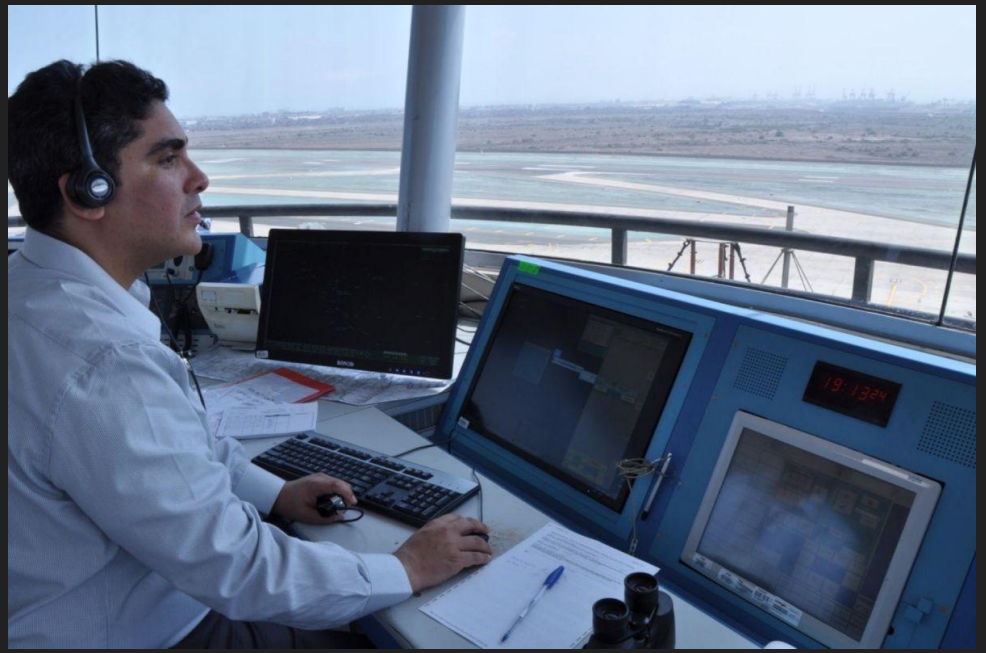
Controlador Aéreo

Sebastián Donoso
Augusto Villanueva
Ignacio Gaete

Descripción del problema

- Pistas.
- Aviones.
- Tiempo de uso.
- Combustible del avión.
- Situaciones inesperadas.



Solución propuesta: Clase AirTrafficControlSimulation

Propósito:

- Punto de entrada de la aplicación
- Crea la interfaz (ATCFrame) y la torre de control (ATC)
- Crea indefinidamente aviones (Plane)
- Período de creación: entre 1 y 3 segundos

Métodos:

- Solo contiene método main

Solución propuesta: Clase AirTrafficControlSimulation

```
// Generate airplanes dynamically
new Thread(() -> {
  Random random = new Random();
  int airplaneCounter = 1;

  while (true) {
    boolean isLanding = random.nextBoolean();
    int fuelLevel = random.nextInt(bound:100) + 1;
    String airplaneId = "A" + airplaneCounter++;

    Airplane airplane = new Airplane(airplaneId, fuelLevel, isLanding, atc);
    airplane.start();

    try {
      Thread.sleep(random.nextInt(bound:2000) + 1000); // New airplane every 1-3 seconds
    } catch (InterruptedException e) {
      Thread.currentThread().interrupt();
    }
  }
}).start();
```

Solución propuesta: Clase ATCFrame

Propósito:

- Crear interfaz gráfica (extiende JFrame)
- Organiza información en 3 paneles:
  1) Registro de aviones
  2) Cola de espera
  3) Estado de pista

Propiedades:

```
private final DefaultTableModel airplaneTableModel;
private final DefaultTableModel stripTableModel;
private final DefaultListModel<String> waitingQueueModel;
```

Solución propuesta: Clase ATCFrame

Métodos:

```
public ATCFrame() {···

public void addAirplaneToRegister(Airplane airplane, String status) {···

public void updateAirplaneStatus(String airplaneId, String status) {···

public void updateStripStatus(int stripId, String airplaneId, String status) {···

public void updateWaitingQueue(PriorityBlockingQueue<Airplane> queue) {···
```

Solución propuesta: Clase ATC

Propósito:

- Organizar aterrizajes y despegues de aviones
- Actualiza GUI con la última información

Propiedades:

```
private final PriorityBlockingQueue<Airplane> landingQueue = new PriorityBlockingQueue<>();
private final LandingStrip[] landingStrips;
private final Lock lock = new ReentrantLock();
private final ATCFrame frame;
```

Solución propuesta: Clase ATC

Métodos:

```
public ATC(int numberOfStrips, ATCFrame frame) {···

public void requestLanding(Airplane airplane) {···

public void requestTakeoff(Airplane airplane) {···

private void assignStrip() {···
```

Solución propuesta: Clase Airplane

Propósito:

- Representar a un avión individual.
- Cada avión se modela como un hilo para simular concurrencia.

Propiedades:

- airplaneId: identificador único para cada avión.
- fuelLevel: Indicador de nivel de combustible.
- isLanding: Indica si se está aterrizando o despegando
- atc: referencia al objeto ATC para solicitar pista.

```
private final String airplaneId;
private final int fuelLevel;
private final boolean isLanding;
private final ATC atc;
```

Solución propuesta: Clase Airplane

Funcionalidad del hilo:

```
public void run() {
    if (isLanding) {
        atc.requestLanding(this);
    } else {
        atc.requestTakeoff(this);
    }
}
```

Controlador Aéreo en funcionamiento!

Qué se podría incorporar a nuestra solución?

- Asignación manual de prioridad.
- Considerar el tiempo que se lleva en la cola para asignar prioridad.
- Disminución de combustible a medida que pase el tiempo en la cola.

Conclusiones finales

¿Qué se demostró con nuestro proyecto?