



Escuela Técnica Superior de  
**Ingeniería Informática**

# DP2

# REFACTORING

Grupo - G2-1
Miembros del grupo
DANIEL ARELLANO MARTÍNEZ
EDUARDO MIGUEL BOTÍA DOMINGO
JOSE MARTÍN SÁNCHEZ
JUAN NOGUEROL TIRADO
JOSÉ MANUEL SÁNCHEZ RUIZ
JAVIER VÁZQUEZ ZAMBRANO

# ÍNDICE

<b>Flight</b>	<b>3</b>
<b>Book</b>	<b>5</b>
<b>Airline</b>	<b>7</b>
<b>Client</b>	<b>8</b>
<b>Plane</b>	<b>9</b>
<b>Security Configuration</b>	<b>11</b>
<b>Airport</b>	<b>12</b>
<b>Runway</b>	<b>16</b>
<b>Todos los tests</b>	<b>18</b>

---

# Flight

Para esta entidad, nos hemos centrado en solventar malos olores y un bug, localizados específicamente en el controlador y un validador. Antes de pasar a enumerar cambios, mostramos una captura de pantalla de un análisis anterior y otro posterior a los cambios y refactorizaciones que hemos llevado a cabo.

<div><div>Type</div><div><div>Bug</div><div>Vulnerability</div><div>Code Smell</div><div>Security Hotspot</div></div><div>1</div><div>2</div><div>47</div><div>5</div></div> <div><div>Severity</div><div><div>Blocker</div><div>Critical</div><div>Major</div></div><div><div>0</div><div>12</div><div>4</div></div><div><div>Minor</div><div>Info</div></div><div><div>34</div><div>0</div></div></div> <div><div>Resolution</div><div>Status</div><div>Security Category</div><div>Creation Date</div><div>Language</div><div>Rule</div><div>Tag</div><div>Directory</div></div> <div><div>File</div><div>8 SELECTED</div><div>Clear</div></div> <div><div>flight</div><div>src/.../acmevolar/model/Flight.java</div><div>src/.../acmevolar/web/FlightControll... 24</div><div>src/.../projections/FlightListAttributes.java</div><div>src/.../repository/FlightRepository.java 10</div><div>src/.../service/FlightService.java 10</div><div>src/.../acmevolar/model/FlightStatusTyp...</div><div>src/.../acmevolar/web/FlightStatusType...</div><div>src/.../acmevolar/web/FlightValidator.... 4</div><div>src/.../springdatajpa/SpringDataFligh... 7</div></div>	<div><div>Type</div><div><div>Bug</div><div>Vulnerability</div><div>Code Smell</div><div>Security Hotspot</div></div><div>0</div><div>2</div><div>31</div><div>5</div></div> <div><div>Severity</div><div><div>Blocker</div><div>Critical</div><div>Major</div></div><div><div>0</div><div>3</div><div>1</div></div><div><div>Minor</div><div>Info</div></div><div><div>29</div><div>0</div></div></div> <div><div>Resolution</div><div>Status</div><div>Security Category</div><div>Creation Date</div><div>Language</div><div>Rule</div><div>Tag</div><div>Directory</div></div> <div><div>File</div><div>8 SELECTED</div><div>Clear</div></div> <div><div>Flight</div><div>src/.../acmevolar/model/Flight.java</div><div>src/.../acmevolar/web/FlightControll... 11</div><div>src/.../projections/FlightListAttributes.java</div><div>src/.../repository/FlightRepository.java 10</div><div>src/.../service/FlightService.java 10</div><div>src/.../acmevolar/model/FlightStatusTyp...</div><div>src/.../acmevolar/web/FlightStatusType...</div><div>src/.../acmevolar/web/FlightValidator.java</div><div>src/.../springdatajpa/SpringDataFligh... 7</div></div>
---	---

A continuación, analizamos el bug que hemos encontrado, que podría haber causado un error de tipo *NullPointerException*.

```
if (flight.getPublished() || flight.getAirline().getName().equals(airline.getName())) {  
    if (flight.getPublished() || airline != null &&  
flight.getAirline().getName().equals(airline.getName())) {
```

Por otra parte, hemos solucionado malos olores, de los que destacamos los que mostramos a continuación.

