Part I
Analysis

Well begun is half done
-Aristotle

In this section, we will attempt to give the reader insight into the rationale behind the decisions made in the course of the project, by reflecting on challenges and surprises uncovered by analysing the problem area. In other words, the section will try to explain *why* the system has been designed as it has, why the process has been managed the way it has, and so on.

The section is structured by the various techniques and artefacts used in conducting the analysis. The primary topics covered in this section are:

- Business modelling
- Requirements analysis
- Data modelling
- Architectural analysis

We begin the section by investigating and discussing the activities aimed at uncovering requirements for the system.

1 Requirements Analysis

Generally speaking, requirements can be divided into two categories; functional and non-functional. The functional requirements for the RentIt server are expressed by use-cases, that each encapsulate some functionality the system must posses in order to be a useful solution. The non-functional requirements are captured by the factor-tabel. These artefacts will serve as the basis for the discussion of the requirements in this section.

1.1 Use Cases

As mentioned, the use-cases capture the functional requirements of the system. The list of use cases was composed by examining the business model and identifying user goals associated with uploading, viewing and buying videos, creating user accounts and so on.

When writing use cases for the RentIt system, we noted that they could be grouped into three categories of usage:

- **User management**
  Use cases concerning user profile creation, editing user profiles and authorizing users

- **Video management**
  Use cases concerning downloading, viewing and rating videos
- **Transaction management**
  Use cases concerning transactions between account balances and buying more RentIt credits.

In other words, the use cases reveal at least three major areas of responsibility within the required functionality of the system. These responsibilities imply a large scale organization of namespaces, by way of the GRASP principles.

1.2 Factor Table

As opposed to the functional requirements, the non-functional requirements are not expressed by some functionality the system must offer, but rather some behaviour or quality the system must posses. These are captured by a factor table, as described in **Larman**. The list of factors (requirements) the factor table consists of, was devised by using the FURPS+ mnemonic, and brainstorming on predictable requirements based on the business model. This facilitated a fruitful discussion on topics such as security, reliability and performance requirements of the system.

Some of the usefulness of the factor table is its recording of quality scenarios, because it encouraged the group to reflect on the quantifying the requirement, as to make it testable. Realistically, it would not be possible to satisfactory implement and test all of the factors, but for the sake of exercise, the factor table was composed as if it was meant to be used in real world situation. As a consequence of the limited timeframe of the project, we decided to limit the factors that we would *actually* emphasize to:

- accuracy of search results
- documentation of web service interface
- persistence of user data

These factors were consensually understood as the most crucial and/or interesting requirements to pursue.

2 Data Modelling

Another major part of our problem analysis was devising a model of the data, partly in order to construct and refine a database design for persisting the necessary data, and in part to serve for inspiration for C# classes. This modelling was done using two artefacts, a domain to serve as a visual dictionary for objects and concepts in the problem domain, and an ER-diagram as an aid for reflecting on a useful database design. The main inspiration for these models were the use cases and the business model. These two artefacts serve the basis for discussions in this section.

2.1 Domain Model

The domain model was used to facilitate and document the results of a discussion of the problem domain. By modelling the objects and concepts in the problem domain, the project group came upon a number of questions, such as:

-

2.2 ER-Diagram

3 Architectural Analysis

In this section

3.1 Package Diagram

4 Project Management Plan