CS-230 Software Engineering
Assignment One (A1)
Object-Oriented Software Design

Liam O'Reilly

Number of Credits: 40% of 15 credit module.
Deadlines:
Week 1's Minutes and Contribution Breakdown:
Monday 15th October at 11am
Week 2's Minutes and Contribution Breakdown:
Monday 22nd October at 11am
Week 3's Minutes and Contribution Breakdown:
Monday 29th October at 11am
Week 4's Minutes and Contribution Breakdown:
Monday 5th November at 11am
Final Submission:
Monday 5th November at 11am.
Learning Outcome: To gain experience in designing larger scale software in a group.

1 Overview

In this assignment, each group will complete an **object oriented design** of an application that manages a modern library called Tawe-Lib. This electronic system shall adhere to the detailed list of the functionality and requirement specifications provided in this document (Functional Specification). *Hold on to this document. You will need it for Assignment A2. For those of you on the Software Engineering course, you will need it for assignments in CS-235 as well!*

2 Functional Specification

The main purpose of Tawe-Lib is to manage a modern library, by for example, keeping track of resources such as books, allowing users to borrow and return resources, keeping track of fines, etc. Such a system would normally (these days) be run as a web application, but to make this assignment manageable, Tawe-Lib will run as a stand alone PC application. Only one user can be logged in at a time. All data will be stored in files on the local computer – no networking or web technology will be involved.

It is up to you to design the data structures, algorithms and GUI involved in realising this system.

2.1 Users

There are two categories of users within the system: normal users and Librarians.

Each user has the following information associated with their account:

1. A username that uniquely identifies their account.
2. A first name and a last name.
3. A UK mobile telephone number.
4. A full UK address complete with postcode.
5. A profile image.

Additionally, Librarians also have:

- Employment date.
- Staff Number.

You should design and implement classes to manage this information. Also, you should create an graphical user interface that allows Librarians to create new accounts.

2.2 Resources

Resources available in the library fall into various categories: Books, DVDs, and Laptop Computers.

Different types of resources have different information associated with them. However all resources have the following information associated with them:

- Unique ID.
- Title.
- Year.
- A thumbnail image.

Each Book also has:

- Author.
- Publisher.
- Genre (optional).
- ISBN (optional).
- Language (optional).

Each DVD also has:

- Director.
- Runtime.
- Language (optional).
- List of Subtitle Languages (optional).

Each Laptop Computer also has:

- Manufacturer.
- Model.
- Installed operating system.

You will also need to build a user interface for allowing Librarians to manage resources and copies (see Section 2.3). Librarians shall be able to create new resources and edit existing ones. When a Librarian creates a new resource (or edits an existing one) they must provide all the data (with the exception of optional data).

2.3 Multiple Copies

There are multiple (identical) copies of each resource available within the library. For example, there can be 20 copies of the resource (i.e., Book) *Java for Everyone by Cay Horstmann*.

Each copy of an item has a loan duration associated with it. This can be 1 day, 1 week, 2 weeks, or 4 weeks. When copies are borrowed no due date is set (see Section 2.6).

2.4 Browsing and Searching Resources

Users shall be able to browse all the resources within the library. That is, the user shall be able to look through the resources and have at least the unique ID, title and year displayed. The user shall be able to choose a resource and the system will provide more detailed information. For each resource, the user shall be able to see how many copies of the resource are in the system. For each copy, they shall be able to see if it is available or currently being borrowed (but not by whom).

Users shall also be able to search for resources based on entering partial information in a typical search mechanism. The user shall also be able to filter resources based on the type of resource to only show Books, DVDs, etc.

2.5 Borrowing Copies

Individual copies, not currently borrowed, can be borrowed by users by visiting the Issue Desk and interacting with a Librarian. Only Librarians have the authority to loan copies to users. The system will keep track of which user has borrowed the copy and the date it was borrowed. If the user borrowing the copy has outstanding fines on their account, or overdue copies, then the system will prevent any further items from being borrowed.

2.6 Due Dates and Requesting Items

When a particular copy of a resource is borrowed **no** due date is set. Instead the user may borrow the copy until someone requests it.

If all copies of a resource are currently being borrowed then a user may request that the first copy of the resource to be returned is reserved for them. They will be added to the resource's request queue. As copies become available they will be reserved for people in the queue (in queue order).

