Requirements Document

1. Purpose

The goal of this Project is to create a system to manage the front-desk activities of the “Linnaeus Hotel”, which manages 2 different buildings, 1 within the Växjö Campus and 1 within the Kalmar Campus. Currently they use the software JHotel which according to the client doesn’t comply with the current requirements of the Linnaeus Hotel. Therefore we have been contracted to re-engineer the existing legacy software “JHotel” version 0.62 to comply with the Linnaeus Hotel needs.

2. Scope

The scope will cover the whole software needed for the application to fulfill the requirements. That includes graphical user interface(GUI) as well as a solution of storage of data and communication between the two hotels.

3. Stakeholders

The main stakeholders are management of Linnaeus Hotel which are the project owner and the front desk clerks that will be the main user of the system. The hotel customers can be seen as an indirect stakeholder which get affected of the design decisions of the application.

4. Environment

The application will be run simultaneously from two places, if one change any data, that has to be updated in the other as well. Linnaeus has no preferences how this communication should be done other than it cannot take more than 2 seconds to search for an available room. Linnaeus have requested application to programmed using Java so that Linnaeus Hotel is not bound to any one operating system.

5. Non-Functional Requirements

In this section the non functional requirements can be found.

NonFunc_01: The system must never take more than 2 seconds to do a search of available rooms. The requirement starts when a user presses the search button, and is accepted if the available rooms is shown in the user interface within 2 seconds.

NonFunc_02: The whole check out procedure must take in average less than 60 seconds to complete.

6. Functional Requirements

In this section the functional requirements will be listed.

6.1 Main Requirements

These are requirements that must be fulfilled for project to be approved by client.

Req_01: The user must be able to search for rooms in a specific hotel, matching certain criteria (bed type, smoking policy, view, quality level, adjoined rooms) and room status for a certain day.

Req_02: The user must be able to make a reservation for one or more rooms to a single client over one or more nights.

Req_03: The system must be able to record guest information, such as name, address, telephone number, credit card, passport number and loyalty level.

Req_04: The user must be able to mark a reservation as checked in.

Req_05: The user must be able to mark a reservation as checked out.

Req_06: The system must handle special price to be given to a specific booking, which overrules the maximum price for a room.

Req_07: System must handle cancellation of a reservation, if the cancellation is done to late according to hotel policy a cancellation fee should be generated to the customer.

6.2 Extra Requirements

These are requirements that is not directly specified by client, but will increase usability of the application for the client.

ExReq_01: The system should keep track of how much the client has paid. i.e. paid in advance.

ExReq _02: The user should be able to add extra fees to a booking, i.e. extra bed, mini bar, etc.

7. Scenarios

In this section we detail and explain the scenarios meant to be handled by the application.

Scenario 1 . Booking a room

Anders the plumber is going on a holiday and needs accomodation. Anders calls the “Linneus Hotel” and asks if they have any rooms available for 4 days in June. Anders has asthma so he requires a room without smoking permission. The clerk search for available rooms with a single bed, without smoking permissions and for selected dates. The system replies with a list of available rooms that follows these criteria and tells Anders. Anders decides one of the rooms, and the clerk asks for his name, address, telephone number , passport number and credit card number. The clerk then creates an booking for the reservation, and tells Anders that the reservation has been made.

Scenario 2. Checking in

Olle and his wife have booked a room in advance and have finally arrived at the hotel. They intend to check in so that they may gain access to their room. They give the clerk their booking number and the clerk searches for their reservation. The clerk finds the reservation, and gives the guests the key to the room. The clerk also activate their booking by setting the booking status to “checked in” and it will stay active until they check out.

Scenario 3. Canceling reservation

Nisse who booked a room 3 weeks ago got some complications and have to cancel his visit at the hotel. He calls the hotel desk, and the clerk answers. Nisse gives the clerk his booking number and the clerk searches the system for the given booking number. The clerk finds the reservation and quickly gives a summary of the reservation to Nisse. Nisse confirms that this is the reservation he wants to cancel. If the cancellation is close to the booking date, the clerk will inform Nisse that the cancellation is considered a late cancellation, which means that Nisse will be billed for an amount of the reservation. The clerk then cancels the reservation, and the room is flagged as available for reserved dates, and the reservation is deleted and the booking is changed to a “canceled” status.

