity class
LibItem	class	The book class, CD, DVD, reference book classes	Entity class that will provide as the base class for all library items (DVD, CD, Book, Reference book)
Book	Class		Entity class

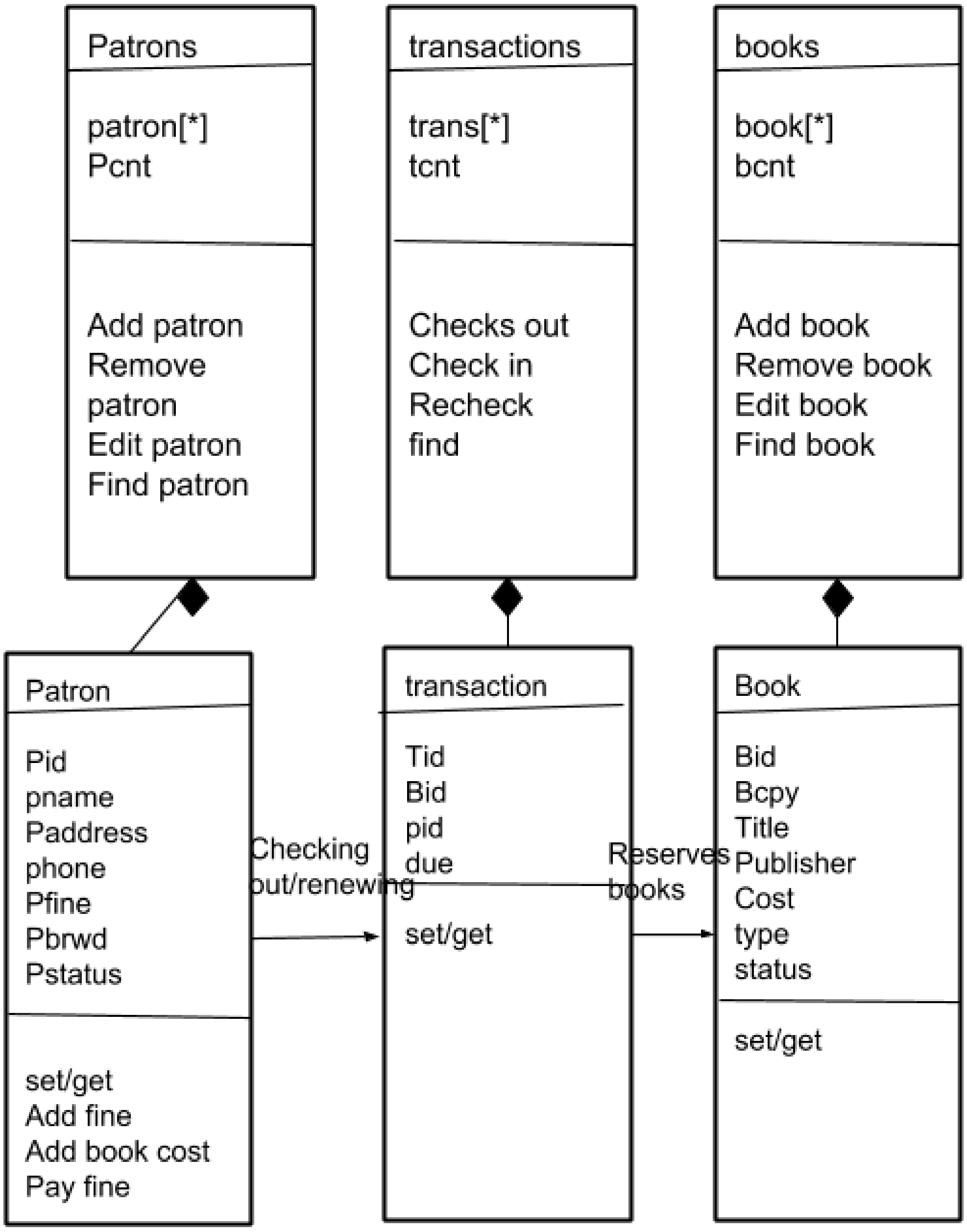
CD	Class		Entity class
DVD	Class		Entity class
Reference book	class		Entity class
Transaction	Class		Entity class
Patrons	Class		Collection classes
Books	Class		Collection classes
CDs	Class		Collection classes
DVDs	Class		Collection classes
Reference books	Class		Collection classes
Transactions	Class		Collection classes
pid	Int	Patron	Patron’s identification number
pname	string	Patron	Patron’s name
address	string	Patron	Patron’s address
phone	int	patron	Patron’s phone number
pfine	Int	patron	Patron’s fine balance
pbrwd	int	patron	Number of books borrowed
pstatus	int	patron	1 = good standing, 2 = blocked
set/get	function	patron	Sets or gets patron information
Add fine	function	patron	Add fines to fine balance
Add book cost	function	patron	Adds the book cost to fine balance
Pay fine	function	patron	Pays off fines from fine balance

