FRAPPER
Colin

Assignment 2 - Analysis, Design and Implementation

02/17/2017

Note : I'm sorry about my english, I'm an Erasmus student.
Note 2 : All diagram are realise with draw.io . It's an open source.

Task 1 - Analysis

Subtask A - Identifying Use Cases

Task : The UML way of documenting requirements is through use cases and in this first task you are to identify and document the use cases used in the system. When documenting a use case, it is customary to specify initiation, pre and post conditions, the primary flow as well as some additional ones.

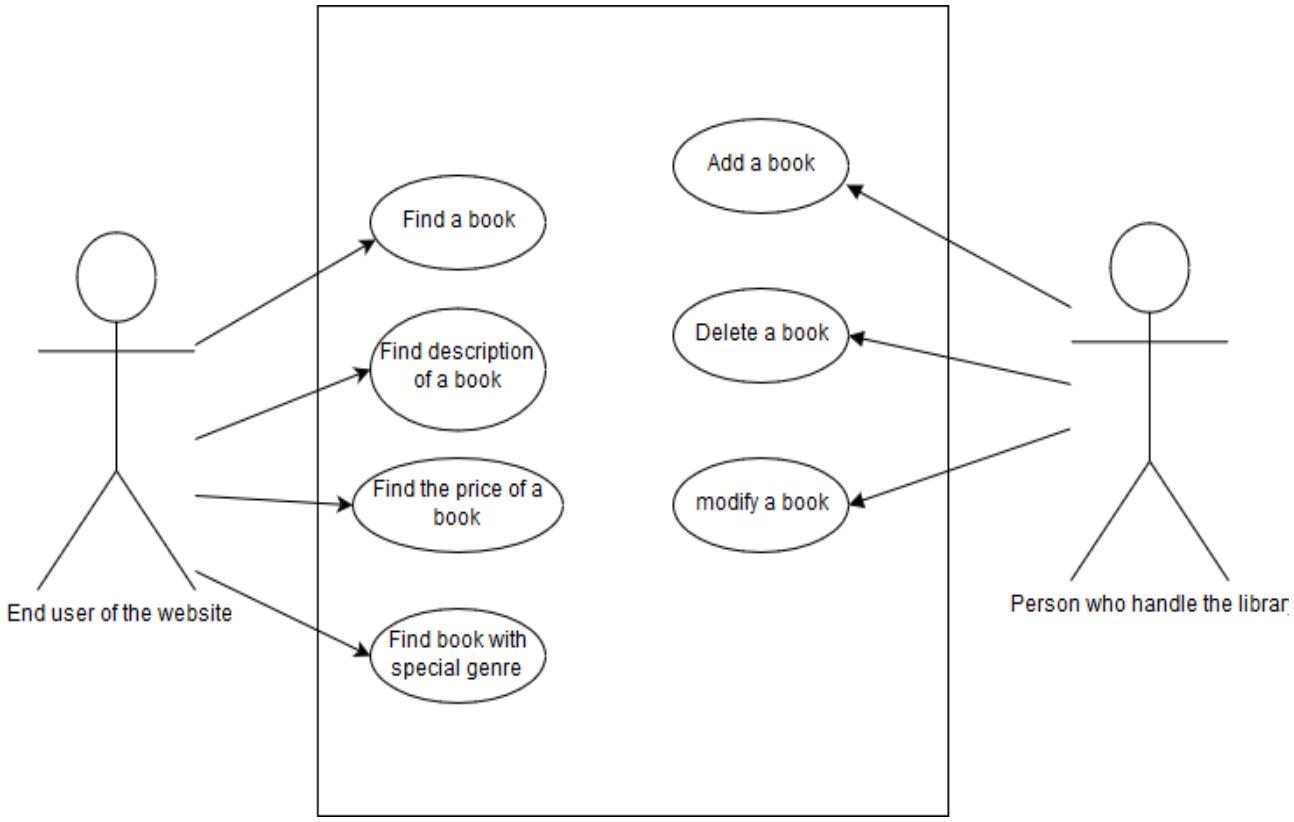

Figure 1 : Use cases model

Use-case : < Modify a book >

1. Brief description

Only the person who handle the web library (let's call it PWL) can modify a book. Someone asking (often the author) to change for example the price of the book to PWL, then PWL will enter the modification and save it. The PWL click on a book, then change information then save.

