The Iceman Cometh

SENG 330: Assignment 1
University of Victoria

Sept 26, 2017

Problem Statement

The arena complex has several rinks used for sports such as hockey, figure skating and other sports. A rink-booking system is required to facilitate the process of booking these rinks to customers and groups or clubs. The system requires initial enrollment before any first booking. Staff and customers can have access to the system with different levels of privileges in order to make operations such as booking, cancelling, payment and viewing or updating information.

Table of Contents

1. Domain Model and Glossary

This section describes all of the terminology the developing team and stakeholders use along with all its definitions. This is also intended to capture data abstraction for the problem domain in a manner that reflects appropriate OOP techniques. This will be achieved via iteratively defining and refining the core domain by asking questions and making assumptions in order to help distill the value of the system and provide focus around the essential core domain.

1.1 Glossary

Booking: A reservation for a Rink, as booked by the Rider.
Rink: A surface of ice used for hockey, ice skating or play other winter sports.
Arena Complex: Sport and Leisure facility that has several full-size rinks.
Community Group: Group of people or organization which works to advance a particular cause or interest.
Customer: Individual person or a group of people.
Pick-up Hockey: Informal type of hockey played on ice with no formal rules or specific positions.
Invoice: Document that shows the service provided, venue, description of booking such as time and date, total amount owed.
Maintenance: The process of keeping arena facility in good condition.
Staff: All employees employed by the arena.
Head Consultant: Lead person who directs the consultant team and is the main point of contact between the client and the consultant team.
Unavailable: Occupied or inaccessible due to certain circumstances**.**
Available: Accessible and can be obtained.
Real-Time Report: Data available immediately as feedback
Ice schedule: Calendar view that renders availability of the arena facilities
Event Type: Hockey, Figure skating or others.
Venue: Location of event within the Arena (Rink1, Rink2 …).
User: Employee or a customer representing a group/club.

1.2 Assumptions and Dependencies

- Customers have no accounts and need registration.
- Groups/clubs are at least 5+ people.
- Rinks are unavailable for short hours every month.
- Customers can book one or multiple rinks.
- The system is accessible online.
- Data are rendered in real-time.
- Rinks can be booked for a month or a couple of months (entire season).
- Payments are due at the beginning of the month or season.
- Online payment accepts credit cards/debit cards.

- Customers have access to features of the system as long as they don't hold any outstanding balances.

1.3 Questions to Client

- What makes the client need a new system?
- Why the client doesn't use a product off the shelf?
- Why not outsource?
- What is the client's goal with this product?
- What is the overall purpose of the product?
- Who is going to use the system? Staff? Customers?
- What is the release date for the system and what is the budget?

1.4 Detailed Description

A customer that is looking at booking a rink for their own group or club registers first for a new account. The customer is able to view the calendar for the availability of the rinks in day/weekly/monthly views. When a rink is booked, a reservation is issued which the person can cancel any time later. Also, an invoice is automatically generated and the customer is forwarded to the payment webpage. If rinks are under maintenance, the system renders them as unavailable and what time they will be accessible again. Staff and management can also create or cancel bookings, create invoices and reports, view and update customer information.

2 Use Case Modelling

This section presents different scenarios where a user such as a player, customer or staff interacts with the rink booking system. It includes 10 informal system use cases and a diagram to demonstrate the top three important use cases.

Name: Customer Books a venue (self-service)
Identifier: UC 1
Basic Course of Action:

- Customer clicks on an available spot in the Calendar.
- System forwards customer to a page with details of booking such as time, date, rate and venue type and location.
- Customer enters/create group type, event type and event name.
- Customer clicks on BOOK Now.
- System forwards customer to payment webpage.