When someone is added to a resource's request queue the system will automatically set the due date of the oldest borrowed copy where no due date is currently set. The due date will be set to the latest of the following:

- tomorrow (the date after this request is being made); or

- the earliest date such that the borrower has had the copy for the loan duration. (This ensures that a borrower has access to the copy for at least the loan duration).

2.7 Returning Copies

Borrowed copies can be returned by users by visiting the Issue Desk and interacting with a Librarian. Only Librarians have the authority to process returns. Upon returning a copy, the system will:

1. Calculate if the returned copy is overdue. If it is overdue then a fine will be applied to the balance of the account as follows:
    - For Books and DVDs, the fine is £2.00 per day late, up to a maximum of £25.
    - For Laptop Computers, the fine is £10.00 per day late, up to a maximum of £100.
2. If the resource's request queue is empty then the copy will be marked as available.
3. If the resource's request queue is not empty then the copy will be reserved for the next person in the queue.

2.8 Paying Fines

A user may pay off current fines (fully or partial) by visiting the Issue Desk and interacting with a Librarian. Only Librarians have the authority to pay off fines. A user will be able make a payment between £0.01 and the full outstanding balance of the account (i.e., the outstanding total amount of fines). The balance of the account will be adjusted accordingly. If all the fines are paid off then the user can now resume borrowing items.

2.9 User Dashboard

Each user shall have their own dashboard where they can view the following:

- Borrowed Items. A list of items they are currently borrowing. This shall show the due date of each item (if set).
- Requested Items. A list of items that the user has requested, but are not currently available.
- Reserved Items. A list of items that they previously requested that are now available to pick up. They have been reserved for this user.
- The current balance of the account (i.e., the current amount of fines).
- Transaction History. A chronologically ordered list showing each transaction on their account. A transaction is either a fine (showing the date and time of the fine, the amount, the item that caused the fine, and the number of days the item was overdue) or a payment (showing the date and time of payment and the amount).

2.10 Viewing a Copies Borrowing History

Librarians shall be able to view a particular copy's borrowing/return history. This will show a chronological log showing each time the copy was borrowed or returned, by whom and the date and time of the event.

2.11 Viewing a Overdue Copies

Librarians shall be able to view a list of all overdue copies showing the borrower, and the number of days overdue.

2.12 Profile Images

Each user must have a profile image. Two kinds of profile images are supported:

- Avatars
- Custom Drawings

The system will have a set (say 5 or 6) built in Avatar images. The user can select one of these as their profile picture.

Alternatively, the user will be allowed to create a custom drawing. The system will provide the user with a built-in basic drawing environment that is similar to a paint program with at least the following functionality:

- Ability to draw straight lines

- Ability to draw a particle trace (i.e., clicking and dragging will draw "filled" circles at the location of the pointer).

Once you have the basic functionality working to a high standard, you are welcome to expand upon it to earn higher marks.

2.13 Statistics

The dashboard shall show users statistics about how many items they borrow per week, per month and per year. These will be displayed both textually and graphically.

2.14 Other Properties of Tawe-Lib

The Tawe-Lib must have the following property:

- Data shall be persisted across runs of the system. For example, if a book is borrowed or other changes made, then those changes are also present after the application has been closed and re-opened. You may save the data to local files when exiting the application. When you restart the application, the data will be loaded from the files. You may also use a database if you so choose.

Tawe-Lib should operate on a single machine. A user logs in to the program and performs actions which are saved locally to disk on that machine. The user logs out. A new user can then log in to view what has changed. When you start the implementation part of the project, please build the system up one step at a time: get one aspect working before starting another.

2.15 Extra Tawe-Lib Features

This section is not technically part of A1, but it will be part of A2. It will be helpful for you to plan for extensibility and hence know about this section.

To achieve top marks in A2, you will need to be creative. At a minimum, all the functionality of the Functional Specification should be completed to a high standard. All features should adhere strictly to the specification. You need to get all this working very well in order to get a low first class mark (in A2). In order to get higher marks, you would be required extend the implementation in a novel way. All extensions that do not violate the specification will be considered. Substantial extensions to the software, extra reading and learning, will be required to achieve a high first class mark (in A2).