2. Triggers

1. PWL receive an email from the author saying that there is some change in the description of the book
2. PWL made a mistake and enter wrong information when adding the book and need to modify the book.

3. Actor Brief descriptions

1. PWL (The person who handle the web library)
2. The author of a book who have been added to the web library (a part of the end user).

4. Preconditions

1. The book who will be modify exist in the web library

5. Basic flow of Events

1. The use case begin when PWL receive information saying that a book should be modify
2. PWL will search the book who should be modify
3. PWL find the book
4. PWL modify the book
5. PWL save changes.
6. Book is modify.

6. Alternative flows

6.1 If in step 5, the PWL can't save changes. (problem system, code is wrong for saving or whatever)

1. PWL call a developper saying that he can't save change
2. The developper fix the problem by coding.

3. The problem system is fixed.

4. The use case resume at step 6.

7. Subflows

S1 : Author call/email the PWL saying that finally he changes his mind and didn't want to modify the book.

8. Key scenarios

8. 1 Book is modify

1. The PWL receive an information from an end user saying that the description of a book is wrong

2. The PWL will look at the book and see if the some information are wrong

3. The PWL will find the book

4. He modify the book.

5. Save changes

8.2 Author asking to change description of his book.

1. The PWL will find the book

2. He modify the book.

3. Save changes

9. Post condition

1. Book is modified

Use-case : < Add a book >

1. Brief description

Only the person who handle the web library(let's call it PWL) can add a book. Someone asking (often the author) to add his book to the web library. Click on new Book, fill information then save.

2. Triggers

1. PWL receive an email from the author saying that they want their book added to the web library.

3. Actor Brief descriptions

1. PWL (The person who handle the web library)
2. The author of a book who have been added to the web library (a part of the end user).

4. Preconditions

1. The book didn't exist already in the web library

5. Basic flow of Events

1. The use case begin when PWL receive information saying that an author wants his book added in the web library.

2. PWL will setup all the information needed with the person who want his book added.

3. PWL add the book will all information.

4. PWL save changes.

5. PWL call/email the author saying that his book has been added.

6. Alternative flows

6.1 If in step 4, the PWL can't save changes. (problem system, code is wrong for saving or whatever)

1. PWL call a developper saying that he can't save change

2. The developper fix the problem by coding.

3. The problem system is fixed.

4. The use case resume at step 6.

6.2 If in step 5, the author is not anymore agree with the description and ask to the PWL to modify the book

1. Go to the use case modify a book.

7. Subflows

S1 : Author call/email the PWL saying that finally he changes his mind and didn't want add his book

8. Key scenarios

8. 1 Book is added

1. The PWL receive an information from an author saying that a book should be add
2. The PWL will look at the description of a book
3. The PWL will add the book
4. Book is added.
5. Save changes

9. Post condition

1. Book is added.

Task : Write down your own reflections on identifying use cases and documenting them in about 100 words

Why use case is useful ?

Identify use case is very important because it's a better form of documentation, each use case has a description and what should be the result of this use case, it's permit to have a description of what we should have once the use case is finish. It's also a good way to avoid the issues and be prepared to them because it's almost sure that they will be issues during the development of the project.
Use case also improve communication between team members and I think the communication is the key to lead to a good project, it's pretty easy to understand and you can refer it when there is a problem.

If a use case is complete, you should know and prepare to what to do in case of problems. A use case is here to tells a story of how to fulfill a goal, or a set of stories of both reaching and failing

The problem here is that I never made a use case, and I think you have to do a lot of project and get experience to make a good use case and a complete one. With experience you will know what problem will come and how to deal with them.

Subtask B Robustness Diagrams

Task : Create robustness diagrams for the use cases you identified in task a.

