**3.2 Subsystem Interfaces Specifications**

The system consists of five interfaces: IUserInteraction, IScheduleManagement, ICourseManagement, IPreferenceManagement and ISequenceManagement. Each of those interfaces provides a set of methods that components can publicly access. The classes that implements those interfaces are the ones presented in the architecture diagram, and the use of the method calls are explained in more detail in dynamic design scenario.

**3.2.1 IUserInteraction**

The user interaction interface provides basic methods that a user, either an administrator or a student, will use to interact with the system. Those interactions include login and logout operations. This interface also allows to consult data related to courses and schedules, but no modification on those data is allowed through this interface. The classes implementing the provided methods are the Student, Administrator and User classes.

***3.2.1.1 Specifications for methods implemented by the Student class***

|  |
| --- |
| **Method** |
| browseCourseList() list of Course objects |
| **Specification** |
| ***Description:***  Retrieves and returns a list of courses that a student can take.  ***Input parameter(s):***  none  ***Return type:***  list of Course objects |

|  |
| --- |
| **Method** |
| getSavedSchedule(scheduleID:long) Schedule |
| **Specification** |
| ***Description:***  Given a schedule ID, retrieves and returns a specific saved schedule that represents the schedule of a semester.  ***Input parameter(s):***  scheduleID:long representing the schedule ID  ***Condition(s) for validity:***  scheduleID ≥ 0  ***Return type:***  Schedule |

|  |
| --- |
| **Method** |
| login(studentID:long, password:String) LogonPage |
| **Specification** |
| ***Description:***  Logs a student into the system, provided that the student ID and password passed as parameters are valid.  ***Input parameter(s):***  studentID:long representing the ID of the student to be logged in  password:String representing the password entered by the user  ***Conditions for validity:***  studentID ≥ 0  password is not empty  ***Return type:***  LogonPage |

|  |
| --- |
| **Method** |
| logout(studentID:long) LogoutPage |
| **Specification** |
| ***Description:***  Logs a student out of the system.  ***Input parameter(s):***  studentID:long representing the ID of the student to be logged out  ***Conditions for validity:***  studentID ≥ 0  ***Return type:***  LogoutPage |

|  |
| --- |
| **Method** |
| searchCourse(courseID:long) list of Section objects |
| **Specification** |
| ***Description:***  Retrieves and returns information about a course, including the course’s sections, given the course ID.  ***Input parameter(s):***  courseID:long representing the course ID  ***Conditions for validity:***  courseID ≥ 0  ***Return type:***  list of Section objects |

|  |
| --- |
| **Method** |
| selectWeek(day:int, month:int) Schedule |
| **Specification** |
| ***Description:***  Provides the caller with a weekly schedule view given the day and month as valid integer values for the targeted week.  ***Input parameter(s):***  day:int representing a calendar day in the targeted week  month:int representing the month of the targeted week  ***Conditions for validity:***  1 ≤ day ≤ 31  1 ≤ month ≤ 12  ***Return type:***  Schedule |

|  |
| --- |
| **Method** |
| viewAcademicRecord(studentID:long) AcademicRecord |
| **Specification** |
| ***Description:***  Allows to view the academic record of a student given the student’s ID.  ***Input parameter(s):***  studentID:long representing the ID of the student  ***Conditions for validity:***  studentID ≥ 0  ***Return type:***  AcademicRecord |

|  |
| --- |
| **Method** |
| viewCourseSequence(studentID:long) Sequence |
| **Specification** |
| ***Description:***  Retrieves and returns the sequence of a student given the ID of the student.  ***Input parameter(s):***  studentID:long representing the ID of the student  ***Conditions for validity:***  studentID ≥ 0  ***Return type:***  Sequence |

***3.2.1.2 Specifications for methods implemented by the Administrator class***

|  |
| --- |
| **Method** |
| addUser(username:String, firstname:String, lastname:String, netname:String, password:String, privilege:int) User |
| **Specification** |
| ***Description:***  Adds a new user to the system, provided its username, first name, last name, net name, password and privilege code. Upon successful completion, the newly create user is returned.  ***Input parameter(s):***  username:String representing the username of the user to be created  firstname:String representing the first name of the user to be created  lastname:String representing the last name of the user to be created  netname:String representing the net name of the user to be created  password:String representing the password of the user to be created  privilege:int representing a code that indicates whether the new user is a student or an administrator  ***Conditions for validity:***  username.length > 0  firstname.length > 0  lastname.length > 0  netname.length > 0  password.length > 5  privilege = 0 1  username does not exist  netname does not exist  ***Return type:***  User |