Scenario 4. Checking out

After staying for the duration of his booking a guest wishes to check out. He returns his key to the clerk and gives the clerk his booking number, name or telephone number. The clerk then finds the booking, and presses “check out”. The system then shows how much the guest is billed. After the client has paid, the system prints out a receipt, and the guest’s booking is marked as “checked out”. The room is now available for booking again and will be shown when searching for rooms.

8. UML profile diagram

In this section we import the Modeling and Analysis of Real Time and Embedded system (MARTE) to this project by profile diagram which is used to model non functional requirements. We will continue to use the MARTE profile in other documents such as the Design document. The diagram can be seen in figure 1.

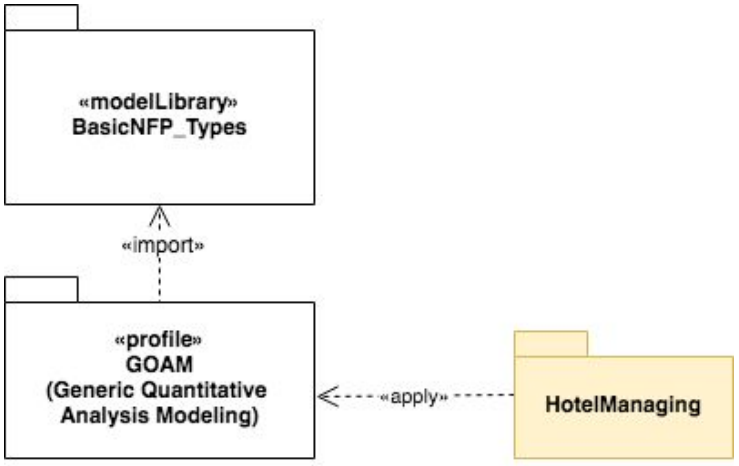

fig.1: A profile diagram showing the import of the MARTE profile to this project UML models.

9. Use cases

These are the current use cases of the system, which covers the main flows of this application. Since the clerk is the only one handling the system, the clerk will be the only actor of the system. Even though the guest is a part of the flow, the clerk is the only actor in the system. The connection between the use cases and also the non functional requirements can be seen in figure 2. One can see in the diagram that to find an available room the only way is to book room. Another use case that is very related to other use cases is the find booking. This can be seen as a stand alone use case but is also a prerequisite to complete the use cases check in, check out and cancel.

