|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  | | Leer desde MCA la información enviada por el BCK | | | | |  |
| Registro de modificaciones | | | | | | | |
| **Versión** | **Descripción** | | **Autor** | **Fecha creación** | **Aprobado por** | **Fecha de aprobación** | |
| 1.0 | UAT Inicial | | everis | 18/10/2019 |  | 18/10/2019 | |
|  |  | |  |  |  |  | |

/

**Contenido**

[1. Descripción 2](#_Toc25596380)

[2. Recursos 2](#_Toc25596381)

[3. Definición 2](#_Toc25596382)

[4. Criterios de aceptación 2](#_Toc25596383)

[5. Tabla de resultados 2](#_Toc25596384)

[6. Tests Automáticos con JUnit 4](#_Toc25596385)

[6.1 Test en MCA 4](#_Toc25596386)

[6.1.1 Guardado en el profile de cabeceras cuando no había nada previamente guardado y la propiedad está activada 4](#_Toc25596387)

[6.1.2 Guardado en el profile de cabeceras cuando ya existe información previa, está relacioanda con ARQUEO y la propiedad está activada 6](#_Toc25596388)

[6.1.3 Guardado en el profile de cabeceras cuando ya existe información previa, no está relacionada con ARQUEO y la propiedad está activada 7](#_Toc25596389)

[6.2 Test en BCK 9](#_Toc25596390)

[6.2.1 Clase PrepareArqueoInfoHandler 9](#_Toc25596391)

[6.2.1.1 Si el profile viene nulo no se ejecuta el seteo de cabeceras 9](#_Toc25596392)

[6.2.1.2 Si el canal no tiene permitido ejecutar el seteo de cabeceras 9](#_Toc25596393)

[6.2.1.3 Si el SN no es arqueable no se ejecuta el seteo de cabeceras 10](#_Toc25596394)

[6.2.1.4 Si el SN es arqueable y tiene un dato de arqueo, se ejecuta el seteo de cabeceras 11](#_Toc25596395)

[6.2.1.5 Si el SN es arqueable pero no tiene datos de arqueo, no se ejecuta el seteo de cabeceras 11](#_Toc25596396)

[6.2.1.6 Si el SN es arqueable y tiene varios datos de arqueo, se ejecuta el seteo de cabeceras 12](#_Toc25596397)

[6.2.2 Clase ArqueoDataImpl 13](#_Toc25596398)

[6.2.2.1 Si el profile es nulo se debe devolver un NullPointerException 13](#_Toc25596399)

[6.2.2.2 Si el profile contiene un mapa pero este está vacío, no se pasará ningún dato a las cabeceras 13](#_Toc25596400)

[6.2.2.3 Si el profile contiene un mapa con un único elemento, la cabecera contendrá ese único valor 14](#_Toc25596401)

[6.2.2.4 Si el profile contiene un mapa con varios elementos, la cabecera contendrá todos los valores 14](#_Toc25596402)

[7. Tests Manuales 15](#_Toc25596403)

[7.1 Autorización sin motivo automático 16](#_Toc25596404)

[7.2 Autorización con un motivo OK 16](#_Toc25596405)

[7.3 Autorización con un motivo NOK 17](#_Toc25596406)

[7.4 Autorización con un motivo OK y dos no automático 18](#_Toc25596407)

[7.5 Autorización con dos motivos NOK 19](#_Toc25596408)

[7.6 Autorización con un motivo OK y uno NOK 20](#_Toc25596409)

[7.7 Autorización con un motivo no automático, pero por defecto es Automático 20](#_Toc25596410)

# Descripción

Tras el desarrollo de la nueva implementación para poder traspasar la información de aquellos SN “arqueable” desde BCK a MCA, se adquiere la necesidad de comprobar que la respuesta dada a la información obtenida en las cabeceras provenientes desde BCK sea la correcta.

Esta nueva versión va a ser probada en el entorno de DSV, y utilizaremos una demo generada para comprobar el correcto funcionamiento del desarrollo realizado y de los test unitarios automáticos creados con JUnit.

El objetivo principal es demostrar que el tratamiento a las cabeceras provenientes desde BCK y hayan sido identificado como “arqueables”, se tratan de la forma adecuada, es decir, guardando una lista de String en un mapa cuya key sea “ARQUEO”, que a su vez será seteado en el “requestData” del profile en la key “outpuheaders”.

# Recursos

* Test automáticos con JUnit
* Test manuales

# Definición

|  |  |
| --- | --- |
| **RTC / Proyecto** | **Leer desde MCA la información enviada por el BCK** |
| **Fecha de ejecución** | 18/09/2019 |
| **Entorno** | LOCAL, DSV |

# Criterios de aceptación

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | **Historias de usuario** | | | **Criterios de Aceptación** | | |
| **Ámbito** | **Id.** | Título | **Descripción** | **Id.** | **Escenario** | **Descripción** | |
|  |  |  |  |  |  |  | |
| **MCA** | F-001 | Envío de información a MCA desde BCK | Tests Automáticos con JUnit | 1 | MCA | Si la propiedad de arqueo está activada, se tendrá que tener en cuenta si ya existe información en el profile. | |
| F-003 | Envío de información a MCA desde BCK | Test manuales | 1 | MCA | Se comprueba que se recoge información del monitor de arqueo de la capa de Backend | |
| **BCK** | F-002 | Recolección y envío de información de Arqueo a MCA | Tests Automáticos con JUnit | 1 | BCK | Se prepara el SI que será recogido desde MCA. | |