- Reutilización de código

En diversas ocasiones, reutilizábamos el nombre de las variables o cadenas de texto con el mismo contenido una y otra vez, y gracias al análisis de Sonar, nos hemos percatado de ello y hemos realizado un reajuste definiendo cadenas de texto *final*, como el que mostramos a continuación.

```
private static final String FLIGHTS_WORD      = "flights";
private static final String FLIGHT_LIST_URI   = "flights/flightList";
private static final String FLIGHT           = "flight";
private static final String REFERENCE         = "reference";
private static final String DEPARTES         = "departes";
private static final String LANDS            = "lands";
private static final String AIRPORTFULLOFPLANES = "airportFullOfPlanes";
private static final String AIRPORTFULLOFPLANES_MESSAGE = "This airport is full of planes this day";
```

A continuación, mostramos un ejemplo de un punto del código en el que hemos efectuado dicha refactorización.

```
model.put("flights", flights);
return "flights/flightList";
model.put(FLIGHTS_WORD, flights);
return FLIGHT_LIST_URI;
```

- No respetar convenciones de estilo de código

Por otra parte, ha habido una variable que no hemos nombrado siguiendo las convenciones y estándares de nombrado, en concreto el siguiente ejemplo:

```
List<Boolean> opciones_publicao = new ArrayList<>();
List<Boolean> opcionesPublicao = new ArrayList<>();
```

- Comentarios de código

Por otra parte, también hemos eliminado fragmentos de código que no estaban en uso, como el siguiente ejemplo.

```
// flight.getPlane().addFlight(flight);
```

# Book

Se han resuelto 3 tipos de Code Smells:

- Literales repetidos:  
Define a constant instead of duplicating this literal "flight".

```
model.put("flight", flight);

private static final String FLIGHT_ATTRIBUTE = "flight";
model.put(FLIGHT_ATTRIBUTE, flight);
```

- Comentarios de código:  
This block of commented-out lines of code should be removed.

```
// Client client = SecurityContextHolder.getContext().getAuthentication();
```

- Repositorios no usados:  
Remove this unused private field.

```
private BookRepository bookRepository;
private AirlineRepository airlineRepository;
private SpringDataPlaneRepository springPlaneRepository;

@Autowired
public BookService(final BookRepository bookRepository, final
SpringDataPlaneRepository springPlaneRepository, final AirlineRepository
airlineRepository) {
    this.bookRepository = bookRepository;
    this.springPlaneRepository = springPlaneRepository;
    this.airlineRepository = airlineRepository;
}
```

Type

Bug

Vulnerability

Code Smell

Security Hotspot

0

2

30

5

Severity

Blocker

Critical

Major

Minor

Info

0

3

9

20

0

Resolution

Status

Security Category

Creation Date

Language

Rule

Tag

Directory

File

7 SELECTED

Clear

book

src/.../acmevolar/model/Book.java

src/.../acmevolar/web/BookControll... 17

src/.../repository/BookRepository.java 7

src/.../service/BookService.java 7

src/.../acmevolar/model/BookStatusTyp...

src/.../acmevolar/web/BookStatusType...

src/.../springdatajpa/SpringDataBook... 6

## Airline

Se han resuelto 3 tipos de Code Smells (pongo un ejemplo de cada tipo):

- Aserciones incompletas:  
Complete the assertion.

```
assertThat(airline.getName().equals("Sevilla Este Airways"));
```

```
assertThat(airline.getName().equals("Sevilla Este Airways")).isTrue();
```

- Comentarios de código:  
This block of commented-out lines of code should be removed.

```
// Client client = SecurityContextHolder.getContext().getAuthentication();
```

Aquí se pueden observar el cambio realizado:

## Client

Se ha resuelto 1 tipo de Code Smells:

The image shows two side-by-side screenshots of a code quality tool's 'Filters' panel, illustrating a change in the number of detected Code Smells.

