# **Call center Application**

## **Descripción General**

El proyecto callcenter, es un proyecto que propone una solución al manejo y asignación de llamadas concurrentes. El proyecto implementa el framework SpringBoot con el gestor de dependencias maven, generado mediante la utilidad Spring Initializr (<https://start.spring.io/>).

## **Proceso**

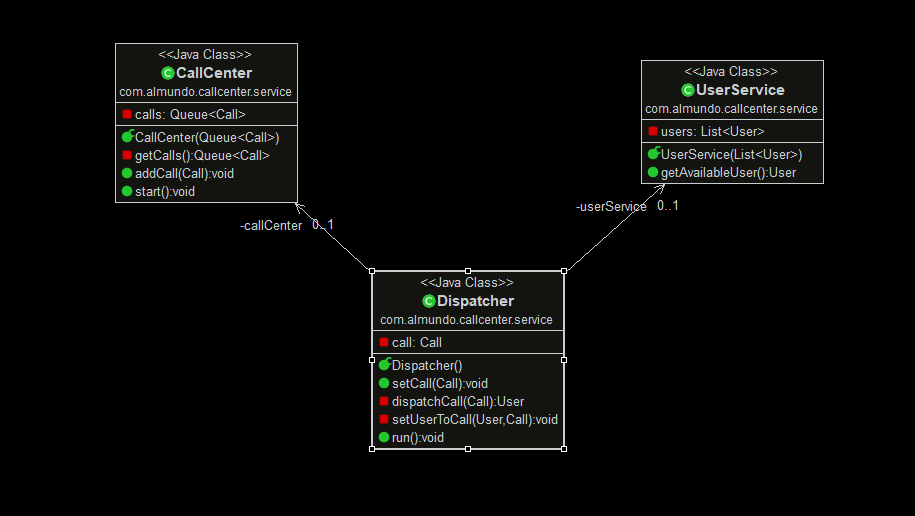
El proceso inicia con una cola de llamadas, las cuales una a una se van desencolando para ser manejadas por el Dispatcher el cual mediante el método dispatchCall manjeado por hilos, busca y asigna un usuario disponible.

Para simular la duración de las llamadas, el hilo que maneja el Dispatcher se pone a dormir el tiempo configurado en la propiedad duration de la clase Call

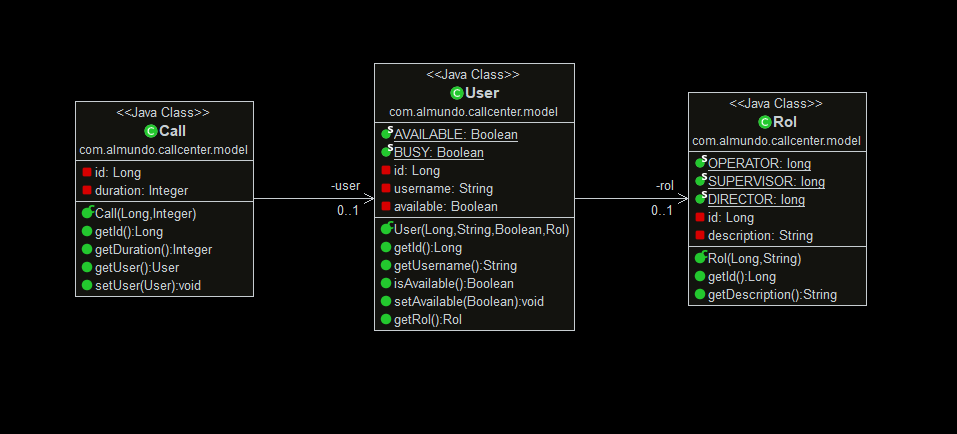
Las llamadas son administradas por la clase CallCenter, los usuarios por la clase UserService y el despacho por la clase Dispatcher.

Tanto las clases que guardan la estructura del modelo de datos son Call, User y Rol, que describen el modelo de las llamadas, usuarios y roles respectivamente.

## **Diagrama de clases**



## **Modelo de Datos**



## **Javadoc**

El Javadoc generado se encuentra en el folder doc del proyecto.

## **Repositorio**

El proyecto está versionado en el siguiente repositorio public de Github:

<https://github.com/georoem/callcenter>

## **Puntos Extra**

* **¿Qué pasa con una llamada cuando no hay ningún empleado libre?:**

En este caso la llamada se vuelve a encolar en la cola de llamadas para ser procesada después.

* **¿Qué pasa con una llamada cuando entran más de 10 llamadas concurrentes?:**

En este caso se chequea el pool de hilos, si hay uno disponible se despacha la llamada, sino se vuelve a encolar para reproceso.

## **Consideraciones**

* Se asume que el presente trabajo tiene fines evaluativos, por tanto, no se utilizó ningún API para el manejo de llamadas.
* El desarrollo no implementa ningún API de persistencia de datos, aunque contiene los POJOs que representan el modelo.
* La cola que se utilizó en el proyecto es de la clase Queue de Java, pero se hubiera podido utilizar un motor de colas como ActiveMQ para entornos productivos.
* Algunas pruebas unitarias no contienen aserciones por la falta de tiempo para averiguar cómo crear aserciones basadas en hilos.
* Queda faltando el diagrama de secuencia por falta de tiempo.
* Agradezco la oportunidad de ser evaluado y ofrezco de ante mano excusas por otras faltas que pudiera tener.