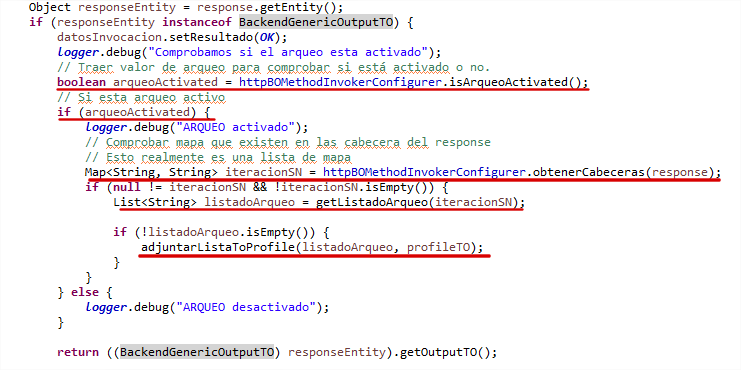
# Tabla de resultados

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Id | Historia | Criterio | Caso de prueba | Descripción | Estado |
| 1 | F-001 | 1 | Guardado en el profile de cabecera cuando no había nada previamente guardado | Se verifica que se guarda correctamente la cabecera en el profile tras obtener cabeceras desde el BCK, siempre y cuando la propiedad de arqueo esté activada. | **OK** |
| 2 | F-001 | 1 | Guardado en el profile de cabecera cuando ya existe información previamente guardada | Se verifica que se guarda correctamente la cabecera en el profile tras obtener cabeceras desde el BCK, siempre y cuando la propiedad de arqueo esté activada. | **OK** |
| 3 | F-001 | 1 | Guardado en el profile de cabecera cuando ya existe información previamente guardada | Se verifica que se guarda correctamente la cabecera en el profile tras obtener cabeceras desde el BCK, siempre y cuando la propiedad de arqueo esté activada. | **OK** |
| 4 | F-002 | 1 | Si el profile viene nulo no se ejecuta el seteo de cabeceras | Se verifica que si el profile viene nulo las cabeceras no serán ejecutadas | **OK** |
| 5 | F-002 | 1 | Si el canal no tiene permitido ejecutar el seteo de cabeceras | Se verifica que si el canal no está permitido el seteo de cabeceras. | **OK** |
| 6 | F-002 | 1 | Si el SN no es arqueable no se ejecuta el seteo de cabeceras | Se verifica que si el SN no es arqueable el seteo de cabeceras no se ejecuta. | **OK** |
| 7 | F-002 | 1 | Si el SN es arqueable y tiene un dato de arqueo, se ejecuta el seteo de cabeceras | Se verifica que si el SN es arqueable y tiene datos, el seteo de cabeceras se ejecuta. | **OK** |
| 8 | F-002 | 1 | Si el SN es arqueable pero no tiene datos de arqueo, no se ejecuta el seteo de cabeceras | Se verifica que si el SN es arqueable y no tiene datos, el seteo de cabeceras no se ejecuta. | **OK** |
| 9 | F-002 | 1 | Si el SN es arqueable y tiene varios datos de arqueo, se ejecuta el seteo de cabeceras | Se verifica que si el SN es arqueable y tiene varios datos, el seteo de cabeceras se ejecuta. | **OK** |
| 10 | F-002 | 1 | Si el profile es nulo se debe devolver un NullPointerException | Se verifica que si el profile es nulo se debe devolver un NullPointerException. | **ERROR** |
| 11 | F-002 | 1 | Si el profile contiene un mapa pero este está vacío, no se pasará ningún dato a las cabeceras | Se verifica que Si el profile contiene un mapa pero este está vacío, no se pasará ningún dato a las cabeceras. | **OK** |
| 12 | F-002 | 1 | Si el profile contiene un mapa con un único elemento, la cabecera contra ese único valor | Se verifica que si el profile contiene un mapa con un único elemento, la cabecera contra ese único valor. | **OK** |
| 13 | F-002 | 1 | Si el profile contiene un mapa con varios elementos, la cabecera contendrá todos los valores | Se verifica que si el profile contiene un mapa con varios elementos, la cabecera contendrá todos los valores. | **OK** |
| 14 | F-003 | 1 | Verificación del filtrado por un motivo que no es automático. | Se verifica que las autorizaciones con un motivo no automático funcionan correctamente. | **OK** |
| 15 | F-003 | 1 | Verificación del filtrado por un motivo que es automático. | Se verifica que las autorizaciones con un motivo automático (OK) funcionan correctamente. | **OK** |
| 16 | F-003 | 1 | Verificación del filtrado por un motivo que es automático pero NOK. | Se verifica que las autorizaciones con un motivo automático pero NOK funcionan correctamente. | **OK** |
| 17 | F-003 | 1 | Verificación del filtrado por un motivo que es automático OK y dos no automáticos. | Se verifica que las autorizaciones con un motivo automático OK y dos no automáticos funcionan correctamente. | **OK** |
| 18 | F-003 | 1 | Verificación del filtrado de dos motivos NOK. | Se verifica que las autorizaciones con dos motivos automáticos NOK funcionan correctamente. | **OK** |
| 19 | F-003 | 1 | Verificación del filtrado de un motivo OK y otro NOK | Se verifica que las autorizaciones con un motivo automático OK y otro NOK funcionan correctamente. | **OK** |
| 20 | F-003 | 1 | Verificación del filtrado de un motivo inexistente pero por defecto es OK | Se verifica que una autorización que no es automática pero su servicio si lo es, devolverá un OK. | **OK** |

# Tests Automáticos con JUnit

## Test en MCA

Anteriormente, las cabeceras no recibían ningún tratamiento en MCA. Con este nuevo desarrollo, se va a controlar si viene alguna “key” con el prefijo “**ARQUEO-**“. En caso afirmativo, se guardará en una lista que posteriormente será utilizada para guardar en el Profile de la arquitectura.

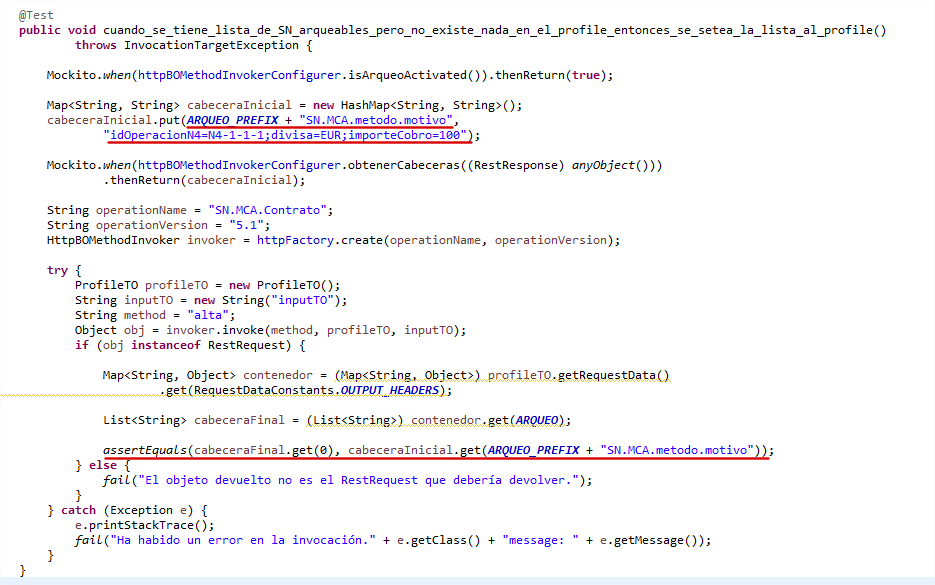


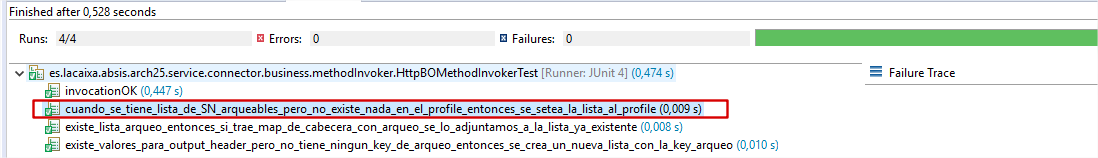
## Guardado en el profile de cabeceras cuando no había nada previamente guardado y la propiedad está activada

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Id | Historia | Criterio | Caso de prueba | Descripción | Estado |
| 1 | F-001 | 1 | Guardado en el profile de cabecera cuando no había nada previamente guardado | Se verifica que se guarda correctamente la cabecera en el profile tras obtener cabeceras desde el BCK, siempre y cuando la propiedad de arqueo esté activada. | **OK** |

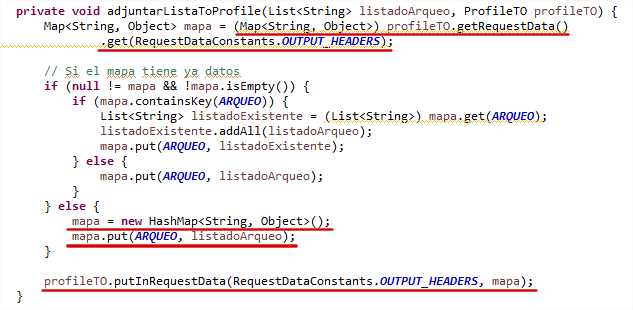
Se va a realizar un test que comprueba que cuando se obtiene las cabeceras del BCK y una de ellas contiene la key “**ARQUEO-**“, su valor (cadena con todos los datos) se guardará en una lista, que posteriormente servirá para añadir a la lista que se obtendrá del mapa que proviene del profile para la key “**ARQUEO**”.

**TEST**





**CLASE TRAZA:**



En esta imagen se ve el método encargado de guardar en el profile la lista de valores obtenido de las cabeceras. Este método reciba la lista de valores y el profile.