**Left Panel (Initial State):**

- Filters:** Clear All Filters
- Type:**
  - Bug: 0
  - Vulnerability: 1
  - Code Smell: 12
  - Security Hotspot: 2
- Severity:**
  - Blocker: 0
  - Critical: 1
  - Major: 3
  - Minor: 9
  - Info: 0
- Resolution:**
- Status:**
- Security Category:**
- Creation Date:**
- Language:**
- Rule:**
- Tag:**
- Directory:**
- File:** 5 SELECTED (Clear)
- Search:** airline
- Results:**
  - src/.../acmevolar/model/Airline.java
  - src/.../acmevolar/web/AirlineControll... 6
  - src/.../projections/AirlineListAttributes.ja...
  - src/.../repository/AirlineRepository.java 5
  - src/.../service/AirlineService.java 4
  - src/.../springdatajpa/SpringDataAirline...
- Footer:** 6 of 6 shown

**Right Panel (After Change):**

- Filters:** Clear All Filters
- Type:**
  - Bug: 0
  - Vulnerability: 1
  - Code Smell: 9
  - Security Hotspot: 2
- Severity:**
  - Blocker: 0
  - Critical: 1
  - Major: 0
  - Minor: 9
  - Info: 0
- Resolution:**
- Status:**
- Security Category:**
- Creation Date:**
- Language:**
- Rule:**
- Tag:**
- Directory:**
- File:** 5 SELECTED (Clear)
- Search:** airline
- Results:**
  - src/.../acmevolar/model/Airline.java
  - src/.../acmevolar/web/AirlineControll... 3
  - src/.../projections/AirlineListAttributes.ja...
  - src/.../repository/AirlineRepository.java 5
  - src/.../service/AirlineService.java 4
  - src/.../springdatajpa/SpringDataAirline...

- Aserciones incompletas:  
Complete the assertion.

```
assertThat(client1.getName().length())>0);
```

```
assertThat(client1.getName().length())>0).isTrue());
```

Aquí se pueden observar el cambio realizado:

The image displays two side-by-side screenshots of a software analysis tool's 'Issues' tab, illustrating the effect of a filter change.

**Left Screenshot:**

- Filters:**
  - Type: Bug (0), Vulnerability (2), Code Smell (22), Security Hotspot (8)
  - Severity: Blocker (0), Critical (2), Major (4), Minor (18), Info (0)
  - Resolution, Status, Security Category, Creation Date, Language, Rule, Tag, Directory: All collapsed
  - File: 10 SELECTED (Clear button)
- Search:** Client
- Results:**
  - src/.../acmevolar/model/Client.java
  - src/.../acmevolar/web/ClientControlle... 8
  - src/.../projections/ClientListAttributes.java
  - src/.../repository/ClientRepository.java 4
  - src/.../service/ClientService.java 5
  - src/.../springdatajpa/SpringDataClientR...
- Footer:** 6 of 6 shown

**Right Screenshot:**

- Filters:**
  - Type: Bug (0), Vulnerability (1), Code Smell (11), Security Hotspot (6)
  - Severity: Blocker (0), Critical (1), Major (2), Minor (9), Info (0)
  - Resolution, Status, Security Category, Creation Date, Language, Rule, Tag, Directory: All collapsed
  - File: 5 SELECTED (Clear button)
- Search:** client
- Results:**
  - src/.../acmevolar/model/Client.java 1
  - src/.../acmevolar/web/ClientControlle... 8
  - src/.../projections/ClientListAttributes.java
  - src/.../repository/ClientRepository.java 4
  - src/.../service/ClientService.java 5
  - src/.../springdatajpa/SpringDataClientR...
- Footer:** 6 of 6 shown

## Plane

Se han resuelto 4 tipos de Code Smells:



- Aserciones incompletas:

Define a constant instead of duplicating this literal "flight".

```
assertThat(nActualAviones.equals(nPrevioAviones));
```

```
assertThat(nActualAviones.equals(nPrevioAviones)).isTrue();
```

- Comentarios de código:

This block of commented-out lines of code should be removed.

```
//private static final String REQUIRED = "required";
```