Name: Customer cancels a venue (self-service)
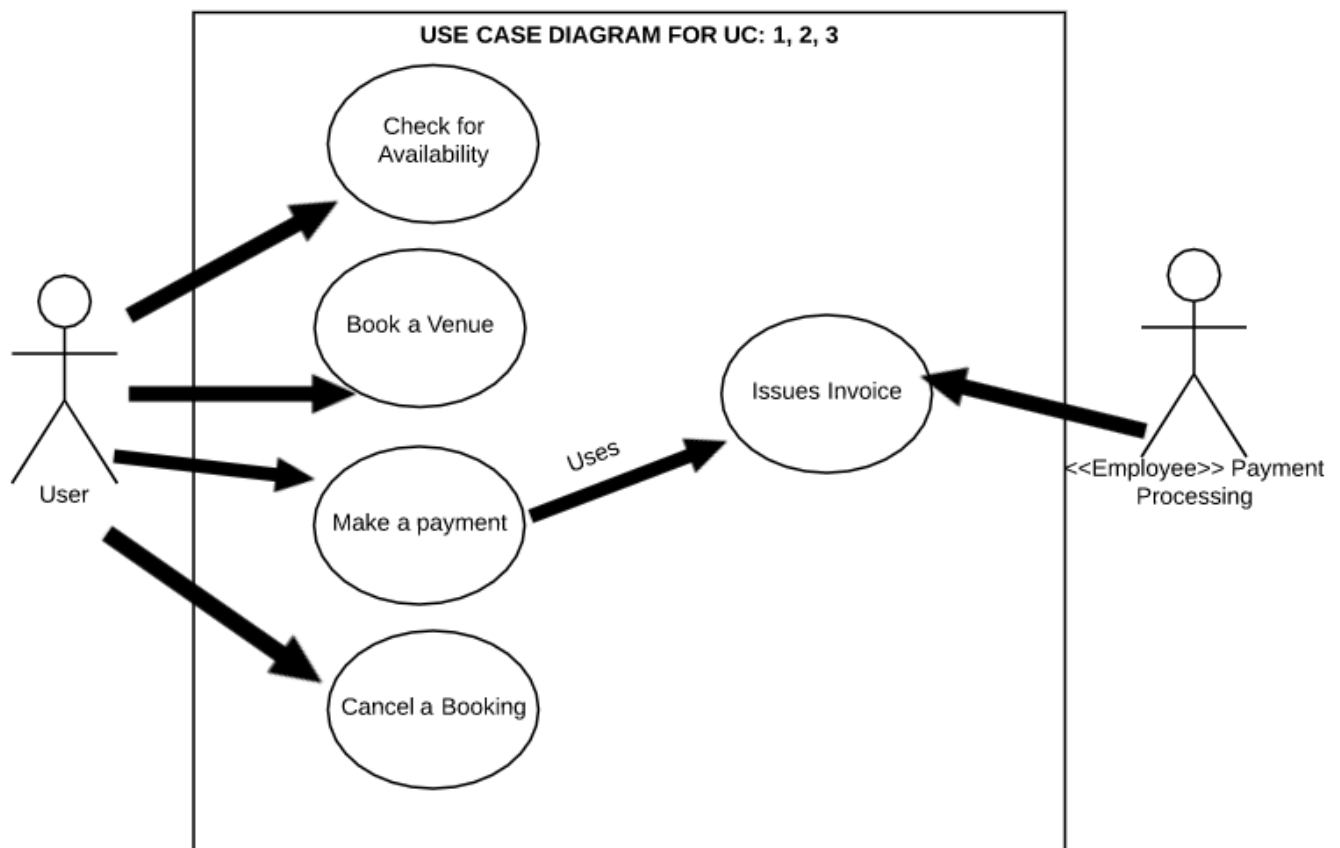
Identifier: UC 2
Basic Course of Action:

- Customer clicks the calendar to view their booking.
- The customer clicks on the X mark on top of their booking to cancel.
- The system displays a pop-up window with details of the booking and an option to cancel.
- The customer clicks on Cancel Booking.
- The system forwards the customer back to the calendar.

Name: Employee Creates Invoice
Identifier: UC 3
Basic Course of Action:

- Employee clicks on Calendar.
- Employee clicks on a booking from the calendar.
- Employee clicks on Create New Invoice.
- System generates an automatic invoice.
- Employee emails or prints invoice to customer.

USE CASE DIAGRAM FOR UC: 1, 2, 3

Check for Availability

Book a Venue

Issues Invoice

Uses

Make a payment

User

<<Employee>> Payment Processing

Cancel a Booking

Name: Employee tracks unpaid invoices

Identifier: UC 4
Basic Course of Action:

- Employee clicks on Booking from the main menu.
- Employee clicks all booking.
- The system displays all booking with many filters to choose from such as Group name, Venue, Event Type, Date & Time, Due Amount, Paid/Unpaid.
- Employee chooses Paid/Unpaid filter.
- The System displays all unpaid bookings.