Patron[ ]	vector	patrons	Patron memory location in a dynamic array
pcnt	int	patrons	Number of patrons
add patron	function	patrons	Add a patron
Remove patron	function	patrons	Remove a patron
Edit patron	Function	patrons	Edit a patron’s information
List patron	function	patrons	List all patrons in alphabetical order
List patron fine	function	patron	List all patrons with outstanding fines
Find patron	Function	patrons	Find a patron’s information
Pay fine	function	patrons	Finds patron balance and then pays fines
tid	int	transaction	Transaction identification number
bid	int	transaction	Book identification number
pid	int	transaction	Patron identification number
due	int	transaction	Due date of book
set/get	function	transaction	set/get transaction info
trans[*]	Dynamic array	transactions	Dynamic array of the transactions made
tcnt	int	transactions	Number of transactions made
Check out	function	transactions	Check out a book
Check in	Function	transactions	Return a book
recheck	function	transactions	Renew a book

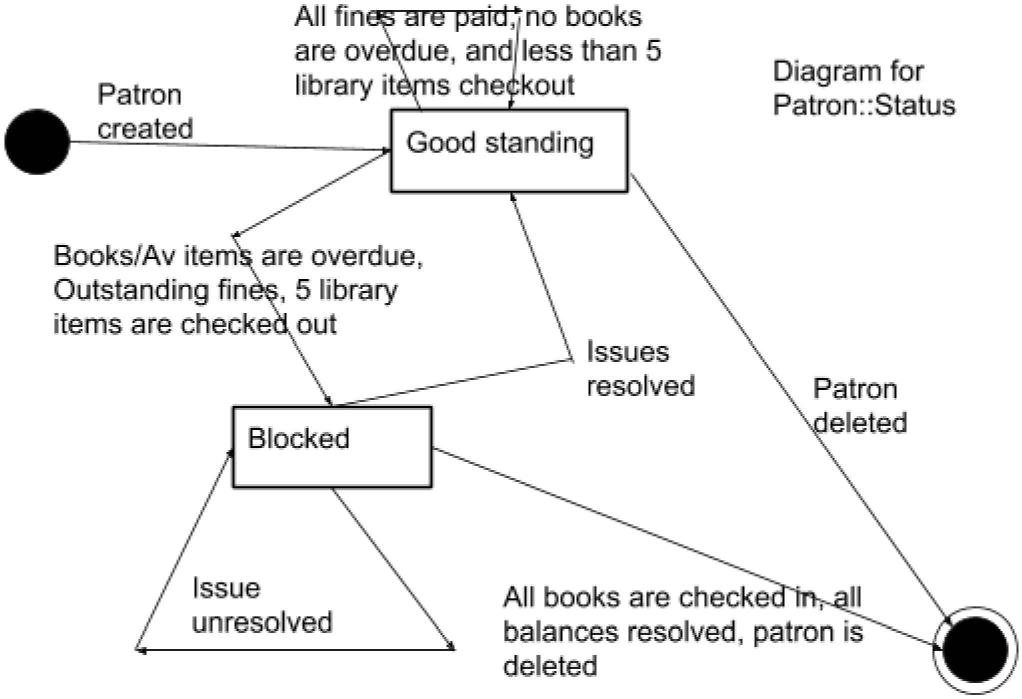
find	function	transactions	Find a transaction
Title	string	LibItem	Title of the library item
id	int	LibItem	Id of library item
cpy	int	LibItem	Copy number of library item
cost	int	LibItem	Cost of library item
Status	Int	LibItem	Status of library item: 0- checked in, 1-checked out, 2-lost
set/get	function	LibItem	Sets or gets information of library item
publisher	string	book	Book publisher/author
rechecks	int	book	Number of rechecks for a book (max 1)
set/get	function	book	set/get information of books
Book[ ]	vector	books	Dynamic array of books
bcnt	int	books	Number of books in catalog
Add book	function	books	Add a book
Remove book	Function	books	Remove book
Edit book	function	books	Edit book information
lost	function	Books	Reports a book as lost
Find book	function	books	Find book information

