

# Requirements and Analysis Document

Table of contents

## [1 Introduction](#)

### [1.1 Purpose of application](#)

#### [1.1.1 Stakeholders](#)

### [1.2 General characteristics of application](#)

### [1.3 Scope of application](#)

### [1.4 Objectives and success criteria of the project](#)

### [1.5 Definitions, acronyms and abbreviations](#)

### [1.6 Licensing](#)

## [2 Proposed application](#)

### [2.1 Overview](#)

#### [2.1.1 Commit standard and syntax for traceability](#)

### [2.2 Functional requirements for base functionality](#)

### [2.3 Non-functional requirements for base-functionality](#)

#### [2.3.1 Usability](#)

#### [2.3.2 Reliability and security](#)

#### [2.3.3 Performance](#)

#### [2.3.4 Supportability](#)

#### [2.3.5 Implementation](#)

#### [2.3.6 Verification](#)

#### [2.3.7 Packaging and installation](#)

#### [2.3.8 Legal](#)

### [2.4 Functional requirements for add-on functionality](#)

### [2.5 Application models](#)

#### [2.5.1 Scenarios](#)

#### [2.5.2 Use case model Use cases priority](#)

#### [2.5.3 Static model](#)

#### [2.5.4 Dynamic model](#)

#### [2.5.5 User interface](#)

### [2.6 Test cases and test plan](#)

### [2.7 Possible future directions](#)

### [2.8 References](#)

*Version: 1.3*

*Date: 2012-04-16*

*Authors: Oscar Brodefors, Filip Askviken, Rikard Andersson,  
Emil Nyström*

*This version overrides all previous versions.*

## **1 Introduction**

When attending courses at Chalmers you tend to need books. These books are expensive and mostly useful only under the course of the course. At this moment the authors of this document have at least 10 books at home, collecting dust. These could be useful to other students who are now taking the courses connected to those particular books. Wouldn't it be swell if we could find those students and sell them our

old books for a lower price than it would cost them to buy new once, preferably with minimum work effort?

## **1.1 Purpose of application**

A simple application for smartphones where students can gather to sell and buy used books needed at courses given at Chalmers (and possibly other Universities). Users can register books for sale and other users can search and find these books. The unique selling point compared to existing sites like blocket or tradera is the niche market. When buying from our application you know that the seller is close by and that you can set up a meeting to get the book the same day.

### **1.1.1 Stakeholders**

There are several stakeholders for our application.

- Students (buyers and sellers called ST)
- Student organizations (called SNI)
- Book stores (Cremona in particular)
- Teachers
- Developers (called DEV)
- Testers
- Beta-testers
- Supervisors
- Maintenance
- Server-side supplier

We aim to sell our product to student organisations and therefore view them as our most important stakeholder along with, of course, the actual users of the application. Most of our requirements will be connected to them. Since we develop towards users we anticipate that a lot of test cases will be needed.

We have already been in contact with the student study organization at Industrial Engineering and Management and we have got some requirements from this stakeholder. We have also talked to some students at Chalmers regarding the application in order to get some actual requirements from real stakeholders.

## **1.2 General characteristics of application**

No transactions are handled by the application. The application simply matches buyer with seller and provides the opportunity for market mechanisms to function in the interaction between these two parties.

Functionality to add a plethora of attributes about the book and for users to search for books on these criterias. Also the ability to describe a book in general and to upload a photo of the book taken with the mobile camera. A book could be uploaded by filling in a ISBN-number and then obtaining book title, author and edition.

The ability for other users to comment on the book.

Possible functions to be added in future versions:

- Auctions
- Connection to Cremona to compare price and to buy new books
- A website connected to the application

### 1.3 Scope of application

There is no platform available for buying and selling books but exchanges happens anyway in other forums, for example "bokbytarmingel" arranged by SNI. However this application aims to find more sellers and to build a natural platform where buyers can meet sellers without knowing each other.

The application is to be sold to different student organisations who can then brand it and market it to their users.

An alternative market plan is to offer it for free to start with and to build a user base. When there is a bulk of users a small fee can be charged for adding books for sale.

### 1.4 Objectives and success criteria of the project

A objective for the application is to complete and fulfill all the requirements for the application. Another objective is to launch the application on time, according to the market plan.

