sked-it
Analysis Model

Submitted to:

Asst. Prof. Ma. Rowena C. Solamo
Faculty Member
Department of Computer Science
College of Engineering
University of the Philippines, Diliman

Submitted by:
Dela Sierra, Joshua Joseph Riki V.
Garcia, Patric Charles M.
Granda, Justin Tristan Gabriel R.

In partial fulfillment of Academic Requirements
for the course
CS 191 Software Engineering I
of the
1st Semester, AY 2019-2020

Unique Reference:

The documents are stored in the https://github.com/jvdelasierra/sked-it/tree/master/02-Requirements%20Engineering/Project%20Deliverables referenced 3-sked-it-Analysis Model.pdf..

Purpose:

The purpose of this document is to formalize the Analysis Method of sked-it by deriving it from the Use Case Model and Specifications

Audience:

The target audience of this document includes the developers of the sked-it project and any person who wishes to learn more about the project.

Revision Control:

Revision Date	***Person Responsible***	***Version Number***	***Modification***
09/28/2019	Joshua V. Dela Sierra	1.0	Prepared Initial Document
10/01/2019	Joshua V. Dela Sierra	1.1	Added Boundary Classes
10/02/2019	Joshua V. Dela Sierra	1.2	Added Entity Classes
10/02/2019	Justin Tristan Gabriel R. Granda	1.3	Added Control Classes
10/02/2019	Justin Tristan Gabriel R. Granda	2.0	Updated Boundary Class Descriptions Added Initial Data to Behavioral Model
10/03/19	Justin Tristan Gabriel R. Granda	3.0	Added Diagrams for Behavioral Model Use Case 3.0
10/04/19	Justin Tristan Gabriel R. Granda	4.0	Added Diagrams for Behavioral Model Use Case 2.2

System Name: Sked-it

Description: Sked-it is an app that we are proposing to make to help people manage their own personal schedules while being able to set-up meetings with other people. The app aims to provide users with a way to make a "routine" schedule which they could update if they have other things to do. One of the main features that we will implement is a Group Scheduling system where people in the same group will be allowed to view a portion of another person's schedule (controlled by the owner of the schedule) within the same group so that the people involved in the group can easily set-up their meetings based on their common free time. A person will then be notified of their schedule with a reminder sent to their phones.

***Analysis Model*:**

Place here the class diagram of the analysis model.

Boundary Classes:

Class Name	Description
MonthCalendarUI	This is the interface of the end user to the system whenever he/she needs to manage his/her monthly calendar. Responsibilities: public void addEventCalendar(boolean addOrDelete, String date, String time, String description, String groupPrivacy) public void updateEventCalendar(String eventID, String date, String time, String description, String groupPrivacy) public void deleteEventCalendar(String eventID, boolean addOrDelete)
WeeklyTemplateCalendarUI	This is the interface of the end user to the system whenever he/she needs to manage his/her weekly template calendar. Responsibilities: public void addEventCalendar(boolean addOrDelete, String date, String time, String description, String groupPrivacy) public void updateEventCalendar(String eventID, String date, String time, String description, String groupPrivacy) public void deleteEventCalendar(String eventID, boolean addOrDelete)
WeeklyCalendarUI	This is the interface of the end user to the system whenever he/she needs to manage his/her weekly calendar. Responsibilities: public void addEventCalendar(boolean addOrDelete, String date, String time, String description, String groupPrivacy) public void updateEventCalendar(String eventID, String date, String time, String description, String groupPrivacy) public void deleteEventCalendar(String eventID, boolean addOrDelete)
DailyCalendarUI	This is the interface of the end user to the system whenever he/she needs to manage his/her daily calendar. Responsibilities: public void addEventCalendar(boolean addOrDelete, String date, String time, String description, String groupPrivacy) public void updateEventCalendar(String eventID, String date, String time, String description, String groupPrivacy) public void deleteEventCalendar(String eventID, boolean addOrDelete)
LoginUI	This is the interface of the end user to the system whenever he/she needs to enter the system. Responsibility: public String enterLoginDetails(String email, String password)
CreateProfileUI	This is the interface of the end user to the system whenever he/she needs to create an account for the system. Responsibility: public String enterProfileDetails(String email, String password, boolean agreeOnTerms)
ForgetPasswordUI	This is the interface of the end user to the system whenever he/she forgot the password for the system. Responsibility: public String enterDetails(String email)
LinkScheduleUI	This is the interface of the end user to the system whenever he/she needs to link a

	schedule to a group. Responsibility: public String enterDetails(String groupCalendarID, string eventID)
ConfirmScheduleIO.	This is the interface of the end user to the system whenever he/she needs to link a schedule to a group. Responsibility: public String confirmDetails(String groupCalendarID, string eventID) public String cancelDetails(String groupCalendarID, string eventID)