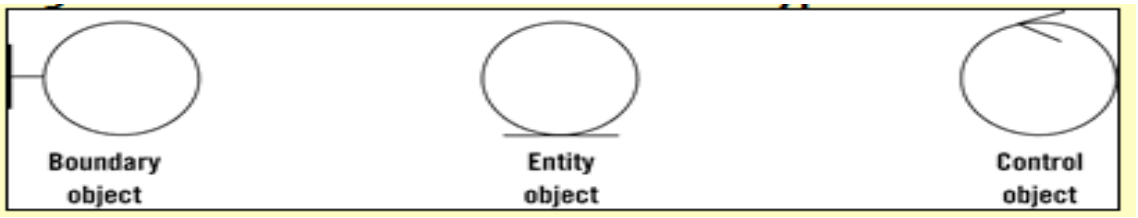

Figure 2 : Definition of robstness diagram

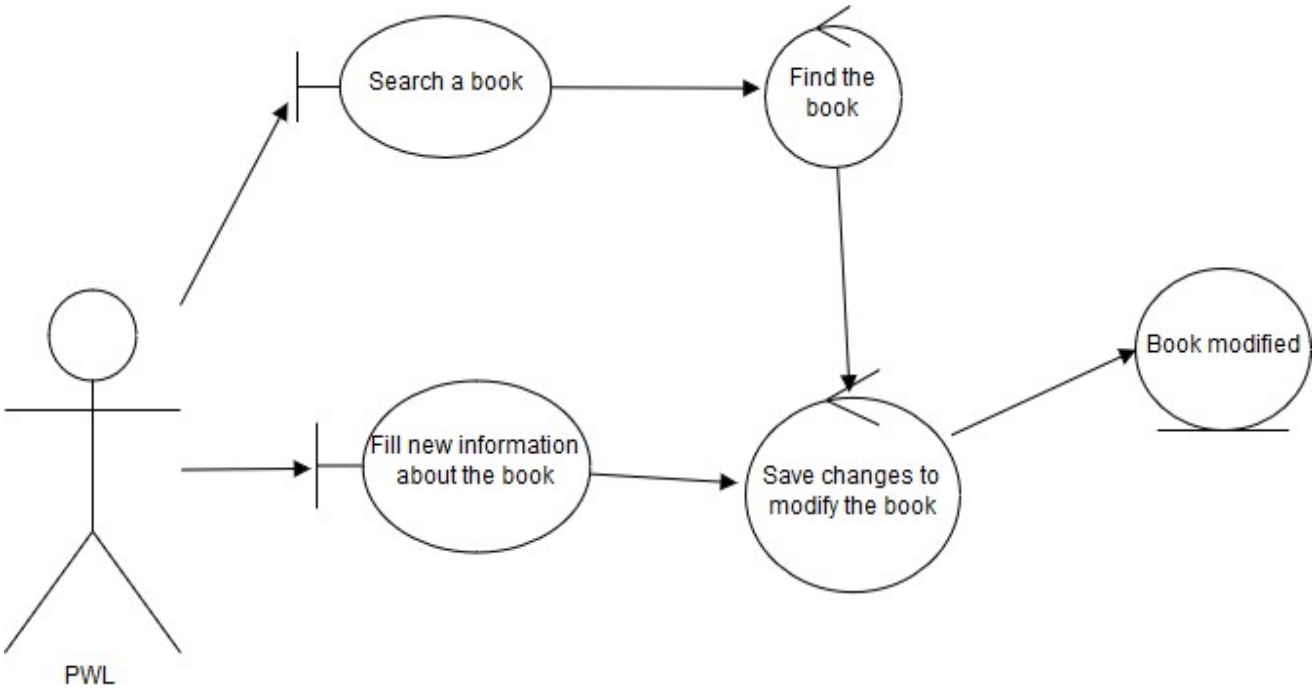

Figure 3 : Robustness diagram <modify a book>

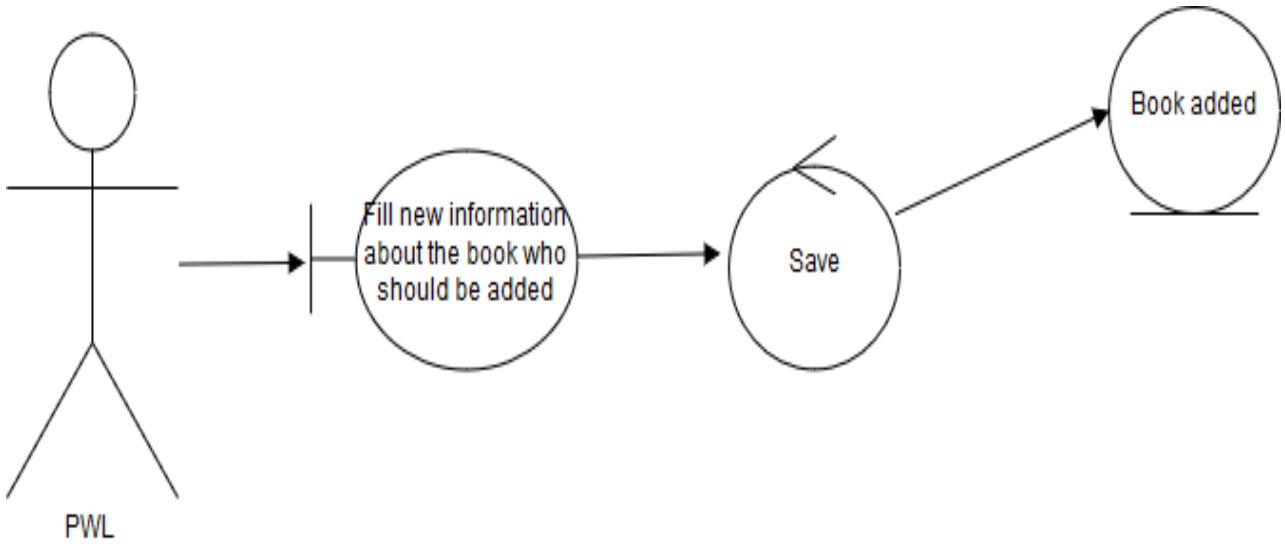

Figure 4 : Robustness diagram <add a book>

Task : In your reflection document you write down about 100 words on your experience using robustness diagrams.

Why robustness diagram is useful ?

Clearly robustness diagram is a simplified UML communication/collaboration diagram using ‘stereotyped objects’.

Robustness diagram will help to make the connection and bridge the gap from analsis to design. It's also a way to analyse use case text. It's like a picture of use case, and it ensure that the use case you made is correct. It permit to see what actors use when communicating with the system (the boundary object). The entity objects are the thing we need to keep tracking “the final result” and between them there is the control objects often implemented as methods.

As I said robustness diagram is here to make the connection from requirements to design, for example discover the objects you need to implement the use cases, and decide which methods belong to which objects.

In this project (and because it's a “small” project), it will not be that useful but I can imagine that for a huge project with lot of actors it can be really interesting to realise what method should be implement and what the actor will do, what should be the “end product” (entity object).

Subtask C Use Case Realization

Task : Show a use case realization of that in the form of a sequence diagram for the use case list books. In addition to that, do the same for the use case "Delete Book".