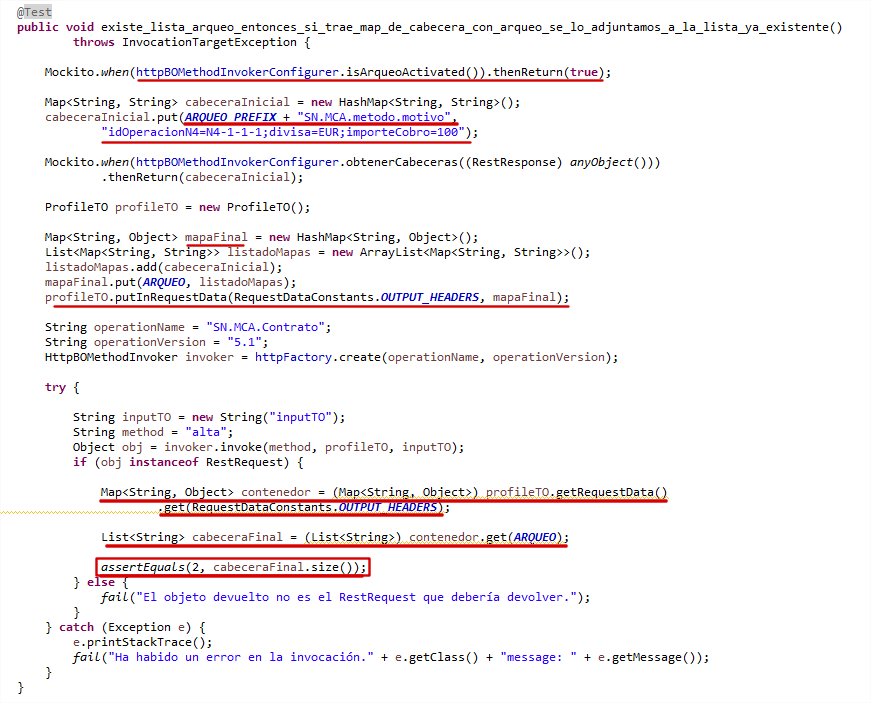
Como vemos, primero se obtiene el mapa del profile para la key “outputHeaders”. Si este trae datos, comprobamos si ya contiene alguna key de “**ARQUEO**”. En este caso, la respuesta es negativa, por lo que directamente añadiremos al mapa la lista con la key “**ARQUEO**”.

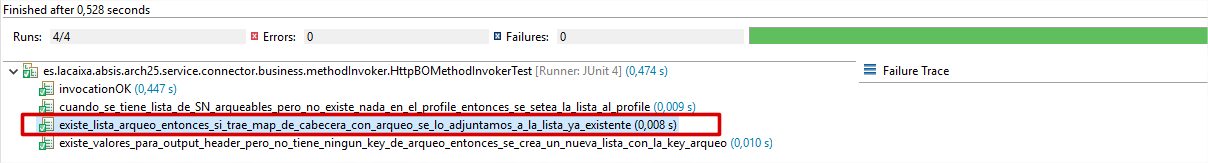
## Guardado en el profile de cabeceras cuando ya existe información previa, está relacioanda con ARQUEO y la propiedad está activada

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Id | Historia | Criterio | Caso de prueba | Descripción | Estado |
| 2 | F-001 | 1 | Guardado en el profile de cabecera cuando ya existe información previamente guardada | Se verifica que se guarda correctamente la cabecera en el profile tras obtener cabeceras desde el BCK, siempre y cuando la propiedad de arqueo esté activada. | **OK** |

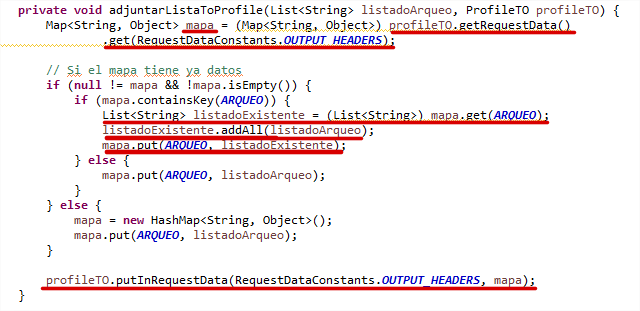
Se va a realizar un test que comprueba que cuando se obtiene las cabeceras del BCK y una de ellas contiene la key “**ARQUEO-**“, su valor (cadena con todos los datos) se guardará en una lista, que posteriormente servirá para añadir a la lista que se obtendrá del mapa que proviene del profile para la key “**ARQUEO**”.

**TEST**





**CLASE TRAZA:**



En esta imagen se ve el método encargado de guardar en el profile la lista de valores obtenido de las cabeceras. Este método reciba la lista de valores y el profile.

Como vemos, primero se obtiene el mapa del profile para la key “outputHeaders”. Si este trae datos, comprobamos si ya contiene alguna key de “**ARQUEO**”. En este caso, la respuesta es afirmativa, por lo que obtendremos la lista de valores existentes en el mapa para la key “**ARQUEO**”. Añadiremos la lista nueva de valores a la ya existente y luego volveremos a setear al mapa obtenido inicialmente la lista para la key “**ARQUEO**”.

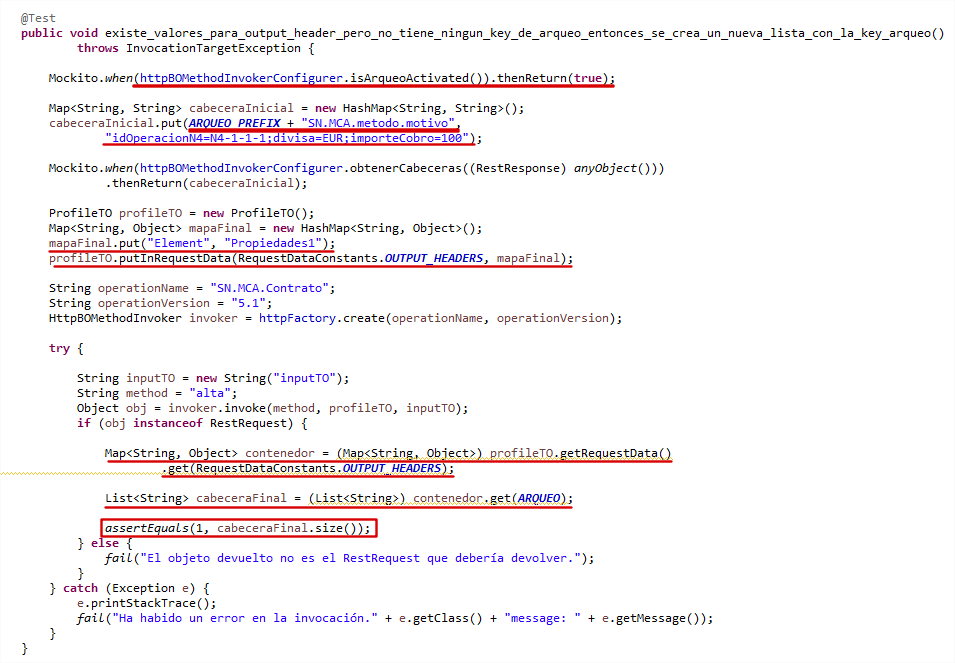
Por último, volvemos a setear este mapa al profile para la key “outputHeaders”.

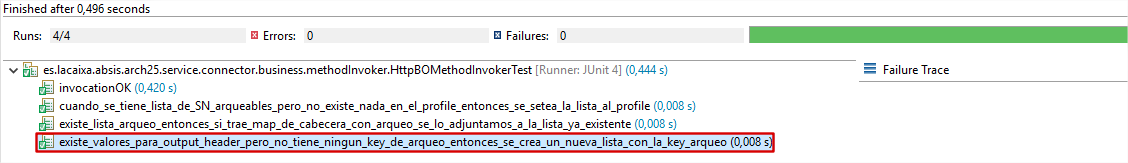
## Guardado en el profile de cabeceras cuando ya existe información previa, no está relacionada con ARQUEO y la propiedad está activada

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Id | Historia | Criterio | Caso de prueba | Descripción | Estado |
| 3 | F-001 | 1 | Guardado en el profile de cabecera cuando ya existe información previamente guardada | Se verifica que se guarda correctamente la cabecera en el profile tras obtener cabeceras desde el BCK, siempre y cuando la propiedad de arqueo esté activada. | **OK** |

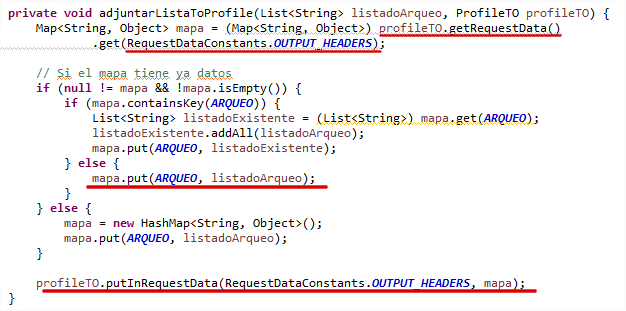
Se va a realizar un test que comprueba que cuando se obtiene las cabeceras del BCK y una de ellas contiene la key “**ARQUEO-**“, su valor (cadena con todos los datos) se guardará en una lista, que posteriormente servirá para añadir a la lista que se obtendrá del mapa que proviene del profile para la key “**ARQUEO**”.