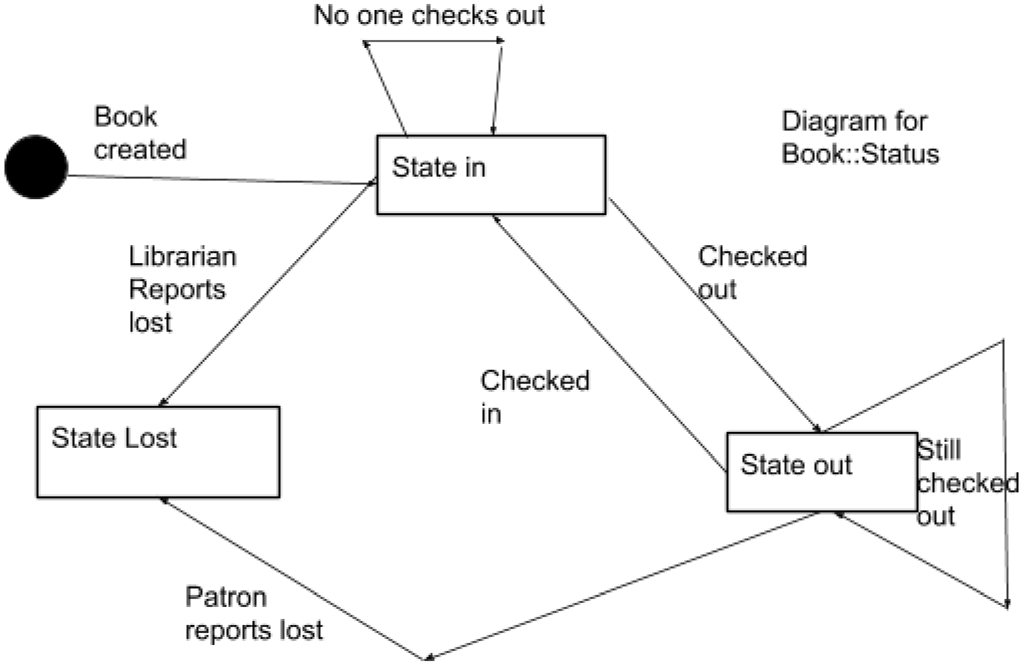
To avoid redundancy, I did not specify what was inside the references, cds, and dvds classes because they will be very similar to book class.

2. Class diagram of entities in the system



3. State diagrams for the status of book class and patron class


All fines are paid, no books are overdue, and less than 5 library items checkout
Patron created
Good standing
Diagram for Patron::Status
Books/Av items are overdue, Outstanding fines, 5 library items are checked out
Issues resolved
Patron deleted
Blocked
Issue unresolved
All books are checked in, all balances resolved, patron is deleted



