|  |  |  |
| --- | --- | --- |
| CS3343-LA3 2014/15 Sem A December 5, 2014 | B.T.  Analysis and Design Report | |
| [Type the abstract of the document here. The abstract is typically a short summary of the contents of the document. Type the abstract of the document here. The abstract is typically a short summary of the contents of the document.] | | Prepared by  B.T. |

Contents

[Introduction 2](#_Toc405496358)

[Design Constraints 3](#_Toc405496359)

[User Stories 3](#_Toc405496360)

[Use Case 4](#_Toc405496361)

[Use Case Diagram 5](#_Toc405496362)

[Analysis Class Diagram 6](#_Toc405496363)

[Initial 6](#_Toc405496364)

[After first refactoring 6](#_Toc405496365)

[After second refactoring 7](#_Toc405496366)

[After third refactoring 7](#_Toc405496367)

[Final 8](#_Toc405496368)

[Class Diagram 9](#_Toc405496369)

[Sequence Diagrams 10](#_Toc405496370)

# Introduction

Timetable Scheduling System aims to generate a timetable which fulfills a student’s personal preferences and constraints so as to ease the pain of him/her during add/drop period. A student has to prepare for deciding which courses he/she is going to take in the coming semester. Courses offered in the same semester may collide with each other. There are many possible lecture-tutorial combinations. It is tedious and difficult for a student to draw all possible combinations in Excel, which is actually much like the reality. Especially some courses’ lectures overlap with another, it is difficult for a student to decide which to take, simply based on their time slots. With our system, it is possible to generate a timetable based on the constraints provided by the user.

For example, a student would like to take 6 courses, each of them has 1 lecture and 3 tutorials offered. One should only select one tutorial.

# Design Constraints

# User Stories

User story 1 : User input the txt file which contains the information of the courses and then a possible time table will be generated for the user.

User story 2 : User input a wrong txt file and nothing will be generated to the user but only error.

User story 3 : User input the txt file which contains the information of the courses and then choose the preference that he wants, e.g. no class before 11 am, and then a possible time table will be generated for the user according to his preference.

# Use Case

|  |  |
| --- | --- |
| Use case | Description |
| Use case number | 1 |
| Application | Optimal Timetable |
| Use case name | Generate a timetable |
| Use case description | The user prepares the txt file of the courses which contains the information of the courses and put them into the working directory of TimetableSchedulingSystem.jar. |
| Primary actor | User |
| Basic flow | 1. User prepares the data files of the courses which contains the information of the courses and put them into the working directory of TimetableSchedulingSystem.jar. 2. The system processes the data file and constraint files. 3. The system displays a tailor made timetable. |
| Exception | 3.1 The system cannot generate a possible timetable with matched time slots. 3.2 The system shows an error message. |