**TEST**





**CLASE TRAZA:**



En esta imagen se ve el método encargado de guardar en el profile la lista de valores obtenido de las cabeceras. Este método reciba la lista de valores y el profile.

Como vemos, primero se obtiene el mapa del profile para la key “outputHeaders”. Si este trae datos, comprobamos si ya contiene alguna key de “**ARQUEO**”. En este caso, el mapa tendrá datos, pero no con la key de “ARQUEO”, por lo que simplemente asignaremos la lista de valores a la key “ARQUEO” y lo setearemos en el mapa, el cual posteriormente se guardará en el profile con la key “outputHeaders”.

## Test en BCK

## Clase PrepareArqueoInfoHandler

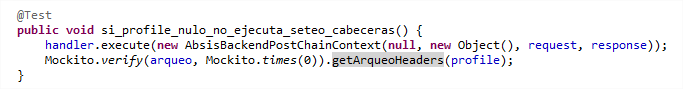
En BCK, hemos tenido que implementar, después de conocer los datos del SN que se está tratando, ejecutaremos la clase PrepareArqueoInfoHandler. En esta clase veremos si el método del SN que se está tratando es arqueable y si la canalidad es 001 (Canal oficina), si esto se cumple, setearemos en las cabeceras la lista de datos de objetos de ArqueoTO que se han guardado en el RequestData para la key “arqueo-current-backend-data”.

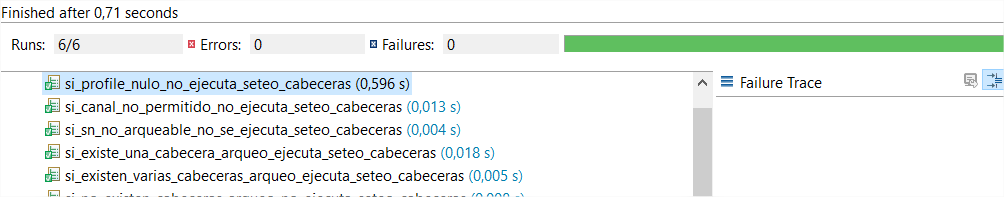
## Si el profile viene nulo no se ejecuta el seteo de cabeceras

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Id | Historia | Criterio | Caso de prueba | Descripción | Estado |
| 4 | F-002 | 1 | Si el profile viene nulo no se ejecuta el seteo de cabeceras | Se verifica que si el profile viene nulo las cabeceras no serán ejecutadas | **OK** |

Se va a realizar un test que comprueba que cuando quiere setear a las cabeceras pero el profile viene nulo, no se ejecuta esta acción.

**TEST**



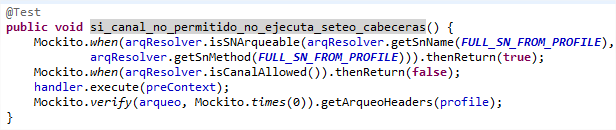


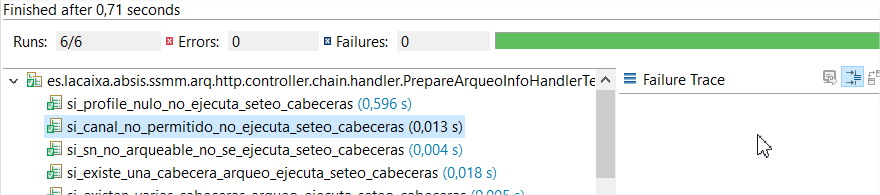
## Si el canal no tiene permitido ejecutar el seteo de cabeceras

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Id | Historia | Criterio | Caso de prueba | Descripción | Estado |
| 5 | F-002 | 1 | Si el canal no tiene permitido ejecutar el seteo de cabeceras | Se verifica que si el canal no está permitido el seteo de cabeceras. | **OK** |

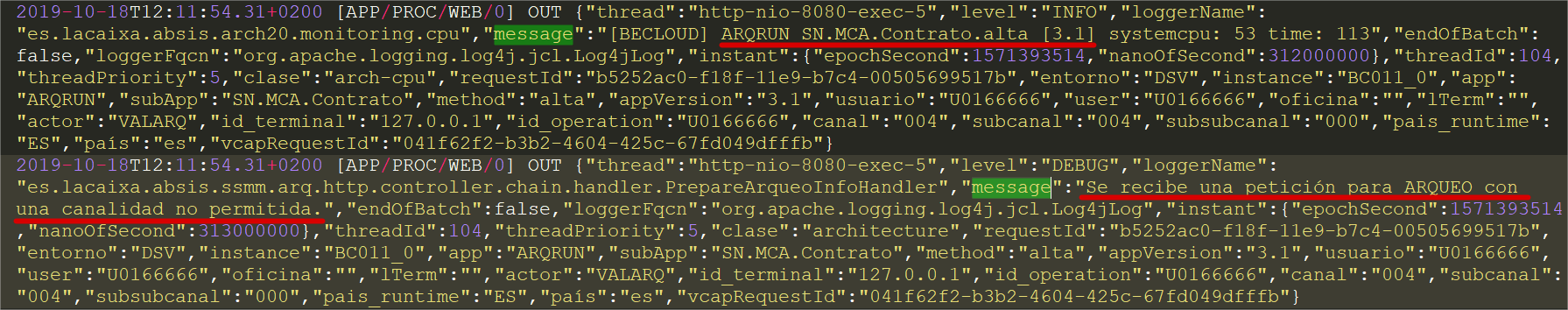
Se va a realizar un test que comprueba que cuando quiere setear a las cabeceras pero el canal no tiene permitido usar la funcionalidad de ARQUEO, no se ejecuta esta acción.

**TEST**





En los logs aparece de la siguiente forma:

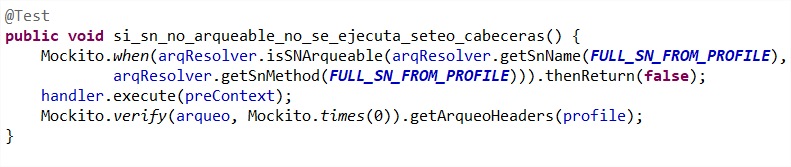


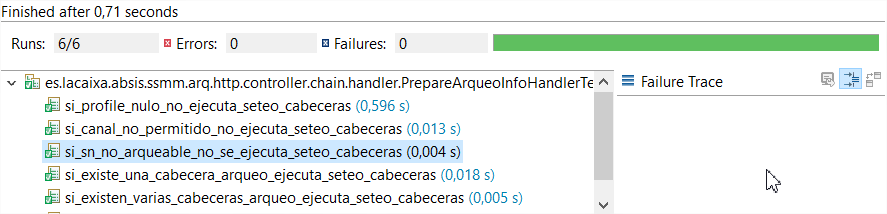
## Si el SN no es arqueable no se ejecuta el seteo de cabeceras

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Id | Historia | Criterio | Caso de prueba | Descripción | Estado |
| 6 | F-002 | 1 | Si el SN no es arqueable no se ejecuta el seteo de cabeceras | Se verifica que si el SN no es arqueable el seteo de cabeceras no se ejecuta. | **OK** |

Se va a realizar un test que comprueba que cuando se quiere setear a las cabeceras pero el SN no es arqueable, no se ejecuta esta acción.

