**Software Architecture Document**

Version 1.4

for

*Quickbook* Conference Room Reservation System

Prepared by

|  |  |  |
| --- | --- | --- |
|  |  |  |
| Piratheeban Annamalai | 27755708 | piratheeban95@hotmail.com |
| Andrew Laramee | 27050925 | zephix109@gmail.com |
| Jacqueline Luo | 26938949 | luo.jackie@outlook.com |
| Michael Mescheder | 27202202 | meschedermichael@gmail.com |
| Hoang Khang Nguyen | 27079427 | alphakennyn@gmail.com |
| Eric Payette | 27008058 | eric.payette24@gmail.com |

|  |  |
| --- | --- |
| Instructor: | Dr. Constantinos Constantinides |
| Course: | SOEN 344 |
| Date: | 04/04/2017 |

**Document history**

|  |  |  |  |
| --- | --- | --- | --- |
| **Date** | **Version** | **Description** | **Author** |
| 10/12/2016 | 1.0 | - Addition of UML Class Diagram with respective associations during team meeting | - All team members |
| - Design of communication diagram for critical use cases | - Ideawin, Philip, Hannah |
| 10/25/2016 | 1.1 | - Modification of UML class Diagram | - All team members |
| - Addition of Mappers, TDGs, unit of work diagram |
| -Ideawin, Philip, Hannah |
| 11/21/2016 | 1.2 | - Modification of visibilities for subsystems | -Hannah, Ideawin |
| - Addition of methods in Class Diagram for respective restrictions | -Armine, Ideawin, Hannah, Aline |
| 29/03/2017 | 1.3 | -Addition of new requirements including updated diagrams | Eric, Michael, Jacqueline |
| 03/04/2017 | 1.4 | -Modified descriptions in SAD to include new entity | Kenny |
| 04/04/2017 | 1.5 | Updated Class diagram, domain model, sequence diagram and communication diagram | Eric |

**Table of contents**

[1. Introduction](#_30j0zll) 4

[Purpose](#_1fob9te) 4

[Scope](#_3znysh7) 4

[Definitions, acronyms, and abbreviations](#_2et92p0) 4

[2. Architectural representation](#_tyjcwt) 5

[2.1 Scenarios (Use Case View)](#_3dy6vkm) 5

[2.2 Logical View](#_26in1rg) 7

[2.2.1 Layers](#_4d34og8) 8

[2.2.2 Subsystems](#_2s8eyo1) 9

[2.2.3 Use Case Realizations 14](#_3rdcrjn)

[2.3 Data View 16](#_lnxbz9)

[3. Architectural requirements: goals and constraints](#_35nkun2) 16

[3.1 Functional requirements (Use case view)](#_1ksv4uv) 18

[3.2 Non-functional requirements](#_44sinio) 18

**List of figures**

Figure 1: UML Use Case Diagram 6

Figure 2: UML Class Diagram 7

Figure 3: Server-side architecture: logical view: layered architectural style 8

Figure 4: Relationship between Core package and Mappers 9

Figure 5: Relationship between Mappers package and Unit of Work 10

Figure 6: Relationship between Mappers and Identity Maps 11

Figure 7: Relationship between Mappers, TDGs and Tables 12

Figure 8: Communication Diagram 13

Figure 9: Sequence Diagram to make a new reservation 13

Figure 10: Sequence Diagram when user is done and mapper is ready to commit to UoW 14

Figure 11: Sequence Diagram to make a new TimeSlot 15

Figure 12: Sequence Diagram when user is ready to commit to UoW 15

Figure 13: Complete Sequence Diagram for Make Reservation 16

Figure 14: Entity-Relationship (ER) Model 17

# Introduction

This document contains a high-level design overview and provides an overall architecture of QuickBook, a web application tool that helps ENCS faculty members reserve a conference room. With high-level descriptions of the goals, the SAD explains the underlying architecture behind some of the use cases such as when a user tries to create, modify and cancel a reservation. This document provides the goals of the architecture, a view of the use cases supported by the system and architectural styles and components that have been selected to best achieve the use cases.

## 1.1 Purpose

This document provides a comprehensive architectural overview of the QuickBook. To describe different aspects of the system, we have presented three different architectural views such as: Logic view, Data Model view and Use-Case view. The purpose of this documentation is to record and express our architectural decisions which have been made on the system.

## 1.2 Scope

SAD describes the architecturally significant design aspects of QuickBook. This document can be used to achieve a good understanding of the fundamentals of the system as well as a good guiding tool for duplicating or building the system. Any stakeholder who wants to have a good technical knowledge of QuickBook are encouraged to read this document in order to be able to follow up with the source code.

## 1.3 Definitions, acronyms, and abbreviations

**UML**: Unified Modeling Language

**SAD**: Software Architecture Document

**TDG:** Table Data Gateway

**UoW:** Unit of Work

**ER:** Entity Relationship

**2. Architectural representation**

*QuickBook* is implemented as a web application; it is developed using multi-layered architecture which is a client-server architecture. The multilayered architecture is composed of a presentation layer, application (logic) layer, data source (storage) layer.