Name: Employee Changes a Booking
Identifier: UC 5
Basic Course of Action:

- Employee clicks on Calendar and chooses a booking.
- Employee drags a booking to a different spot on the calendar.
- System displays a pop-up window with details of the change and an option to notify the customer.
- Employee clicks on Confirm Change & Notify.

Name: Employee Cancels a Booking
Identifier: UC 6
Basic Course of Action:

- Employee clicks on Calendar and chooses a booking.
- Employee clicks on the X on top of the booking to cancel.
- The system displays a pop-up window with details of the cancelation and an option to notify the customer.
- Employee clicks on Cancel & Notify.

Name: Customer Registers a New Account
Identifier: UC 7
Basic Course of Action:

- Customer visits the system's website.
- The system displays options to log in or register.
- Customer clicks on Register.
- Customer enters their name, email, phone number, group/club name
- Customer clicks on Submit.
- System forwards customer to log in webpage.

Name: Customer Searches Booking
Identifier: UC 8
Basic Course of Action:

- Customer clicks on Calendar.
- Customer browses by day/week/month to view bookings
- Customer clicks on the desired booking to view its details.

Name: Employee Views Facility Reporting
Identifier: UC 9
Basic Course of Action:

- Employee clicks on the dashboard from the main menu.
- The system displays graphs and live feed for user/admin bookings and invoices.

Name: Customer Edits Personal Information
Identifier: UC 10
Basic Course of Action:

- Customer logs in into the system.
- Customer clicks on settings on the main menu.
- Customer clicks on Edit Account.
- Customer update their information such as name, email, phone number, address, group/club name.
- Customer clicks Update.
- System forwards customer back to main page.

3 Requirements Gathering

This section lists three requirements in detailed use case approach in order to illustrate the series of interaction between the user and the system. The system is specified to a high level of completion using these use cases assuming that they capture everything.

Name: Customer Books a venue (self-service)
Identifier: UC 1
Description:
Book a venue for an existing customer.

Preconditions:
The Customer is registered in the system.

Post-conditions:
A venue or multiple venues will be booked for the Customer if they don't have any holds on their account and there's an available venue.

Basic Course of Action:
1. The use case begins when a customer wants to book a venue.
2. The customer logs in to the online system using their username and password.

3. The system verifies the customer is eligible to book venues and performs other tasks. [Alt A: The Customer Is Not Eligible for Booking]
4. The system displays the main page where the customer can click on the Calendar to view bookings and availability of venues.
5. The customer chooses the available spot for their booking. [Alt B: The Customer Decides Not to book]
6. The system displays a webpage with details of the type of the venue, location, time, date.
7. The customer enters their name, group/club name, event type and event name and the amount of time of the booking.
8. The system validates the venue's chosen period of time fits into the existing schedule of the calendar and there are no overlaps.
9. The system calculates the fees for the booking based on the fees published in the policy.
10. The customer clicks on Confirm Booking.
11. The system informs the customer that booking was successful by displaying a message and then generates an automatic invoice.
12. The system forwards the customer to the payment page to *Pay Now or Pay Later*.
13. The use case ends when the booking is successful and the customer is forwarded to the payment page.

Alternate A: The customer is not eligible to book venues due to holds.
A.3. The system determines the customer is not eligible to book a venue.
A.4. The system informs the customer of any holds such as outstanding balances.
A.5. The use case ends here.

Alternate B: The Customer Decides Not to book a venue.
B.5. The customer views the list of bookings in the calendar and does not see a suitable time for their booking.
B.6. The use case ends.

Name: Customer cancels a venue (self-service)
Identifier: UC 2
Description:
Cancels a venue for an existing customer.

Preconditions:
The Customer is registered in the system.

Post-conditions:
A venue or multiple venues will be canceled for the Customer if they don't have any holds on their account and are following the cancelation policy of the facility (14 days notice). Otherwise, customers are charged full amount and no refunds are issued.

Basic Course of Action:
1. The use case begins when a customer wants to cancel a venue.
2. The customer logs in to the online system using their username and password.

