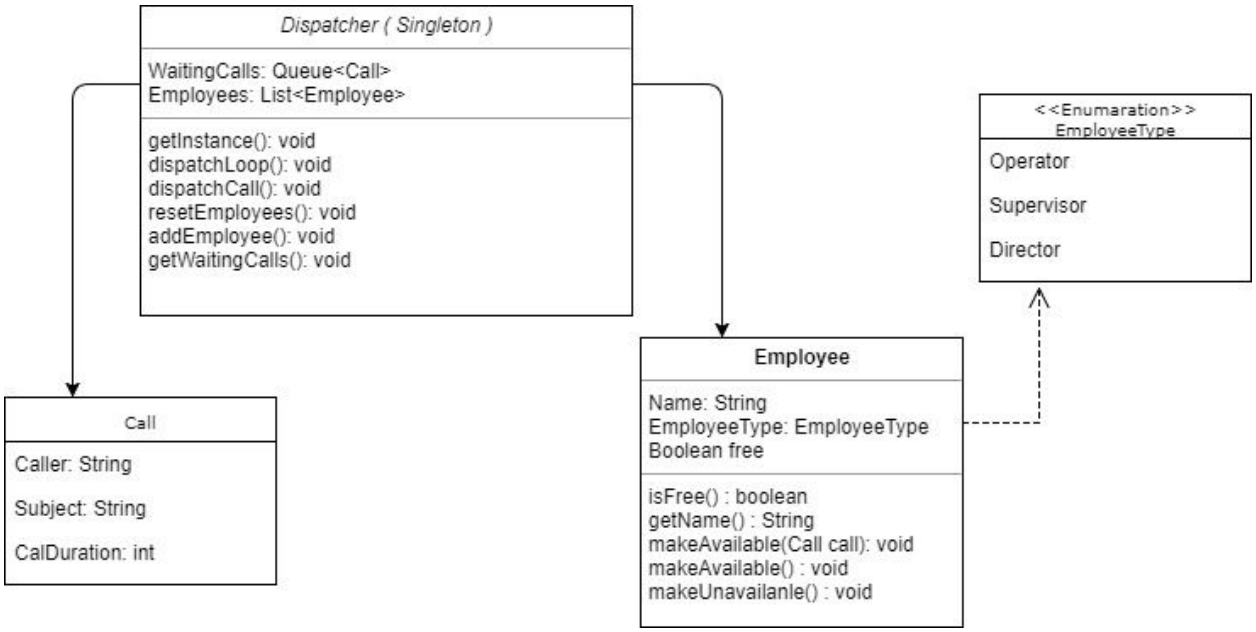
Diagrama UML



Otras Aclaraciones

- La implementación planteada usa un thread pool de 10 para permitir a la función dispatchCall atender las 10 llamadas simultáneamente

- Luego de ejecutar los hilos se usa la función awaitTermination del executorService encargado de los hilos, esto es para esperar la finalización de todos los hilos y dejar el sistema en un estado consistente.
  Esto permite en los test verificar condiciones como el estado de la cola de llamadas pendientes o la lista de empleados, imagino que existe una mejor forma de testear la ejecución de los threads, pero dado las limitaciones de tiempo y conocimiento sobre el manejo de threads fue la opcion que elegi, me parecio una mejor alternativa a un threadSleep sobre cada test.

- Inicialmente arme la clase Employee con tres clases hijas, una para cada tipo de empleado (Supervisor, Director y Operador) pero al ver que en esta instancia no tenian logica que amerite esa separación opte por usar una diferenciación usando el Enum EmployeeType

- La lógica de dispatchCall usa una cola de llamadas en espera, y de no haber empleados disponibles para atender la llamada se mantienen en la cola, esto resolvería la situación problemática de no tener empleados disponibles.
  Lo mismo se puede aplicar a la situación de recibir más de 10 llamadas recurrentes, la idea sería encolar las mismas y no mandarlas a procesar inmediatamente. Al usar una Blockingqueue no deberia haber problema respecto de la recurrencia de la entrada.

- El manejo de empleados libres/en call fue con un atributo de estado. Como la función dispatch call funciona con 10 threads en simultáneo y cada uno necesita pedir un empleado libre, cada thread está haciendo una consulta por el estado de los empleados y luego una actualización cuando el mismo pasa a estar ocupado.
  Es por esto que se usa un semáforo en la función availableEmployee.

- Deje algunos System.out.println en el codigo para poder mirar la ejecución(todos especifican el id del thread que ejecuta ese output) en la consola se puede ver que empleado fue asignado a que caller y se puede ver un string cuando la llamada está en espera, la cantidad de puntos suspensivos luego de la frase Calling in progress representa la cantidad de segundos que esa llamada duro.