|  |
| --- |
| **Method** |
| browseCourses() list of Course objects |
| **Specification** |
| ***Description:***  Returns a list of all courses offered by the university.  ***Input parameter(s):***  none  ***Return type:***  list of Course objects |

|  |
| --- |
| **Method** |
| createCourse(courseCode:int, description:String, credits:double, type:String) Course |
| **Specification** |
| ***Description:***  Adds a new course along with its description into the system provided the course code, the description, the credits and the type of course. Returns the newly create course.  ***Input parameter(s):***  courseCode:String representing the code of the course to be added  description:String representing the description of the course to be added  credits:double representing the number of credits allocated to the course  type:String representing the type of the course  ***Conditions for validity:***  courseCode ≥ 100  description is not empty  credits ≤ 0  type is not empty  ***Return type:***  Course |

|  |
| --- |
| **Method** |
| deleteCourse(courseID:long) list of Course objects |
| **Specification** |
| ***Description:***  Deletes a course given its course ID and returns an updated list of courses.  ***Input parameter(s):***  courseID:long representing the ID of the course to be deleted  ***Conditions for validity:***  courseID ≥ 0  ***Return type:***  list of Course objects |

|  |
| --- |
| **Method** |
| deleteUser(userID:long) list of User objects |
| **Specification** |
| ***Description:***  Deletes a user given its user ID and returns an updated list of users.  ***Input parameter(s):***  userID:long representing the ID of the user to be deleted  ***Conditions for validity:***  userID ≥ 0  ***Return type:***  list of User objects |

|  |
| --- |
| **Method** |
| login(adminID:long, password:String) LogonPage |
| **Specification** |
| ***Description:***  Logs an administrator into the system, provided that the admin ID and password passed as parameters are valid.  ***Input parameter(s):***  adminID:long representing the ID of the administrator to be logged in  password:String representing the password entered by the user  ***Conditions for validity:***  adminID ≥ 0  password is not empty  ***Return type:***  LogonPage |

|  |
| --- |
| **Method** |
| logout(adminID:long) LogoutPage |
| **Specification** |
| ***Description:***  Logs an administrator out of the system.  ***Input parameter(s):***  adminID:long representing the ID of the administrator to be logged out  ***Conditions for validity:***  adminID ≥ 0  ***Return type:***  LogoutPage |

|  |
| --- |
| **Method** |
| updateCourse(courseID:long, courseCode:String, courseDescription:String, credits:double, courseType:String) Course |
| **Specification** |
| ***Description:***  Given the ID of a course, allows to update the course information such as the course code, course description, credits allocated and type of course. Once completed, the updated course is returned.  ***Input parameter(s):***  courseID:long representing the ID of the course to be updated  courseCode:String representing the new code of the course  credits:double representing the new number of credits allocated for the course  courseType:String representing the new type of course  ***Conditions for validity:***  courseID ≥ 0  credits ≥ 0  ***Return type:***  Course |

|  |
| --- |
| **Method** |
| updateUser(userID:long, username:String, firstname:String, lastname:String, netname:String, privilege:int) User |
| **Specification** |
| ***Description:***  Given the ID of a user and a set of username, first name, last name, net name and privilege information, updates the user’s information accordingly. Once updated, returns the updated user.  ***Input parameter(s):***  userID:long representing the ID of the user to be updated  username:String representing the new username to be set  firstname:String representing the new first name to be set  lastname:String representing the new last name to be set  netname:String representing the new net name to be set  privilege:int representing the new privilege level (student or administrator) to be set  ***Conditions for validity:***  username.length > 0  firstname.length > 0  lastname.length > 0  netname.length > 0  privilege = 0 1  username does not exist  netname does not exist  ***Return type:***  User |

|  |
| --- |
| **Method** |
| userSearch(userID:String, username:String, firstname:String, lastname:String, netname:String) list of User objects |
| **Specification** |
| ***Description:***  Based on filtering search criteria such as the user ID, the username, first name, last name and net name, this method returns a list of user corresponding to the search criteria. Note that the search criteria are based on a ‘contains’ relationship, and not ‘exactly is’.  ***Input parameter(s):***  userID:String representing the filtering criteria for the user ID, can be preceded by comparative operators  username:String representing a part of, or the complete username to be filtered when looking for users  firstname:String representing a part of, or the complete first name to be filtered when looking for users  lastname:String representing a part of, or the complete last name to be filtered when looking for users  netname:String representing a part of, or the complete net name to be filtered when looking for users  ***Conditions for validity:***  none  ***Return type:***  list of User objects |

|  |
| --- |
| **Method** |
| viewCourse(courseID:long) Course |
| **Specification** |
| ***Description:***  Returns a course to be viewed by the administrator given its course ID.  ***Input parameter(s):***  courseID:long representing the ID of the course to be viewed  ***Conditions for validity:***  courseID ≥ 0  ***Return type:***  Course |

|  |
| --- |
| **Method** |
| viewUser(userID:long) User |
| **Specification** |
| ***Description:***  Returns the user corresponding to the valid given user ID.  ***Input parameter(s):***  userID:long representing the ID of the user to be viewed  ***Conditions for validity:***  userID ≥ 0  ***Return type:***  User |