For this assignment (A1), **do not include these extra features**. Simply design without the extra features, but be aware that they will be required for the implementation (in A2).

3 A1 Tasks

Your team is asked to provide a complete **design** for the entire system. This design must include at least one complete class hierarchy. Each team member is required to contribute to at least one class in the design. The designer of the class does not necessarily need to implement it in A2.

3.1 Design Document

Your Design Document proposes an object-oriented design for the *entire* application. In other words, your team incorporates the full functional specification into the design. The Design Document should be no more than 30 pages including text and diagrams.

The Design Document consists of the sections as detailed below.

3.1.1 Candidate Classes and Responsibilities

Provide a list of candidate classes and their responsibilities. This list is like the CRC cards in lectures but with a bit more detail. For each candidate class the team has identified, the following information (i.e., card) must be provided:

1. **Class Name** (in bold)
2. Rough idea behind the class.
3. Author.
4. SuperClass.
5. SubClasses.

6. Responsibilities: a list of services this class provides.
7. Collaborations: a list of classes with which the class communicates.

There should be one of these cards for every class that appears in your class diagrams (see Section 3.1.2).

3.1.2 Class Diagrams

After specifying the information in Section 3.1.1, draw the classes using UML Class Diagram notation. Your class diagrams must use appropriate arrows and UML syntax to represent relationship such as collaborations (i.e., associations, compositions, and aggregations) and generalisations/specialisations (i.e., inheritance).

You should carefully think about navigability and multiplicities when drawing the collaboration relationships. If you would like, you can use UML tools such as Papyrus[1] (or others).

When providing details about the attributes and operations, you must think carefully about type information and your design. The classes and methods should fit together and function to achieve the intended behaviour. Do not just add operations and collaborations without thinking about how the operations can actually carry out their jobs. You need to take time to think about what collaborations your classes need to have in order to provide the services they offer.

Each class in your design (See Section 3.1.1) must appear in at least one UML Class Diagram. Each class hierarchy identified during your design process must be depicted in a hierarchy in a UML Class diagram.

3.1.3 Hierarchy Descriptions

Each generalisation/specialisation (i.e., inheritance) relationship (see Section 3.1.2) must be accompanied by a short textual description that describes the "is-kind-of" relationship used. Make sure that your team provides justifications that backs up the choices. Make sure that abstract classes are distinguishable from concrete classes in your diagrams. Your team should be able to identify two (or more) class hierarchies.

[1] `http://www.eclipse.org/papyrus/`

3.1.4 Collaboration Descriptions

Each composition and aggregation relationship (see Section 3.1.2) must be accompanied by a short textual description that describes the "is-made-up-of" relationship used. Make sure that your team provides justifications that backs up its choices.

3.1.5 Operation Descriptions

For each of your 5 most complex operations (within your classes) you must provide a short paragraph explaining what the operation should do overall, how the operation can be implemented, and how it has access to the data it needs (e.g., through collaborations).

3.2 Deliverables

There are multiple deliverables during this assignment:

- **Weekly Contribution Breakdowns and Minutes**

  You must submit weekly Contribution Breakdowns (see CS-230 Assignment Overview document). These will include the Contribution Breakdown itself and the minutes of the weekly meeting where you create the Contribution Breakdown. These will help document the contribution of members and the progress over the duration of the assignment.

  The Contribution Breakdown is a physical paper submission. The minutes that accompany them will be submitted to Blackboard.

- **Final Submission**

  The final submission will take place digitally via Blackboard. The deadlines can be found at the start of this document. Please submit your files in the following formats:

  – Design Document (PDF) – see Section 3.1

  – Contributions Report (PDF) – see Section 3.2.3

You should attach both files to the group submission on Blackboard. Please do not zip the files.

All group members must review the final documents prior to submission. Each individual group member is responsible for understanding the entire contents of all documents. Submission indicates that all group members have read and approved the document unless a conversation that has been documented by your Academic Mentor indicates otherwise.

One member of each group, the **Secretary**, should lead the submission of the weekly Contribution Breakdowns and the final submission (i.e., the Design Document and Contributions Report) on behalf of the group. Points will be deducted for those submissions that do not follow the file naming conventions and required file formats.