Control Classes:

Class Name	Description
ManageCalendarController	This controller handles creating, updating and deleting calendars for the user. Responsibilities: public void updateCalendar(String eventID, String date, String time, String description, String groupPrivacy)
AccountController	This controller is responsible for maintaining existing user accounts, including logging in and managing forgotten passwords. Responsibilities: public String loginSystem(String email, String password) public void RenewPassword(String email)
LinkScheduleController	This controller handles schedule linking (i.e. finding common free time between schedules). Responsibilities: public Schedule linkSchedules(String eventID_1, String eventID_2) public boolean isCommonTimeFound(String eventID)
ConfirmationController	This controller handles cases when users need to confirm their schedule with the group. Responsibilities: public String setConfirmedSched(Schedule a) public void cancelConfirmedSched(Schedule a)

Entity Classes:

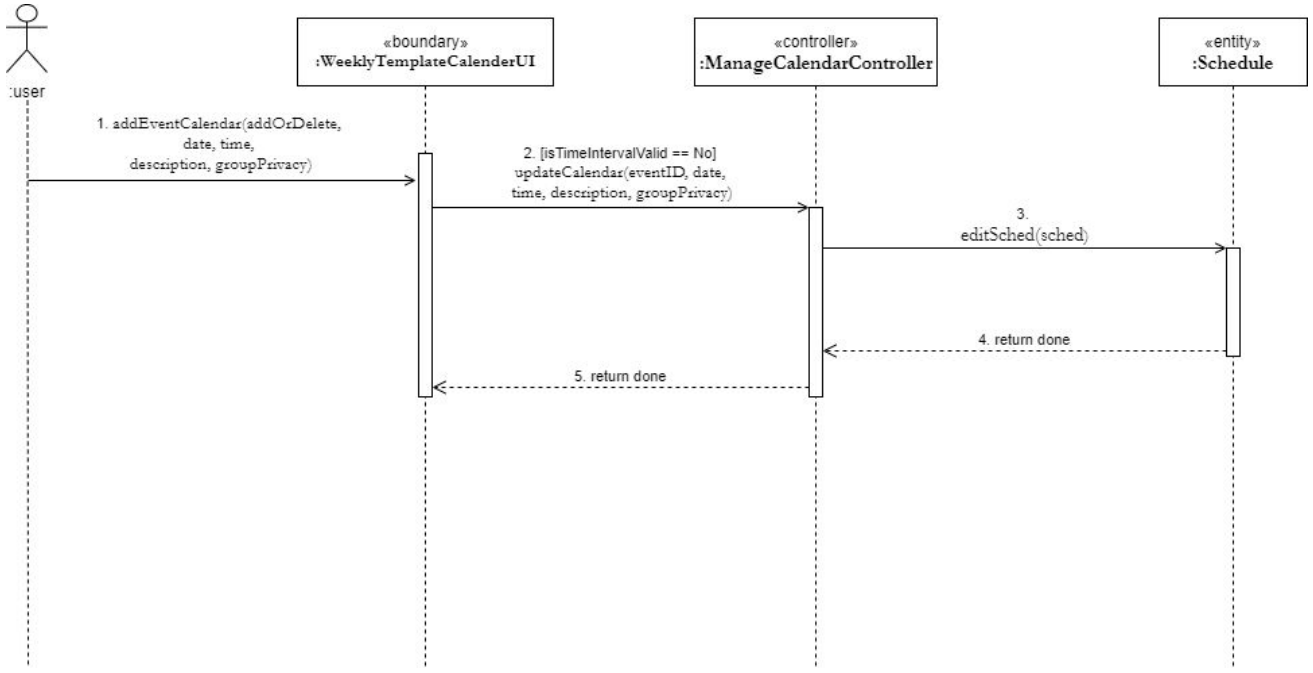
Class Name	**Description**
Individual	This is the entity class Individual, which contains the data about an individual user. Attributes: private String userID private String email private String password private Schedule schedules Responsibilities public void addUser(Individual a) public void editUser(Individual a) public void deleteUser(Individual a)
Schedule	This is the entity class Schedule, which contains the data about an individual user. Attributes: private String eventID private String date private String time private String description private String groupPrivacy Responsibilities: public void editSched(Schedule a)
Group	This is the entity class Group, which contains the data about groups. Attributes: private String groupCalendarID private Individual individual private String nameOfLeader Responsibilities: public void addGroup(Group a) public void editGroup(Group a) public void deleteGroup(Group a)

Behavioral Model:

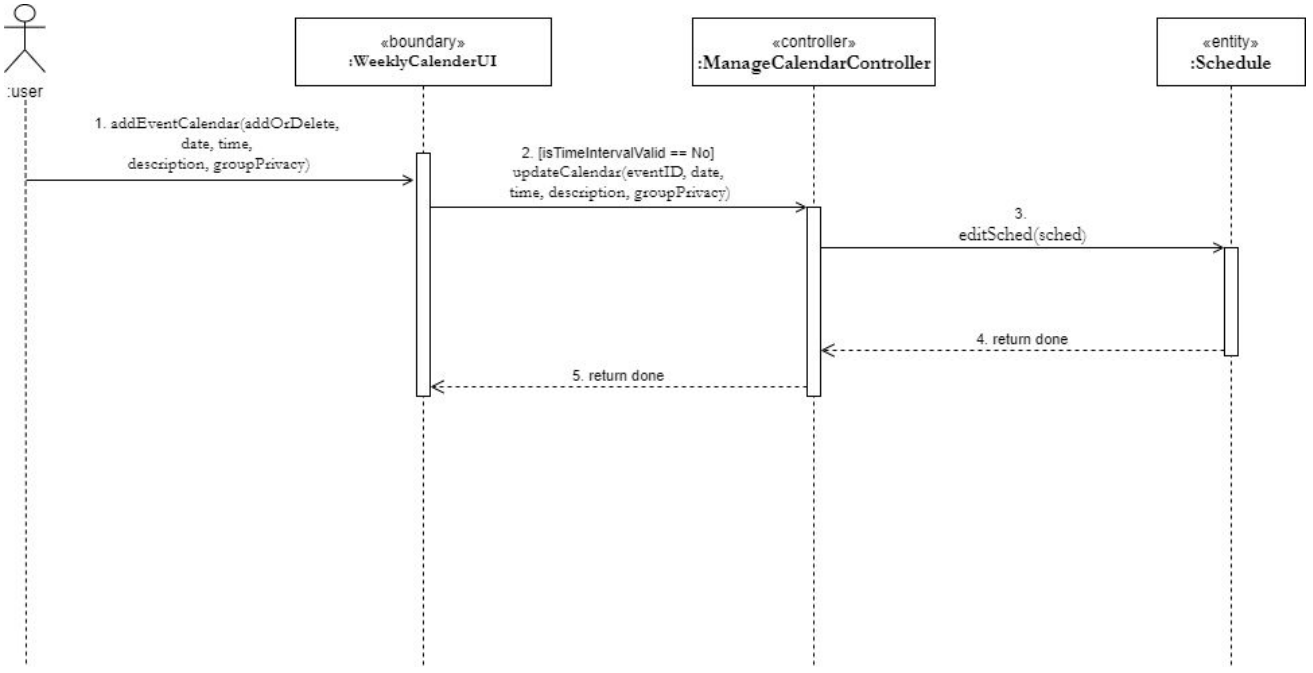
Use-Case Name: 1.3 Create Schedule
Description:

Individual users are able to create and upload their schedule/s. They simply input the times per day of the week that they are free/busy. The user can create a general schedule and have different schedules for different groups.

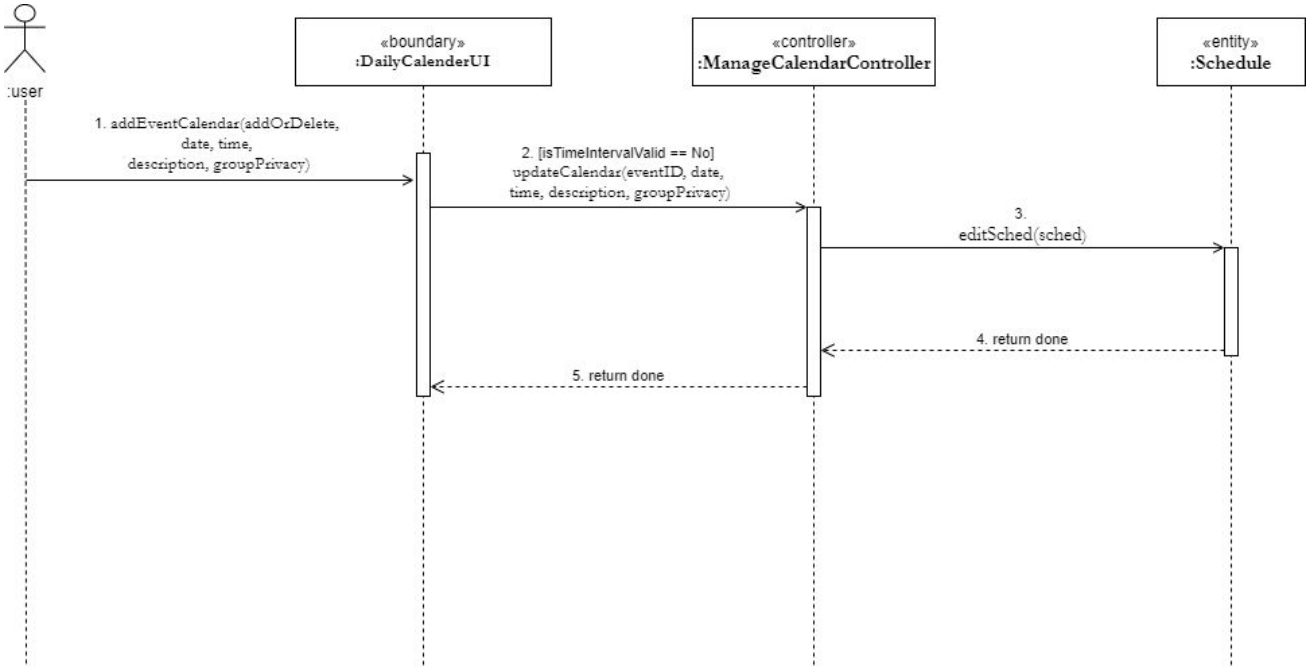
Scenario 1: Successful Creation of Routine Schedule