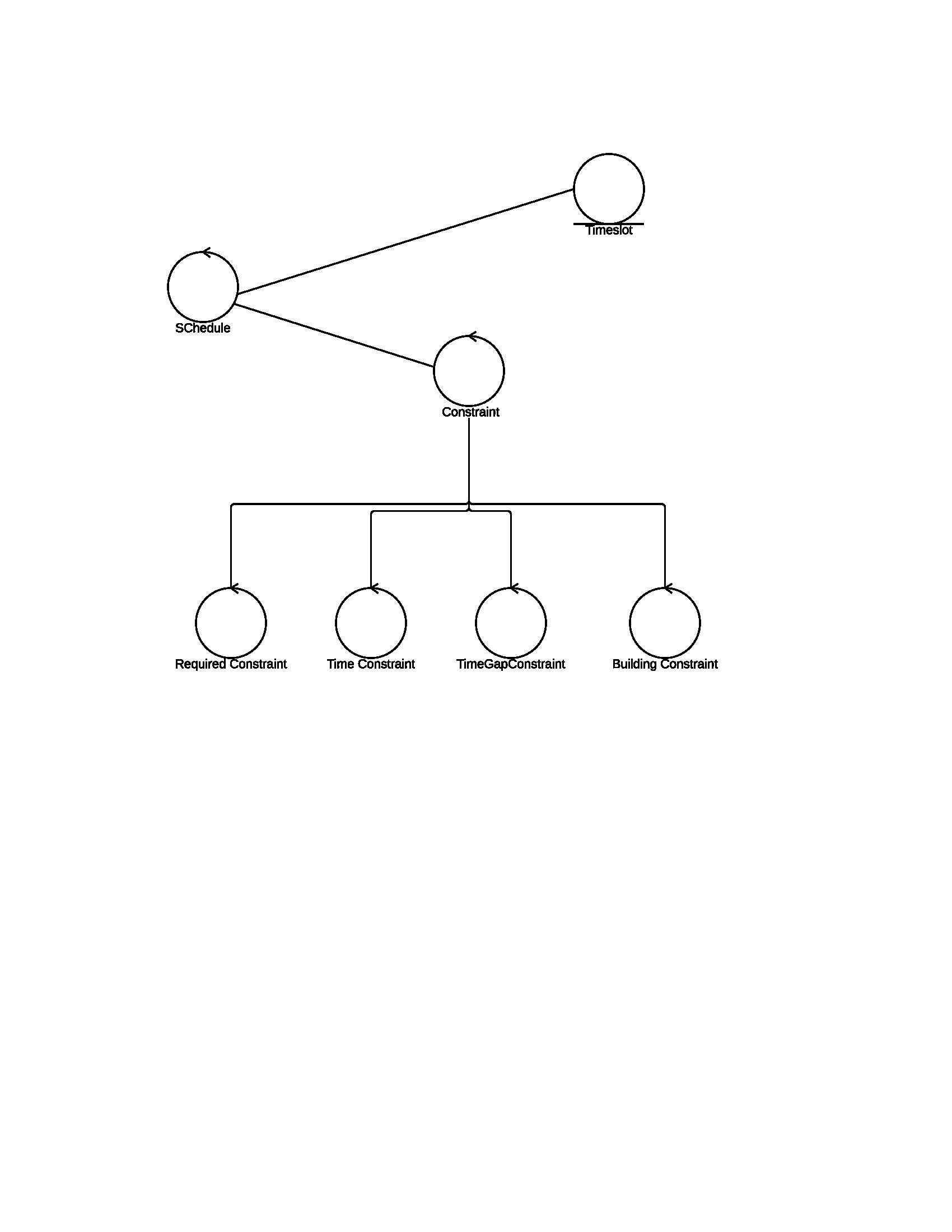
|  |  |
| --- | --- |
| Use case | Description |
| Use case number | 2 |
| Application | Optimal Timetable |
| Use case name | Generate a timetable with time constraints |
| Use case description | The user prepares the txt file of the courses which contains the information of the courses and the txt file of time constraints. Then the user put them into the working directory of TimetableSchedulingSystem.jar. |
| Primary actor | User |
| Basic flow | 1. User prepares the data files of the courses which contains the information of the courses and the txt file of time constraints and put them into the working directory of TimetableSchedulingSystem.jar. 2. The system checks the time of the courses and the required time constraints from the file. 3. The system displays a tailor made timetable. |
| Exception | 3.1 The system cannot generate a possible timetable with matched time slots according to the time constraints. 3.2 The system shows an error message. |