***3.2.1.3 Specifications for methods implemented by the User class***

|  |
| --- |
| **Method** |
| isNameValid(name:String) boolean |
| **Specification** |
| ***Description:***  Verifies whether the given name, either first name or last name, is valid. For the name to be valid it must be non-empty.  ***Input parameter(s):***  name:String representing the name to verify  ***Conditions for validity:***  none  ***Return type:***  boolean |

|  |
| --- |
| **Method** |
| isNetnameValid(netname:String) boolean |
| **Specification** |
| ***Description:***  Verifies whether the given net name is valid. For the net name to be valid, it must be non-empty and unique. Returns true if the username is valid, false otherwise.  ***Input parameter(s):***  netname:String representing the netname to verify  ***Conditions for validity:***  none  ***Return type:***  boolean |

|  |
| --- |
| **Method** |
| isPrivilegeValid(privilege:int) boolean |
| **Specification** |
| ***Description:***  Verifies whether the privilege level set to a user is valid. For the privilege level to be valid, it must be either 0 or 1 (student or administrator). Returns true if the privilege is valid, false otherwise.  ***Input parameter(s):***  privilege:int representing the privilege level to verify  ***Conditions for validity:***  none  ***Return type:***  boolean |

|  |
| --- |
| **Method** |
| isUsernameValid(username:String) boolean |
| **Specification** |
| ***Description:***  Verifies whether the given username is valid. For the username to be valid, it must be non-empty and unique. Returns true if the username is valid, false otherwise.  ***Input parameter(s):***  username:String representing the username to verify  ***Conditions for validity:***  none  ***Return type:***  boolean |

**3.2.2 IScheduleManagement**

The IScheduleManagement interface consists of a set of methods that allow another component to access and modify data about a schedule. The classes Schedule and SavedSchedules implement the provided methods.

***3.2.2.1 Specifications for methods implemented by the Schedule class***

|  |
| --- |
| **Method** |
| addCourse(courseID:long, sectionID:long) Schedule |
| **Specification** |
| ***Description:***  Adds a course given the course ID and the section ID to a student’s schedule.  ***Input parameter(s):***  courseID:long representing the ID of the course  sectionID:long representing the ID of the section  ***Conditions for validity:***  courseID ≥ 0  sectionID ≥ 0  ***Return type:***  Schedule |

|  |
| --- |
| **Method** |
| bookTimeSlot(startTime:int, endTime:int) |
| **Specification** |
| ***Description:***  Sets time slots in a scheduled as occupied given the starting and ending time of the time block to book.  ***Input parameter(s):***  startTime:int representing the time at which the time block starts  endTime:int representing the time at which the time block ends  ***Conditions for validity:***  0 ≤ startTime ≤ endTime  ***Return type:***  void |

|  |
| --- |
| **Method** |
| deleteSchedule(scheduleID:long) |
| **Specification** |
| ***Description:***  Deletes a schedule given its schedule ID.  ***Input parameter(s):***  scheduleID:long representing the ID of the schedule to be deleted  ***Conditions for validity:***  scheduleID ≥ 0  ***Return type:***  void |

|  |
| --- |
| **Method** |
| freeTimeBlock(startTime:int, endTime:int) |
| **Specification** |
| ***Description:***  Frees a time block of a schedule given the start and end time of the block.  ***Input parameter(s):***  startTime:int representing the time at which the time block starts  endTime:int representing the time at which the time block ends  ***Conditions for validity:***  0 ≤ startTime ≤ endTime  ***Return type:***  void |

|  |
| --- |
| **Method** |
| getFreeCredits(studentID:long) int |
| **Specification** |
| ***Description:***  Returns the number of free credits a student is still allowed to take.  ***Input parameter(s):***  studentID:long representing the ID of the student  ***Conditions for validity:***  studentID ≥ 0  ***Return type:***  int |

