Testing Report

C1.017

https://github.com/vicgrabru/Acme-SF-D04

Carlos García Ortiz
cargarort3@alum.us.es

Tabla de contenidos

Resumen ejecutivo

Las pruebas de un sistema software son esenciales para garantizar el funcionamiento deseado del producto, permitiendo gran variedad de beneficios como el descubrimiento temprano de bugs, el análisis y comprensión del código, y muchas otras ventajas.

Sin embargo, un testing de calidad implica un procedimiento riguroso comprobando hasta el último recorrido del código a probar mediante un alto coverage, pues sino es así es posible que queden escondidos algunos fallos entre las sombras del sistema.

Tabla de revisiones

Revisión	Fecha	Descripción
1.0	27-05-2024	Primera versión del documento
2.0	27-05-2024	Versión final del documento

Introducción

En este documento se ofrece una visión detallada del proceso de elaboración del conjunto de pruebas creadas por el estudiante 2 para las diferentes funcionalidades y características que ofrece el sistema al rol cliente respecto a las entidades Contract y Progress Log.

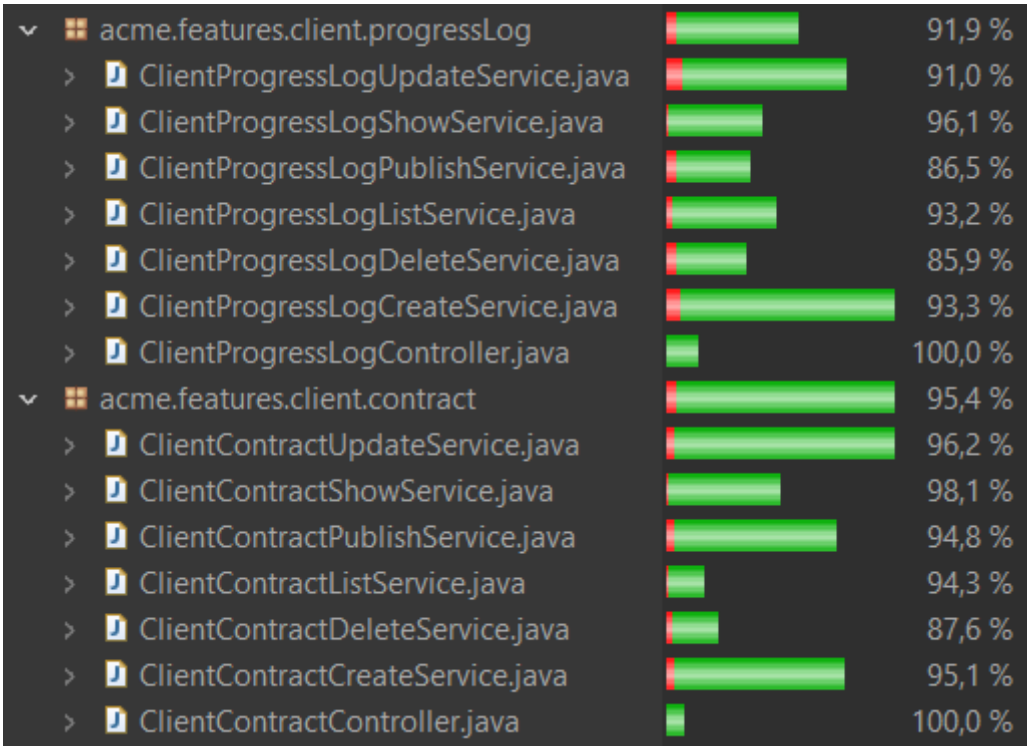
Además se incluye un análisis de las pruebas de rendimiento en diferentes ordenadores para comprobar el desempeño del sistema en dichas pruebas.

Añadir que el conjunto de pruebas y los datos de muestra necesarios para estas se encuentra en esta rama del repositorio online:

https://github.com/vicgrabru/Acme-SF-D04/tree/individual/cargarort3

Contenido

En primer lugar se muestra el coverage obtenido al reproducir todas las pruebas, puede observarse que todos los servicios superan el 80% sin problemas. De hecho, las líneas marcadas como “amarillas” no muestran ningún tipo de problema ya que la mayoría son “assert object != null;” y algunos cálculos del status para la comprobación de autorización que irremediablemente muestran esta cobertura.