# Use Case Diagram

# Analysis Class Diagram

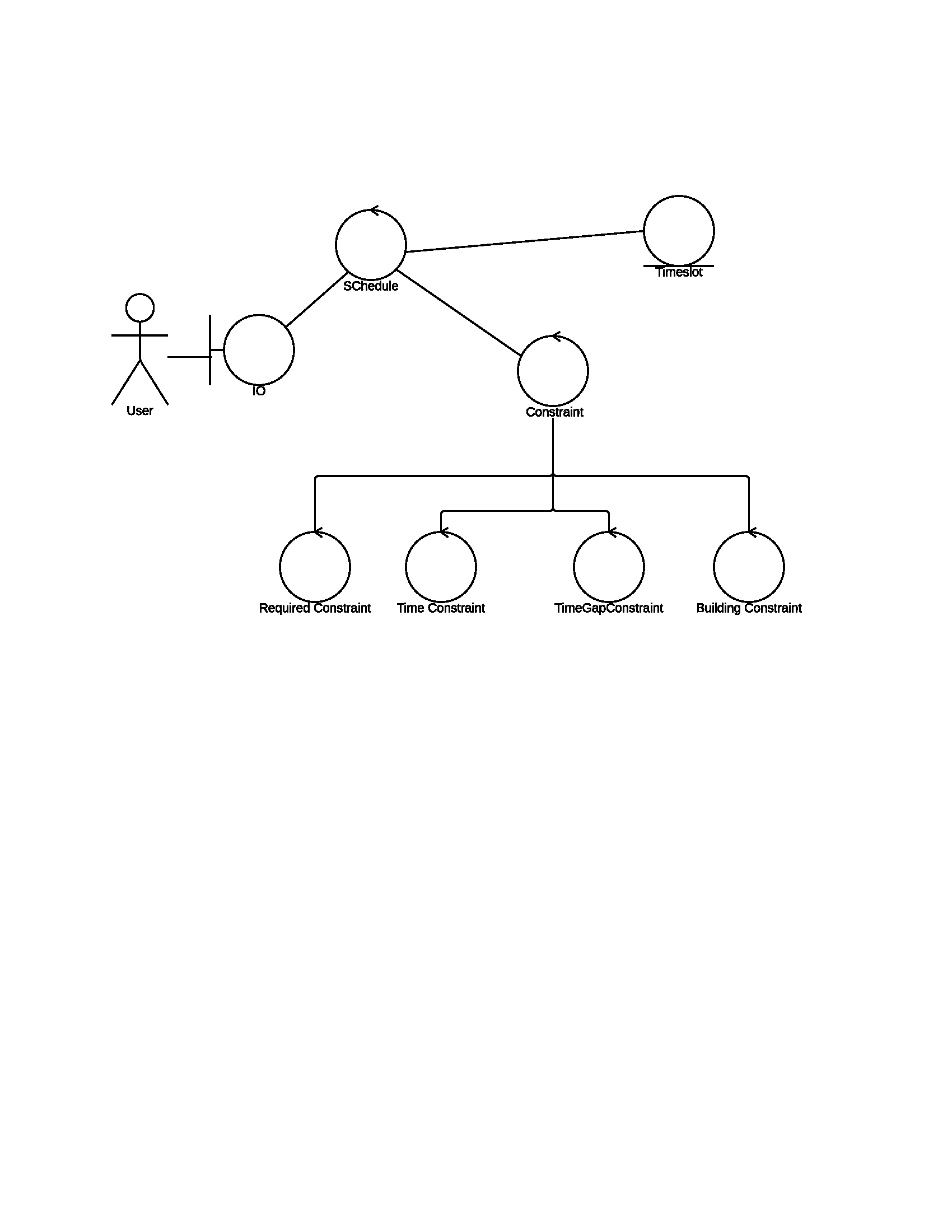
## Initial

At first, our program encapsulates all helper functions and so in the Schedule class, which stores the main function. This is not a good design, that’s why refactor is needed.