|  |
| --- |
| **Method** |
| getWeeklyView(startDay:int, startMonth:int) Schedule |
| **Specification** |
| ***Description:***  Generates a weekly view of a schedule given the starting date.  ***Input parameter(s):***  startDay:int representing the day on which the week starts  startMonth:int representing the month in which the week is found  ***Conditions for validity:***  1 ≤ startDay ≤ 31  1 ≤ startMonth ≤ 12  ***Return type:***  Schedule |

|  |
| --- |
| **Method** |
| isBlockFree(startTime:int, endTime:int) boolean |
| **Specification** |
| ***Description:***  Verifies whether a time block defined by the given starting and ending time is free in a schedule.  ***Input parameter(s):***  startTime:int representing the time at which the time block starts  endTime:int representing the time at which the time block ends  ***Conditions for validity:***  0 ≤ startTime ≤ endTime  ***Return type:***  boolean |

|  |
| --- |
| **Method** |
| removeCourse(sectionID:long, studentID:long) Schedule |
| **Specification** |
| ***Description:***  Removes a given course section from the schedule of a student.  ***Input parameter(s):***  sectionID:long representing the ID of the section to be removed  studentID:long representing the ID of the student  ***Conditions for validity:***  sectionID ≥ 0  studentID ≥ 0  ***Return type:***  Schedule |

***3.2.2.2 Specifications for methods implemented by the SavedSchedules class***

|  |
| --- |
| **Method** |
| getAllSavedSchedules(studentID:long) list of Schedule objects |
| **Specification** |
| ***Description:***  Retrieves and returns all saved schedules of a student given its student ID.  ***Input parameter(s):***  studentID:long representing the student ID  ***Condition(s) for validity:***  studentID ≥ 0  ***Return type:***  list of Schedule objects |

|  |
| --- |
| **Method** |
| getSchedule(scheduleID:long) Schedule |
| **Specification** |
| ***Description:***  Retrieves and returns the schedule of a semester given the ID of the schedule.  ***Input parameter(s):***  scheduleID:long representing the ID of the schedule  ***Conditions for validity:***  scheduleID ≥ 0  ***Return type:***  Schedule |

|  |
| --- |
| **Method** |
| saveSchedule(scheduleID:long) list of Schedule objects |
| **Specification** |
| ***Description:***  Saves a new schedule related to a student given the schedule ID, then returns an updated list of schedules that have been saved.  ***Input parameter(s):***  scheduleID:long representing the ID of the schedule to be saved  ***Conditions for validity:***  scheduleID ≥ 0  ***Return type:***  list of Schedule objects |

**3.2.3 ICourseManagement**

The ICourseManagement interface allows to consult information about academic courses as well as the sections of each course. It also allows to perform modification on courses, such as adding and removing students from a course. The ICourseManagement interface is also responsible for creating course sequences and generating schedules. The Course, CoursesPassed and Section classes are the ones implementing the provided methods.

***3.2.3.1 Specifications for methods implemented by the Course class***

|  |
| --- |
| **Method** |
| createSequence() Sequence |
| **Specification** |
| ***Description:***  Creates and returns a course sequence for a student.  ***Input parameter(s):***  none  ***Return type:***  Sequence |

|  |
| --- |
| **Method** |
| generateSchedule() Schedule |
| **Specification** |
| ***Description:***  Generates and returns a student’s schedule.  ***Input parameter(s):***  none  ***Return type:***  Schedule |

|  |
| --- |
| **Method** |
| getAllSections(courseID:long) list of Section objects |
| **Specification** |
| ***Description:***  Returns a list containing all the sections of a course.  ***Input parameter(s):***  courseID:long representing the ID of the course to be consulted  ***Conditions for validity:***  courseID ≥ 0  ***Return type:***  list of Section objects |

|  |
| --- |
| **Method** |
| getCourseList() list of Course objects |
| **Specification** |
| ***Description:***  Returns a list containing all courses that a student can take.  ***Input parameter(s):***  none  ***Return type:***  list of Course objects |

|  |
| --- |
| **Method** |
| getCredits(courseID:long) int |
| **Specification** |
| ***Description:***  Returns the number of assigned credits of a course.  ***Input parameter(s):***  courseID:long representing the ID of the course  ***Conditions for validity:***  courseID ≥ 0  ***Return type:***  int |

|  |
| --- |
| **Method** |
| getPrerequisites() list of Prerequisite objects |
| **Specification** |
| ***Description:***  Returns a list of prerequisites for a course, or an empty list if no prerequisite courses are required for a course.  ***Input parameter(s):***  none  ***Return type:***  list of Prerequisite objects |

|  |
| --- |
| **Method** |
| selectCourse(courseID:long) list of Section objects and Prerequisite objects |
| **Specification** |
| ***Description:***  Given the ID of the selected course, returns information including the sections and prerequisites of that course.  ***Input parameter(s):***  courseID:long representing the ID of the course  ***Conditions for validity:***  courseID ≥ 0  ***Return type:***  (mixed) list of Section objects and Prerequisite objects |

***3.2.3.2 Specifications for methods implemented by the CoursesPassed class***

|  |
| --- |
| **Method** |
| checkPassed(prereqs:Course, studentID:long) boolean |
| **Specification** |
| ***Description:***  Verifies that a student has passed a prerequisite course. Returns true if the student has passed, false otherwise.  ***Input parameter(s):***  courseID:long representing the ID of the prerequisite course  studentID:long representing the ID of the student to verify  ***Conditions for validity:***  courseID ≥ 0  studentID ≥ 0  ***Return type:***  Boolean |

|  |
| --- |
| **Method** |
| gatherCoursesPassed(studentID:long) list of Course objects |
| **Specification** |
| ***Description:***  Retrieves and returns a list of courses that a student has successfully completed.  ***Input parameter(s):***  studentID:long representing the ID of the student  ***Conditions for validity:***  studentID ≥ 0  ***Return type:***  list of Course objects |

***3.2.3.3 Specifications for methods implemented by the Section class***

|  |
| --- |
| **Method** |
| addToList(studentID:long) |
| **Specification** |
| ***Description:***  Adds a student to the list of registered student for a section given the student ID.  ***Input parameter(s):***  studentID:long representing the ID of the student  ***Conditions for validity:***  studentID ≥ 0  ***Return type:***  void |

|  |
| --- |
| **Method** |
| getSection(sectionID:long) Section |
| **Specification** |
| ***Description:***  Retrieves and returns information about a course given the section ID.  ***Input parameter(s):***  sectionID:long representing the ID of the section  ***Conditions for validity:***  sectionID ≥ 0  ***Return type:***  Section |

|  |
| --- |
| **Method** |
| getSections() list of Section objects |
| **Specification** |
| ***Description:***  Retrieves and returns a list of section associated with a course.  ***Input parameter(s):***  none  ***Return type:***  list of Section objects |

|  |
| --- |
| **Method** |
| getTimes(sectionID:long) list of int |
| **Specification** |
| ***Description:***  Returns a list of integers representing the starting and ending time of each time block that a section holds.  ***Input parameter(s):***  sectionID:long representing the ID of the section  ***Conditions for validity:***  sectionID ≥ 0  ***Return type:***  list of int |

|  |
| --- |
| **Method** |
| removeStudent(studentID:long) |
| **Specification** |
| ***Description:***  Removes a student from the list of registered student of a course.  ***Input parameter(s):***  studentID:long representing the ID of the student to be removed  ***Conditions for validity:***  studentID ≥ 0  ***Return type:***  void |

**3.2.4 IPreferenceManagement**

The preference management interface allows to access and modify data related to the scheduling preferences of a student. The only class implementing the methods provided by this interface is the Preferences class.

***3.2.4.1 Specifications for methods implemented by the Preferences class***

|  |
| --- |
| **Method** |
| choosePreference(pref:Preference) Sequence |
| **Specification** |
| ***Description:***  Adds a preference to the list of scheduling preferences of a student and returns the list of all chosen preferences.  ***Input parameter(s):***  pref:Preference representing a chosen preference  ***Conditions for validity:***  pref != null  ***Return type:***  list of Preference objects |

|  |
| --- |
| **Method** |
| getPreferenceList() list of Preference objects |
| **Specification** |
| ***Description:***  Retrieves and returns a list of preferences assigned to a student.  ***Input parameter(s):***  none  ***Return type:***  list of Preference objects |

**3.2.5 ISequenceManagement**

The sequence management interface provides methods to access and modify the course sequence of a student. The only class implementing the provided methods is the Sequence class.

***3.2.5.1 Specifications for methods implemented by the Sequence class***

|  |
| --- |
| **Method** |
| getSequence() Sequence |
| **Specification** |
| ***Description:***  Returns an object representing a course sequence of a student.  ***Input parameter(s):***  none  ***Return type:***  Sequence |

|  |
| --- |
| **Method** |
| updateSequence(courseID:long) |
| **Specification** |
| ***Description:***  Updates a course sequence after a course removal given the ID of the course removed.  ***Input parameter(s):***  courseID:long representing the ID of the course removed  ***Conditions for validity:***  courseID ≥ 0  ***Return type:***  void |