Iteración 2 Manejo Transaccional de Información

Sebastián García 201630047, Nicolas Sotelo 201623026

Grupo D-05

Sistemas Transaccionales

Ingenieria de Sistemas y Computación

{js.garcial1, n.sotelo[}@uniandes.edu.co](mailto:%7d@uniandes.edu.co)

Abril 24 de 2018

Tabla de contenido

[1 Introducción 1](#_Toc512092829)

[2 Diseño y Construcción de la Aplicación 1](#_Toc512092830)

[2.1 Modificaciones al modelo conceptual 1](#_Toc512092831)

[3 Conclusiones 2](#_Toc512092832)

# Introducción

El objetivo de la siguiente iteración es desarrollar habilidades en el proceso de diseño de una aplicación transaccional, a partir de la descripción de un caso de negocio. En este proyecto, el caso de negocio gira en torno al alojamiento de la comunidad universitaria: Alohandes. Alohandes es un facilitador de opciones de alojamiento para la comunidad universitaria.

La principal meta de la siguiente iteración es integrar requerimientos funcionales y no funcionales relacionados con los aspectos ACID de una aplicación transaccional desarrollada en una arquitectura de tres niveles con manejo de persistencia en base de datos.

# Diseño y Construcción de la Aplicación

## Modificaciones al modelo conceptual

Para realizar los requerimientos funcionales de la presente iteración 2, se agregaron atributos a las relaciones Reservas y Propuestas para facilitar la consulta de datos y poder llevar a cabo requerimientos de modificación más complejos. El la relación Reserva se agregó el atributo ID\_COLECTIVO que representa el identificador de una reserva realizada de forma colectiva. Si una reserva se realiza de forma conjunta con otras, las anteriores tendrán en común el atributo que los identifica como una reserva colectiva, sin perder su identificador como reserva individual. En el caso de que una reserva se realice de forma individual, el valor de su atributo ID\_COLECTIVO será NULL.

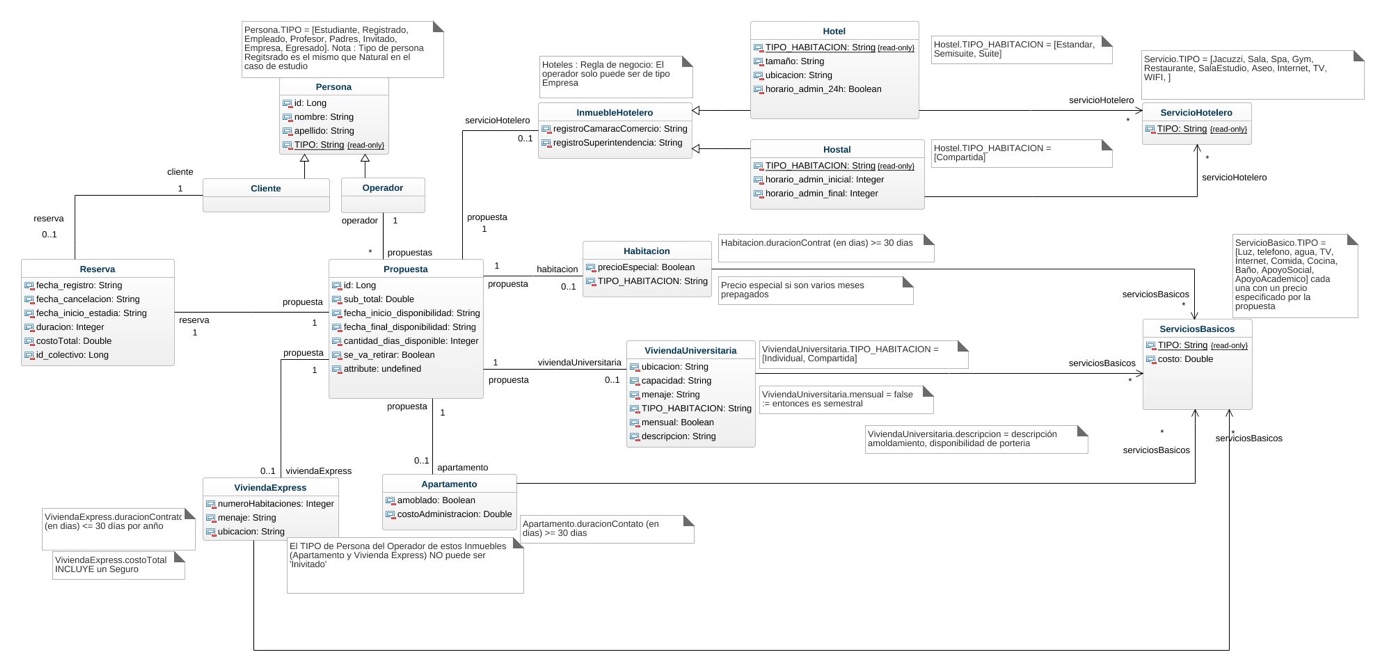


Figura 1. Modelo de clases UML actualizado luego de los resultados obtenidos en la iteración 1.