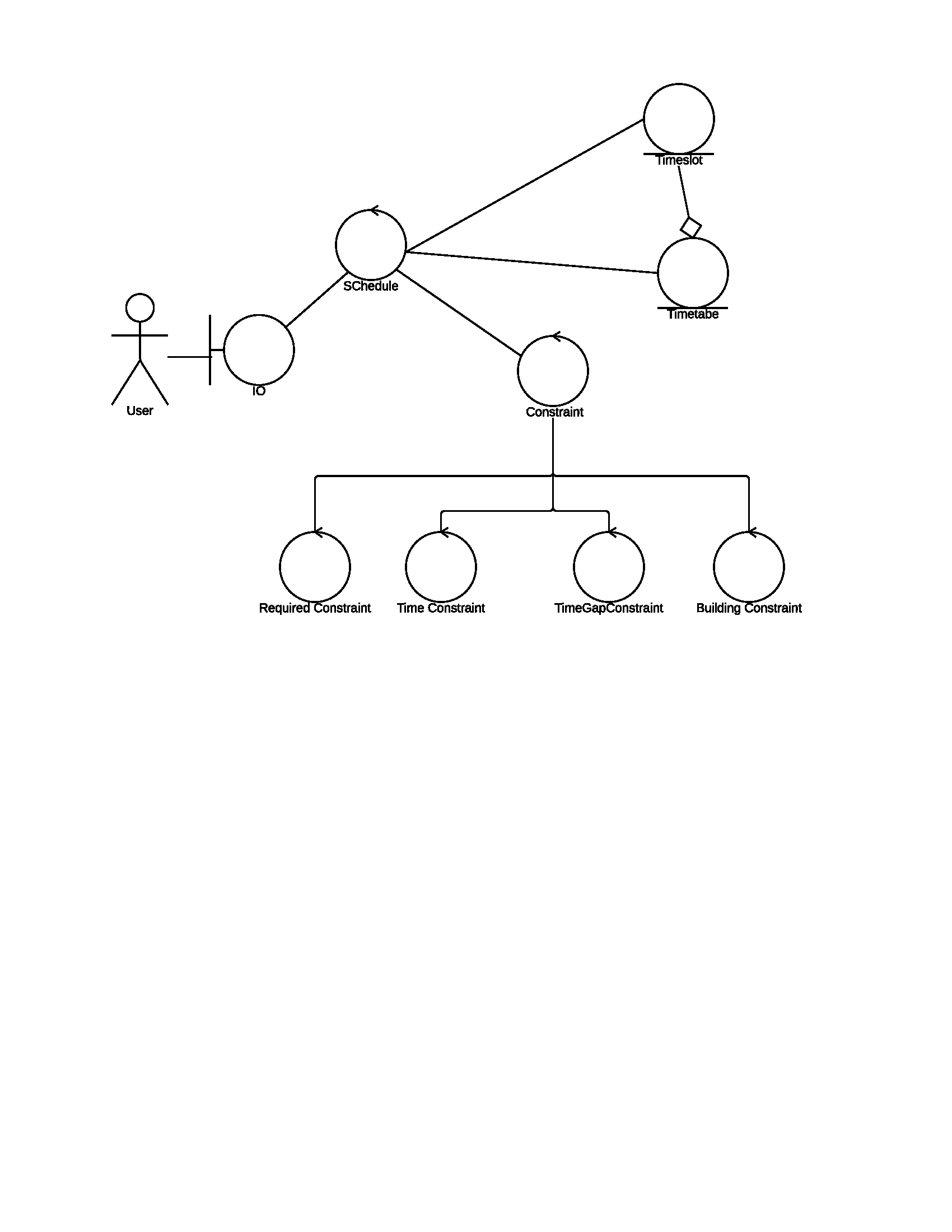
## After first refactoring

Some functions are extracted to a new boundary class (“IO”) from Schedule class.