**TEST**



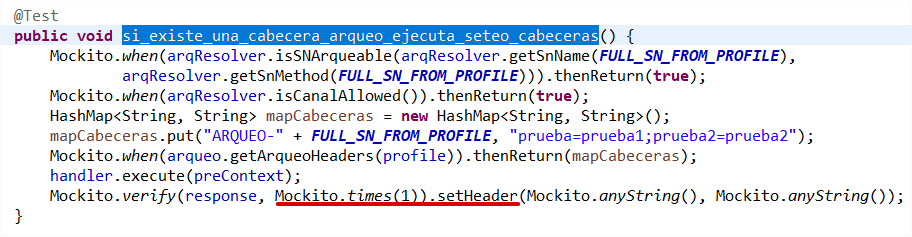


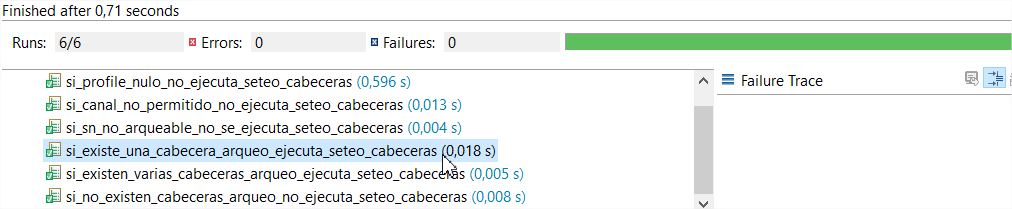
## Si el SN es arqueable y tiene un dato de arqueo, se ejecuta el seteo de cabeceras

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Id | Historia | Criterio | Caso de prueba | Descripción | Estado |
| 7 | F-002 | 1 | Si el SN es arqueable y tiene un dato de arqueo, se ejecuta el seteo de cabeceras | Se verifica que si el SN es arqueable y tiene datos, el seteo de cabeceras se ejecuta. | **OK** |

Se va a realizar un test que comprueba que cuando el SN tiene un dato de arqueo, se ejecuta la acción del seteo de cabeceras.

**TEST**



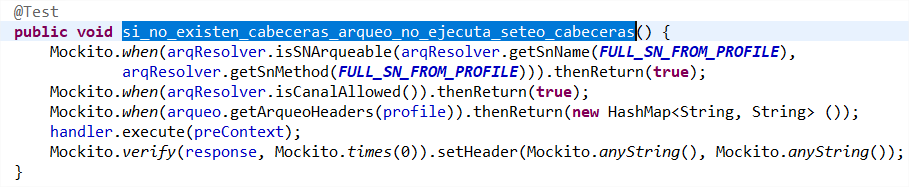


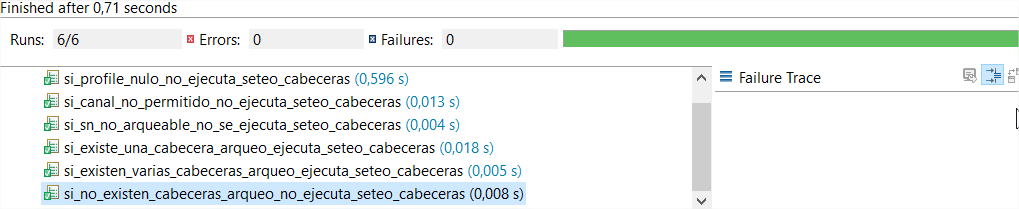
## Si el SN es arqueable pero no tiene datos de arqueo, no se ejecuta el seteo de cabeceras

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Id | Historia | Criterio | Caso de prueba | Descripción | Estado |
| 8 | F-002 | 1 | Si el SN es arqueable pero no tiene datos de arqueo, no se ejecuta el seteo de cabeceras | Se verifica que si el SN es arqueable y no tiene datos, el seteo de cabeceras no se ejecuta. | **OK** |

Se va a realizar un test que comprueba que cuando el SN no tiene datos de arqueo, no se ejecuta la acción que setea las cabeceras.

**TEST**



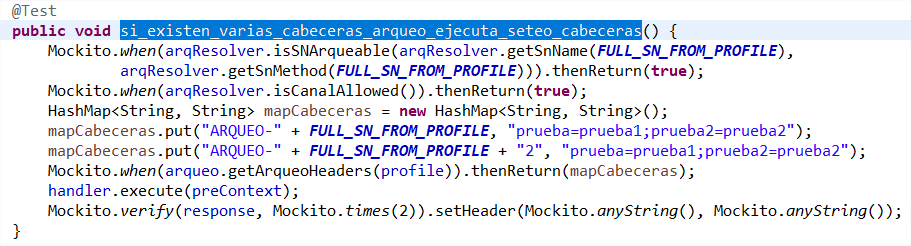


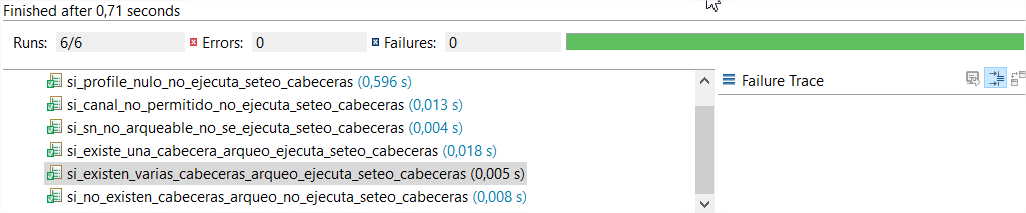
## Si el SN es arqueable y tiene varios datos de arqueo, se ejecuta el seteo de cabeceras

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Id | Historia | Criterio | Caso de prueba | Descripción | Estado |
| 9 | F-002 | 1 | Si el SN es arqueable y tiene varios datos de arqueo, se ejecuta el seteo de cabeceras | Se verifica que si el SN es arqueable y tiene varios datos, el seteo de cabeceras se ejecuta. | **OK** |

Se va a realizar un test que comprueba que cuando el SN tiene varios datos de arqueo, se ejecuta la acción del seteo de cabeceras.

**TEST**





## Clase ArqueoDataImpl

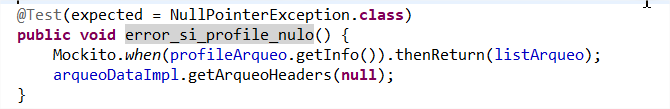
Una vez se haya validado que el SN es arqueable y que contiene datos referentes a arqueo, convertiremos la lista de objetos de Arqueo que viene en el profile para poder añadirlo a las cabeceras. Se va a mostrar aquellos test de JUnit que se han creado para demostrar la estabilidad y buen funcionamiento del desarrollo.

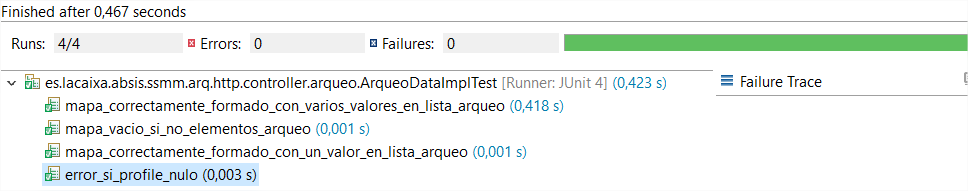
## Si el profile es nulo se debe devolver un NullPointerException

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Id | Historia | Criterio | Caso de prueba | Descripción | Estado |
| 10 | F-002 | 1 | Si el profile es nulo se debe devolver un NullPointerException | Se verifica que si el profile es nulo se debe devolver un NullPointerException. | **ERROR** |

Se va a realizar un test que comprueba que cuando el profile es nulo se debe devolver un NullPointerException y parar la preparación de los datos del SN arqueable.

