GL02 - Fondements d'Ingénierie Logicielle

ZHANG Xizhu

WANG Ziyan

MAQUEDA Soraya

01 Novembre 2021

Contents

Preface

This document aims to facilitate developers of the system here described in their task of correctly implementing it. It contains specifications regarding the language syntax, the data types that each abstract data type manages as inputs and renders as output, as well as some general non-technical specifications that describe the expected behavior of the system.

The developers will find detailed specifications regarding the pre and postconditions of each of the previously described behavior specifications. These specifications also detailed the objective, so that developers can try to answer some doubts that they might have that are not explicitly written.

They will also find the required inputs and its data type for each method that help fulfill the objectives. It is important for developers to try to utilize the axioms as well in order to assert that the implementation that they have is the right one, as they describe the expected behavior in case of an error.

We hope that this document is as explicit and self-explanatory as possible, and that the team of developers that gets to implement this system can carry that task out smoothly.

Introduction

The Central University of the Republic of Sealand (SRU) hopes to facilitate the management and organization of its premises by providing users (teachers and students) with tools to monitor classroom occupancy. The General Course Planning System (PGC) exports timetables in CRU format and establishes weekly course slots for each available room.

We decided to develop this software by creating a command line interface tool, where users can request the associated room for a given course, and restore the maximum capacity based on the number of locations in a given room. Users will be able to see when a certain room is free during the week or which rooms are free during a given time period of the week.

With this new monitoring tool, a student or teacher can also generate an iCalendar file between two given teaching dates in which he participates, thereby integrating it into his own agenda software.

Considering that a slot in the room can only be used by a single teaching and cannot overlap, the software can also check the quality of the timetable data,

In addition, at the request of the school, the software can help monitor the occupancy of the premises. It needs to generate a comprehensive visualization of room occupancy rates and classify them by reception capacity. The idea is to allow venue managers to determine which rooms are underused or overused so that they can investigate problems and decide future arrangements.

General specification of requirements

By studying the context of SRU software development, we have identified the following functional as well as non functional user requirements.

Functional Specifications

SPEC01. Search for the Timeslot and all related information for a given course.

SPEC02. Check the schedule and maximum Capacity of a precise room.

SPEC03. Search for which rooms are available for a defined Daytime.

SPEC04. Exporter a file in iCalendar format between two given dates.

SPEC05. Visualize room occupancy rates.

SPEC06. Visualize a chart of room types by Capacity.

Non-functional Specifications

SPEC_NF_01. The software is able to check the quality of the university schedule data to avoid overflow of rooms.

Detailed specification

Details of requirements

Title	Search for the Timeslot and all related information for a given course.
Identifier	SPEC01
Objective(s)	The user can find the Timeslot of the course by the CourseName, including all the information of the course: CourseName, its Type, the Capacity, the Daytime, the SubgroupIndex and the RoomName.
Precondition(s)	The user has logged in to the software and opened the query interface.
Postcondition(s)	The user has found the Timeslot and all related information of the corresponding course.
Input	Course name(M), course type(O)
Process	The user enters the CourseName. The system searches for the corresponding data in the database according to the name, and displays the Timeslot containing all its information.
Output	The Timeslot of the course is displayed for the user, including CourseName, its Type, the Capacity, the Daytime, the SubgroupIndex and the RoomName.
Error handling	When the Coursename entered by the user does not exist, the system displays that the relevant information cannot be found, and the user is required to re-enter the CourseName.
Comment(s)	Mandatory input (M), Optional input (O). If the user only enters the CourseName (M), the system will display the Timeslot of all types of courses under the course. If the course Type is entered at the same time (C/D/T), only one corresponding Timeslot will be displayed.

Title	Check the schedule and maximum Capacity of a precise room.
Identifier	SPEC02
Objective(s)	The user can find the schedule of the room within a week by entering the RoomName and therefore know the free period of the room. Its Capacity is also confirmed.
Precondition(s)	The user has logged in to the software and opened the query interface.
Postcondition(s)	The user has found the Capacity and schedule of the corresponding room.
Input	RoomName
Process	The user enters the RoomName. The system finds its Capacity and occupied periods in the room within a week according to the RoomName. Then arrange it into a schedule according to the date, and display it to the user.
Output	The Capacity and schedule of the room within a week is displayed for the user.
Error handling	When the RoomName entered by the user does not exist, the system displays that the relevant information cannot be found, and the user is required to re-enter the RoomName.
Comment(s)	None.