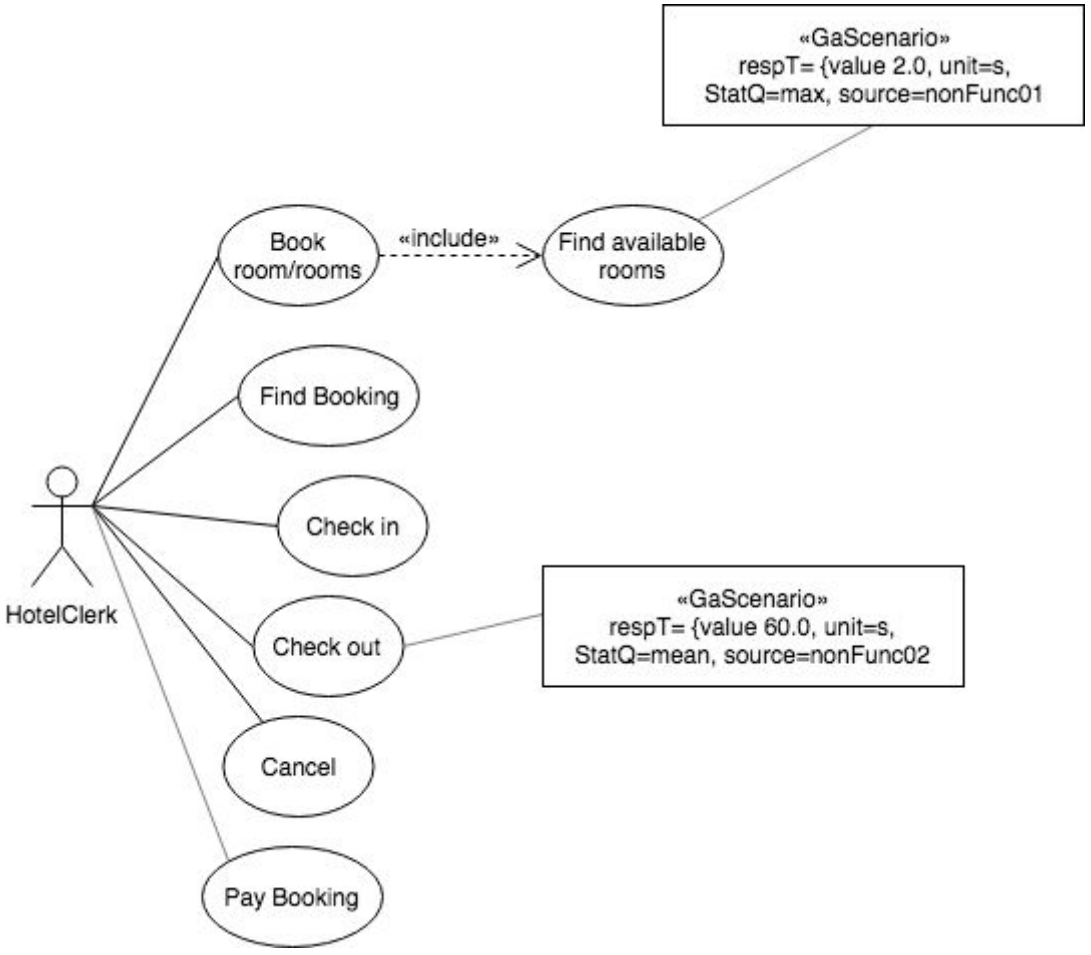

Fig 2. The uses cases of the Linnaeus Hotel management system.

Use case 1: Check in

Actors

- The user (hotel clerk)

Goals

- The guest wishes to check in and activate their booking.

Pre conditions

- The system is up and running
- The guest has an existing booking.

Summary

The user searches the system for an existing booking using a phone number, name or booking number. System presents the existing booking, and the user activates the booking.

Related use cases

- Use case 3: Find reservation.

Basic flow

1. Use case 3: Find reservation
2. User clicks the check-in button.
3. The system presents that the check in was successful.

Exception flow

3a. There was a connection problem and the server couldn't finalize the request
4a. System gives an error message.

Post conditions

- The booking status of the customer is changed to "checked in".

Use case 2: Check out

Actors

- The user (hotel clerk)

Goals

- The user wishes to end an activated booking.

Preconditions

- The system is up and running
- There booking to be checked out, is checked in.
- The customer has paid for the room.

Summary

The user searches the system for an existing booking using a phone number, name or booking number. System presents the existing booking, user click a button to check out the customer.

Related use cases

- Use case 3: Find reservation.

Basic flow

1. Use case 3: Find reservation.
2. User presses “check out” button.
3. The system presents that the check out was successful.

Alternative flow

3a The booking is not paid.
4b. The system shows an error message that the booking can’t be checked out since the booking isn’t paid.
5b. Use case 7: Pay booking
6b User clicks check out.
7b System shows that the checkout was successful.

Exception flow

4e. The server fails to mark the user as check out.
5e. System gives an error message.

Postconditions

- The guests booking has ended, and the room booked is then returned to “available” status.

Use case 3: Find reservation

Actors

- The user (hotel clerk)

Goals

- The user wants to find out if a certain type of room is available on a certain day.

Preconditions

- The system is up and running
- The booking to be found is registered in the system.

Summary

The user searches the system for one or more existing bookings using a phone number, name or booking number. System presents the existing bookings and displays them.

Related use cases

- None

Basic flow

1. The user searches for a reservation with parameters given by the guest.
2. The system presents one or more existing bookings with the parameters given.
3. The user clicks the booking he/she want to view.
4. The system displays info about the booking.