Casos de pruebas positivos, negativos y hackeos para Contract

List.safe

Con client1 accedemos a su lista de contratos y observamos que todo está bien. Puntualizar que para que se haga todo el coverage se observa una lista que tenga contratos publicados y no publicados ya que muestran diferentes símbolos.

List.hack

Sin iniciar sesión entramos en localhost:8082/Acme-SF-D04/client/contract/list para ver que no estamos autorizados.

Show.safe

Con client1 visualizamos el contrato A-000 (id 542) y luego el BP-000 (id 579), publicado y no publicado respectivamente. Esto es debido a que si está en modo borrador debe mostrar un campo adicional indicando cuánto dinero del coste del contrato queda disponible.

Show.hack

Sin iniciar sesión y con client2, entramos en *localhost:8082/Acme-SF-D04/client/contract/show?id=542* para ver que no estamos autorizados **(por ser contrato de client1)**.

Sin iniciar sesión y con client2, entramos en *localhost:8082/Acme-SF-D04/client/contract/show?id=2000* para ver que no estamos autorizados **(por ser contrato inexistente)**.

Delete.safe

Con client1 borramos los contratos BP-000 (id 579), BP-001 (id 580) y BP-002 (id 581), los cuales tienen 0, 1 y 2 progress logs hijos respectivamente. Obviamente todos estos contratos están en borrador.

Delete.hack

Sin iniciar sesión entramos en localhost:8082/Acme-SF-D04/client/contract/delete para ver que no estamos autorizados.

Con client2, visualizamos cualquier contrato en borrador e intentamos eliminarlo modificando la id a 579 con inspeccionar elemento, esperando no estar autorizados **(por ser contrato de client1 sin publicar)**. Se replica con el contrato de id 542 **(de client1 publicado)** y con id 2000 **(por contrato inexistente)**.

Finalmente con client1, visualizamos cualquier contrato en borrador e intentamos eliminar modificando la id a 542 con inspeccionar elemento esperando no estar autorizados **(por contrato de client1 publicado)**.

Publish.safe

Con client1 intentamos publicar el contrato BP-000 (id 579), esto falla ya que el presupuesto de este contrato hará que se supere el coste del proyecto con la suma de los presupuestos de sus contratos publicados.

Tras esto intentamos publicar el contrato BP-001 (id 580), funcionará ya que no se supera el coste.

Finalmente visualizamos e intentamos publicar el contrato BP-002 (id 581), esto resulta bien ya que se llegará justo al límite del coste del proyecto sin superarlo. Obviamente todos estos contratos están en borrador.

Publish.hack

Sin iniciar sesión y con client2, entramos en *localhost:8082/Acme-SF-D04/client/contract/publish?id=579* para ver que no estamos autorizados **(por contrato de client1 sin publicar)**.

Sin iniciar sesión y con tanto client1 como client2, entramos en *localhost:8082/Acme-SF-D04/client/contract/publish?id=542* para ver que no estamos autorizados **(por contrato de client1 publicado)**.

Sin iniciar sesión y con client2, entramos en *localhost:8082/Acme-SF-D04/client/contract/publish?id=2000* para ver que no estamos autorizados. **(por contrato inexistente)**

Create.safe

Con client1 intentamos crear un contrato sin datos para que nos de fallo en los campos no opcionales.

Luego rellenamos los campos con valores correctos y vamos atributo por atributo comprobando sus casos negativos:

- **code:** Sin respetar patrón (**AAAA-0000**), código repetido (**A-000**).

- **goals/providerName/CustomerName:** Por encima del tamaño máximo (***texto de tamaño 101/76/76***), con spam (**sex sex sex**).

- **budget:** Por debajo del valor mínimo (**EUR -0.01**), por encima del valor máximo de su proyecto (**EUR 300**, ya que el proyecto seleccionado tiene EUR 299.99), por encima del valor máximo para Money (**EUR 10,000,000,000**), divisa no aceptada por el sistema (**JPY 5**).