Title	Search for which rooms are available for a defined Daytime.
Identifier	SPEC03
Objective(s)	The user finds the available rooms during a defined Timeslot and their Capacities by giving the Daytime.
Precondition(s)	The user has logged in to the software and opened the query interface.
Postcondition(s)	The user has found the available rooms with their Capacities for the defined Daytime.
Input	Daytime(a date and a period).
Process	The user enters a Timeslot. The system searches the database for rooms that have not been occupied during this time period according to the given Daytime, and returns a list of RoomNames and Capacities, then displays it to the user.
Output	A list of available rooms and their respective Capacities is displayed to the user.
Error handling	If the Daytime entered by the user is outside the school opening hours, the system will prompt the user to re-enter the Daytime.
Comment(s)	The maximum Capacity is displayed at the same time as the RoomName, which allows users to confirm whether the size of room is what they want when booking a room later.

Title	Exporter a file in iCalendar format between two given dates.
Identifier	SPEC04
Objective(s)	The user can obtain the iCalendar file created according to the courses he participated in during the defined period and export it to his own agenda software.
Precondition(s)	The user has entered two dates, and the system generates the user's schedule based on these two dates.
Postcondition(s)	The schedule file in iCalendar format is created and exported.
Input	Schedule to export
Process	The schedule is loaded, and then an iCalendar format file of .ics is created.
Output	The file is created and a confirmation message is displayed.
Error handling	If any error occurs during the export, the action is interrupted and then a message is shown to the user.
Comment(s)	None.

Title	Visualize room occupancy rates.
Identifier	SPEC05
Objective(s)	The user can see the occupancy rate of all rooms in order to identify which rooms are overused or underused.
Precondition(s)	The user has logged in to the software and opened the query interface.
Postcondition(s)	The user has visualized the occupancy rates of all the rooms.
Input	None.
Process	The system calculates the occupancy rate of all rooms according to the usage frequency of each room in the database. Then display the RoomName and occupancy rate to the user in a certain order.
Output	All room names and occupancy rates are displayed to the user.
Error handling	If any error occurs, the action is interrupted and then a message is shown to the user.
Comment(s)	There are two ways for the system to display the RoomName and occupancy rate: in descending order of occupancy rate or in alphabetical order.

Title	Visualize a chart of room types by capacity.
Identifier	SPEC06
Objective(s)	The user can obtain the types of all rooms and the number of each type classified by their Capacities.
Precondition(s)	The user has logged in to the software and opened the query interface.
Postcondition(s)	The user has visualized the types of all rooms and the number of each type classified by their Capacities.
Input	None.
Process	The system searches for the rooms with the same Capacity and classifies them into one type, then integrates all types into one chart and displays it to the user.
Output	The chart is displayed to the user.
Error handling	If any error occurs, the action is interrupted and then a message is shown to the user.
Comment(s)	The chart shows the number of rooms for each Capacity (room type). e.g. 3 rooms for 38 persons.

Title	The software is able to check the quality of the university schedule data to avoid overflow of rooms.
Identifier	SPEC_NF_01
Objective(s)	The software must be able to check the schedule data of all courses in the university to ensure that only one course is scheduled in a room at the same time. There must be no conflicts in the use of rooms.
Precondition(s)	The schedule data of all courses have been loaded into the database.
Postcondition(s)	The quality of the university schedule data is all confirmed, and there is no conflict in the use of rooms.
Input	The university schedules data for all rooms.
Process	The system compares all the schedule information of each room in turn to determine whether there is a conflict in the schedule. The confirmation message shows to the user after all data processing is over.
Output	A message which tells all the schedule data has been confirmed is displayed to the user.
Error handling	If there is a conflict, the Timeslot of the two courses which have the time conflict will be filtered out and displayed. The user adjusts the schedule of the room.
Comment(s)	None.

Data Format

Data format--Timeslot

The specification respects the Augmented Backus-Naur Form.

File's name:

Filename = 'AB' / 'CD' / 'EF' / 'GH' / 'IJ' / 'KL' / 'MN' / 'OP' / 'QR' / 'ST'

File context:

(Instruction = '+UVUV' CRLF 'Seance 1 S=1 / Seance 2 S=3' CRLF 'Seance 1 S=2 / Seance2 S=4' CRLF) au min 2 séances

Timeslots[] = Instruction Coursename 1*(Type Capacity Daytime Time Subgroupindex Roomname End)

Coursename = '+' 1*(ALPHA/DIGIT) CRLF

Type = DIGIT ',' (D/C/T) DIGIT ','

Capacity = 'P =' 1*DIGIT ','

Daytime = 'H=' ('L'/'MA'/'ME'/ 'J /'V') WSP Time ','