A description for the QuickBook system will be provided through the representation of different views. The 4+1 view model is composed of Logical View, Process View, Development View,

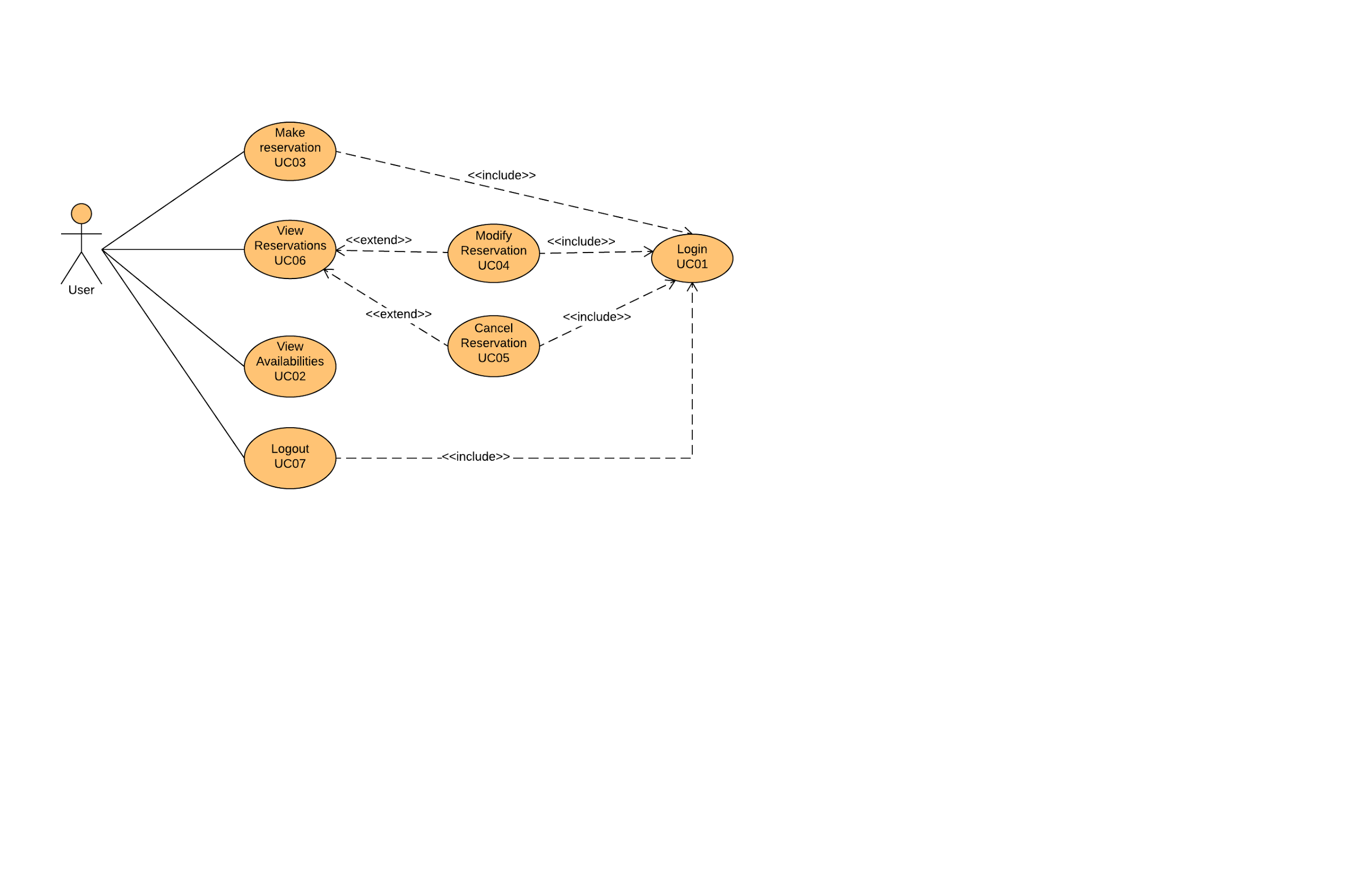
Physical View and Use Case View. This document will only be describing two of the five architectural views: Use Case View and Logical View. An extra view, Data View, will also be described.

In the **Logical View**, a class diagram shows the relationship between classes with their specific associations and dependencies. Also, the logical view includes the communication diagram, which illustrates the interaction of objects, for a given system operation, in a network format.

In the **Use Case View**, a use case model illustrates the functionality the system must provide; its behavior. Furthermore, the use case model displays  the relationship between the system's intended functions and the actors (the user).

In the **Data View**, an Entity Relationship (ER) diagram is used to show a visual representation of  the logical relationship between the data entities (or objects) of the system in order to build a database.

**2.1 Scenarios (Use Case View)**



**Figure 1: UML Use Case Diagram**

Use case 3 **(UC03)** is a *critical use case* of the system because the system depends on its functionality. If users are not able to make a reservation, then the system is entirely non- functional and therefore it doesn’t meet its requirements.