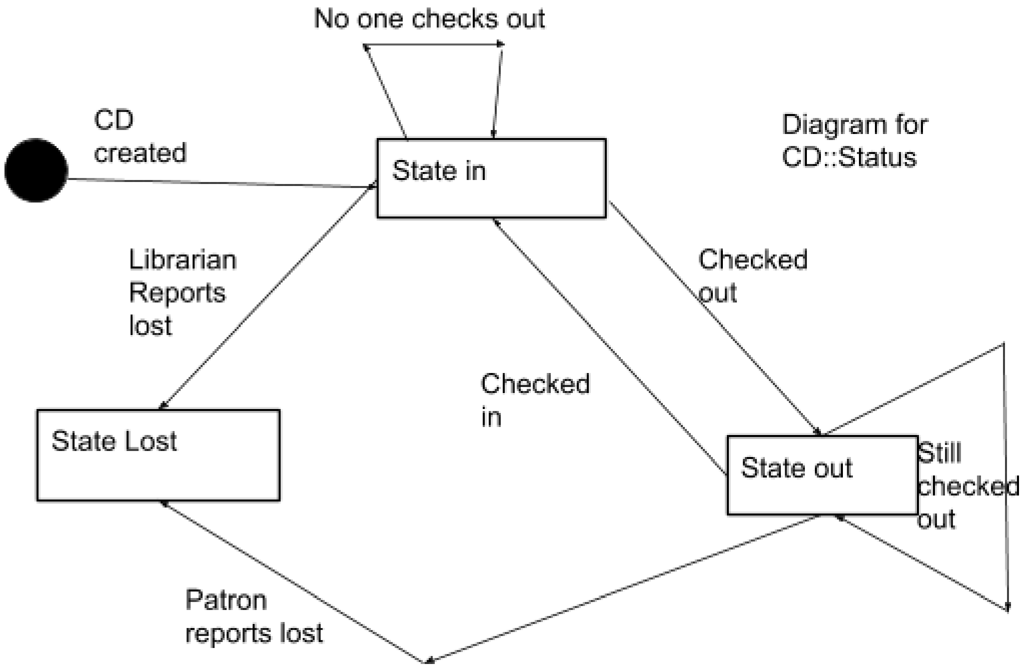


Diagram for CD::Status

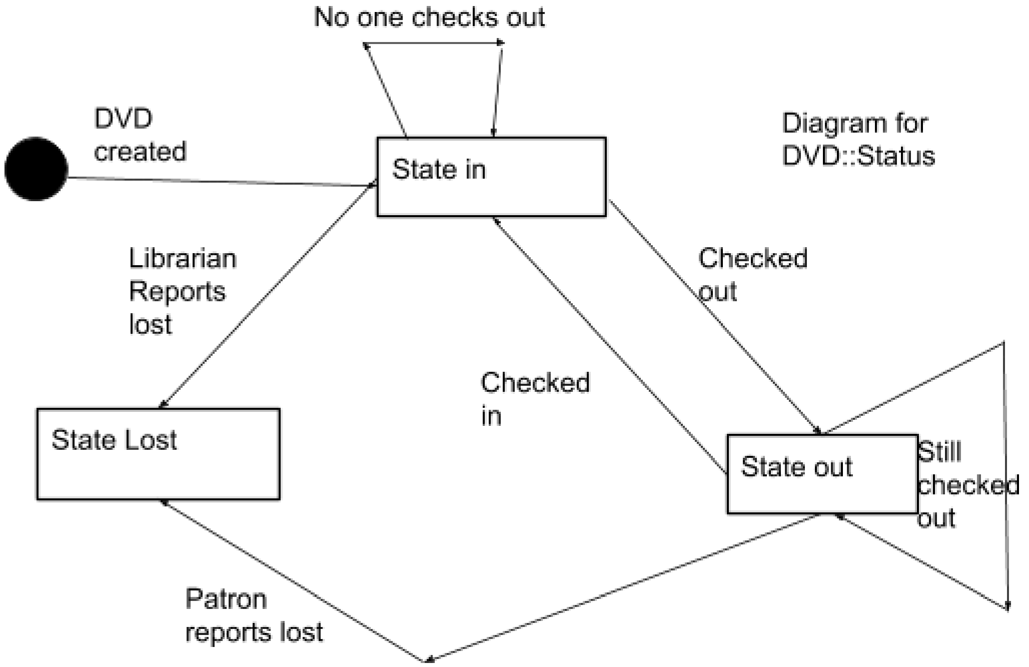
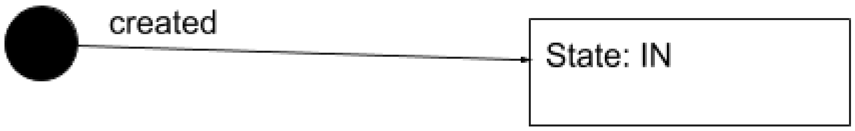


Diagram for DVD::Status

Diagram for Reference::Status



4. Approach to implement the system for limitless expansion

This system uses STL or vectors to handle expansion

5. Questions or issues about the system

A question I had was how would I continue to reallocate data to my dynamic array?

Do I need a third class that links books and patrons, or does 2 suffice?

6. Discussion how designing now can be incorporated into future plans and expansion

This design can be incorporated into future plans and expansion. I can easily build onto the variable list and the class list. The diagrams are easy follow and easy to add more classes/variables onto it. It'll be easier to implement because replacing variables becomes simple. This can also be incorporated into future systems that require a collection and a single class. This can also expand using inheritance.