Server side

Database

Books table

Users table

Server

Client side

GUI

Communication layer

All logic layer

Another objective for the project is that all members of the project should be able to work in many different areas (Database connection, GUI, Scrum, testing etc) of the project in order to learn new things.

## 1.5 Definitions, acronyms and abbreviations

GUI - Graphical user interface

GPLv3 - General Public License, v3

LGPL - Lesser General Public License

## 1.6 Licensing

Since we may be using code that is copylefted in the project and the fact that we are using Github for version control, we believe that a GPLv3 license is appropriate for our project. We support all four requirements for free software. We also want to impose similar requirements on those who use our software and therefore chooses not to use the licence LGPL.

## 2 Proposed application

We propose a client-server application where users communicate with a database. Users can add and get information from the database and get contact information about other users. The application will be available for phones with Google's Android Operating System. In the future we intend to port the application to work on phones with Apple's iOS Operating System.

### 2.1 Overview

#### 2.1.1 Commit standard and syntax for traceability

We will use the following standard for our commit messages:

*RID: TID: m:*

Where RID is Requirement ID, TID is Task ID which is optionally, and then m for message. The message should include what you have done in your commit and in which file.

### 2.2 Functional requirements for base functionality

Functional requirements for a system describes what the system should do. The requirements can be written at different levels of detail and can be based on both user requirements and system requirements (Sommerville, 2007).

#### Requirement ID 1

A book store that provides the possibility for a **seller** to register a book for sale and that this book is then marketed to potential buyers

Requirement ID 1.1, stakeholder ST

Use case: **The seller can register a book for sale**

Trigger: A seller press the button for "add a book"  
Precondition: The seller has the application installed  
Basic path: The seller presses the button for uploading a book and then gives the possibility to fill in ISBN, title, author, edition of the book, actual course, quality of the book, minimum price, email to seller, and phone number to seller. The seller also fills in a password which enables editing of the advertisement later. If ISBN is filled in, the fields of title, author, and edition is filled in automatically.  
Exception path: The seller should get a exception if some of the required fields are not filled in. The seller should also have the possibility to quit the uploading process at any time through a cancel-button.  
Post condition: A new book is uploaded to the market.

**Test case:**

*Positive test:* The tester is uploading a new book with the following information: (Robinson Crusoe, Daniel Defoe, 2nd edition, literature, excellent quality, 300 SEK, telephone: 1234567, password: muffins.  
The book should then be uploaded to the market. The tester then tries to update/change the price to 100 SEK by pressing the edit-button and use his password.  
*Negative test:* The tester is uploading a new book with the following information: (, Daniel Defoe, 2nd edition, literature, excellent quality, , telephone: 1234567, password: muffins. The tester should get a error message saying "missing title and price".

Requirement ID 1.1.1, stakeholder ST, SNI

Use case: **The seller can set a price for the book**

Trigger: A seller is in the process of uploading a book  
Precondition: The seller has the application installed  
Basic path: The seller presses the button for adding a book and then needs to fill in the required field regarding price. The price should be in SEK and made by 1-4 integers.  
Exception path: The seller should get a exception if no INT-character is typed into the price-field.  
Post condition: A new book is uploaded to the market with a minimum price.

**Test case:**

*Positive test:* The tester uploads books with following testnumbers: 99 SEK, 300 SEK, 1000 SEK, 0 SEK  
*Negative test:* The tester uploads books with following testnumbers:  
hundra kronor, 133700 SEK and should get a error message stating: "Please specify a price within the range of 0-10000 SEK."

Requirement ID 1.1.2, stakeholder ST, SNI

Use case: **The seller can upload a picture of the book for sale**

Trigger: A seller are in the process of uploading a book

Precondition: The seller has the application installed  
Basic path: The seller presses the button for uploading a book and then gives the possibility to upload a picture of the book. Either from the phones picture library or by taking a new photo with the phone-camera.  
Exception path: The seller should get a exception if the picture is too big to handle or in a wrong format.  
Post condition: A new book is uploaded to the market with a picture showing the book.  
**Testcase:**  
Positive test: The tester takes a photo of a book and upload it together with the book.  
Negative test: N/A

Requirement ID 1.1.3, stakeholder ST, DEV

Use case: **The seller has to set a title for the book**  
Trigger: A seller presses the button for "uploading a book"  
Precondition: The seller has the application installed  
Basic path: The seller presses the button for uploading a book and then has to fill in a book title. The title must be more than 10 characters long and maximum of 200 characters.  
Exception path: The seller should get a exception message if a title is not filled in or the title have less than 10 characters (the 200 character limit should be implemented in the text box).  
Post condition: A book for upload have a title.  
**Testcase:**  
*Positive test:* The tester should write "Robinson Crusoe" as title and upload a book.  
*Negative test:* The tester should write the title "xxyy" and get a exception message "Title must be greater than 10 characters".