En el caso de la relación Propuestas (entendida como oferta) se crearon varios atributos que contienen la información de la fecha desde la cual una propuesta comienza a estar disponible, la fecha en la cual la propuesta deja de estar disponible, la cantidad de días de disponibilidad de una propuesta, el estado actual de la propuesta (disponible o no) y el estado a futuro (si se va a retirar la propuesta, teniendo reservas vigentes).

Cada vez que se registra una reserva, el atributo de FECHA\_FINAL\_DISPONIBILIDAD cambia de valor. Similarmente, cuando se da por culminada una reserva, el valor del atributo FECHA\_INICIO\_DISPONIBILIDAD se actualiza. Por medio de las funciones de SQL Developer: to\_char(), to\_number() y to\_date(), se obtiene la cantidad de días disponibles luego de realizar la diferencia en días entre el atributo FECHA\_FINAL\_DISPONIBILIDAD y el atributo FECHA\_FINAL\_DISPONIBILIDAD. Más adelante esta información será útil al momento se hacer el análisis sobre las propuestas.

Como se mencionó anteriormente, se agregaron los atributos SE\_VA\_RETIRAR y DISPONIBLE que se actualizan cada vez que se registra una reserva o se deshabilita un inmueble.

Para la siguiente iteración no se crearon nuevas relaciones.

// TABLAS EXCEL Y CASOS DE USO

## Lógica de los requerimientos funcionales

* RF 7 Registrar Reserva Colectiva

Para este requerimiento se codificó un método que consume un archivo JSON en donde se especifica: una lista de usuarios que hacen parte de la reserva colectiva (en donde cada usuario especifica el id de su reserva individual y la cantidad de personas que ocuparan el inmueble que se reserva a nombre del usuario respectivo), el identificador de la reserva colectiva, la fecha de inicial de ocupación del inmueble, el tipo de inmueble deseado, la duración de la reserva, la privacidad (un inmueble compartido o sencillo) y una lista de servicios deseados.

En primer lugar, se buscan todas las propuestas (ofertas) que cumplan con las condiciones especificadas (tipo de inmueble y servicios deseados). Si el sistema no cuenta con las suficientes propuestas, se le informa por medio de un mensaje al usuario la cantidad de propuestas disponibles y se realizan las reservas respectivas para esas propuestas. Como ejemplo del caso anterior, si el usuario quiere reservar 10 apartamentos sencillos pero el sistema solo cuenta con 6 apartamentos, luego se le informa al usuario que se realizarán solamente 6 de las 10 reservas que inicialmente solicitó y finalmente el sistema continua con el proceso normal de registrar una reserva.

En caso de que se realicen las reservas de forma exitosa se finaliza el requerimiento con un COMMIT. De lo contrario se aborta la anterior operación con una ROLLBACK en la base de datos. De la misma forma, los siguientes requerimientos realizan sus respectivas verificaciones para decidir su deben finalizar guardando o abortando las modificaciones a la base de datos.

* RF 8 Cancelar Reserva Colectiva

Este es el proceso inverso al requerimiento anterior. En este caso se hace uso del nuevo atributo creado en la relación RESERVAS para obtener el identificador de la reserva colectiva que se piensa cancelar. Luego, haciendo uso del requerimiento funcional de la iteración 1 que cancela reservas, se procede a eliminar las respectivas reservas que se hicieron de manera conjunta y se procede a penalizar a los respectivos usuarios de cada reserva con una multa en el caso de ser necesario.

* RF 9 Deshabilitar una propuesta

El requerimiento funcional 9 recibe como parámetro el identificador de la propuesta que se piensa deshabilitar. Por medio del atributo ID\_PROPUESTA presente en la relación RESERVAS, se obtienen las reservas que se verán afectadas por la cancelación de la propuesta. Luego se procede a buscar propuestas que estén disponibles y que sean del mismo tipo de inmueble que la propuesta original para comenzar a reubicar las reservas afectas. Primero se reubican las reservas vigentes, aquellas reservas o reserva que hacían uso del inmueble en el momento en el que se decidió deshabilitarlo. Por medio de un comparador se organizan las reservas por prioridad para ser reubicadas antes que las demás. La prioridad es establecida por la fecha en la que se registró en el sistema la reserva. Luego, se procede a realizar el anterior proceso con las reservas colectivas afectadas y finalmente con las reservas generales.

* RF 10 Rehabilitar una propuesta

Por medio del atributo DISPONIBLE en la relación PROPUESTAS se pueden obtener las propuestas que se encuentran disponibles o no. En el anterior requerimiento, el valor de este atributo es modificado para avisarle al sistema que no se encuentra disponible. De forma análoga, al rehabilitar una propuesta, se modifica el valor de este atributo nuevamente para poder recibir reservas en el futuro.

# Conclusiones

Con estos ejemplos se espera que haya un ejemplo y guía para la elaboración de los documentos solicitados como reporte técnico en cursos y proyectos.