Una vez comprobados los casos negativos creamos un contrato con datos válidos intermedios: **(FG-456, Goals, EUR 5, Provider, Customer, العظمى واعتلاء (proyecto id 295))**

Ahora crearemos múltiples contratos en los que se comprueben todos estos datos de casos positivos:

- **code:** Varios casos: **Z-000, AA-567, AAA-999, ZZZ-000, ZZ-123, AZ-456, ZA-789, ABC-987**

- **goals/providerName/customerName:** Mínimo tamaño (**G/P/C**), mínimo tamaño + 1 (**Go/Pr/Cu**), máximo tamaño (***texto de tamaño 100/75/75***), mínimo tamaño (***texto de tamaño 99/74/74***), carácteres extranjeros (**العظمى واعتلاء** y 국민경제의 발전을), intento de injection para que lo tome como un simple string (**<h1>!</h1**> y **' or 'A'='A**).

- **budget:** Valor mínimo (**EUR 0**), valor mínimo + 1 (**EUR 0.01**), valor máximo respecto al proyecto (**EUR 299.99**, ya que el proyecto seleccionado tiene dicho límite), valor máximo respecto al proyecto - 1 (**EUR 299.98**), valor máximo (**EUR 9999999999.99**, usando un proyecto con coste en el límite), valor máximo - 1 (**EUR 9999999999.98**, usando un proyecto con coste en el límite), otra divisa aceptada por el sistema (**USD 0**).

Create.hack

Sin iniciar sesión entramos en *localhost:8082/Acme-SF-D04/client/contract/contract/create* para ver que no estamos autorizados.

Con client1, intento crear un contrato introduciendo unos datos aceptados y alteramos el instantiationMoment y el draftMode mediante inspeccionar elemento. Luego visualizamos el contrato creado y comprobamos que esos campos no se han modificado por el hack ya que al ser atributos no modificables en el create no se les hace bind.

Update.safe

Con el cliente1 se comprueba exactamente lo mismo que en create.safe editando el contrato BP-000 (id 579), aunque en este caso no hay que comprobar code porque no puede modificarse una vez asignado.

Para comprobar la edición de budget máximo absoluto y su justo inferior usamos el client2 ya que su contrato A-031 (id 575) pertenece a un proyecto con esa cantidad de coste.

Update.hack

Sin iniciar sesión y con client2, entramos en *localhost:8082/Acme-SF-D04/client/contract/update?id=579* para ver que no estamos autorizados **(por contrato de client1 sin publicar)**.

Sin iniciar sesión, con client2 y client1, entramos en *localhost:8082/Acme-SF-D04/client/contract/update?id=542* para ver que no estamos autorizados **(por contrato de client1 publicado)**.

Sin iniciar sesión y con client2, entramos en *localhost:8082/Acme-SF-D04/client/contract/update?id=2000* para ver que no estamos autorizados **(por contrato inexistente)**.

Con client1 visualizamos cualquier contrato en borrador e intentamos actualizarlo alteramos el instantiationMoment y el draftMode mediante inspeccionar elemento. Luego visualizamos el contrato creado y comprobamos que esos campos no se han modificado por el hack.

Casos de pruebas positivos, negativos y hackeos para Progress Log

List.safe

Con client1 entramos en el listado de progress logs del contrato A-000 (id 542) y observamos que todo está bien. Puntualizar que para que se haga todo el coverage se observa una lista que tenga progress logs publicados y no publicados ya que muestran diferentes símbolos.

List.hack

Sin iniciar sesión y con client2, entramos en *localhost:8082/Acme-SF-D04/client/progress-log/list?contractId=542* para ver que no estamos autorizados **(por contrato de client1)**.

Sin iniciar sesión y con client2 entramos en *localhost:8082/Acme-SF-D04/client/progress-log/list?contractId=2000* para ver que no estamos autorizados **(por contrato inexistente)**.

show.safe

Con client1 visualizamos el progress log PG-A-0000 (id 776) y luego el PG-BP-0000 (id 800), que respectivamente está publicado y sin publicar. Ambos se han podido encontrar a través de la lista de progress logs de A-000 (id 542).

show.hack

Sin iniciar sesión y con client2 entramos en *localhost:8082/Acme-SF-D04/client/progress-log/show?id=776* para ver que no estamos autorizados **(por progress log de contrato de client1)**.