Requirement ID 1.1.4, stakeholder ST, DEV

Use case: **The seller can set an ISBN for the book for sale**  
Trigger: A seller press the button for "uploading a book"  
Precondition: The seller has the application installed  
Basic path: When uploading a book, the seller has the possibility to fill in ISBN-code for the book in a text box.  
Exception path: The seller should get a exception message if the number of characters in the ISBN-number is any other than 10 or 13 digits.  
Post condition: An uploaded book can have a ISBN-number.  
**Test case:**  
*Positive test:* The tester should fill in "9789144060743" as ISBN (Java direkt med Swing).  
*Negative test:* The tester should fill in 123456 as a ISBN-number and get a exception message: "The ISBN-number must be 10 or 13 digits long."

Requirement ID 1.1.5

Use case: **The seller can set a publishing year for the book**

Trigger: A seller press the button for "add a book"

Precondition: The seller has the application installed

Basic path: The seller types in the publishing year of the book, four digits.

Exception path: The seller should get a exception message if the "filled in year" is not a valid year (type YEAR)

Post condition: An uploaded book have a publishing year.

**Test case:**

*Positive test:* The tester should fill in 1999 as publishing year.

*Negative test:* The tester should fill in 2015 as a publishing year and get a exception message: "this is not a valid year".

Requirement ID 1.1.6, stakeholder SNI

Use case: **The seller can set a course for which he/she used the book**

Trigger: A seller press the button for "add a book"

Precondition: The seller has the application installed

Basic path: The seller can fill in which course the book is used in.

Exception path: The seller have a possibility to not fill in the course, therefore no specific exception path.

Post condition: An uploaded book have a connection to a course.

**Test case:**

*Positive test:* The tester should fill in "Databases" as Course and "TDA357" as a course.

*Negative test:* N/A

Requirement ID 1.1.7, stakeholder ST

Use case: **The seller can add a comment about the book**

Trigger: A seller press the button for "add a book"

Precondition: The seller has the application installed

Basic path: The seller types in comments about a book in a text box.

Exception path: -

Post condition: An uploaded book might have comments.

**Test case:**

*Positive test:* The tester should fill in "excellent quality" as a comment.

*Negative test:* N/A

Requirement ID 1.2

Use case: **The seller can update information about a book or take a book off of the market**

Trigger: A book is sold (not for sale any more) or not up to date.

Precondition: The seller has the application installed

Basic path: The seller looks at his/hers uploaded books and press one of them. The seller then sees the information concerning the book and have the possibility to update information or delete the

book from the market, by pressing a "edit-button" and typing in a password.

Exception path: If a book is not deleted or updated within two months, the book will be deleted from the market.

Post condition: A book is updated with new information or deleted from the market.

**Test case:**

*Positive test:* The tester is uploading a new book with the following information: (Robinson Crusoe, Daniel Defoe, 2nd edition, literature, excellent quality, 300 SEK, telephone: 1234567, password: muffins.

The book should then be uploaded to the market. The tester then tries to delete the ad by pressing the edit-button and use the password. The book should be deleted from the market.

*Negative test:* The tester should fill in a wrong password and the not be able to delete the book from the market. The book will automatically disappear from the market after two months.

**Requirement ID 2**

A book store that provides the possibility for a **buyer** to search for and browse books and put him/her in touch with sellers

Requirement ID 2.1, stakeholder SNI

Use case: **A buyer can search the database for available books**

Trigger: A buyer needs information about a specific book or course

Precondition: The buyer has the application installed

Basic path: A buyer fills in a attribute in the search field and then gets the answers from the search presented in a list.

Exception path: If the searched text does not match any attribute, the searcher should get a exception message "your search did not match any information at the market"

Post condition: The searcher finds the information he/she wants.