Time = 1*2DIGIT ':' 1*2DIGIT '-' 1*2DIGIT ':' 1*2DIGIT ','

Subgroupindex = ALPHA DIGIT ','

Roomname = 'S=' 1*(ALPHA/DIGIT)

End = '//' CRLF

Data format---iCalendar

File’s name:

File context:

```
iCalendar            = ‘BEGIN’ ‘:’ ‘VCALENDAR’ CRLF
                        iEvent
                       ‘END’ ‘:’ ‘VCALENDAR’ CRLF
iEvent               = ‘VERSION’ ‘:’ DIGIT 1*(‘.’/DIGIT) CRLF
                        ‘PRODID’ ‘:’ TEXT CRLF
                        ‘CALSCALE’ ‘:’ ‘GREGORIAN’ CRLF
                        iEventbody
iEventbody           = ‘BEGIN’ ‘:’ ‘VEVENT’ CRLF
                        ‘UID’ ‘:’ TEXT CRLF
                        ‘DTSTAMP’ ‘:’ DATETIME CRLF
                        ‘DTSTART’ ‘:’ DATETIME CRLF
                        ‘DTEND’ ‘:’ DATETIME CRLF
                        ‘CLASS’ ‘:’ (‘PUBLIC’ / ‘PRIVATE’ / ‘CONFIDENTIAL’) CRLF
                        ‘DESCRIPTION’ ‘:’ TEXT CRLF
                        ‘GEO’ ‘:’ FLOAT ‘;’ FLOAT CRLF
                        ‘LOCATION’ ‘:’ TEXT CRLF
                        ‘STATUS’ ‘:’ (‘TENTATIVE’ / ‘CONFIRMED’ / ‘CONFIRMED’)
                CRLF
```

```
                         ‘SUMMARY’ ‘:’ TEXT CRLF

                         ‘TRANSP’ ‘:’ (‘OPAQUE’ / ‘TRANSPARENT’) CRLF

                         ‘CATEGORIES’ ‘:’ TEXT *(‘,’ TEXT) CRLF

                         ‘COMMENT’ ‘:’ TEXT CRLF

                         ‘CONTACT’ ‘:’ TEXT CRLF


TEXT                     = *(VCHAR / WSP / DQUOTE / DIGIT / ALPHA)

FLOAT                    = ([‘+’] / ‘-’) 1*DIGIT [‘.’ 1*DIGIT]

DATETIME                 =  8DIGIT ‘T’ 6DIGIT ‘Z’
```

Data Semantics

<Specification Name>

sort	
imports	
Description	
Operators	
Axioms	

<Timeslot>

sort	Timeslot
imports	Date, Integer, String
Description	Defines a space of time available for reserving a classroom.
Operators	● Create → Empty Timeslot ● Cons → Creates a new Timeslot with one timestamp ● Hour → Renders the hour portion of the timestamp ● Date → Renders the date portion of the timestamp ● Room → Renders the room that has assigned this timestamp
	● Create: → Timeslot ● Cons: Timeslot → Timeslot ● Hour: Timeslot → Hour (Integer) ● Date: Timeslot → Date (String) ● Room: Timeslot → RoomName (String)

Axioms	ex. 'August 19, 2021; 23:15:30; N101'
	
	Create('August 19, 2021; 23:15:30; N101')
	Create(Cons('August 19, 2021; 23:15:30; N101'))
	Cons(Cons('August 19'),Create(' 2021; 23:15:30; N101'))
	Cons(Date('August 19, 2021'), Create('23:15:30; N101'))
	Cons(Date('August 19, 2021'), Hour('23:15:30'), Room('N101'))

<Calendar>

sort	Calendar
imports	Date, Hour, String, Integer, Timeslot
Description	Defines all the available days, eventually grouped in month number, and year, to position the timeslot of the reservation
Operators	● Create → Empty Calendar ● Cons → Creates a new Calendar with one Timeslot ● Hour → Renders the hour portion of the incoming Timeslot ● Date → Renders the date portion of the incoming Timeslot ● Room → Renders the room that has been assigned to the incoming Timeslot ● Timeslot → Creates a new Timeslot and adds it to the Calendar object
	● Create: → Calendar ● Cons: Timeslot → Calendar ● Hour: Timeslot → Hour (Integer) ● Date: Timeslot → Date (String) ● Room: Timeslot → RoomName (String) ● Timeslot: Date x Hour → Timeslot

Axioms	ex. 'August 19, 2021; 23:15:30; N101'
	