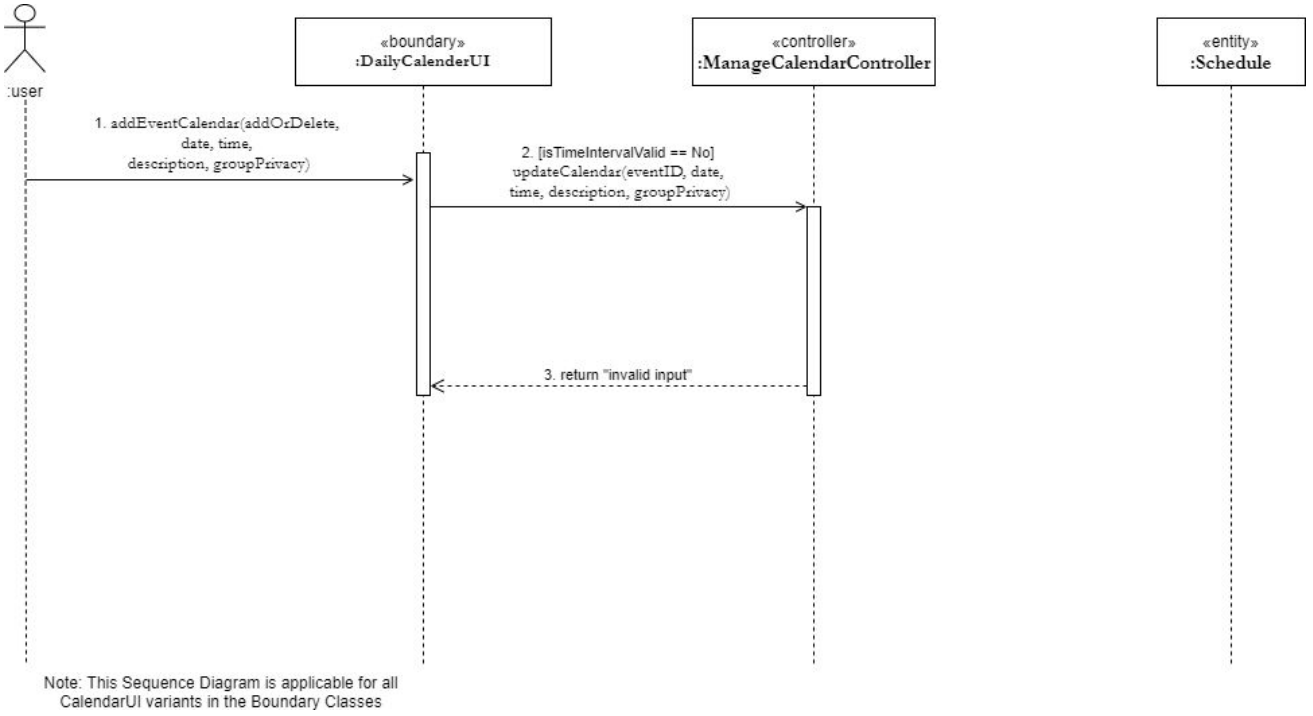
Scenario 2: Successful Creation of Group Free Time Schedule