**Test case:** N/A

Requirement ID 2.1.1, stakeholder SNI

Use case: **In a search, a buyer can specify a book's title**

Trigger: A buyer needs information about a specific book

Precondition: The buyer has the application installed

Basic path: A buyer fills in a book title in the search field and then gets the answers from the search presented in a list.

Exception path: If the searched text does not match any book titles, the searcher should get a exception message "your search did not match any information at the market"

Post condition: The searcher finds the information he/she wants.

**Test case:**

*Positive test:* The tester is searching for the title "Robinson Crusoe" and then get a match.

*Negative test:* The tester should fill in "Petter och hans fyra getter" as a title and not get a match in the market with the



following message: "your search didn't match any books at the market".

#### Requirement ID 2.1.2

Use case: **In a search, a buyer can specify a book's author**

Trigger: A buyer needs information about a specific book

Precondition: The buyer has the application installed

Basic path: A buyer fills in a author in the search field and then gets the answers from the search presented in a list.

Exception path: If the searched text does not match any authors, the searcher should get a exception message "your search did not match any information at the market"

Post condition: The searcher finds the information he/she wants.

**Test case:**

*Positive test:* The tester is searching for the author "Daniel Defoe" and get a match.

*Negative test:* The tester should fill in "Emil Nyström" as a author and not get a match in the market with the following message: "your search didn't match any books at the market".

#### Requirement ID 2.1.3

Use case: **In a search, a buyer can specify which course the book should be used in**

Trigger: A buyer needs information about a specific course

Precondition: The buyer has the application installed

Basic path: A buyer fills in a course in the search field and then gets the answers from the search presented in a list.

Exception path: If the searched text does not match any authors, the searcher should get a exception message "your search did not match any information at the market".

Post condition: The searcher finds the information he/she wants.

**Test case:**

*Positive test:* The tester is searching for the course "objektorienterad programmering" and then get a match on the book "Java direkt med Swing". The tester searches for the course "TMV215" (linear algebra) and get a match on the book "Linear algebra and its applications".

*Negative test:* The tester should fill in "Harry Potter Knowledge" as a course and not get a match in the market with the following message: "your search didn't match any books at the market".

#### Requirement ID 2.1.4

Use case: **A search can be sorted on any of the available attributes that a book can have. Originally it is sorted by price**

Trigger: A buyer finds some attributes more important than others.

Precondition: The buyer has the application installed.

Basic path: a buyer searches for a book and then get the books presented in a list. The buyer then have the possibility to sort by other attributes, for example "recently added", "latest

edition" etc.

Post condition: The buyer finds a book with the right attributes.

**Test case:**

*Positive test:* The tester is searching for the title "Robinson Crusoe" and then get three books with the cheapest first.

*Negative test:* N/A

Requirement ID 2.2

Use case: **Buyer can upload a buy request (see ID 1)**

Trigger: A buyer needs a book which is not found at the market

Precondition: The buyer has the application installed.

Basic path: A buyer press the button "add buy request" and fills in the information concerning uploading a book.

Exception path: The buyer needs to have a book title and/or course, otherwise the buyer will get an exception message and the buying request will not be conducted.

Post condition: A buying request is uploaded to the market.

**Requirement ID 3**

ISBN-connection

Requirement ID 3.1

Use case: **A book can be uploaded by ISBN-number**

Trigger: Adding a book.

Precondition: A user have a cell phone and wants to use the application and presses the button "add a book".

Basic path: A user fills in ISBN-number and then the information regarding title, author, and version fills in automatically.

Exception path: The seller should get a exception message if the number of characters in the ISBN-number is any other than 10 or 13 digits.

Post condition: A book is added by ISBN-number.

**Test case:**

*Positive test:* The tester should fill in 9789144060743 as ISBN (Java direkt med Swing and the get the title: "Java direkt med Swing", the author "Jan Skansholm", and the edition "6".

*Negative test:* The tester should fill in "1234567" as ISBN and get a error message saying: "Not a valid ISBN-number".

**Requirement ID 4**

Settings view

Requirement ID 4.1, stakeholder SNI

Use case: **The user should be able to store his/hers user details on the internal memory**

Trigger: "save data" button pressed

Precondition: A user must have the application open at the settings-view and has entered the information to save.

Basic path:

Post condition: File has been created in the internal storage, containing the information entered by the user.