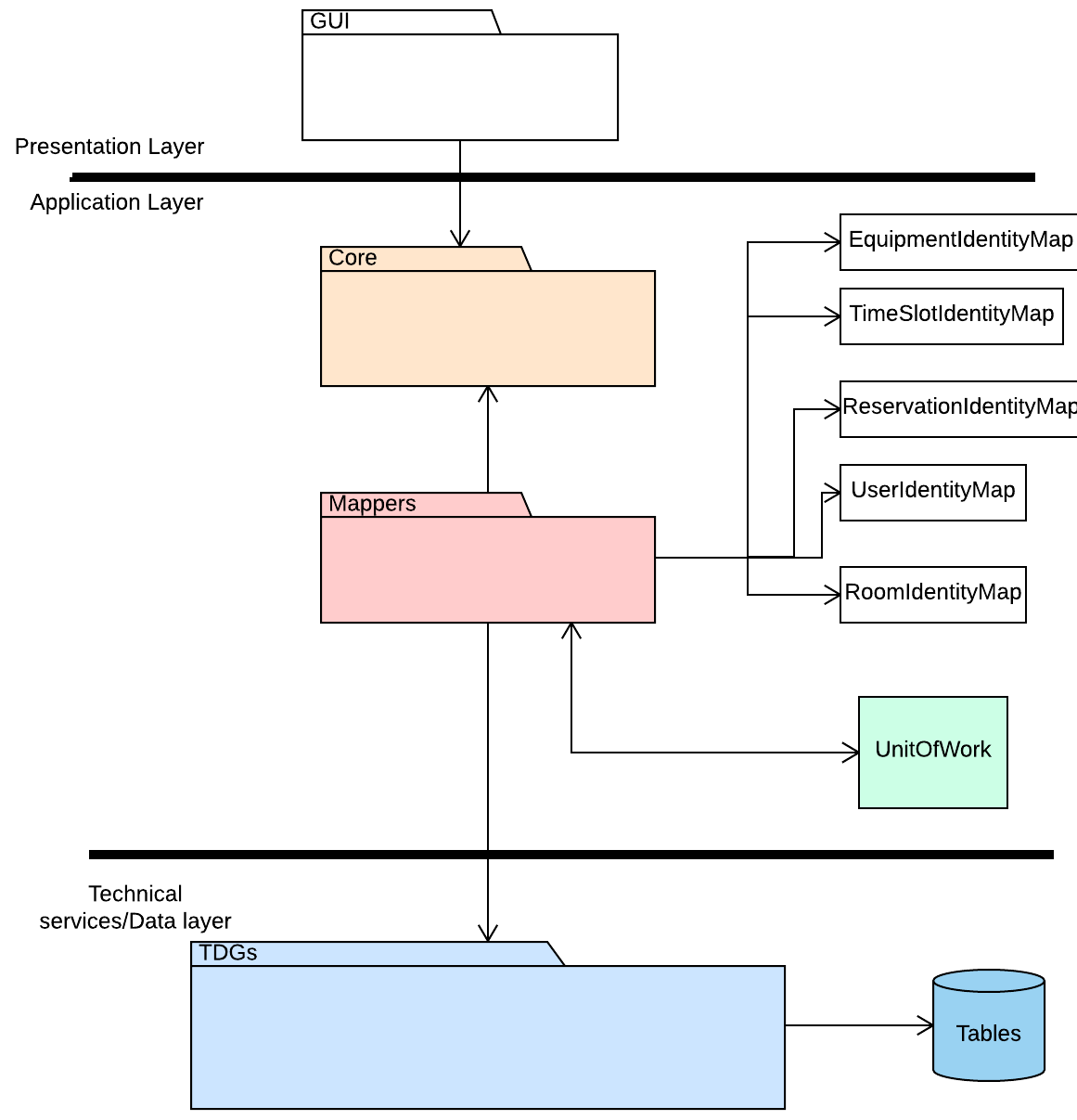
The communication diagram (fig. 8) describes the flow of the critical use case operation *makeNewReservation*. The interaction between domain objects is presented, showing how to successfully make a reservation.

## 2.2 Logical View Copy of DONE 11%2F20%2F16 Class Diagram - Page 1.png

**Figure 1: UML Class Diagram**

**2.2.1 Layers**

The system’s design uses a layered architectural style. There are three layered views of the system.

Figure 3: Server-side architecture: logical view: layered architectural style

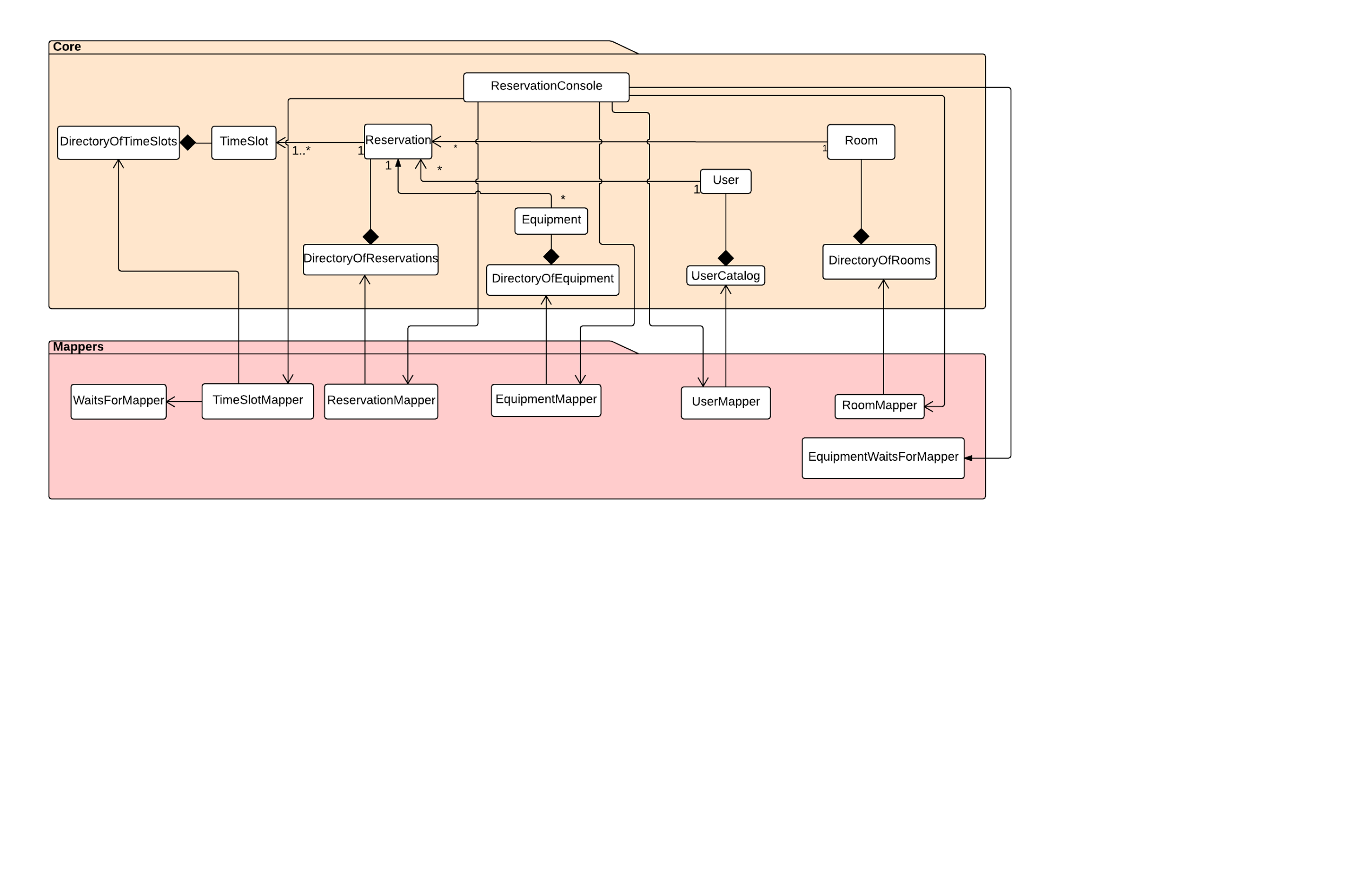
**2.2.2 Subsystems**Decomposition of the system in subsystems and their relation.  