- Literales repetidos:

Define a constant instead of duplicating this literal "plane" 3 times

```
private String planeString = "plane";
```

```
model.put("plane", plane);
```

```
model.put(planeString, plane)
```

- Sentencias if mal colocadas

Move this "if" to a new line or add the missing "else"

```
} if(plane.getLastMaintenance()==null) {
```

```
} else if (plane.getLastMaintenance() == null) {
```

- Excepciones generales:

Define and throw a dedicated exception instead of using a generic one.

```
if (!planes.contains(plane)) {
```

```
    throw new Exception("No está autorizado para modificar un vuelo que  
no es suyo.");
```

```
}
```

```
if (!planes.contains(plane)) {
```

```
    throw new AccessDeniedException("No está autorizado para  
modificar un vuelo que no es suyo.");
```

```
}
```

Aquí se pueden observar el cambio realizado:

FiltersClear All Filters

Type

Bug0

Vulnerability4

Code Smell62

Security Hotspot14

Severity

Blocker0

Critical18

Major11

Minor37

Info0

Resolution

Status

Security Category

Creation Date

Language

Rule

Tag

Directory

File17 SELECTEDClear

plane

src/.../acmevolar/model/Plane.java

src/.../acmevolar/web/PlaneControll...13

src/.../acmevolar/web/PlaneFormatte...2

src/.../projections/PlaneListAttributes.java

src/.../repository/PlaneRepository.java6

src/.../service/PlaneService.java9

src/.../acmevolar/web/PlaneValidato...15

src/.../springdatajpa/SpringDataPlan...3

0 of 8 shown

FiltersClear All Filters

Type

Bug0

Vulnerability2

Code Smell23

Security Hotspot6

Severity

Blocker0

Critical3

Major2

Minor20

Info0

Resolution

Status

Security Category

Creation Date

Language

Rule

Tag

Directory

File7 SELECTEDClear

plane

src/.../acmevolar/model/Plane.java

src/.../acmevolar/web/PlaneControlle...9

src/.../acmevolar/web/PlaneFormatte...2

src/.../projections/PlaneListAttributes.java

src/.../repository/PlaneRepository.java6

src/.../service/PlaneService.java9

src/.../acmevolar/web/PlaneValidator...2

src/.../springdatajpa/SpringDataPlan...3

8 of 8 shown

## Security Configuration

Se ha resuelto 1 tipo de Code Smells:

- Literales repetidos:

Define a constant instead of duplicating this literal "plane" 3 times

```
private String airlineString = "airline";  
private String clientString = "client";
```

```
.antMatchers("/my_planes").hasAuthority("airline").antMatchers("/planes/new").hasAuthority("airline").antMatchers("/planes/{^[\\d]}").hasAnyAuthority("client", "airline").antMatchers("/planes/{^[\\d]}/edit").hasAuthority("airline")
```

```
.antMatchers("/my_planes").hasAuthority(this.airlineString).antMatchers("/planes/new").hasAuthority(this.airlineString).antMatchers("/planes/{^[\\d]}").hasAnyAuthority(this.clientString,  
this.airlineString).antMatchers("/planes/{^[\\d]}/edit").hasAuthority(this.airlineString)
```

## Airport

Para esta entidad, hemos resuelto varios malos olores y un bug. Aquí se presentan dos capturas de pantalla. La primera muestra el análisis de SonarCloud antes de la refactorización. La segunda muestra el análisis después de esta.

Filters

Clear All Filters

Type

Bug

1

Vulnerability

2

Code Smell

21

Security Hotspot

11

NEW Security Hotspots

X

Security Hotspots aren't necessarily issues, but they need to be reviewed to make sure they aren't vulnerabilities.