**Test case:**

*Positive test:* The user should go into settings through the menu-hard key → Add proper information and press "Save data"

*Negative test:* N/A

Requirement ID 4.2, stakeholder DEV

Use case: **The previously saved data should be loaded upon entering the settings-tab and autofilled into the fields**

Trigger: Settings-tab accessed

Precondition: User details should previously have been saved.

Basic path:

Post condition: The information previously entered and saved by the user should be autofilled in the correct fields at the settings-screen.

**Test case:**

*Positive test:* N/A

*Negative test:* N/A

Requirement ID 4.3, stakeholder DEV

Use case: **The user should be able to delete previously saved user details**

Trigger: "Delete data" button pressed

Precondition: There is already saved data

Basic path:

Post condition: UserDetails-file deleted and fields emptied.

**Test case:**

*Positive test:* N/A

*Negative test:* N/A

**Requirement ID 5**

About the application view

Requirement ID 5.1

Use case: **A user can read about how to use the application in the settings view**

Trigger: A user needs help with the functionality of the application

Precondition: A user have the application

Basic path: A user needs help with something and goes into Settings and "help". Here, the user can read a tutorial document regarding the functionality of the application.

Post condition: A user understand the functionality in the application.

**Test case:**

*Positive test:* The user should go into Settings → help and find information regarding "how to add a book to the market"

*Negative test:* N/A

Requirement ID 5.2, stakeholder DEV

Use case: **A user can read about the application developers in the settings view**

Trigger: A user is interested in application details

Precondition: A user have the application

Basic path: A user goes into Settings → Application details and finds contact details to the developers, the application version in use, details regarding licenses etc.

Post condition: A user can read the application details

**Test case:**

*Positive test:* A tester goes into Settings → Application details and reads about the application

*Negative test:* N/A

Requirement ID 5.3

Use case: **A user can read his/hers details in the settings view**

Trigger: A user needs to see his/hers user details

Precondition: A user have the application

Basic path: The first time a user adds a book, the user needs to fill in: mail, phone number and password. This information is stored under Settings.

Post condition: User information is stored in settings

**Test case:**

*Positive test:* The tester should fill in the following details when adding a book for the first time: [emil@vitdesign.se](mailto:emil@vitdesign.se), 0703-200491, muffins. The information should then be stored in Settings.

*Negative test:* N/A

## 2.3 Non-functional requirements for base-functionality

Non-functional requirements are constraints on the services of functions offered by the system, for example regarding timing och the development process. Non-functional requirements often apply to the system as a whole (Sommerville, 2007).

### 2.3.1 Usability

*Requirement ID: NF1*

The system will have to be intuitive and a user should be able to work the system without the use of a manual.

A user should be able to register a book for sale with ease. It should take no more than 5 minutes for a first time user to register a book for sale with the minimum amount of required information.

A user should easily be able to find if the book he/she needs is available for sale. This can be done by doing an extensive search for the exact information by a book or by browsing different available books in different categories. The search of a book should not take more than five seconds.

The design should be minimalistic and easy on the eyes.

**Test case:**

*Positive test:* The tester should ask someone who hasnt used the application before and ask them to add a book. and take time of the process and ask questions afterwards, in order to understand if the application is considered easy to understand.

### *2.3.2 Reliability and security*

*Requirement ID: NF2*

The application will experience the most pressure during the first weeks of the four "läsperioder" during the Chalmers year. During this time there needs to be a very low probability of failure.

Since no actual transactions are handled by the application the severity of a failure has little economic effect on the users which would indicate a lower need for reliability. However the system needs to be stable in order to attract users.

The system has to be reliable in the sense that information about the users have to be kept safe. Information stored about user is always sensitive so the system will keep a minimum amount of information about the users. The only required information stored about the users are: name, email address, and phone number.

The application users should only get information from a web-based layer/server instead of information directly from the server. This is due to safety reasons.

**Test case:**

*Positive test:* The tester will add books at the same time as three other testers. This small test will indicate if the server can handle four uploading books at the same time.

### *2.3.3 Performance*

*Requirement ID: NF3*

Performance needs will vary over different periods. As stated under reliability, the system will experience the most pressure during the first few weeks of every period ("läsperiod"). In the middle of these periods bandwidth and server capacity can be limited. A solution where bandwidth and server capacity is paid for depending on usage, should be considered.