3.2.1 Weekly Minutes

10% of A1 Each week you will write up the minutes of the meetings where you discuss and create the Contribution Breakdown for that week. The minutes will be submitted **each week via Blackboard as a PDF**, along with the Contribution Breakdowns via a **paper submission**, see CS-230 Assignment Overview document). The deadlines can be found at the start of this document.

Your minutes files must be in PDF format and be named *"XXGroupGNMinutes-yyyy-mm-dd.pdf"* where XX is replaced by either "CS" or "SE" depending on if you are a Computer Science Group or a Software Engineering Group, GN is replaced by your group number, $yyyy - mm - dd$ is replaced by the calendar date of the deadline for the minutes(with $yyyy$ replaced by the year, mm with the month, and dd by the day), e.g., *CSGroup3minutes-2018-10-15.pdf*.

The minutes format should follow the standard as set out in lectures.

3.2.2 Design Document

85% of A1 (25% presentation, 60% content) Your Design Document is to be included in your final submission to Blackboard. The file must be named *"XXGroupGNDesignReport.pdf"* where XX is replaced by either "CS" or "SE" depending on if you are a Computer Science Group or a Software Engineering Group and GN is replaced by your group number. PDF format is required. Word document format (.doc or .docx) is not acceptable.

3.2.3 Contributions Report

5% of A1 A Contribution Report is to be included in your final submission to Blackboard. This document contains a description of what and how each group member contributed (overall) to the project design. The file must be named *"XXGroupGNMemberContributions.pdf"* where XX is replaced by either "CS" or "SE" depending on if you are a Computer Science Group or a Software Engineering Group and GN is replaced by your group number.

Each group member is obliged to contribute at least one class. The intention is that the classes each student chooses to design are also the classes they implement (in A2). However, students can change between assignments if necessary.

The Contribution Report, no more than 5 pages, describes who contributed to classes, hierarchies, and other contributions, e.g., the minutes etc. The Contribution Report is also written collectively. In other words, each group member describes their respective contribution to the project.

The Contribution Report also informs the reader if any unexpected problems arose during the course of the assignment. For example, if a group member skipped too many lectures, and as a result didn't have the background necessary to contribute, this should be stated in the Contribution Report. Likewise, if the group experienced success in some areas, both expected and unexpected, this should be included.

3.2.4 Overall Marks

The 3 components above (Weekly Minutes, Design Document, and Contributions Report) will be used to produce a Group Mark. You will submit 4 Contribution Breakdowns over the duration of this assignment. The average of the weekly contribution scores will form each student's own Contribution Score. The final individual mark (for A1) for each

student will be calculated by weighting the Group Mark with their Contribution Score.

Academic staff can overrule marks and change the process of peer assessment for a group in conjunction with the Year Head in atypical situations.

4 Issues with Contribution

Assignment 1 (A1) and Assignment 2 (A2) assess the performance of the group when delivering a non-trivial piece of software. As mentioned in the course, real software is developed in teams and therefore group work is a necessary skill that needs to be developed. Individual contribution levels will be taken in to account via the Contribution Breakdown system and also at the end of term when the groups are interviewed and each member has the opportunity to demonstrate their understanding of the course material in reference to the project.

Under normal circumstances, the Contribution Breakdowns and the interview will be used to adjust marks and determine the level of contribution. However, we realise that abnormal circumstances during group work may occur.

In the event of any non-contribution, please refer to the CS-230 Assignment Overview document and also follow this procedure:

- As early as possible, discuss the situation with your Academic Mentor.
- Work with your Academic Mentor to try and get the non-contributor participating.
- Keep your Academic Mentor posted on the situation.

Likewise, if any particular group member disruptively dominates group meetings and the assignment outputs in a way that is damaging to the team, the Academic Mentor should be notified immediately.

5 Project Hints

- Start early. This cannot be over emphasised.
- All group members are expected to contribute to all phases of the development including design and implementation. The author fields in both the class design and implementation help reflect individual group member contributions.
- Ask questions in your Academic Mentoring sessions.
- Do not email the module lecturer with questions, use the Blackboard forum where possible.
- You could use Discord (`discordapp.com`) or Slack (`slack.com`) for communication as a group. Many software development teams and companies use systems such as these a main form of communication.
- You could Skype for some of your group meetings?