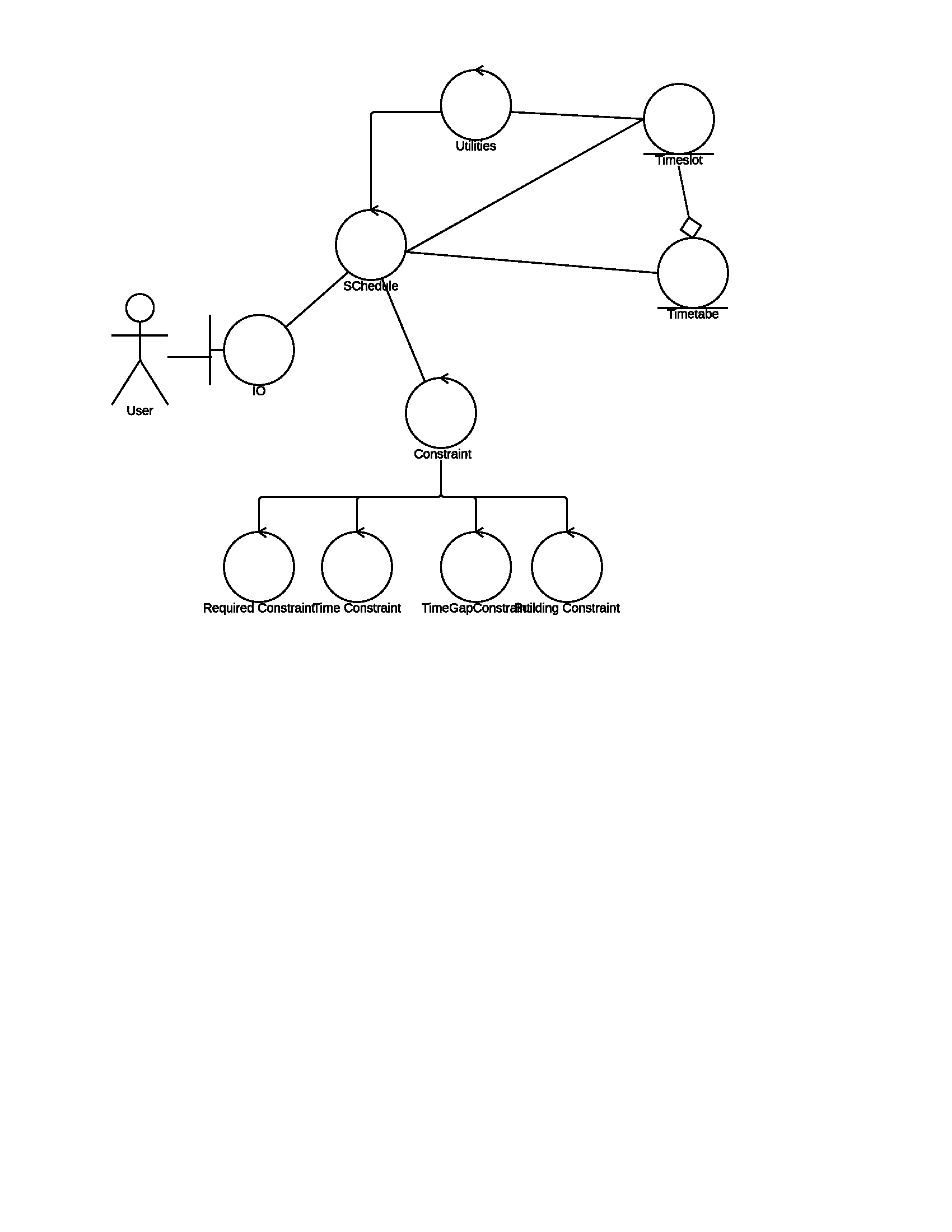
## After second refactoring

Originally, the resultant timetable is in a form of arraylist of Timeslot. Some of them are refactored as a newly created class object, Timetable, so as to fit with the semantic meaning of “a collection of timeslots”. Not all multiple timeslots are regarded as timetable, so only those of such meaning were refactored.