Learn More

Severity

Blocker

0

Minor

19

Critical

3

Info

0

Major

2

Resolution

Status

Security Category

Creation Date

Language

Rule

Tag

Directory

File

9 SELECTED

Clear

airport

X

sro/.../acmevolar/model/Airport.java

sro/.../acmevolar/web/AirportContr... 22

sro/.../acmevolar/web/AirportFormatt... 4

sro/.../projections/AirportListAttributes.j...

sro/.../repository/AirportRepository.java 4

sro/.../service/AirportService.java 5

sro/.../acmevolar/web/AirportValidator.j...

sro/.../exceptions/DuplicatedAirportNa...

sro/.../springdatajpa/SpringDataAirport...

9 of 9 shown

Filters

Clear All Filters

Type

Bug

0

Vulnerability

2

Code Smell

13

Security Hotspot

11

NEW Security Hotspots

X

Security Hotspots aren't necessarily issues, but they need to be reviewed to make sure they aren't vulnerabilities.

Learn More

Severity

Blocker

0

Minor

13

Critical

2

Info

0

Major

0

Resolution

Status

Security Category

Creation Date

Language

Rule

Tag

Directory

File

9 SELECTED

Clear

airport

X

sro/.../acmevolar/model/Airport.java

sro/.../acmevolar/web/AirportContr... 16

sro/.../acmevolar/web/AirportFormatter...

sro/.../projections/AirportListAttributes.j...

sro/.../repository/AirportRepository.java 4

sro/.../service/AirportService.java 5

sro/.../acmevolar/web/AirportValidator.j...

sro/.../exceptions/DuplicatedAirportNa...

sro/.../springdatajpa/SpringDataAirt...

9 of 9 shown

A continuación, se muestra una imagen con los detalles del bug.



Este se encontraba en la clase AirportController y era causado por el código:

```
@PreAuthorize("hasAuthority('airline')")
@GetMapping(value = "/airports/{airportId}/delete")
public String deleteAirport(@PathVariable("airportId") final int airportId) throws
NonDeletableException {

    Optional<Airport> airport = this.airportService.findById(airportId);

    boolean deletable = this.flightService.findFlights().stream().noneMatch(f-
>f.getDepartes().getAirport().equals(airport.get()) ||
f.getLands().getAirport().equals(airport.get()));

    if (deletable) {
        this.airportService.deleteAirport(airport.get());
    } else {
        throw new NonDeletableException();
    }

    return "redirect:/airports";
}
```

Este bug indicaba un acceso a un objeto Optional sin verificar antes que este objeto estuviese presente, es decir, que no fuese nulo. Esto se consigue con .isPresent(), que devuelve true si el objeto no es nulo. Así, se añadió airport.isPresent() a la sentencia if:

```
if (deletable && airport.isPresent()) {
    this.airportService.deleteAirport(airport.get());
} else {
    throw new NonDeletableException();
}

return "redirect:/airports";
```

Una vez resuelto el bug, también se prestó especial atención a un mal olor catalogado como de "Major Severity". Aquí se presenta una captura de pantalla de este mal olor:

src/main/java/acmevolar/web/AirportController.java

☐ This block of commented-out lines of code should be removed. Why is this an issue?

2 days ago ▾ L124 🔗 🔍

Code Smell Major Open Not assigned 5min effort [Comment](#)

unused

Indicaba que existía un comentario que incluía código, el cuál fue eliminado:

```
//Airport airportToUpdate = this.airportService.findAirportById(airportId);
```

El resto de elementos solucionados fueron malos olores, de los cuales algunos se presentan a continuación, junto a sus respectivas soluciones:

- Diamond operator (“<>”)

```
Collection<AirportListAttributes> airports = new ArrayList<AirportListAttributes>();
```

```
Collection<AirportListAttributes> airports = new ArrayList<>();
```

- Definir constante en lugar de repetir “airport”

```
private static final String AIRPORT_CONSTANT = "airport";
```

```
model.put("airport", airport);
```

```
model.put(AirportController.AIRPORT_CONSTANT, airport);
```

- Usar .isEmpty() para comprobar si una lista está vacía

```
airport.getName().size() != 0
```

```
!airport.getName().isEmpty()
```