	Create('August 19, 2021; 23:15:30; N101') = error
	Create(Cons('August 19, 2021; 23:15:30; N101'))
	Cons(Cons('August 19'),Create(' 2021; 23:15:30; N101'))
	Cons(Date('August 19, 2021'), Create('23:15:30; N101'))
	Cons(Date('August 19, 2021'), Hour('23:15:30'), Room('N101'))
	
	Create(Timeslot(Date('August 19, 2021; 23:15:30; N101'), Hour('August 19, 2021; 23:15:30; N101')) = Calendar

<Student>

sort	Student
imports	String, Integer
Description	Describes the entity Student.
Operators	• Cons → New Student • Create → Empty Student Object • FirstName → String of Student's First Name • LastName → String of Student's Last Name • StudentId → Alphanumeric String of the Student • Age → Age of Student
	• Cons: String x String x String x Int → New Student • Create: → Student • FirstName: String → Student.FirstName (String) • LastName: String → Student.LastName (String) • StudentId: String → Student.StudnetId (String) • Age: Int → Student.Age
Axioms	Cons(234, 'Williams', 190, 23) → 0 Cons('Jake', 'Williams', 'STD092301', 23) → New Student Obj Create(Cons('Jake', 'Williams', 'STD092301', 23)) → New Student Obj Age('12.fr') → 0 Age(Create(Cons('Jake', 'Williams', 'STD092301', 23))) → 23 FirstName(Create(Cons('Jake', 'Williams', 'STD092301', 23))) → 'Jake' LastName(Create(Cons('Jake', 'Williams', 'STD092301', 23))) → 'Williams'

<Teacher>

sort	Teacher
imports	String, Integer
Description	Describes the entity Teacher
Operators	• Cons → New Teacher • Create → Empty Teacher Object • FirstName → String of Teacher's First Name • LastName → String of Teacher's Last Name • StudentId → Alphanumeric String of the Teacher Id • Age → Age of Teacher
	• Cons: String x String x String x Int → New Teacher • Create: → Teacher • FirstName: String → Teacher.FirstName (String) • LastName: String → Teacher.LastName (String) • StudentId: String → Teacher.TeacherId (String) • Age: Int → Teacher.Age
Axioms	Cons(Create) → New Teacher FirstName(23) → 0 FirstName('Joe') → 0 FirstName(Cons('Joe', 'Doe', '320SJE', 34)) → 'Joe' Cons('Joe', 'Doe', '320SJE', 34) → Teacher Age('wd') → 0

<Class>

sort	Class
imports	String, Teacher, Student, Timeslot, Calendar, Boolean
Description	Defines the name of the Class taken.
Operators	● Cons: String x Teacher x Student x Timeslot x Calendar x Int x Boolean → Class ● Create: → Class ● ClassName: Class → String ● Teacher: Class → Teacher ● NumStudents: Class → Int ● ClassCalendar: Class → Calendar ● ClassTimeslot: Class → Timeslot ● RoomAssigned: Class → Boolean
	● Cons → New Class ● Create → New Class ● ClassName → String of the Name of the assigned Class ● Teacher → Teach Object assigned to the Class ● NumStudents → Integer of the amount of Student Objects assigned to the Class ● ClassCalendar → Calendar Object assigned to the Class ● ClassTimeslot → Timeslot Object assigned to the Class ● RoomAssigned → True if a Room has been assigned to the class; else False
Axioms	

Conclusion

The goal of the project is to develop a new software to track the occupancy of all rooms in the university and simplify the arrangement and management of the rooms.

This software plays a very important role in both teachers and students, as well as administrative staff.

First of all, this software allows teachers and students to search for relevant information (Timeslot) of the course, especially the room where they are located, which promotes the sharing of information. In addition, school personnel can also find rooms that are free at a certain period of time, as well as the schedule for a certain room. This software makes room information clearer, which simplifies the arrangement and management of the room.

In addition, this software can export schedules in iCalendar format. It is very important that the exported iCalendar file is compatible with the formats of various agenda software. Because students and teachers will export their schedule from this software as a school timetable.

For administrative staff, a very important function of this software is to visualize room occupancy rates. This software must be able to visualize rooms occupancy rates and a chart of room types by capacity, so that administrative staff can better determine room resource allocation and plan the construction of future rooms.

In the future new version of this software, the function of booking a room can be added, so that students and teachers can directly make a reservation for the room for collective work after checking which time periods are available, and update the room schedule status in the software.

References

https://en.wikipedia.org/wiki/ICalendar

https://datatracker.ietf.org/doc/html/rfc5545