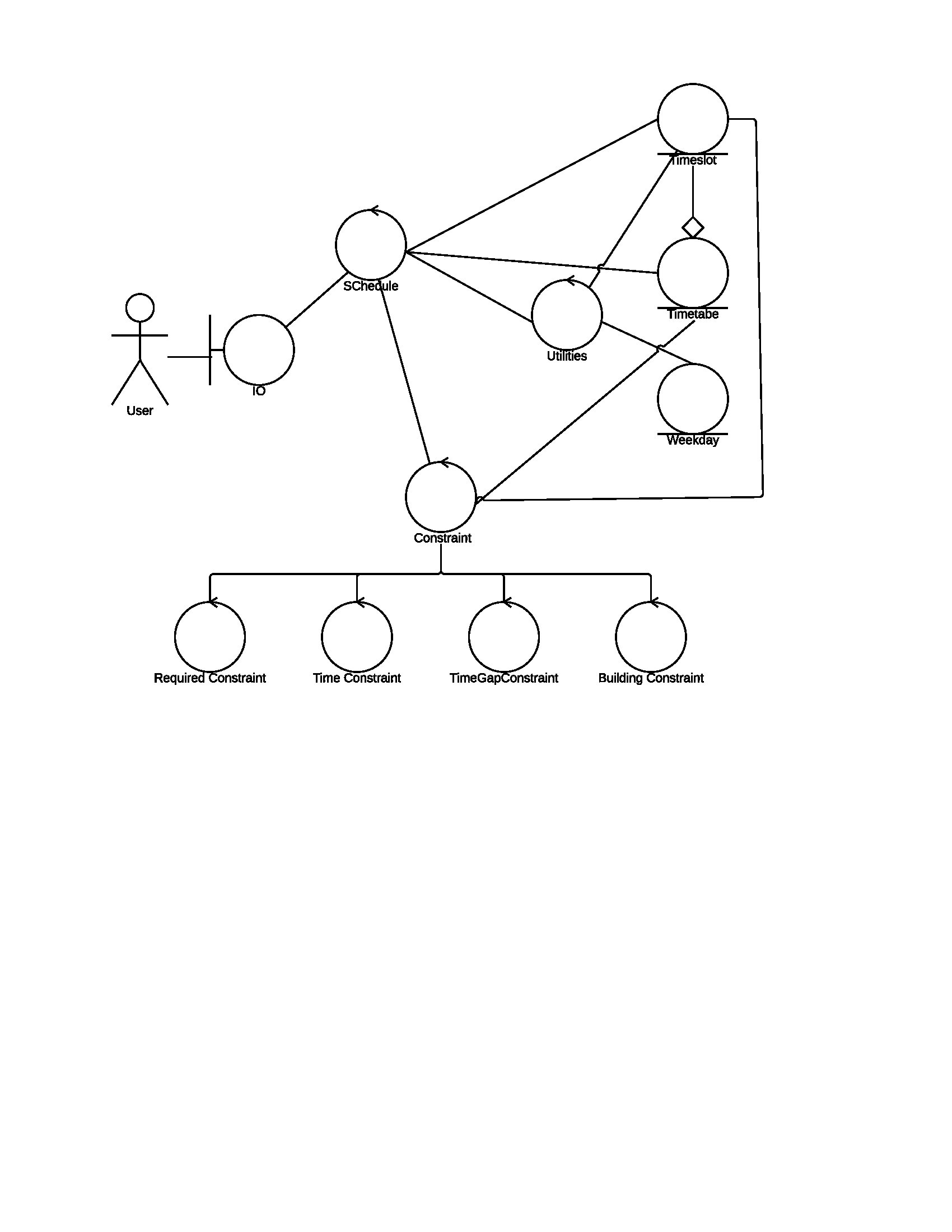
## After third refactoring

All helper functions were extracted to a new control class called Utilities.