- Sustituir una negación junto a .anyMatch() por .noneMatch

```
boolean deletable = !this.flightService.findFlights().stream().anyMatch(f->f.getDepartes().getAirport().equals(airport.get())||f.getLands().getAirport().equals(airport.get()));
```

```
boolean deletable = this.flightService.findFlights().stream().noneMatch(f->f.getDepartes().getAirport().equals(airport.get())||f.getLands().getAirport().equals(airport.get()));
```

- Eliminar imports innecesarios (Minor)

## Runway

Para esta entidad, el análisis de SonarCloud no ha detectado ningún bug potencial, pero ha indicado la existencia de varios malos olores. Nos hemos centrado en solucionar estos. Estas

capturas de pantalla muestran los resultados de los análisis de SonarCloud antes y después de la refactorización:

Aquí se enumeran algunos de los malos olores y sus soluciones:

- Definir constante en lugar de repetir “runway” (Critical)

```
private static final String RUNWAY_CONSTANT = "runway";
```

Filters Clear All Filters

Type

- Bug 0
- Vulnerability 2
- Code Smell 47
- Security Hotspot 8

**NEW** Security Hotspots ×

Security Hotspots aren't necessarily issues, but they need to be reviewed to make sure they aren't vulnerabilities. [Learn More](#)

Severity

- Blocker 0
- Critical 4
- Major 8
- Minor 37
- Info 0

Resolution

Status

Security Category

Creation Date

Language

Rule

Tag

Directory

File 9 SELECTED Clear

runway ×

- sro/.../acmevolar/model/Runway.java 3
- sro/.../acmevolar/web/RunwayContr... 19
- sro/.../acmevolar/web/RunwayFormatte... 11
- sro/.../repository/RunwayRepository... 11
- sro/.../service/RunwayService.java 10
- sro/.../acmevolar/model/RunwayType.java 2
- sro/.../acmevolar/web/RunwayValidat... 5
- sro/.../springdatajpa/SpringDataRun... 7

0 of 0 shown

Filters Clear All Filters

Type

- Bug 0
- Vulnerability 2
- Code Smell 27
- Security Hotspot 8

**NEW** Security Hotspots ×

Security Hotspots aren't necessarily issues, but they need to be reviewed to make sure they aren't vulnerabilities. [Learn More](#)

Severity

- Blocker 0
- Critical 2
- Major 0
- Minor 27
- Info 0

Resolution

Status

Security Category

Creation Date

Language

Rule

Tag

Directory

File 9 SELECTED Clear

runway ×

- sro/.../acmevolar/model/Runway.java 3
- sro/.../acmevolar/web/RunwayContr... 11
- sro/.../acmevolar/web/RunwayFormatte... 11
- sro/.../repository/RunwayRepository... 11
- sro/.../service/RunwayService.java 8
- sro/.../acmevolar/model/RunwayType.java 2
- sro/.../acmevolar/web/RunwayValidat... 5
- sro/.../springdatajpa/SpringDataRun... 7

0 of 0 shown



```
model.put("runway", runway);
```

```
model.put(RunwayController.RUNWAY_CONSTANT, runway);
```

- Eliminar comentarios que contienen código:

```
//Optional<Airport> airport = this.airportService.findById(airportId);
```

- Usar .isEmpty() para comprobar si una lista está vacía

```
runway.getName().size() != 0
```

```
!runway.getName().isEmpty()
```

- Eliminar imports innecesarios
- Eliminar argumentos que no se estén usando en algún método

```
public void insertData(final Map<String, Object> model,  
@PathVariable("airportId") int airportId) {
```

```
List<RunwayType> runwayTypes =  
this.runwayService.findRunwaysTypes();
```

```
model.put("runwayTypes", runwayTypes);
```

```
}
```

```
public void insertData(final Map<String, Object> model,  
@PathVariable("airportId") int airportId) {
```

```
List<RunwayType> runwayTypes =  
this.runwayService.findRunwaysTypes();
```

```
model.put("runwayTypes", runwayTypes);
```

```
}
```

## Todos los tests

Se ha resultado un tipo de Code Smells para todos los tests en general:

- Clases públicas:

Remove this 'public' modifier.

```
public class ClientServiceTests {  
    class ClientServiceTests {
```