The function and use of the application is dependent on internet access. Since the phone can have limited bandwidth it is desirable to minimise the size of data being transferred. Users must be able to get an overview of the availability of books quickly. More detailed information like images can be allowed to take more time and bandwidth but must then be optional to view. When you lose your internet connection on the phone, you should be able to see the information viewed at your last login.

**Test case:**

*Positive test:* The tester should view information about a book and then turn off the internet connection on the phone. The tester should then open the application and still see the information on the book viewed last time.

### *2.3.4 Supportability*

*Requirement ID: NF4*

Maintenance, upgrades and changes have to be made during periods of low usage, that is not during the first few weeks of every period. This is to be done by professionals with insight into software engineering. However the documentation should allow for any such professional to do it, hence the life of the system is not dependent on the engagement of the creators.

### *2.3.5 Implementation*

*Requirement ID: NF5*

The implementation needs to be done without any major investments. The solutions used will probably be free to use and must fit within the confines of the law. Software must be possible to release under GPLv3. Client-side software must be runnable on Android OS and adhere to the concept of object oriented programming.

The client-side application cannot exceed 10 MB.

**Test case:**

*Positive test:* After every release, the tester should control that the client-side application cannot exceed 10 MB.

### **2.3.6 Verification**

*Requirement ID: NF6*

A user is verified by its phone number and his/her name connected to that phone number. Every book-ad is verified by a advertisement number and editing a advertisement is possible if you type in a correct password for the advertisement.

### **2.3.7 Packaging and installation**

*Requirement ID: NF7*

The application will be available through the application outlets available electronically and accessible directly through your phone. This includes, to begin with, Android-market and in future releases also via Appstore.

Installation must be automatic and application needs to be executable once it is downloaded.

### **2.3.8 Legal**

*Requirement ID: NF8*

The application is to be sold to student organisations which in turn can market it with their brand. The application will be sold under GPLv3 license and should be compatible with the requirements forced upon software under this license.

## **2.4 Functional requirements for add-on functionality**

### **Requirement ID 6, stakeholder ST**

Auction

Requirement ID 6.1

If the time for an auction runs out the book is automatically taken off the market.

Requirement ID 6.2

The seller can specify if he/she wants to have an auction, a buyout-price or both

Requirement ID 6.3

Use case: A book out for auction can have a reservation price

Requirement ID 6.4

A book out for auction have an ending date

Requirement ID 6.5

Anyone can make a bid for a book out for auction except for the owner of the auction/book

Requirement ID 6.6

A bid have to exceed the previous bid with at least 10 kr and in even 10 kr intervals.

Requirement ID 6.7

A book on the market that is not on auction can be reserved for 5 minutes during which it is not visible to other buyers.

#### **Requirement ID 7**

Cremona Bookstore integration

Requirement ID 7.1

If a buyer search for a book without any sellers, the buyer should get information regarding if the book is available at Cremona Bookstore at Chalmers

#### **Requirement ID 8**

Reservations of books

Requirement ID 8.1

When a reservation is made, the user making the reservation gets access to the buyer's contact information

Requirement ID 8.2

A buyer can only make one book reservation at the time.

#### **Requirement ID 9**

Saving statistics

Requirement ID 9.1

Saving statistics for book price

#### **Requirement ID 10, stakeholder SNI**

Renting

Requirement ID 10.1

A book can either be sold or rented (or both)

## **2.5 Application models**

### **2.5.1 Scenarios**



### *2.5.2 Use case model Use cases priority*

### *2.5.3 Static model*

### *2.5.4 Dynamic model*

### *2.5.5 User interface*

## **2.6 Test cases and test plan**

Please see test cases connected to requirements above. The test plan is connected to every requirement and the sprint where the requirement is in.

The working in progress sprint for the moment in number three, consisting of Requirements 1.1.1, 1.1.3, 1.1.4, 1.1.5, 1.1.6, 1.1.7, 1.2 and test cases for every requirement. So the tests for these requirements have the same deadline as the sprint, 2012-04-24 (see document Project planning and methodology for sprint information).

## **2.7 Possible future directions**

This application can be used in various ways. For example, all universities in Sweden could have usage, and benefit, from this application. The aim is to make a quite generic application which easily can be customized for another university. The application could also be used in other areas, for example selling other products than second-hand books.

## **2.8 References**

Sommerville, Ian (2007). *Software engineering*. Eight edition, Pearson Education Limited, Edinburgh Gate, England.