Alternative flow

1a. The user choses a date on main page.
2a. The user interface displays all bookings overlapping that date for one of the hotels.
3a. The user double click on the booking he/she want to view.
4a. The system presents info about that booking.

Exception flow

3e. The system cannot find a booking with the requested parameters.
4e. The user interface presents that no bookings fulfilled the search.

Postconditions

- The user found an existing booking.

Use case 4: Find available rooms

Actors

- The user (hotel clerk)

Goals

- The user wants to find out if a certain type of room is available on a certain day.

Preconditions

- The system is up and running

Summary

The user searches the system for a specific room type during a specific time. The system responds with the number of such rooms that are available.

Related use cases

- Use case 5: Book room/rooms

Basic flow

1. The user presses add new booking in main page
2. The user fills in what dates, type of bed, which hotel, if the room has adjacent room, that the customer wishes to book.
3. The user interface displays a list of rooms matching the criterias.

Alternative flow

3a. The system cannot find a room with the requested parameters.
4a. The user changes the search to another hotel.
5a. System found available rooms and displays a list of these.

Exceptional flow

3e. The system cannot find a room with the requested parameters.
4e. User interface displays that no rooms mathed the criteria.
5e. The customer will try another hotel.
6e. User clicks the return to menu.

Postconditions

- The user knows if rooms that fit the parameters are available for reservation.

Use case 5: Book room/rooms

Actors

- The user (hotel clerk)

Goals

- The user wants to find out if a certain type of room is available on a certain day.

Preconditions

- The system is up and running

Summary

The user searches the system for a specific room type during a specific time. The system responds with the number of such rooms that are available.

Related use cases

- Use case 4: Find available rooms.

Basic flow

1. User click create new booking
2. Use case 4: Find available rooms.
3. The user selects one or multiple rooms fitting the parameters that the guest wish to book and presses “Book room/rooms”.
4. The user then fills in a form with the guest information.
5. The system generates a price which is calculated by sum of the price for the rooms booked * the number of nights - discount for client.
6. The user interface shows the generated price as well as the booking id.
7. The user presses “Create booking” to finalize the booking.

Exceptional flow

4a. When user clicks a room and presses continue another user has booked the same room.
5a. The system presents that the chosen room is already reserved.

Postconditions

- The user creates a booking for a guest and it is saved into the system.

Use case 6: Cancel reservation

Actors

- The user (hotel clerk)

Goals

- The user wants to find a reservation and close it

Preconditions

- The system is up and running
- There is an existing booking

Summary

The user wishes to cancel a reservation. User searches for the reservation and the system responds with a reservation matching the provided parameters. The user then presses “cancel booking”

Related use cases

- Use case 3: Find reservation.

Basic flow

1. Use case 3: Find reservation.
2. The user selects the booking he/she wishes to cancel, and press “cancel booking”
3. The system set the status to the booking as cancelled.

Exception flow

3a. If the cancellation is done close to the booked date, the cancellation is referred to as a “late cancellation” and the guest is fined for a amount of the booking cost.

4a. The system register the fee to the booking before marking the booking as cancelled.

Postconditions

- The booking is canceled and the room is available again for new bookings.

Use case 7: Pay Booking

Actors

- The user (hotel clerk)

Goals

- The user wants to pay the fees for his staying at the hotel.

Preconditions

- The system is up and running
- There is an existing booking

Summary

The user wishes to pay his/her expenses for staying at the Linnaeus Hotel. User searches for the reservation and the system responds with a reservation matching the provided parameters. The user then presses “Pay booking”

Related use cases

- Use case 3: Find reservation.

Basic flow

1. Use case 3: Find reservation.
2. The user use a separate system for handling payment from client.
3. The user selects the booking he/she wishes to pay, and press “Pay Booking”
4. The system saves that the user has paid his/her expenses.

Exception flow

3e. The system is down and cannot register the payment.

4e. The user notes that this transaction is payed and does the basic flow once the system is running.

Postconditions

- The amount paid matches the price of the booking