**TEST**



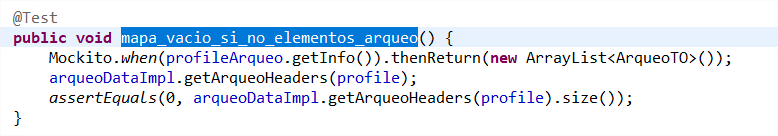


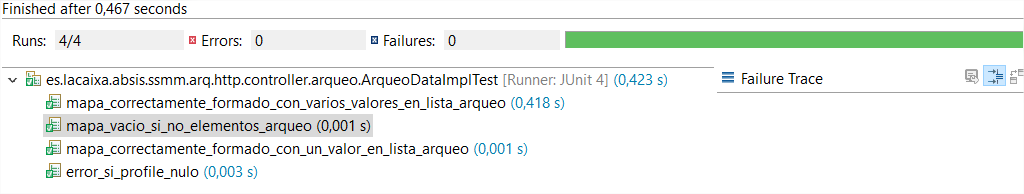
## Si el profile contiene un mapa pero este está vacío, no se pasará ningún dato a las cabeceras

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Id | Historia | Criterio | Caso de prueba | Descripción | Estado |
| 11 | F-002 | 1 | Si el profile contiene un mapa pero este está vacío, no se pasará ningún dato a las cabeceras | Se verifica que Si el profile contiene un mapa pero este está vacío, no se pasará ningún dato a las cabeceras. | **OK** |

Se va a realizar un test que comprueba que cuando el profile no es nulo pero no contiene un mapa de datos de SN arqueables, no se pasará ningún dato a las cabeceras.

**TEST**



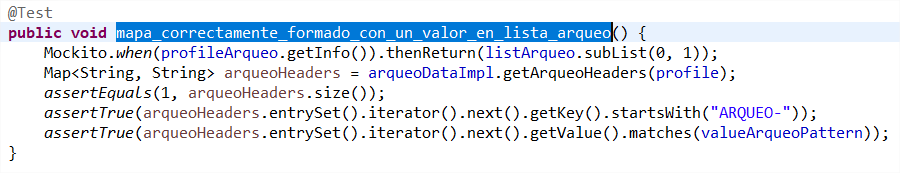


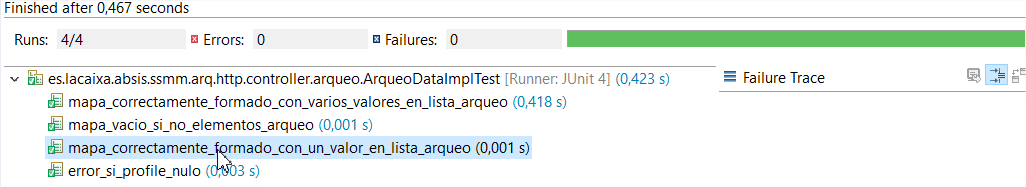
## Si el profile contiene un mapa con un único elemento, la cabecera contendrá ese único valor

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Id | Historia | Criterio | Caso de prueba | Descripción | Estado |
| 12 | F-002 | 1 | Si el profile contiene un mapa con un único elemento, la cabecera contra ese único valor | Se verifica que si el profile contiene un mapa con un único elemento, la cabecera contra ese único valor. | **OK** |

Se va a realizar un test que comprueba que cuando el profile no es nulo y contiene un mapa de datos de SN arqueables, en la cabecera existirá esos datos tratados de forma correcta. La key estará compuesta por el nombre del SN con un prefijo “ARQUEO-“ y los valores para ese key será una concatenación con los campos del SN arqueable.

**TEST**



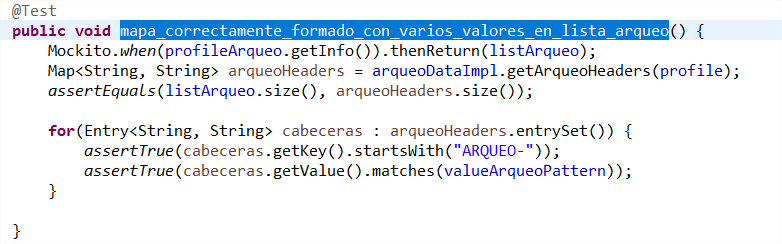


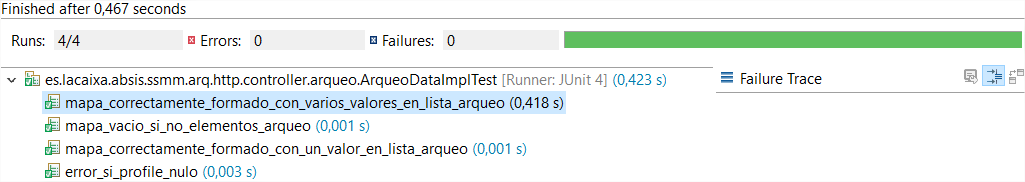
## Si el profile contiene un mapa con varios elementos, la cabecera contendrá todos los valores

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Id | Historia | Criterio | Caso de prueba | Descripción | Estado |
| 13 | F-002 | 1 | Si el profile contiene un mapa con varios elementos, la cabecera contendrá todos los valores | Se verifica que si el profile contiene un mapa con varios elementos, la cabecera contendrá todos los valores. | **OK** |

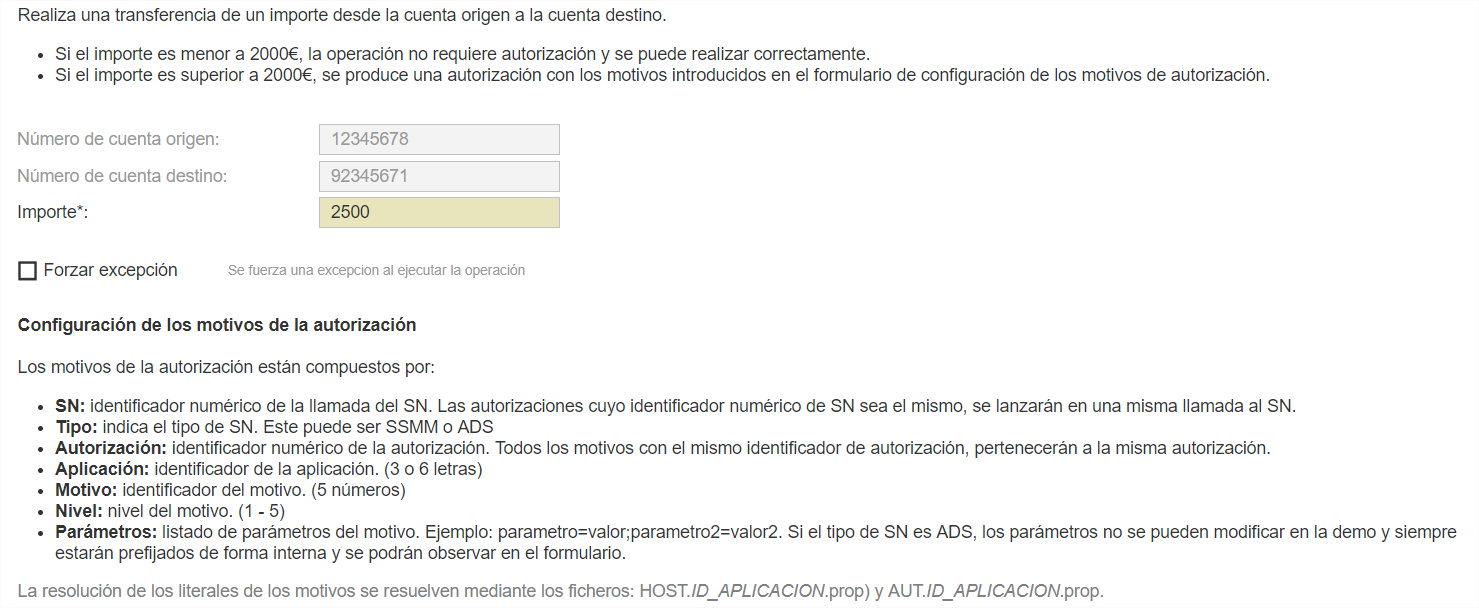
Se va a realizar un test que comprueba que cuando el profile no es nulo y contiene un mapa de datos de SN arqueables, en la cabecera existirá esos datos tratados de forma correcta. La key estará compuesta por el nombre del SN con un prefijo “ARQUEO-“ y los valores para ese key será una concatenación con los campos del SN arqueable.