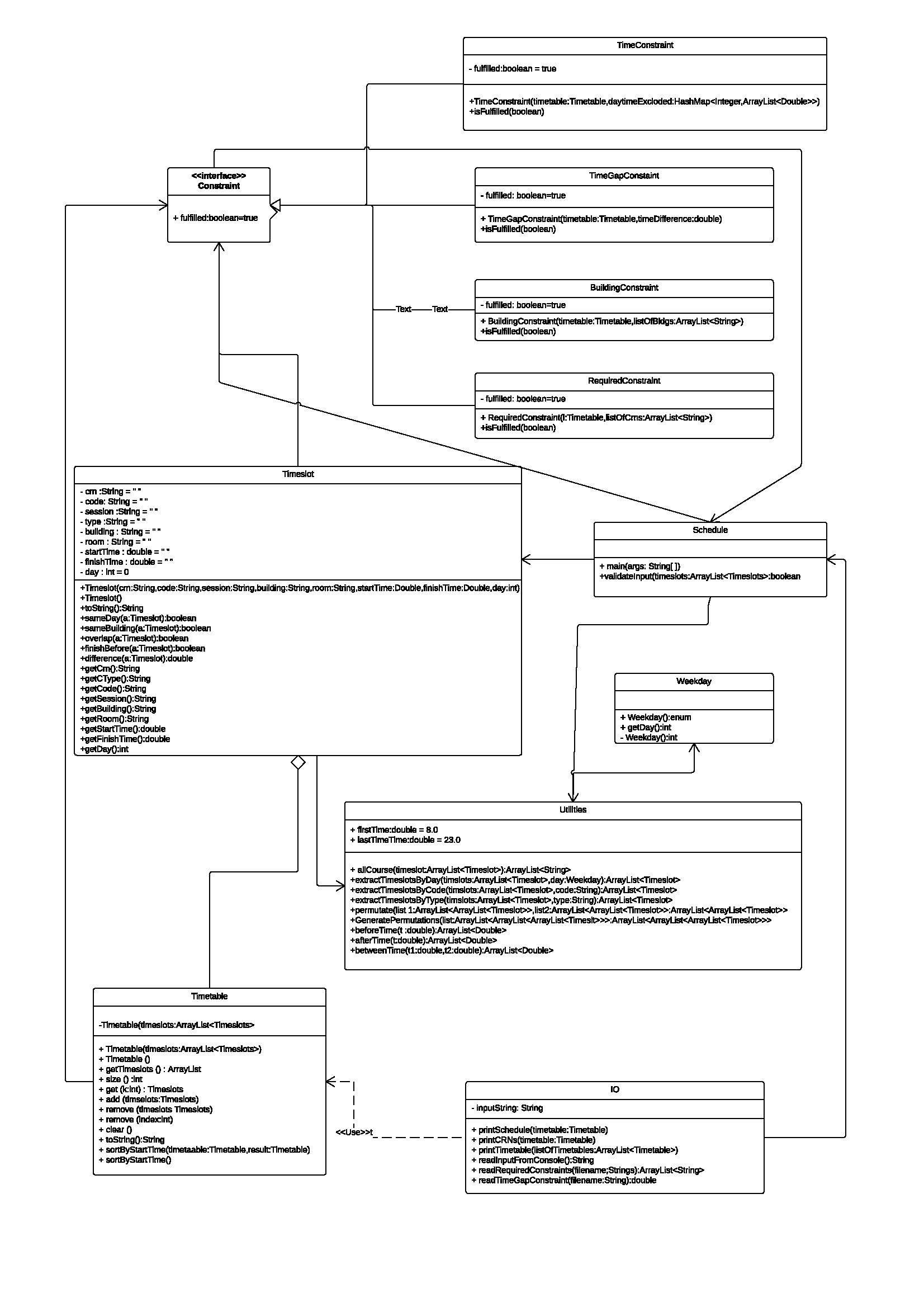
## Final

The final version of analysis class is as the following:



The final design is MVC model. The view class is IO, the model classes are Timeslot, Timetable and Weekday. The rest are controller classes.

# Class Diagram



# Sequence Diagrams