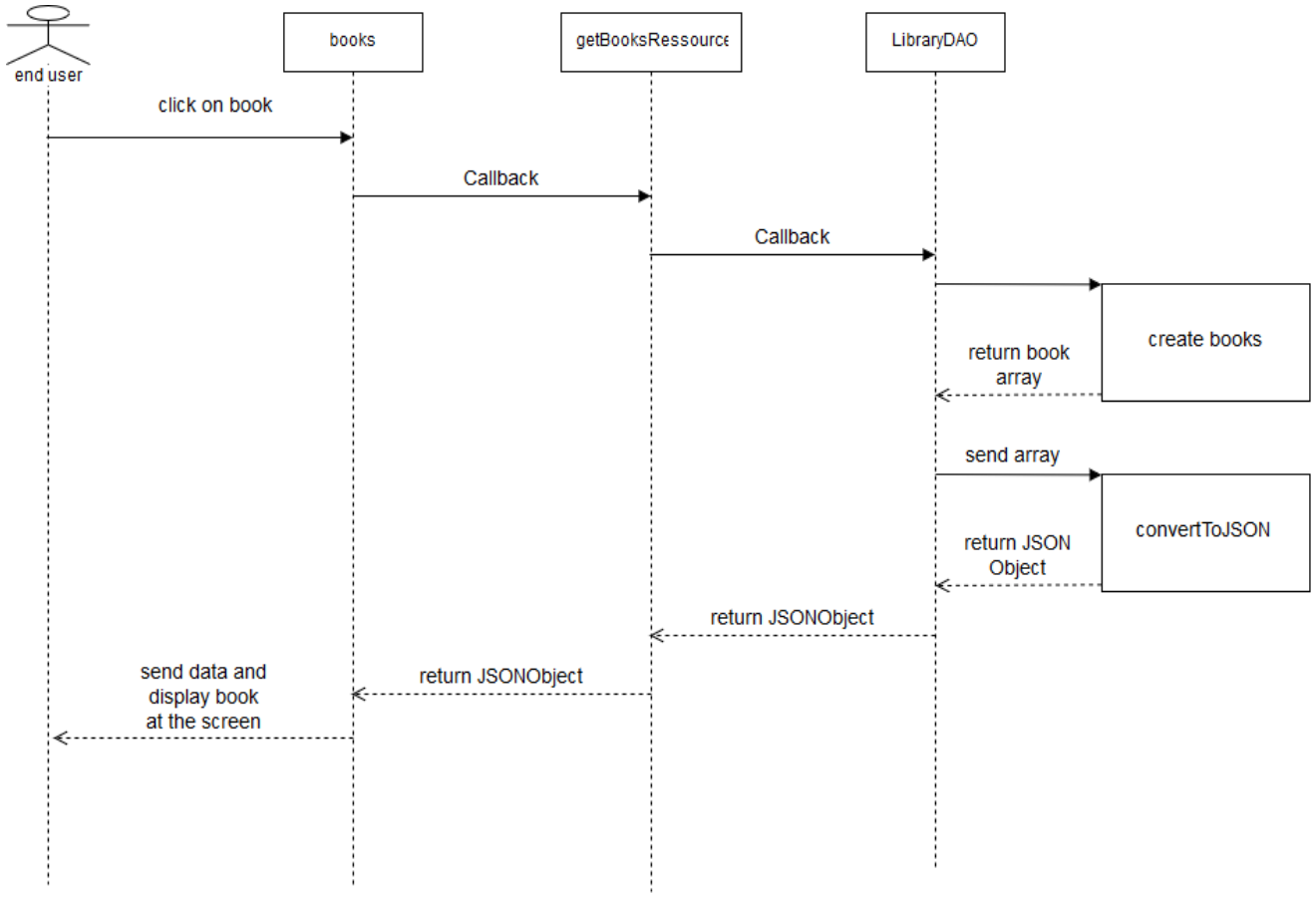

Figure 5 : Sequence diagram of the use case list books

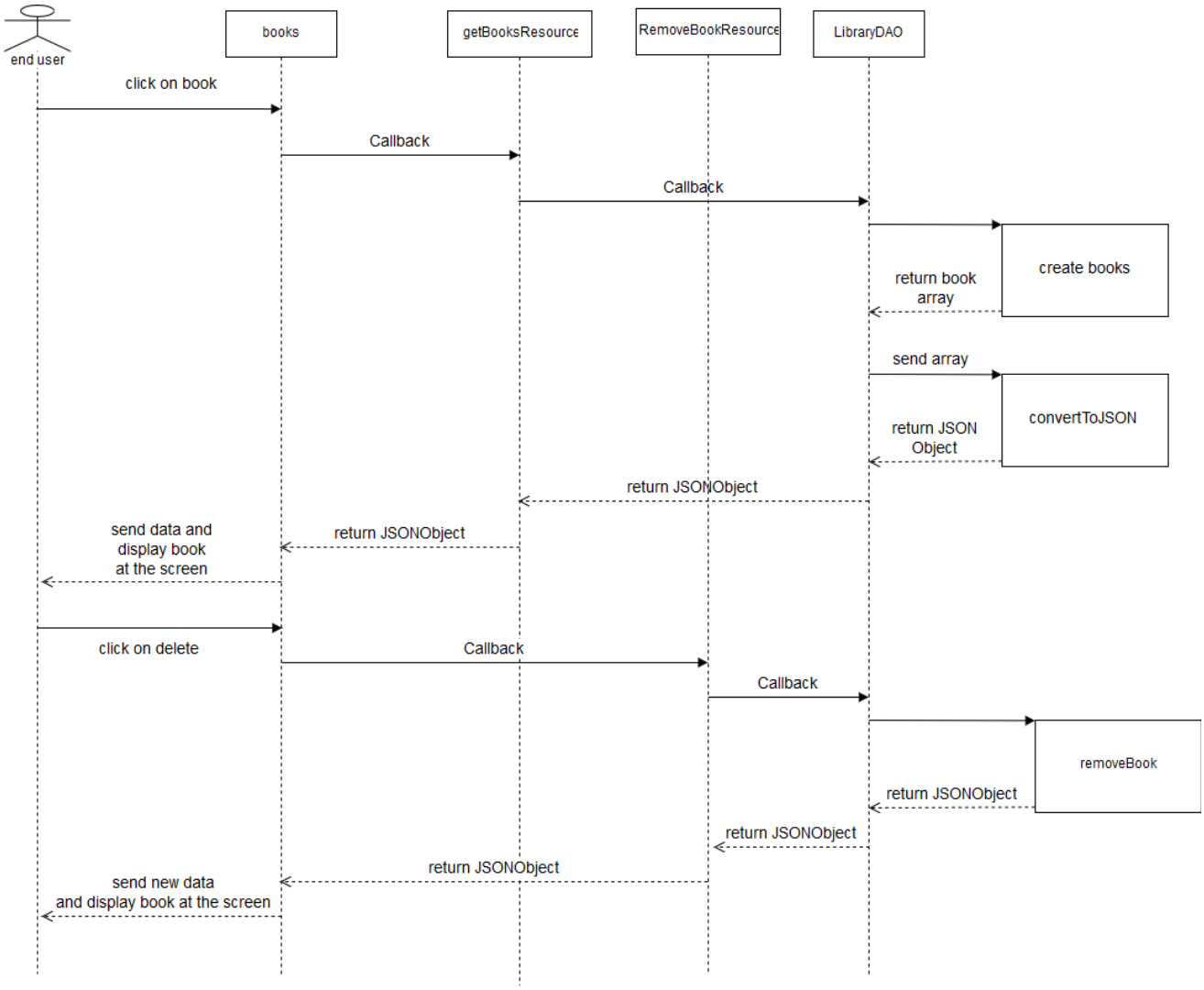

Figure 6 : Sequence diagram of the use case delete a book

Task : Write down your reflections on use case realisations in about 100 words.

Why sequence diagram is useful ?

The sequence diagram is an interaction diagram that shows how objects operate with one another and in what order. A sequence diagram permit to show how objects operate with another and what is the “path” of a use case.
It permit to show how the user interact and which method will be call throught the “timeline life” which can be really interesting to identify which method will be call, it's a part of the analysis before the design/coding of a project.

Task 2 - Design

Task : After reading the XML file it should be converted into objects in the running system and lastly translated into JSON for the web browser. Notice that this is a design task and therefore suitable design diagrams from UML are to be used to describe your solution (i.e. not code). You need to identify objects used in this and what messages they are sending, and when.

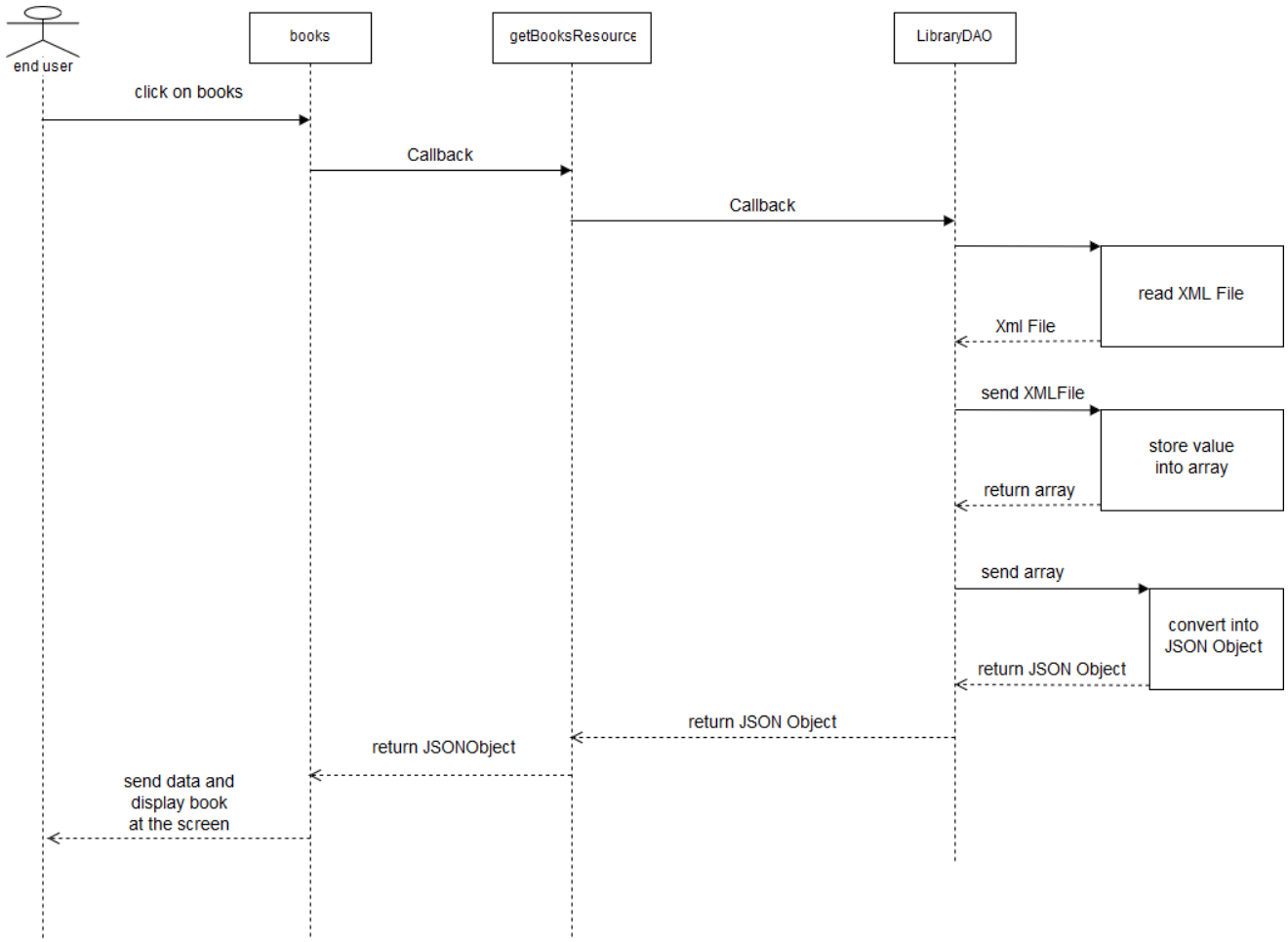

Figure 7 : sequence diagram

Task : As with the previous tasks, we also like you to write down your reflections in about 100 words.

This kind of task permit to before starting coding see how the implementation will be, and which method wo should use, should be call etc. It's very important to have this kind of diagram even if he will (can) change during the implementation and if so, it means that he wasn't that good or complete and so you must think why I did wrong to improve your skills in term of UML, for example asking yourself what I did wrong or what should I do to improve those diagram. Here the goal is to identify the object used, what and when messages should be send.

Task 3 – Implementation

Task : Take the design you created in task 2 and implement it in the system. In addition to that, also implement "Delete Book" without first designing it. Describe your reflections on the two approaches in about 100 words.

First, when I see how much time I spend on this task and the result, there is a problem. I tracked my time with toggl, in the time log we can see that I spent a lot of time on this task and few time and the previous task, and I think I made a huge mistake by not taking enough time to make the design in the previous and even now I realise that the diagram was false but this is how I can improve but next time I will take more time to make the design.

Conclusion :

Well, this project was interresting, I discover how work UML and how powerful the differents diagrams can be, even if in this project and because it's not a huge project it's not that "useful". But I imagine that it's really important to first think about the implementation before starting coding and that kind of diagram can help you to have a global vue of what methods who will gonna use and which method will be call, in which class. UML is the perfect "tool" before starting coding.

Also that was the first time I used toggl and I must say it's pretty impressive to see (you can find it in the time_log.pdf) how much I spend time COMPARE to how many time I spent on the diagrams. And I think it's a mistake from me, and next time I will try to spend more time on the diagram to think what I can improve. Anyway, was really interresting and now I must learn from my mistake for the third assignment.