**TEST**

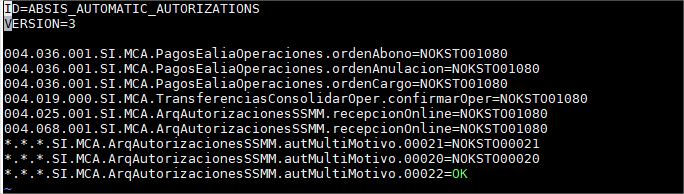




# Tests Manuales

Como se indicaba en la descripción, vamos a probar las respuestas dadas para los motivos de una autorización. Estas respuestas poder ser automáticas (OK o NOK) y no automáticas. Para ello simularemos la transferencia de un importe superior a 2000€, que requiere autorización.

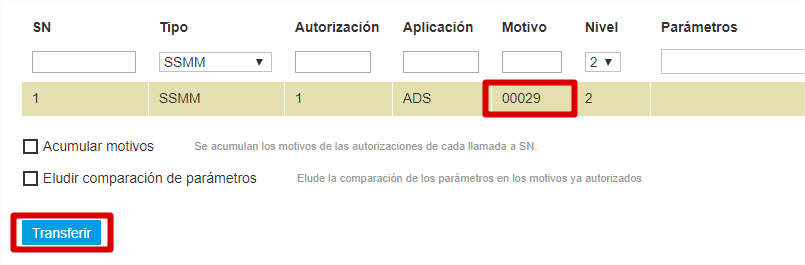
Previamente, el archivo de autorizaciones habrá sido cargado y podremos tener acceso desde BCK a las autorizaciones.



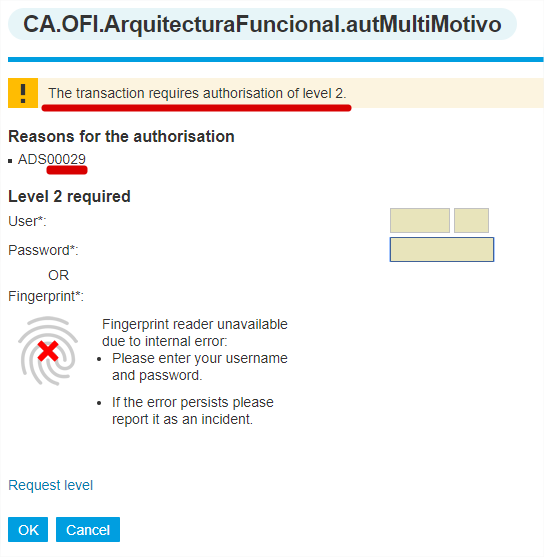
## Autorización sin motivo automático

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Id | Historia | Criterio | Caso de prueba | Descripción | Estado |
| 14 | F-003 | 1 | Verificación del filtrado por un motivo que no es automático. | Se verifica que las autorizaciones con un motivo no automático funcionan correctamente. | **OK** |

Probamos con una autorización cuyo motivo es NO automático.



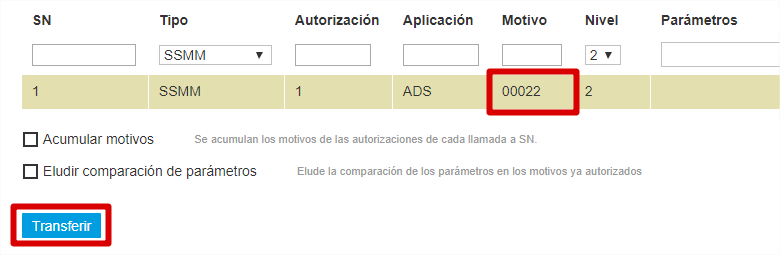
Nos pide una segunda autorización de forma manual por parte de un empleado con nivel 2.



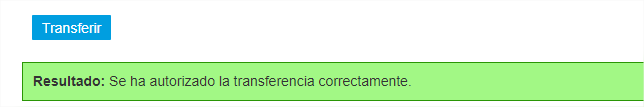
## Autorización con un motivo OK

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Id | Historia | Criterio | Caso de prueba | Descripción | Estado |
| 15 | F-003 | 1 | Verificación del filtrado por un motivo que es automático. | Se verifica que las autorizaciones con un motivo automático (OK) funcionan correctamente. | **OK** |

Probamos con una autorización cuyo motivo es AUTOMÁTICO



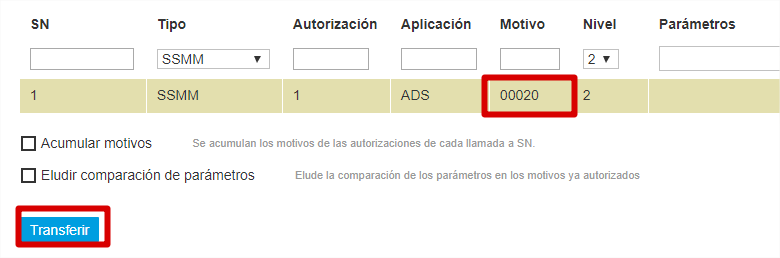
Se nos devuelve la siguiente respuesta:



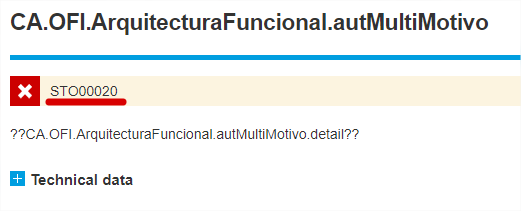
## Autorización con un motivo NOK

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Id | Historia | Criterio | Caso de prueba | Descripción | Estado |
| 16 | F-003 | 1 | Verificación del filtrado por un motivo que es automático pero NOK. | Se verifica que las autorizaciones con un motivo automático pero NOK funcionan correctamente. | **OK** |

Probamos con una autorización cuyo motivo es AUTOMÁTICO pero es un NOK.



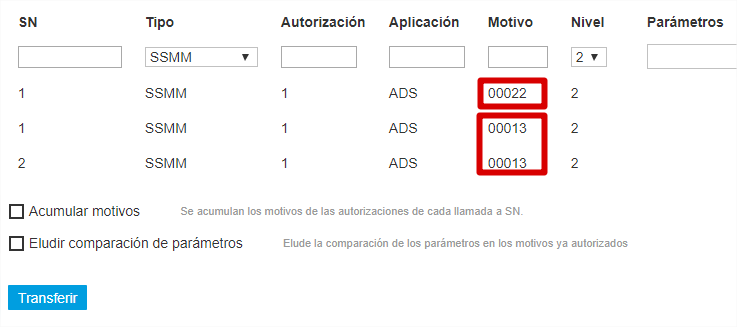
Devolviendo un error porque la transferencia no es posible para esa autorización



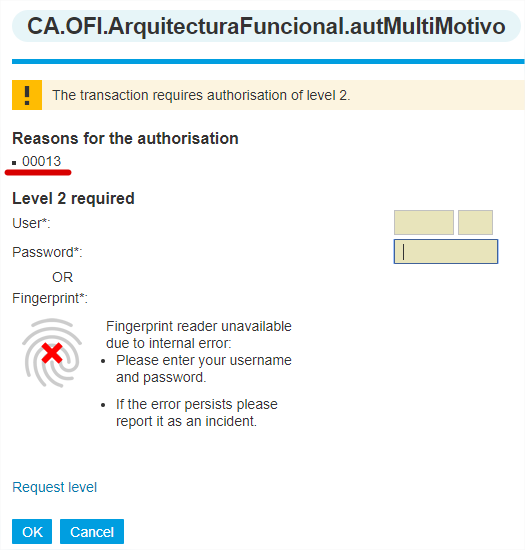
## Autorización con un motivo OK y dos no automático

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Id | Historia | Criterio | Caso de prueba | Descripción | Estado |
| 17 | F-003 | 1 | Verificación del filtrado por un motivo que es automático OK y dos no automáticos. | Se verifica que las autorizaciones con un motivo automático OK y dos no automáticos funcionan correctamente. | **OK** |

Uno de los motivos es AUTOMÁTICO y con respuesta OK (00022) y otros dos, son NO automáticos (00013).