Figure 4: Relationship between Core and Mappers Packages

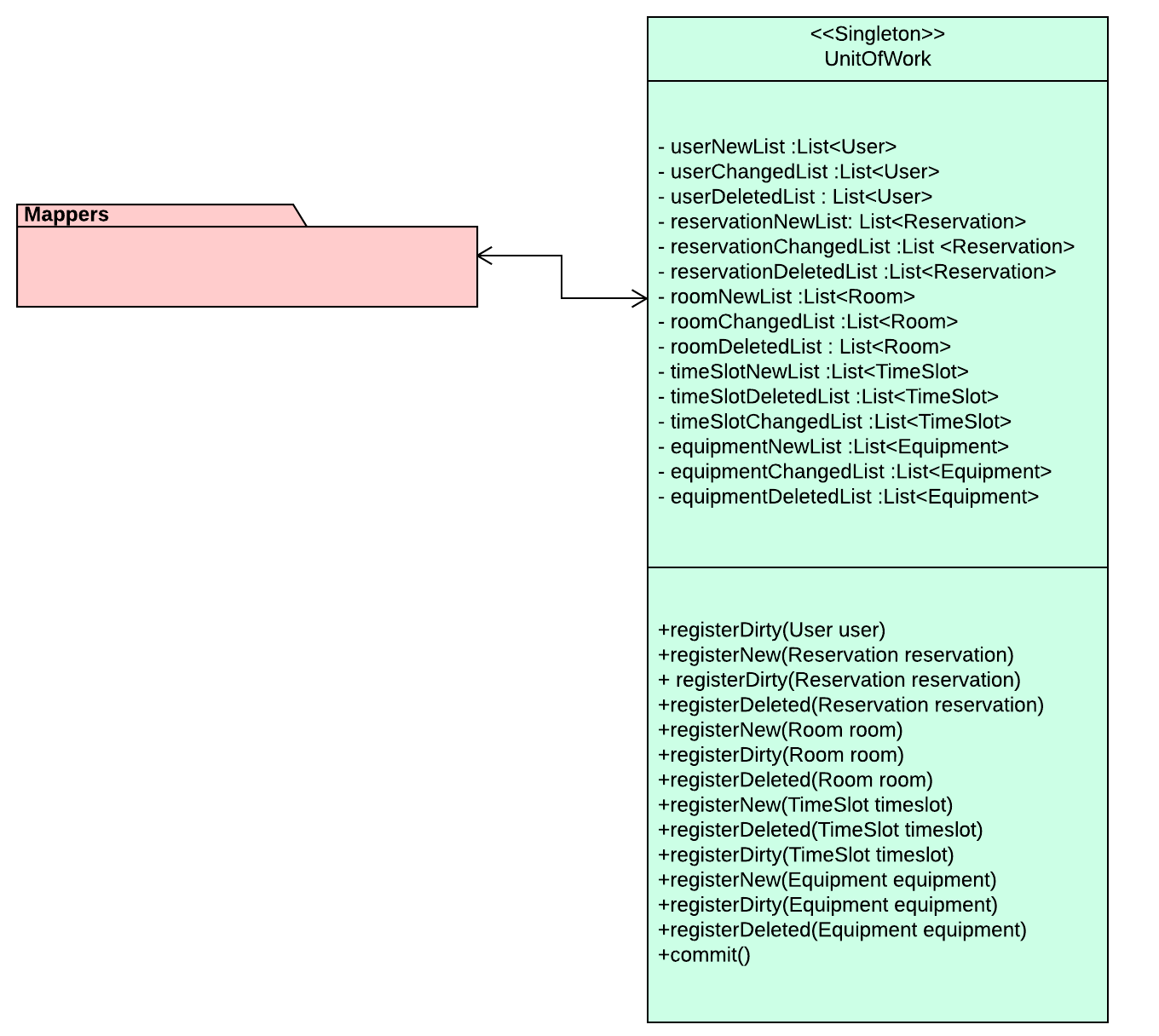


Figure 5: Relationship between Mappers Package and Unit of Work

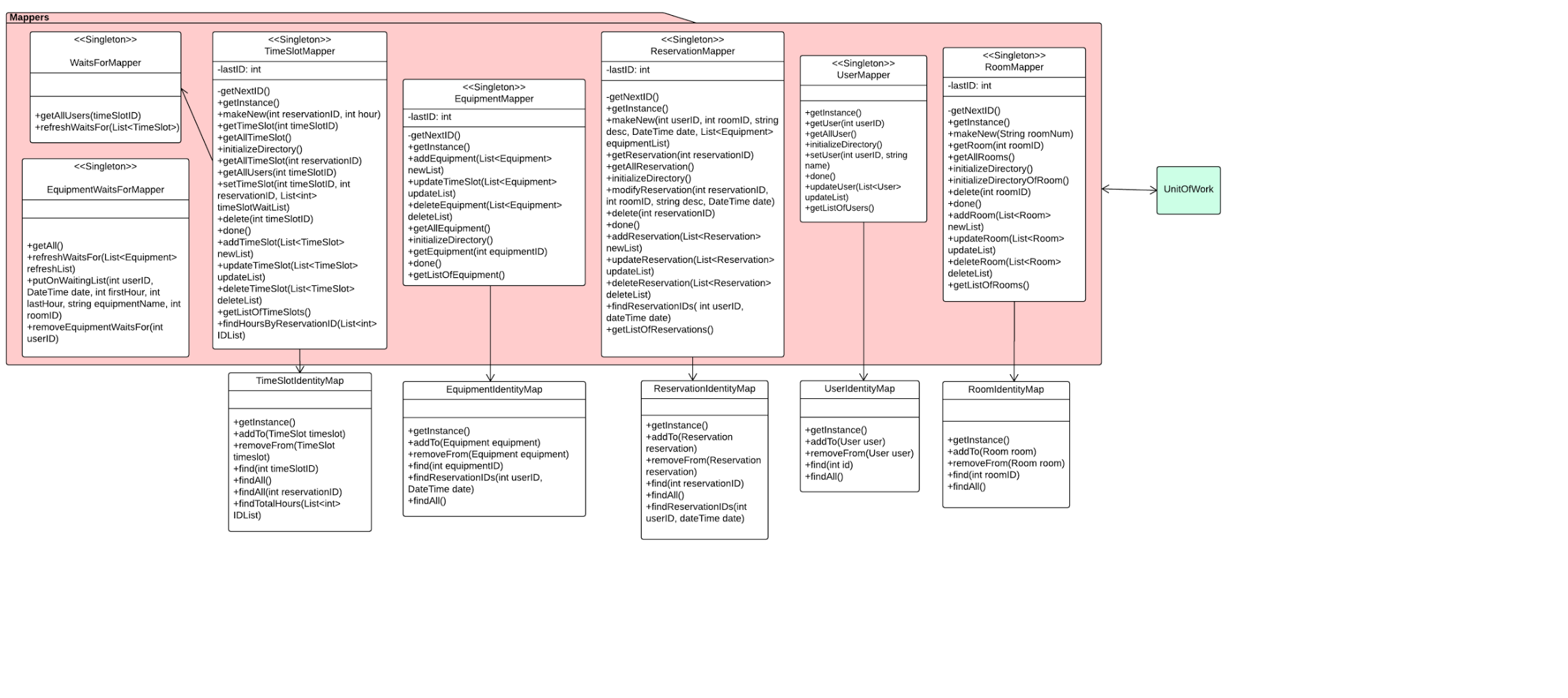


Figure 6: Relationship between Mappers and Identity Maps

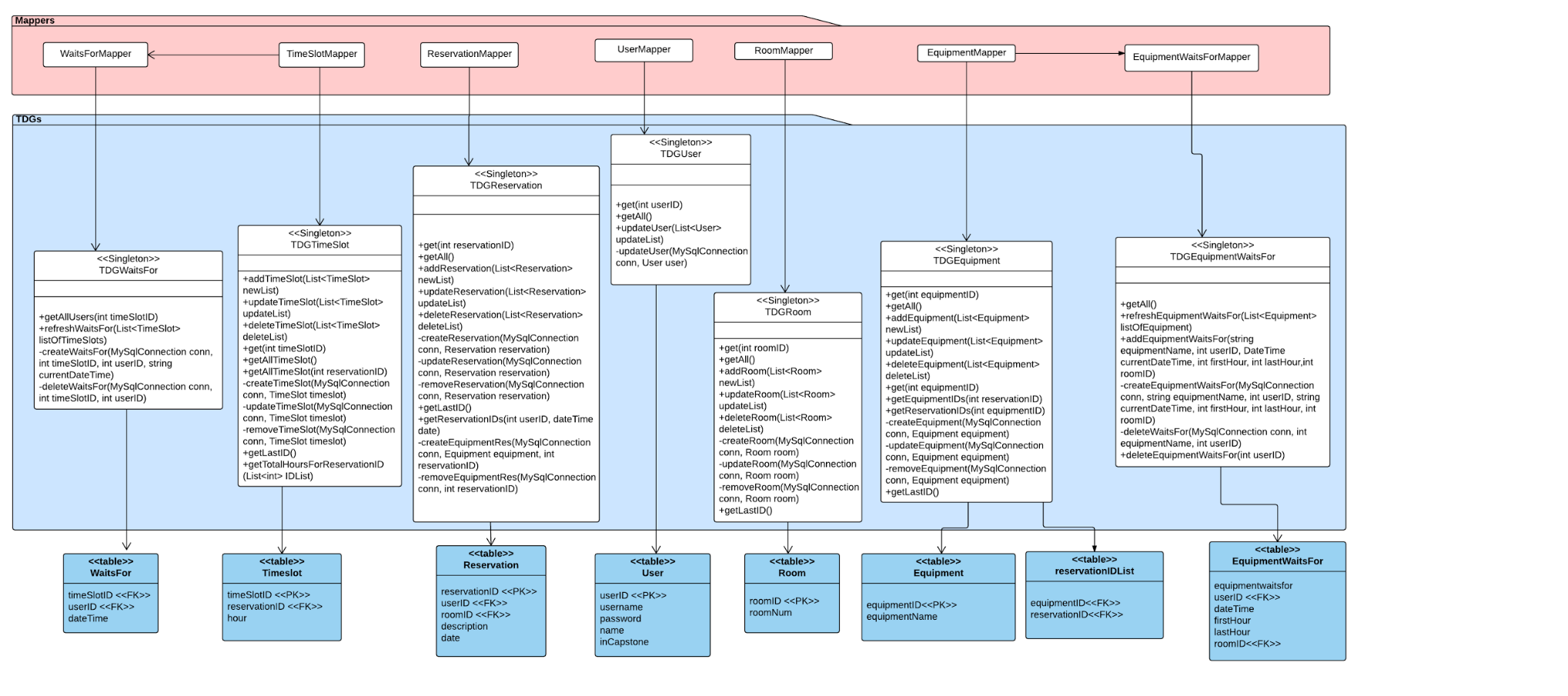


Figure 7: Relationship between Mappers, TDGs and Tables

**2.2.3 Use Case Realizations**

To clearly describe the important architectural elements of *QuickBook,* interaction diagrams are provided for the critical use case Make Reservation: **UC03.**

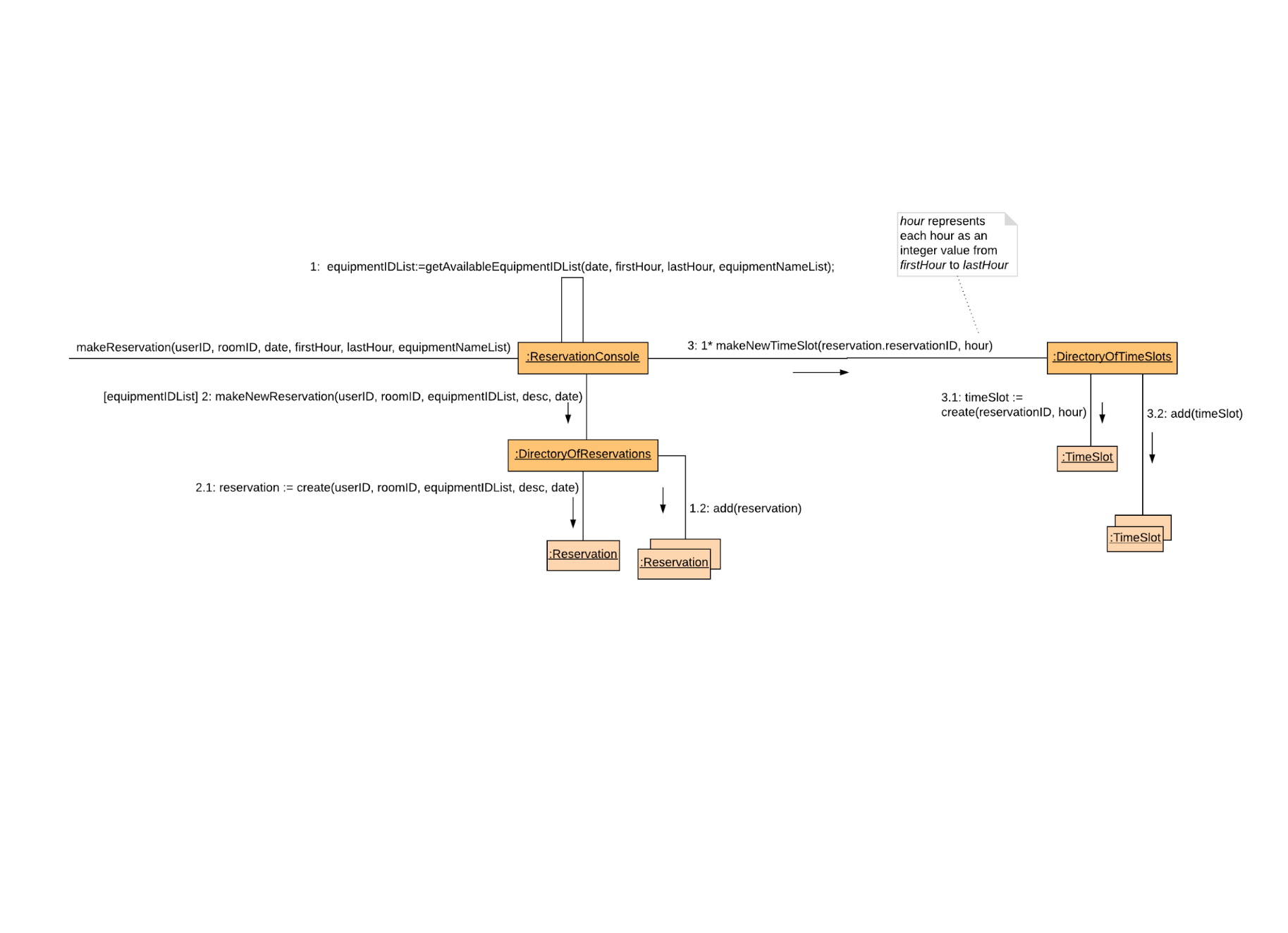
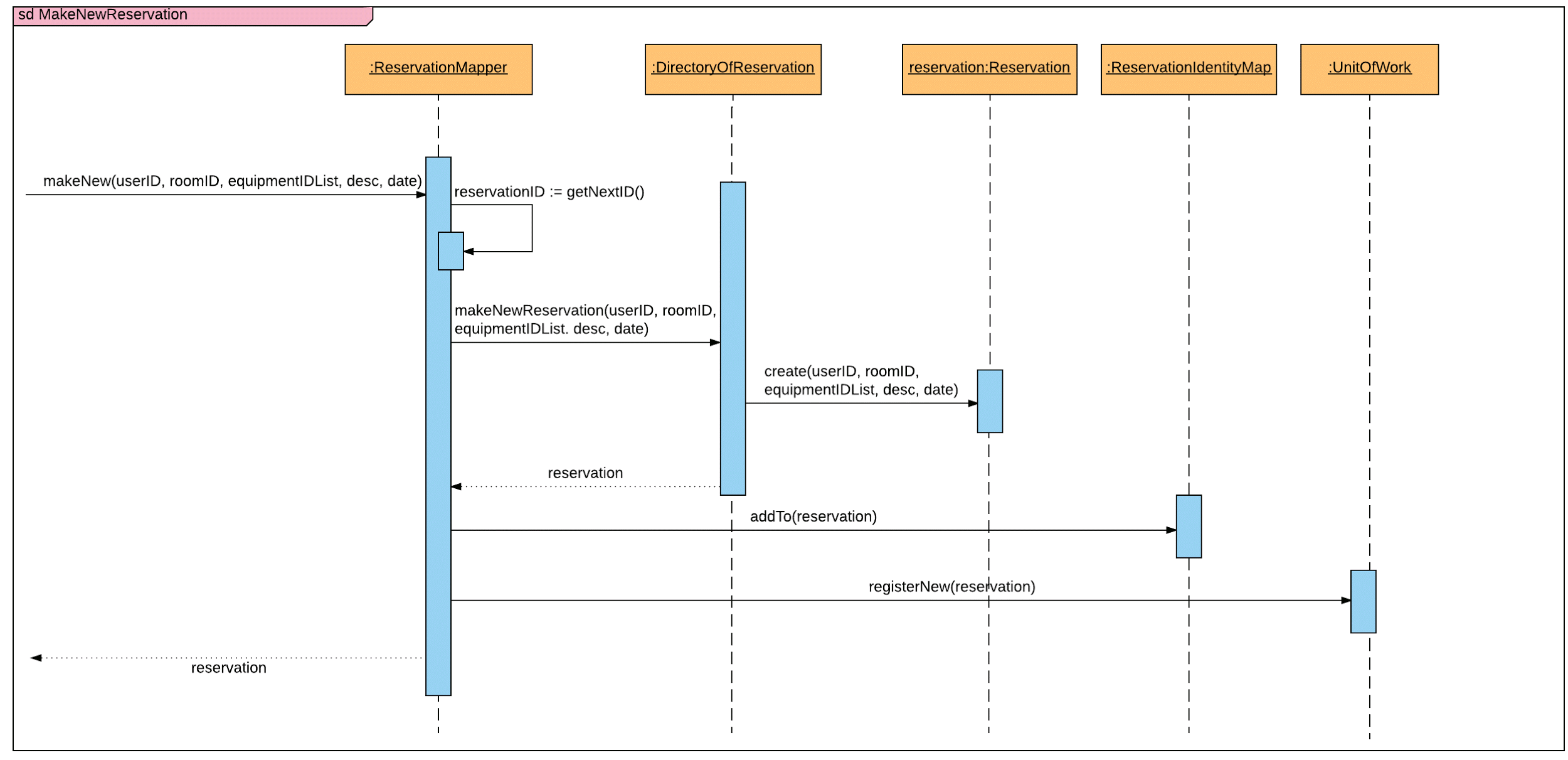


Figure 8: Communication Diagram

The following Sequence diagrams illustrate the UMl interactions between the Domain Object, Mappers, Identity Maps, Unit of Work (UoW), Table Data Getaways (TDG) and the database table.

  
Figure 9: Sequence Diagram to Make a New Reservation

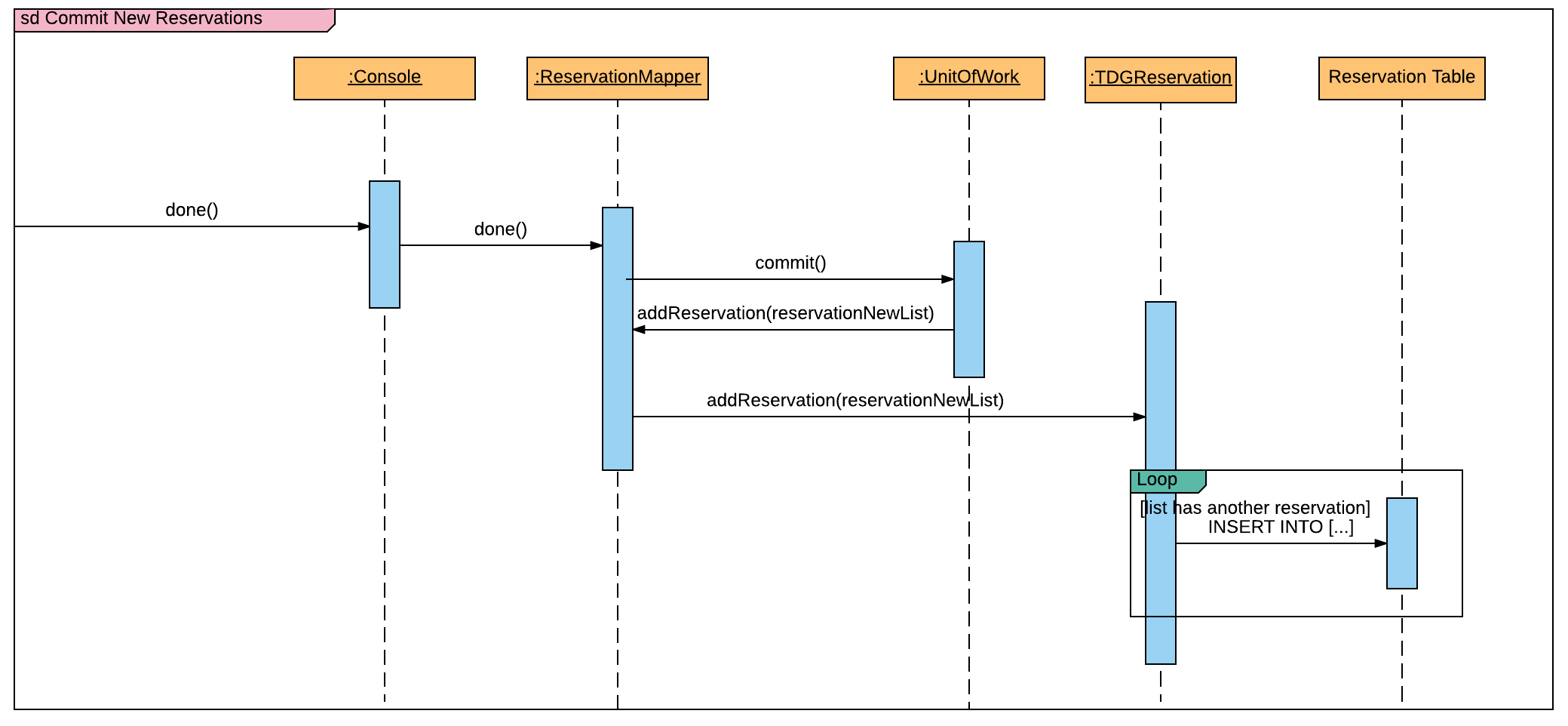


Figure 10: Sequence Diagram When a User is Done and Mapper are Ready to Commit to Unit of Work

## 

## Figure 11: Sequence Diagram to make a new TimeSlot

## 

## Figure 12: Sequence Diagram When User is Ready to Commit to Unit of Work

## 

## Copy of SD_MakeReservation - Page 1.png

## Figure 13: Complete Sequence Diagram for Make Reservation

## 2.3 Data View

Five main entities are present in the system: The User, the Reservation, the Room, the Equipment and the TimeSlot. The below Entity-Relationship (ER) model shows the attributes of each entity and the relationships between them.

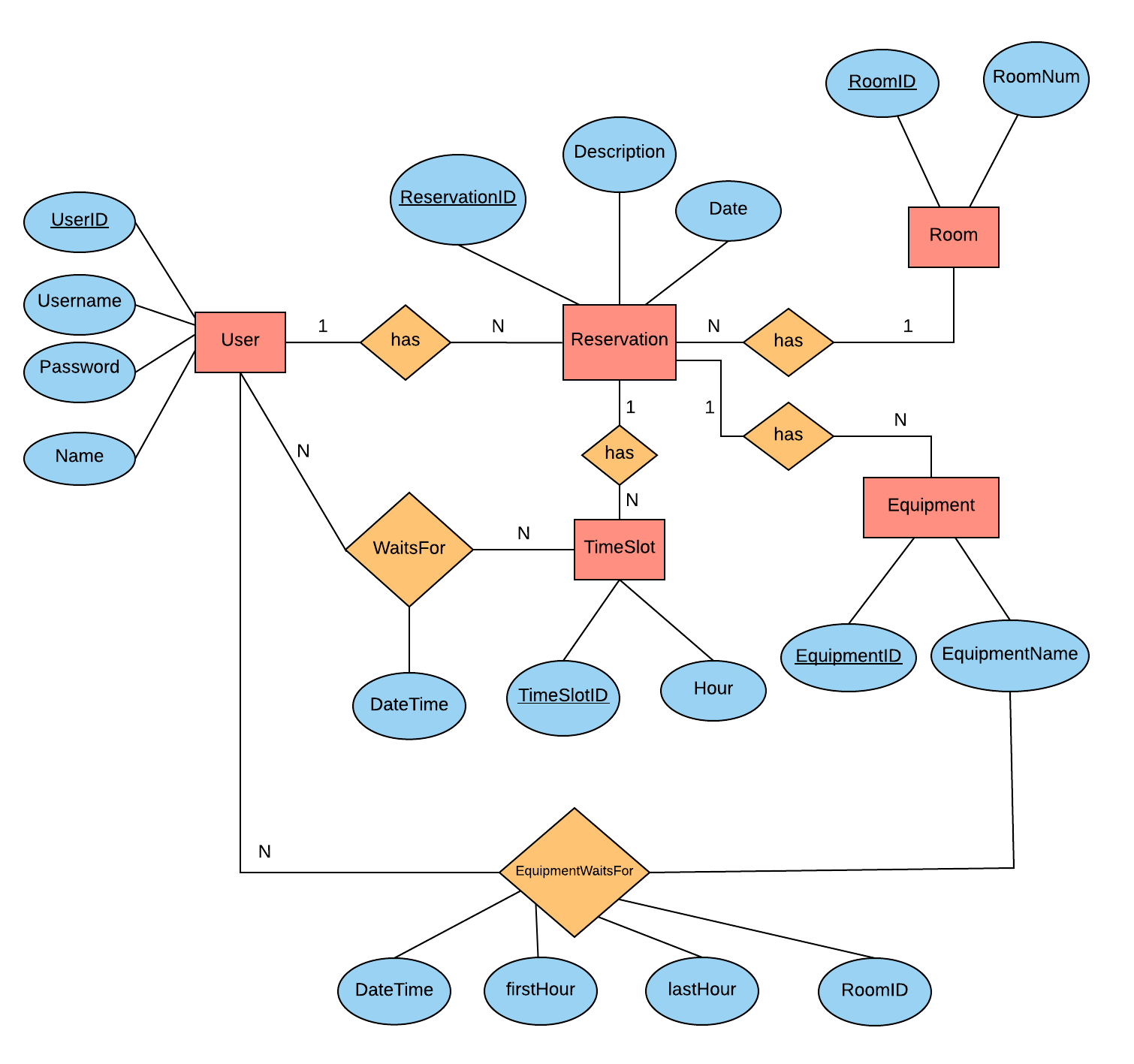


Figure 14: Entity-Relationship Model

**3. Architectural requirements: goals and constraints**

Functional Requirements and Non-Functional requirements can affect the architecture of a software system.

**3.1 Functional requirements (Use case view)**

This section describes which Use Cases from the Use Case Model are relevant to the software architecture.

|  |  |  |  |
| --- | --- | --- | --- |
| **Source** | **Name** | **Architectural relevance** | **Addressed in:** |
| *SRS* | *Use Case Login* | *-A user is needed for most methods* | *Section 2.1* |
| *SRS* | *Use Case Make Reservation* | *-Making a reservation must be functional in the system for most methods to function.* | *Section 2.1* |

**3.2 Non-functional requirements (NFRs)**

This section describes the non-functional requirements that are relevant to the architecture of the *Quickbook* Software. The two most important type of technical NFRs are Usability and Maintainability.

|  |  |  |  |
| --- | --- | --- | --- |
| **Source** | **Name** | **Architectural relevance** | **Addressed in:** |
| *SRS* | *Usability* | *-Consistency of both the user interface and the functionality of the system.*  *-Natural Mapping : ease of navigation* | *Section 3.3.2* |
| *SRS* | *Maintainability* | *-When adding or changing of the functionality and meeting new requirements, the system endures these changes with a degree of ease.* | *Section 3.3.4* |