3. The system verifies the customer is eligible to make operations such as booking and cancelation. [Alt A: The Customer Is Not Eligible for Cancelation]
4. The system displays the main page where the customer can click on the Calendar to view their booking.
5. The customer chooses their booking and click on the X on top of the booking to cancel. [Alt B: The Customer Decides Not to cancel]
6. The system displays a pop-up window with details of the booking.
7. The system validates that the 14-days notice cancelation policy is applied here [Alt C: Customer is Not Eligible to Cancel].
8. The customer views the details of the booking and clicks on Cancel.
9. The system removes the booking from the calendar and forwards the customer back to the main page.
10. The system informs the customer that cancelation was successful by displaying a message.
11.The use case ends when the cancelation process is successful and the customer is forwarded back to the main page.

Alternate A: The customer is not eligible to cancel a booking due to holds.
A.3. The system determines the customer is not eligible to cancel a venue.
A.4. The system informs the customer of any holds such as outstanding balances.
A.5. The use case ends here.

Alternate B: The Customer Decides Not to cancel a booking.
B.5. The customer views the list of bookings in the calendar and changes their mind about cancelling.
B.6. The use case ends.

Alternate C: The Customer Is Not Eligible to Cancel
C.7. The system determines the customer is not eligible to cancel a venue because they are past the 14-days notice.
C.8. The system informs the customer of the short cancellation notice.
C.9. The use case ends.

Name: Employee Creates Invoice
Identifier: UC 3
Description:
Creates an invoice for an existing customer.

Preconditions:
The Customer has an existing booking.

Post-conditions:
An invoice is issued and emailed or printed and handed to the customer to assist them with payment process.

Basic Course of Action:

1. The use case begins when a customer requests an invoice.
2. The employee logs in to the online system using their username and password.
3. The system verifies the employee credentials and forward them to the main page.
4. The employee clicks on the calendar and enters the customer's name in the search bar.
5. The system displays all booking under the customer's name.
6. The employee clicks on the desired booking and then clicks on Create New Invoice.
7. The system automatically generates an invoice for that booking.
8. The employee clicks on email invoice or print invoice.
9.The use case ends when the invoice is generated and the employee chooses to email or print the invoice.

4 High Level Module Design

This section is focused on building expressive models that reveal the intention of the domain using DDD/GoF patterns such as entity classes, value objects and services. Also, it will include a class diagram of the entity, boundary and control classes *for three use cases (Book Venue, Cancel Booking, Issue Invoice).*

4.1 List of Entity, Value, and Service classes.
Look at 1st Figure.
4.2 Class Diagram that includes entity, boundary and control classes
Look at 2nd Figure.

Identity & Access Managment Bounded Context

<<Entity>>
User

- username: String
- password: String

Common Types

<<ValueObject>>
UserID

- userID: String

<<ValueObject>>
ArenaID

- userID: String

<<ValueObject>>
CustomerID

- userID: String

Employee Managment Bounded Context

<<Entity>>
Employee

- name: String

+ BookVenue()
+ CancelBooking()
+ Issue Invoice()

<<ValueObject>>
Address

- name: String
- zipcode: String
- city: String
- country: String
- street: String

+ getName()
+ getZipcode()
+ getCity()
+ getCountry()
+ getStreet()

Arena Managment Bounded Context

<<Entity>>
Arena

- name: String

<<Entity>>
Venue

- name: String
- type: String

<<ValueObject>>
Address

- name: String
- zipcode: String
- city: String
- country: String
- street: String

+ getName()
+ getZipcode()
+ getCity()
+ getCountry()
+ getStreet()

Customer Managment Bounded Context

<<Entity>>
Customer

- name: String
- phonenum: Int
- email: String

+ BookVenue()
+ CancelBooking()
+ PayInvoice()

<<ValueObject>>
Address

- name: String
- zipcode: String
- city: String
- country: String
- street: String

+ getName()
+ getZipcode()
+ getCity()
+ getCountry()
+ getStreet()

Customer Managment Domain Layer

<<Domain Service>>
CustomerService

+ AccountExistForCustomer(CusomerID): Boolean

Uses

Booking Managment Bounded Context

<<Entity>>
Invoice

- amount: Float
- transactionID: Int
- userID: Int

+ confirmTransaction()

<<Entity>>
Booking

- name: String
- bookingID: Int
- date: Date
- time: Time
- venue: String

+ updateCalendar()

Booking Managment Domain Layer

<<Domain Service>>
BookingService

+ BookingExistForCustomer(CusomerID): Boolean

Uses

Book or Cancel

Issue Invoice