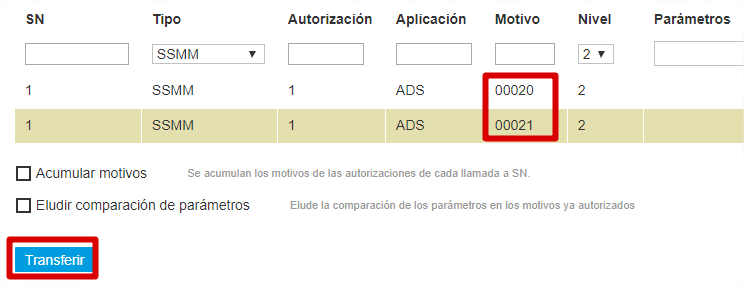
Como al menos existe uno que es NO automático, se va a solicitar que la autorización sea autorizada por un empleado con un nivel 2.



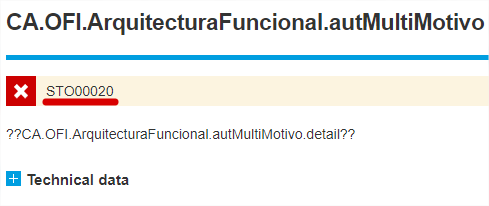
## Autorización con dos motivos NOK

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Id | Historia | Criterio | Caso de prueba | Descripción | Estado |
| 18 | F-003 | 1 | Verificación del filtrado de dos motivos NOK. | Se verifica que las autorizaciones con dos motivos automáticos NOK funcionan correctamente. | **OK** |

Probamos con una autorización que tiene dos motivos AUTOMÁTICOS y ambos son NOK.



La respuesta nos indica que la transferencia no se puede realizar para la primera de la autorización NOK que comprueba.



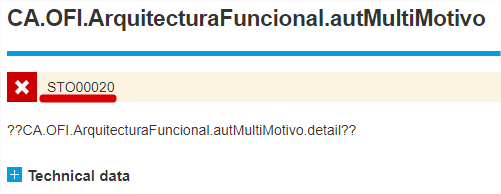
## Autorización con un motivo OK y uno NOK

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Id | Historia | Criterio | Caso de prueba | Descripción | Estado |
| 19 | F-003 | 1 | Verificación del filtrado de un motivo OK y otro NOK | Se verifica que las autorizaciones con un motivo automático OK y otro NOK funcionan correctamente. | **OK** |

Se indica dos motivos, ambos automáticos pero un motivo OK (00021) y otro NOK (00022)



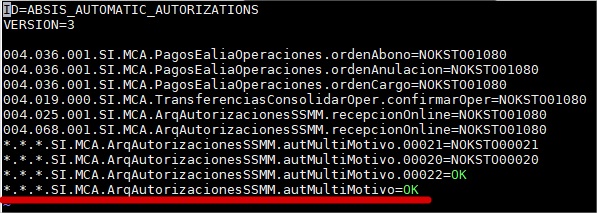
La respuesta indica que no se puede realizar la transferencia ya que uno de los motivos es NOK.



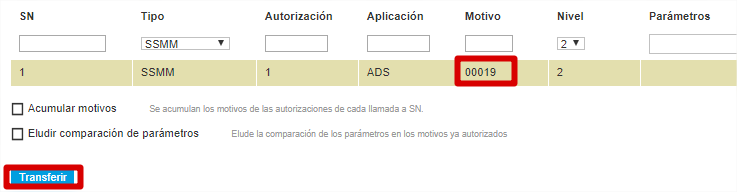
## Autorización con un motivo no automático, pero por defecto es Automático

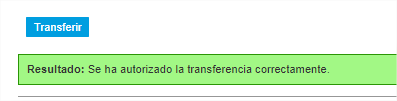
|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Id | Historia | Criterio | Caso de prueba | Descripción | Estado |
| 20 | F-003 | 1 | Verificación del filtrado de un motivo inexistente pero por defecto es OK | Se verifica que una autorización que no es automática pero su servicio si lo es, devolverá un OK. | **OK** |

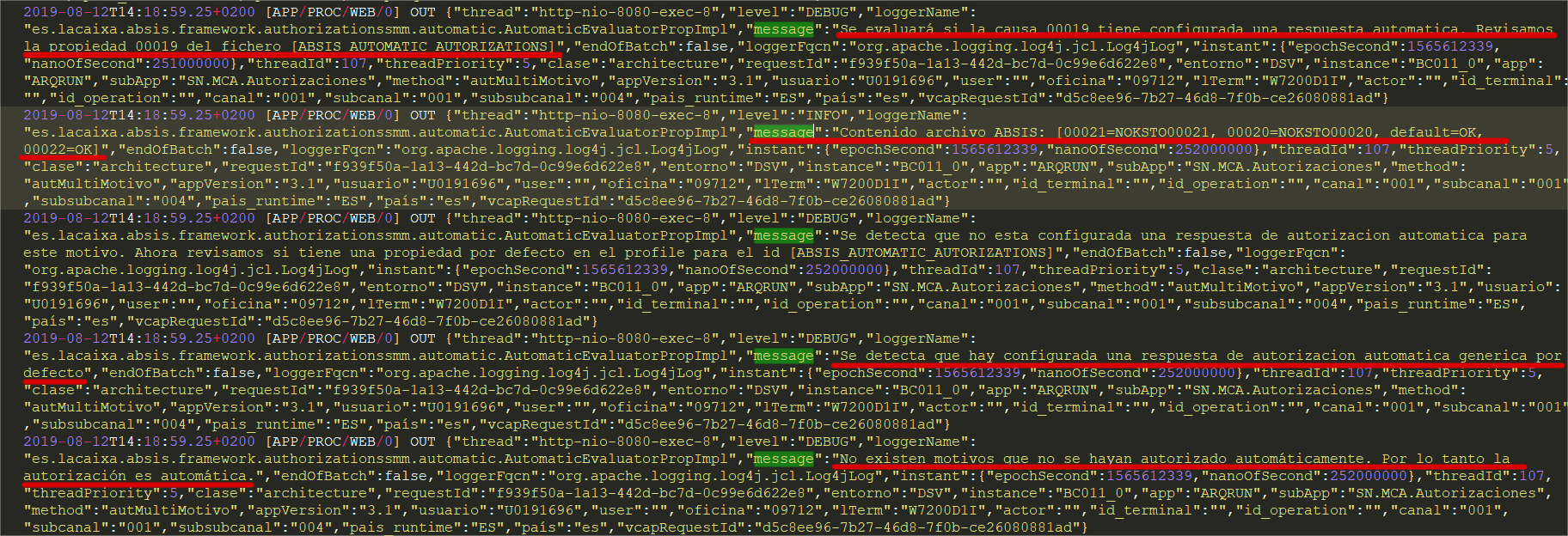
Se indica un motivo que no es automático pero por defecto, el servicio si es automático, por lo que no requiere de autorización de segundo nivel y la autorización es automática y OK.



Como se ve en la imagen, hemos añadido una nueva autorización para en caso de que no se encuentre la autorización entre los motivos dados, se compruebe la de por defecto y según su valor, devolver una respuesta. En este caso, indicado un OK para la autorización automática.





El log en consola nos muestra lo siguiente: