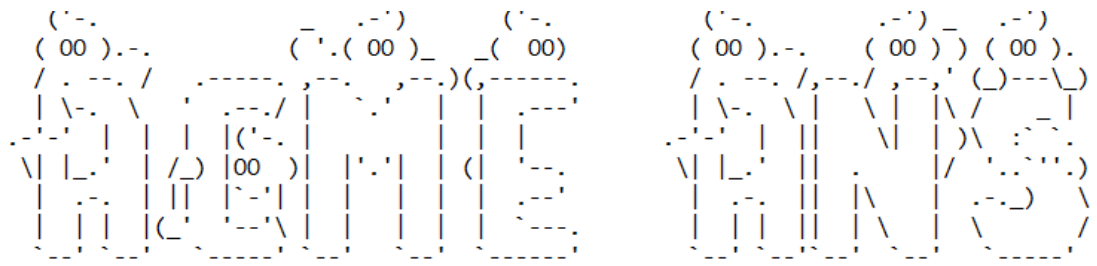
REPORTE DE TESTING Y MUTACIONES DEL CÓDIGO

Acme-ANS-D04



Repositorio: https://github.com/FranciscoFernandezN/Acme-ANS

Creado por el grupo C1.022, del G1

Participantes	
Nombres	Correos
Benito Merchán, Manuel Jesús	manbenmer1@alum.us.es

18 de Mayo de 2025

Índice

Resumen ejecutivo

Este es el proyecto del grupo C1.022 sobre Acme AirNav Solutions, S.A. (Acme ANS, S.A. abreviado), la cual es una compañía ficticia especializada en ayudar a aeropuertos a organizar y coordinar sus operaciones a partir de soluciones desarrolladas en software. La logística de los vuelos (la programación de los vuelos, la organización de reservas y de tripulación, etc.) se gestionan mediante el desarrollo de un WIS.

Con esto queda implícito que no solo será necesario desarrollar la aplicación en sí, sino también una documentación apropiada en la que se refleje la evolución de esta. En este caso, se tratará de un reporte de la planificación y progreso del proyecto por parte del estudiante Manuel Jesús Benito Merchán para el entregable D04.

Tabla de revisiones

Número	Fecha	Descripción

Introducción

En este reporte quedarán expuestos los resultados de los tests realizados durante el entregable D04 sobre las funcionalidades desarrolladas en grupo que tengan asociadas un requisito de conjunto de pruebas, por lo que se tendrá en cuenta tanto los conjuntos de pruebas realizados sobre el requisito 11 como del requisito 39.

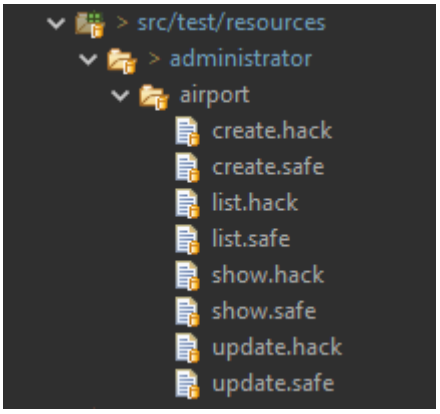
Se reportarán 2 tipos de resultados, los relacionados con el testeo funcional y los relacionados con el testeo del rendimiento. A su vez también se reportarán las 5 mutaciones realizadas sobre el código y sus distintos efectos.

Este reporte está organizado de la siguiente forma:

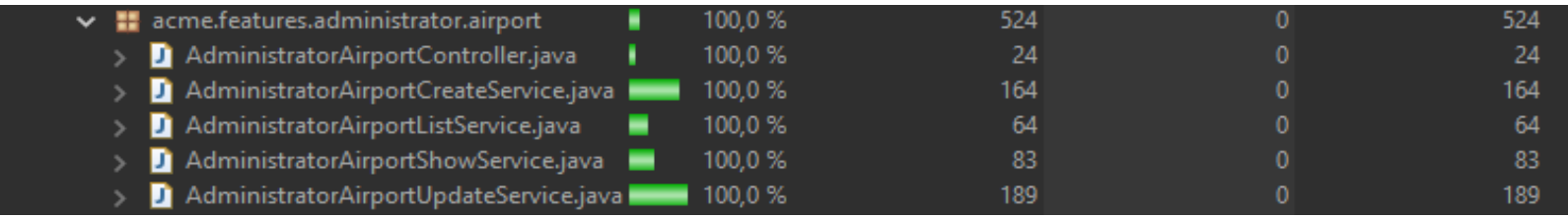
1. **Resumen ejecutivo**: Introducción breve sobre el reporte.
2. **Tabla de revisiones**: Historial de revisiones del documento.
3. **Introducción**: Contextualización de la planificación y progreso además de su importancia.
4. **Funcional testing**: Resultados obtenidos de la fase de testing y posibles bugs encontrados.
5. **Performance testing**: Resultados obtenidos de analizar el rendimiento de los test en 2 ordenadores.
6. **Mutaciones en los aeropuertos**: Resultados de introducir bugs intencionales en el código.
7. **Conclusiones**: Resumen de los hallazgos y la importancia de este reporte.
8. **Bibliografía**: Fuentes consultadas durante la investigación.

Functional testing

- Resultados sobre el testeo en los aeropuertos:

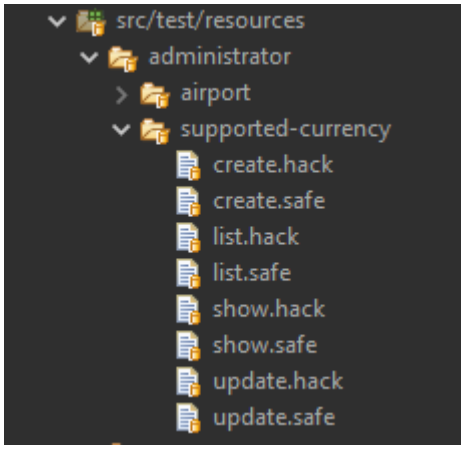


No fue detectado ningún bug durante los tests.

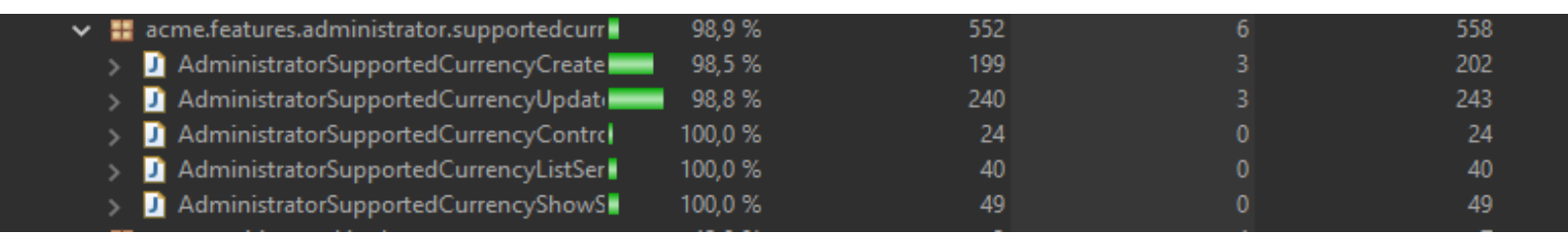


Se ha logrado alcanzar una cobertura total de la feature, cubriendo todas las instrucciones y posibles ramas/paths del código en cada servicio.

- Resultados sobre el testeo en los cambios de moneda:



Se ha logrado alcanzar una cobertura casi total de la feature, no se ha alcanzado el 100% debido a la necesidad de mockear el servicio, pero el resto de instrucciones se encuentran testeadas correctamente.



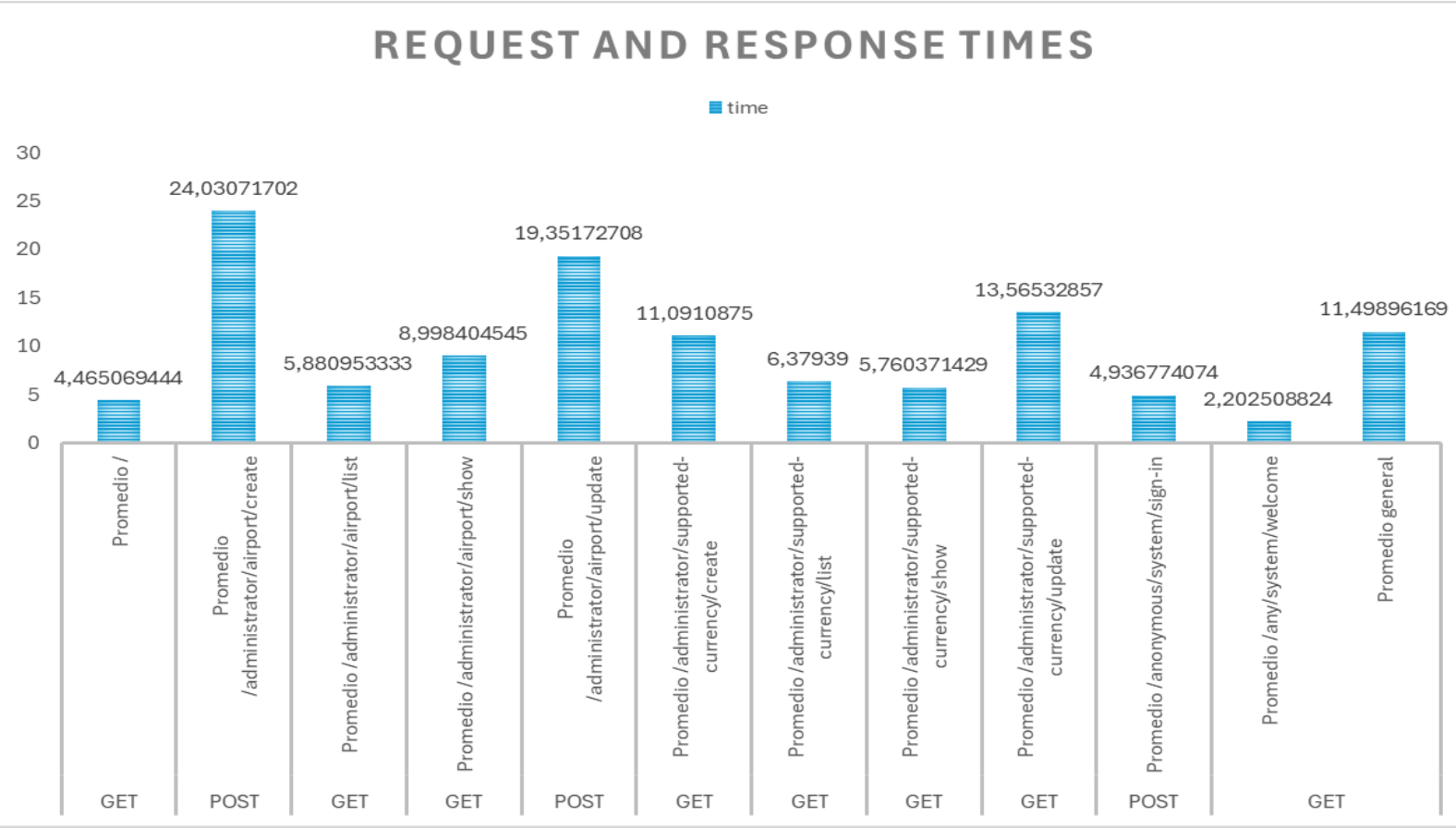
```java
@Override
public void validate(final SupportedCurrency supportedCurrency) {

    List<SupportedCurrency> supportedCurrencies = this.scr.findAllSupportedCurrencies();
    List<String> currencyNames = supportedCurrencies.stream().filter(sp -> sp.getId() != supportedCurrency.getId()).map(sp -> sp.getCurrencyName()).toList();

    super.state(!(this.scr.findDefaultSupportedCurrency().getId() == supportedCurrency.getId() && !supportedCurrency.getIsDefaultCurrency()), "isDefaultCurrency", "administrator.supported-currenc

    if (supportedCurrency.getCurrencyName() != null)
        super.state(!currencyNames.contains(supportedCurrency.getCurrencyName()), "currencyName", "administrator.supported-currency.create.already-exists-currency");
}
```

Como se puede observar, aunque se haya alcanzado la total cobertura de instrucciones, la cobertura de ramas no era total ni en el servicio de Update ni en el de Create por la misma razón, pero gracias a esto, hemos detectado que se podría eliminar la condición ya que en ningún momento ha sido posible que esa condición no se cumpliera por razones que desconocemos, por lo que la solución ha sido eliminarla al ser inservible.

```java
@Override
public void validate(final SupportedCurrency supportedCurrency) {

    List<SupportedCurrency> supportedCurrencies = this.scr.findAllSupportedCurrencies();
    List<String> currencyNames = supportedCurrencies.stream().filter(sp -> sp.getId() != supportedCurrency.getId()).map(sp -> sp.getCurrencyName()).toList();

    super.state(!(this.scr.findDefaultSupportedCurrency().getId() == supportedCurrency.getId() && !supportedCurrency.getIsDefaultCurrency()), "isDefaultCurrency", "ad

    super.state(!currencyNames.contains(supportedCurrency.getCurrencyName()), "currencyName", "administrator.supported-currency.create.already-exists-currency");
```

Performance testing

- Resultados en ordenador 1:

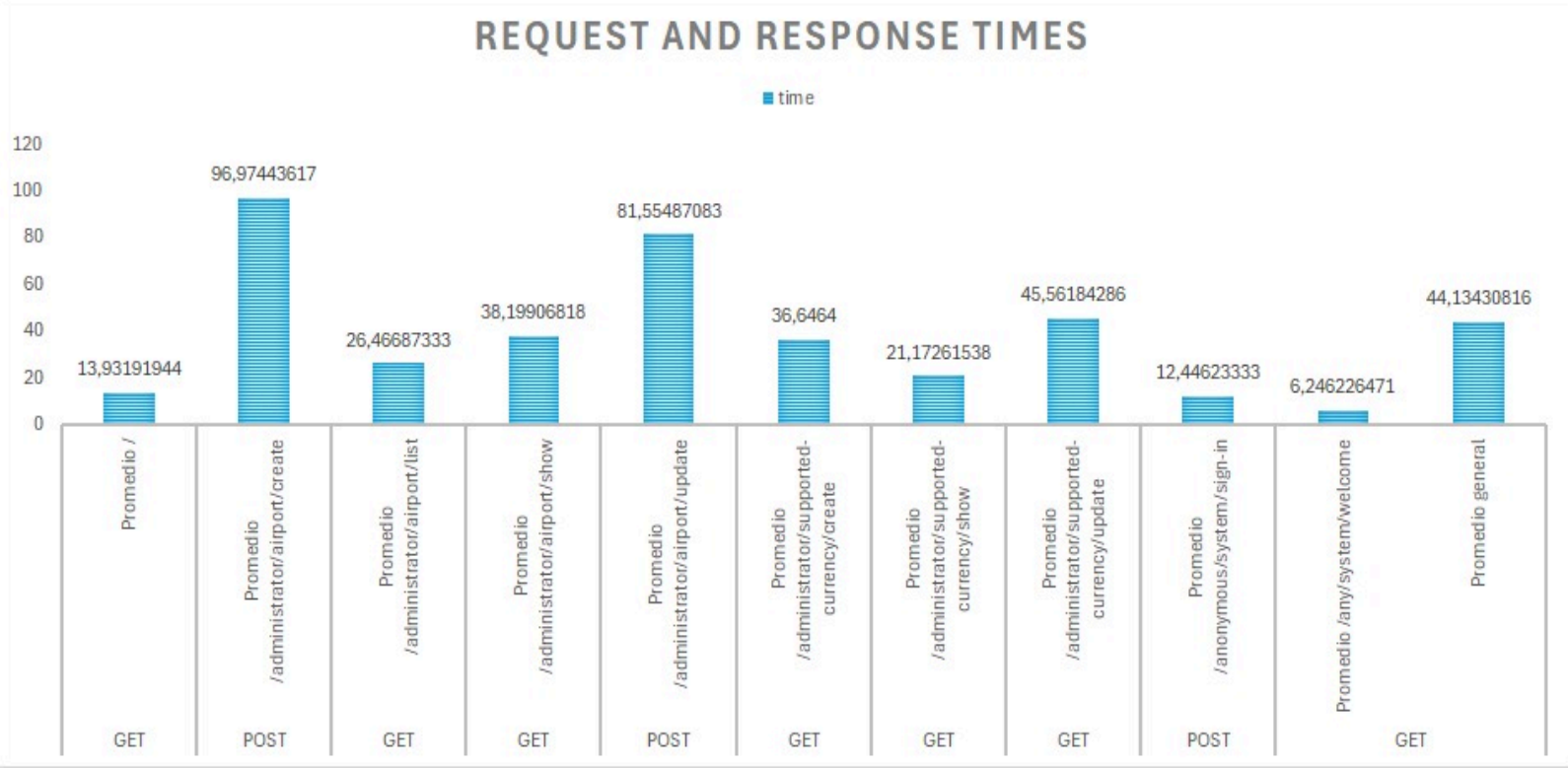


En esta tabla podemos observar los tiempos promedios de las distintas peticiones realizadas tanto en la parte de aeropuertos como en la parte de los cambios de moneda por el ordenador 1.

Solo la parte correspondiente a los aeropuertos, gracias a una cuenta rápida, nos da un promedio de 14.56545049 ya que conforma la parte más amplia de testear entre estas 2 aquí incluídas. Si juntamos esta parte de aeropuertos con el resto de peticiones que no son las relacionadas con la parte de las monedas nos da un promedio de ~9.823593474, un tiempo ligeramente menor a 0.01 segundos. A continuación los resultados del análisis.

	A	B	C	D	E	F
1	*Columna1*				Max(0, Media-confianza)	Suma(Media+confianza)
2				Interval (ms)	10,17006978	12,82785359
3	Media	11,4989617		Interval (s)	0,01017007	0,012827854
4	Error típico	0,67486245				
5	Mediana	6,0432				
6	Moda	#N/D				
7	Desviación estándar	10,9027366				
8	Varianza de la muestra	118,869666				
9	Curtosis	4,19971135				
10	Coeficiente de asimetría	1,61624082				
11	Rango	70,3963				
12	Mínimo	1,1783				
13	Máximo	71,5746				
14	Suma	3001,229				
15	Cuenta	261				
16	Nivel de confianza(95,0%)	1,32889191				

- Resultados del ordenador 2:



En esta tabla podemos observar los tiempos promedios de las distintas peticiones realizadas tanto en la parte de aeropuertos como en la parte de los cambios de moneda por el ordenador 2.

Solo la parte correspondiente a los aeropuertos, gracias a una cuenta rápida, nos da un promedio de 60.79881213 ya que conforma la parte más amplia de testear entre estas 2 aquí incluídas. Si juntamos esta parte de aeropuertos con el resto de peticiones que no son las relacionadas con la parte de las monedas nos da un promedio de ~42.07315536. A continuación los resultados del análisis.

Columna1					
			Interval (ms)	39,0325481	49,2360682
Media	44,1343082				
Error típico	2,59177555				
Mediana	21,9314				
Moda	#N/D				
Desviación es	43,5233127				
Varianza de la	1894,27875				
Curtosis	2,02140747				
Coeficiente de	1,30622214				
Rango	250,7932				
Mínimo	3,697				
Máximo	254,4902				
Suma	12445,8749				
Cuenta	282				
Nivel de confia	5,10176009				

Debido a que el código ya se encontraba en su estado más óptimo al realizar la fase de testing no ha sido posible realizar un hypothesis contrast con valores antes y después de realizar algún cambio.

Aún así podemos sacar conclusiones en base a los datos obtenidos por ambos ordenadores. Como se puede apreciar, hay una diferencia muy significativa en los tiempos de testeo, esto se debe principalmente a la gran diferencia de hardware entre ambos ordenadores. El ordenador 1 es un ordenador de gama alta, gaming y que se encontraba cargando en esos momentos, en cambio el ordenador 2, es un ordenador de gama media, de uso cotidiano y no estaba conectado en esos momentos.

Mutaciones en los aeropuertos

- Mutación 1, eliminar el authorise en AdministratorAirportCreateService

```
@Override
public void authorise() {
    super.getResponse().setAuthorised(super.getRequest().getPrincipal().hasRealmOfType(Administrator.class));
}
```

```
@Override
public void authorise() {
    super.getResponse().setAuthorised(false);
}

@Override
public void load() {
    Airport airport;

    airport = new Airport();

    super.getBuffer().addData(airport);
}

@Override
public void bind(final Airport airport) {
    super.bindObject(airport, "name", "iATACode", "operationalScope", "city", "country", "website", "email", "contactNumber");
}

@Override
public void validate(final Airport object) {

    List<Airport> airports = this.aar.findAllAirports();
    List<String> airportCodes = airports.stream().map(Airport::getIATACode).toList();

    boolean confirmation = super.getRequest().getData("confirm", boolean.class);
    super.state(confirmation, "confirm", "acme.validation.confirmation.message");

    super.state(!airportCodes.contains(object.getIATACode()), "iATACode", "administrator.airport.create.already-exists");

}

@Override
public void perform(final Airport airport) {
    this.aar.save(airport);
}
```

```
FAILED GET /administrator/airport/create (request-id="da5fda4f-2e24-48ba-9640-2b5d74a0ab16", input=""): Expected 'status' to be '200', but got
'500'. Expected 'payload' to be '{confirm=false, id=0, operationalScope=[{"key":"0","label":"----","selected":true,"sealed":true},
{"key":"INTERNATIONAL","label":"INTERNATIONAL","selected":false,"sealed":true},
{"key":"DOMESTIC","label":"DOMESTIC","selected":false,"sealed":true},{"key":"REGIONAL","label":"REGIONAL","selected":false,"sealed":true}],
service=159, version=0}', but got '{service=159}'.
FAILED POST /administrator/airport/create (request-id="46f96c07-27f5-43c5-8da5-faa3b86aa9c6",
input="id=0&version=0&name=&iATACode=&operationalScope=0&city=&country=&website=&email=&contactNumber=&confirm=false"): Expected 'status' to be
'200', but got '500'. Expected 'payload' to be '{city=, city$error=May not be null., confirm=false, confirm$error=Must confirm this.,
contactNumber=, country=, country$error=May not be null., email=, iATACode=, iATACode$error=May not be null., id=0, name=, name$error=May not be
null., operationalScope=[{"key":"0","label":"----","selected":true,"sealed":true},
{"key":"INTERNATIONAL","label":"INTERNATIONAL","selected":false,"sealed":true},
{"key":"DOMESTIC","label":"DOMESTIC","selected":false,"sealed":true},{"key":"REGIONAL","label":"REGIONAL","selected":false,"sealed":true}],
operationalScope$error=May not be null., service=159, version=0, website=}', but got '{service=159}'.
```

El sistema no permite realizar ningún tipo de operación create ya que se le deniega el acceso con un 500 automáticamente.

- Mutación 2, cambiar los rangos máximos de strings en ciertos atributos

```java
@Mandatory
@ValidString(max = 50)
@Automapped
private String              name;

@Mandatory
@ValidString(min = 3, max = 3, pattern = "[A-Z]{3}")
@Column(unique = true)
private String              iATACode;

@Mandatory
@Enumerated(EnumType.STRING)
@Automapped
private OperationalScope    operationalScope;

@Mandatory
@ValidString(max = 50)
@Automapped
private String              city;

@Mandatory
@ValidString(max = 50)
@Automapped
private String              country;
```

```java
@Mandatory
@ValidString(max = 49)
@Automapped
private String              name;

@Mandatory
@ValidString(min = 3, max = 3, pattern = "[A-Z]{3}")
@Column(unique = true)
private String              iATACode;

@Mandatory
@Enumerated(EnumType.STRING)
@Automapped
private OperationalScope    operationalScope;

@Mandatory
@ValidString(max = 49)
@Automapped
private String              city;

@Mandatory
@ValidString(max = 49)
@Automapped
private String              country;
```

```
Resetting application (clear schema, populate sample, reset clock, reset randomiser).
ERROR Validating 'airport-06' ... FAILED. 'name': Length must be between 0 and 49. 'city': Length must be between 0 and 49. 'country': Length must be between 0 and 49.
Replaying .\src\test\resources\administrator\airport\create.hack...
Loaded 34 requests from .\src\test\resources\administrator\airport\list.hack
Resetting application (clear schema, populate sample, reset clock, reset randomiser).
ERROR Validating 'airport-06' ... FAILED. 'city': Length must be between 0 and 49. 'country': Length must be between 0 and 49. 'name': Length must be between 0 and 49.
Replaying .\src\test\resources\administrator\airport\list.hack...
Loaded 154 requests from .\src\test\resources\administrator\airport\show.hack
Resetting application (clear schema, populate sample, reset clock, reset randomiser).
ERROR Validating 'airport-06' ... FAILED. 'name': Length must be between 0 and 49. 'country': Length must be between 0 and 49. 'city': Length must be between 0 and 49.
```

- Mutación 3, cambiar condición de un validador en AdministratorAirportCreateService:
  Se espera que la operación de confirmación funcione, pero al confirmarse, se provoca un fallo.

```
@Override
public void validate(final Airport object) {

    List<Airport> airports = this.aar.findAllAirports();
    List<String> airportCodes = airports.stream().map(Airport::getIATACode).toList();

    boolean confirmation = super.getRequest().getData("confirm", boolean.class);
    super.state(confirmation, "confirm", "acme.validation.confirmation.message");

    super.state(!airportCodes.contains(object.getIATACode()), "iATACode", "administrator.airport.create.already-exists");

}
```

```
@Override
public void validate(final Airport object) {

    List<Airport> airports = this.aar.findAllAirports();
    List<String> airportCodes = airports.stream().map(Airport::getIATACode).toList();

    boolean confirmation = super.getRequest().getData("confirm", boolean.class);
    super.state(!confirmation, "confirm", "acme.validation.confirmation.message");

    super.state(!airportCodes.contains(object.getIATACode()), "iATACode", "administrator.airport.create.already-exists");

}
```

```
FAILED POST /administrator/airport/create (request-id="544dc392-84b1-4e4f-aea0-446adb6f41b4", input="id=0&version=0&name=Aeropuerto
+creado&iATACode=ABC&operationalScope=INTERNATIONAL&city=Ciudad&country=Pa%C3%ADs&website=https%3A%2F%2Fwww.website.com&email=testing
%40hotmail.com&contactNumber=12345678901234&confirm=true"): Expected 'payload' to be '{service=160}', but got '{city=Ciudad, confirm=false,
confirm$error=Must confirm this., contactNumber=12345678901234, country=País, email=testing@hotmail.com, iATACode=ABC, iATACode$error=This IATA
code already exists., id=0, name=Aeropuerto creado, operationalScope=[{"key":"0","label":"----","selected":false,"sealed":true},
{"key":"INTERNATIONAL","label":"INTERNATIONAL","selected":true,"sealed":true},
{"key":"DOMESTIC","label":"DOMESTIC","selected":false,"sealed":true},{"key":"REGIONAL","label":"REGIONAL","selected":false,"sealed":true}],
service=160, version=0, website=https://www.website.com}'.
Loaded 66 requests from .\src\test\resources\administrator\airport\list.safe
Resetting application (clear schema, populate sample, reset clock, reset randomiser).
Replaying .\src\test\resources\administrator\airport\list.safe...
Loaded 330 requests from .\src\test\resources\administrator\airport\show.safe
```

- Mutación 4, eliminar una validación en AdministratorAirportCreateService:

```
@Override
public void validate(final Airport object) {

    List<Airport> airports = this.aar.findAllAirports();
    List<String> airportCodes = airports.stream().map(Airport::getIATACode).toList();

    boolean confirmation = super.getRequest().getData("confirm", boolean.class);
    super.state(confirmation, "confirm", "acme.validation.confirmation.message");

    super.state(!airportCodes.contains(object.getIATACode()), "iATACode", "administrator.airport.create.already-exists");

}
```

```
@Override
public void validate(final Airport object) {

    List<Airport> airports = this.aar.findAllAirports();
    List<String> airportCodes = airports.stream().map(Airport::getIATACode).toList();

    boolean confirmation = super.getRequest().getData("confirm", boolean.class);
    super.state(confirmation, "confirm", "acme.validation.confirmation.message");

}
```

```
FAILED POST /administrator/airport/create (request-id="80d4fae2-a62c-49f5-a532-3f9b653ad8bd",
input="id=0&version=0&name=&iATACode=BCN&operationalScope=0&city=&country=&website=&email=&contactNumber=&confirm=false"): Expected 'payload' to
be '{city=, city$error=May not be null., confirm=false, confirm$error=Must confirm this., contactNumber=, country=, country$error=May not be
null., email=, iATACode=BCN, iATACode$error=This IATA code already exists., id=0, name=, name$error=May not be null., operationalScope=
[{"key":"0","label":"----","selected":true,"sealed":true},{"key":"INTERNATIONAL","label":"INTERNATIONAL","selected":false,"sealed":true},
{"key":"DOMESTIC","label":"DOMESTIC","selected":false,"sealed":true},{"key":"REGIONAL","label":"REGIONAL","selected":false,"sealed":true}],
operationalScope$error=May not be null., service=158, version=0, website=}', but got '{city=, city$error=May not be null., confirm=false, confirm
$error=Must confirm this., contactNumber=, country=, country$error=May not be null., email=, iATACode=BCN, id=0, name=, name$error=May not be
null., operationalScope=[{"key":"0","label":"----","selected":true,"sealed":true},
{"key":"INTERNATIONAL","label":"INTERNATIONAL","selected":false,"sealed":true},
{"key":"DOMESTIC","label":"DOMESTIC","selected":false,"sealed":true},{"key":"REGIONAL","label":"REGIONAL","selected":false,"sealed":true}],
```

El sistema permite IATACodes duplicados gracias a este cambio.

- Mutación 5, cambiar la obligatoriedad de ciertos atributos:

```
@Optional
@ValidUrl
@Automapped
private String              website;

@Optional
@ValidEmail
@Automapped
private String              email;

@Optional
@ValidString(pattern = "^\\+?\\d{6,15}$")
@Automapped
private String              contactNumber;
```

```
@Mandatory
@ValidUrl
@Automapped
private String              website;

@Mandatory
@ValidEmail
@Automapped
private String              email;

@Mandatory
@ValidString(pattern = "^\\+?\\d{6,15}$")
@Automapped
private String              contactNumber;
```

```
FAILED POST /administrator/airport/create (request-id="95657d49-6663-428f-9c04-fc0edb1372b5", input="id=0&version=0&name=&iATACode=%D5%B4%E1%83%9D
%E5%A4%A9%E5%AF%AB.
%C3%81AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA&operationalScope=0&city=&country=&website=&email=&contactNumber=&confirm=false"): Expected
'payload' to be '{city=, city$error=May not be null., confirm=false, confirm$error=Must confirm this., contactNumber=, country=, country$error=May
not be null., email=, iATACode=Մ天寫.ÁAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA, iATACode$error=Length must be between 3 and 3., id=0, name=,
name$error=May not be null., operationalScope=[{"key":"0","label":"----","selected":true,"sealed":true},
{"key":"INTERNATIONAL","label":"INTERNATIONAL","selected":false,"sealed":true},
{"key":"DOMESTIC","label":"DOMESTIC","selected":false,"sealed":true},{"key":"REGIONAL","label":"REGIONAL","selected":false,"sealed":true}],
operationalScope$error=May not be null., service=159, version=0, website=}', but got '{city=, city$error=May not be null., confirm=false, confirm
$error=Must confirm this., contactNumber=, contactNumber$error=May not be null., country=, country$error=May not be null., email=, email$error=May
not be null., iATACode=Մ天寫.ÁAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA, iATACode$error=Length must be between 3 and 3., id=0, name=, name
$error=May not be null., operationalScope=[{"key":"0","label":"----","selected":true,"sealed":true},
{"key":"INTERNATIONAL","label":"INTERNATIONAL","selected":false,"sealed":true},
{"key":"DOMESTIC","label":"DOMESTIC","selected":false,"sealed":true},{"key":"REGIONAL","label":"REGIONAL","selected":false,"sealed":true}],
operationalScope$error=May not be null., service=159, version=0, website=, website$error=May not be null.}'.
```

Conclusiones

Como conclusión de este reporte remarcar que debido a la exhaustiva búsqueda de fallos antes de realizar los tests, estos no fueron capaces de encontrar ningún bug y pasaron perfectamente, por otro lado es claro que la velocidad de ejecución de los tests depende en gran medida del hardware con el que se ejecuten por lo que sería injusto una comparación entre ambos ordenadores.

Bibliografía

Intencionalmente en blanco