Scenario 3: Successful Creation of Date-specific Schedule



Scenario 4: Wrong Time Intervals

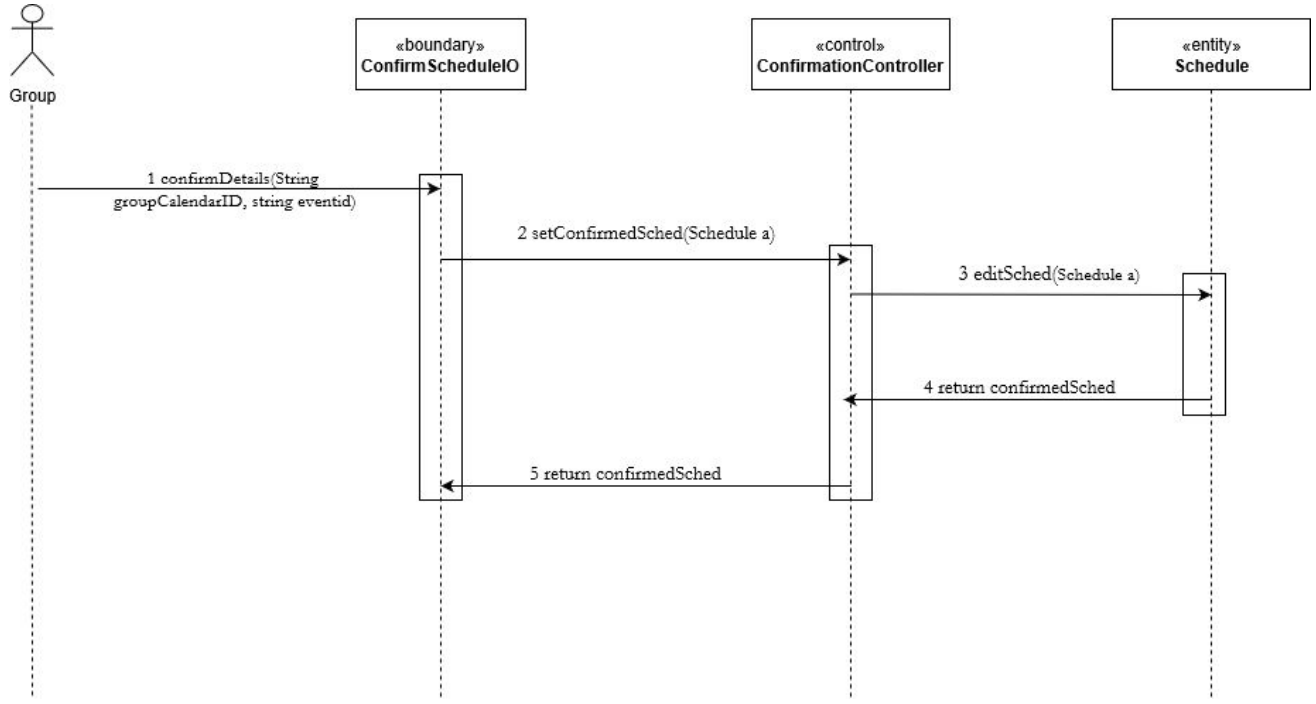


Use-Case Name: 2.2 Confirmation of Common Schedule

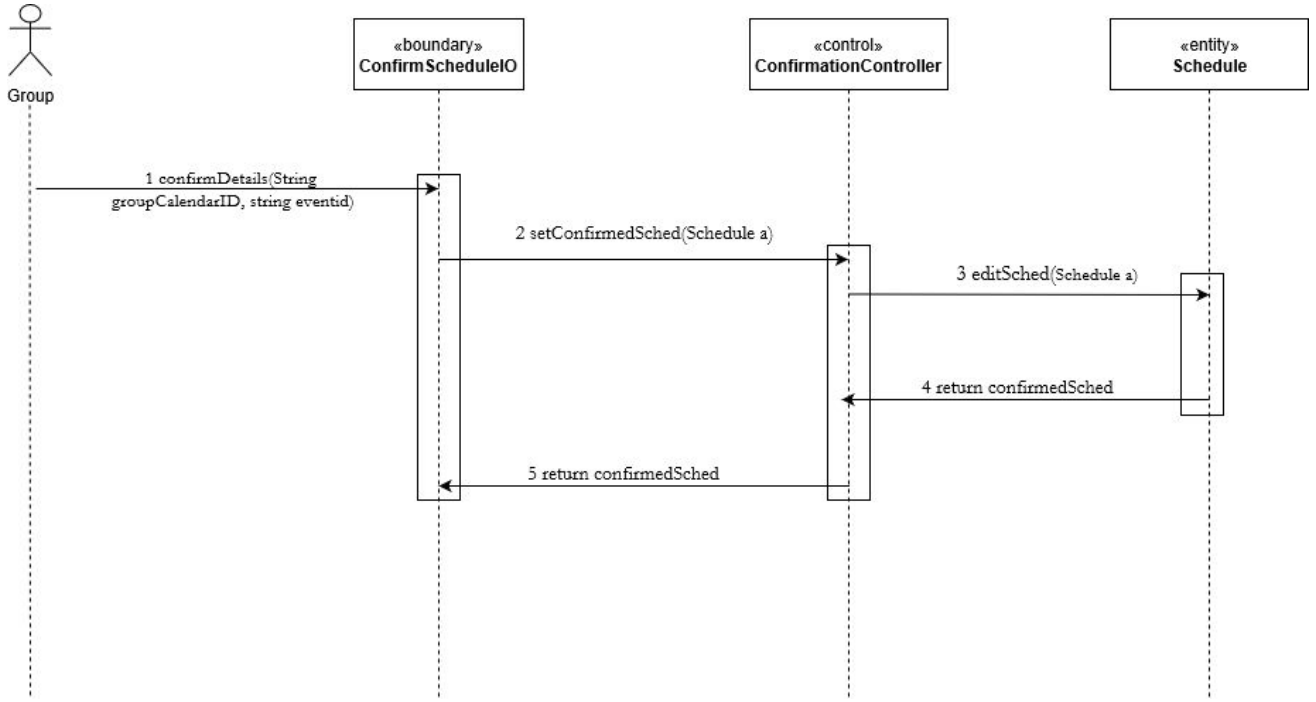
Description:

The group can confirm their common schedule for a certain activity/event. This requires the Link Schedule functionality of the system. An automatic notification (notify group) will be queued shortly.

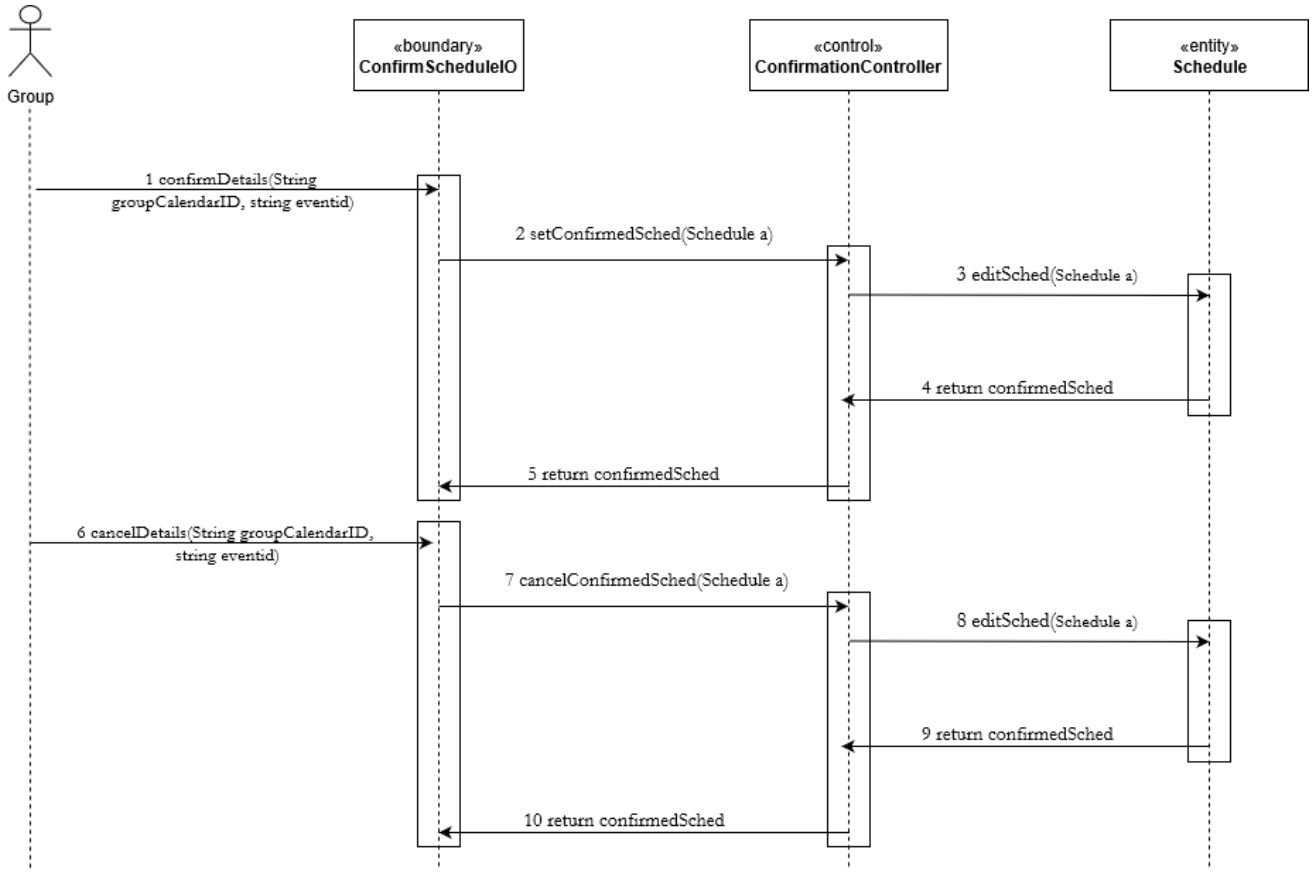
Scenario 1: Everyone Agrees and a time is confirmed



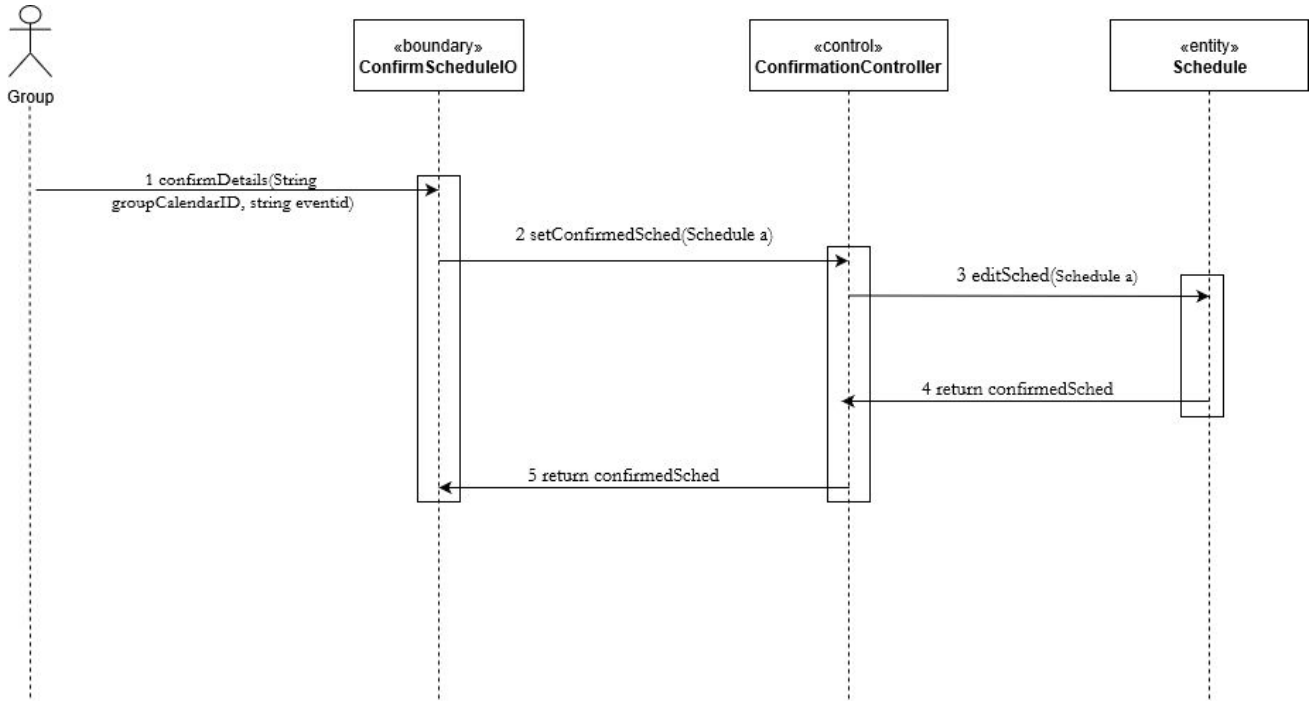
Scenario 2: Some of the members are not available but have confirmed



Scenario 3: Some of the members have confirmed but redacted their confirmation after



Scenario 4: Some of the members have confirmed but overrides the schedule with a schedule of a higher priority

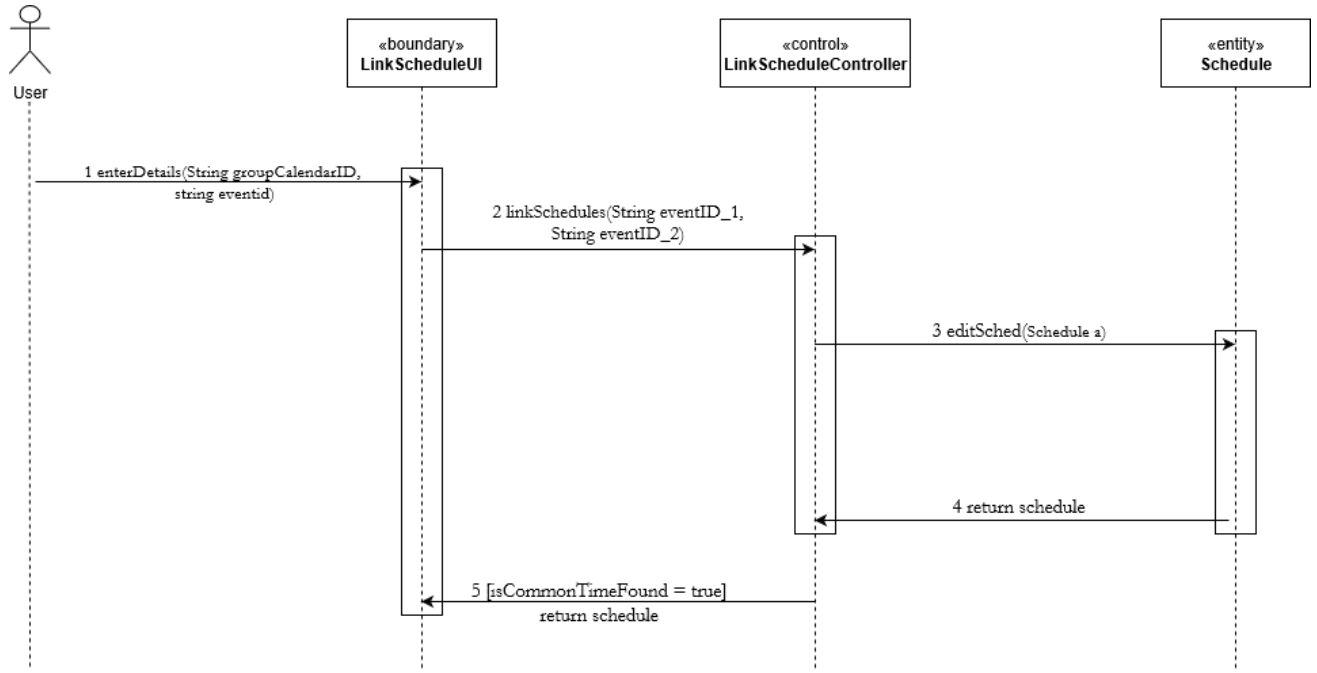


Use-Case Name: 3.0 Link Schedules

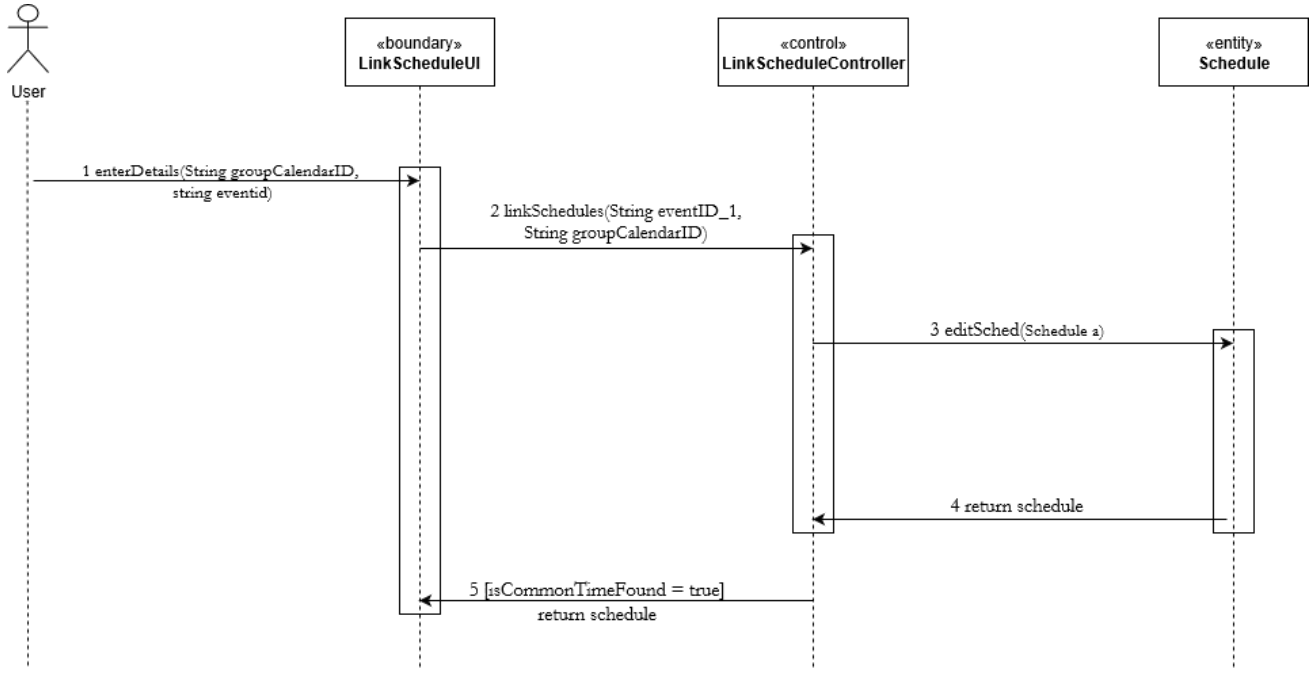
Description:

This is the system's main functionality. Given some schedules, it returns the common free time found between them. This can be done between individual users or within a group.

Scenario 1: Common free time is successfully found between individual users



Scenario 2: Common free time is successfully found between group members



Scenario 3: No common free time was found between the members