Sin iniciar sesión y con client2 entramos en *localhost:8082/Acme-SF-D04/client/contract/show?id=2000* para ver que no estamos autorizados **(por progress log inexistente)**.

delete.safe

Con client1 eliminamos el progress log PG-BP-0000 (id 800) perteneciente al contrato A-000 (id 542). Obviamente dicho progress log está en borrador.

delete.hack

Sin iniciar sesión entramos en *localhost:8082/Acme-SF-D04/client/progress-log/delete* para ver que no estamos autorizados.

Con client2 creamos un progress log en cualquier contrato y lo intentamos eliminar alterando la id a 800 con inspeccionar elemento para ver que no estamos autorizados **(por progress log de contrato de client1 sin publicar)**. Repetir con id 776 **(por progress log de contrato de client1 publicado)** e id 2000 **(contrato inexistente).**

Con client1 repetimos intentando eliminar el progress log de id 776.

publish.safe

Con client1 publicamos el progress log PG-BP-0000 (id 800) perteneciente al contrato A-000 (id 542). Obviamente dicho progress log está en borrador.

publish.hack

Sin iniciar sesión entramos en *localhost:8082/Acme-SF-D04/client/progress-log/publish* para ver que no estamos autorizados.

Iniciamos sesión en client2 e intentamos publicar un progress log en borrador alterando la id a 800 con inspeccionar elemento para ver que no estamos autorizados **(por progress log de contrato de cliente1 sin publicar)**. Repetimos para id 776 **(progress log de contrato de cliente1 publicado)** y para id 2000 **(contrato inexistente)** esperando no estar autorizados.

Repetimos con client1 alterando la id a 776 esperando no estar autorizados **(por progress log de contrato de cliente1 publicado).**

create.safe

Con client1 intentamos crear un progress log sin datos para que nos de fallo en los campos no opcionales.

Luego rellenamos los campos con valores correctos y vamos atributo por atributo comprobando sus casos negativos:

- **recordId**: Sin respetar patrón (**AA-AAA-00000**), código repetido (**PG-A-0000**).

- **completeness:** Por debajo del valor mínimo (**0.00**), por encima del valor máximo (**100.01**), con más decimales (**50.555**).

- **comment/responsiblePerson:** Por encima del tamaño máximo (***texto de tamaño 101/76***), con spam (**sex sex sex**).

Una vez comprobados los casos negativos creamos un progess log con datos válidos intermedios: **(PG-FG-4567, 50.00, Comment, Responsible)**

Ahora crearemos múltiples progress logs comprobando todos estos casos positivos:

- **recordId:** Diferentes posibilidades: **PG-A-1111, PG-BB-2222, PG-ZZ-3333, PG-Z-4444, PG-AA-5555, PG-C-6666, PG-YY-7777, PG-AZ-9999**

- **completeness:** Valor mínimo (**0.01**), valor minimo + 1 (**0.02**), valor máximo (**100.0**), valor máximo - 1 (**99.99**).

- **comment/responsiblePerson:** Mínimo tamaño (**C/R**), mínimo tamaño + 1 (**Co/Re**), máximo tamaño (***texto de tamaño 100/75***), mínimo tamaño (***texto de tamaño 99/74***), carácteres extranjero (**العظمى واعتلاء** y 국민경제의 발전을), intento de injection para que lo tome como un simple string (**<h1>!</h1>** y **' or 'A'='A**).

create.hack

Sin iniciar sesión y con client2, entramos en *localhost:8082/Acme-SF-D04/client/progress-log/create?contractId=542* para ver que no estamos autorizados **(por contrato de client1)**. Repetimos para *localhost:8082/Acme-SF-D04/client/progress-log/create?contractId=2000* **(por contrato inexistente)**.

Con client1 intentamos crear un progress log para cualquier contrato alterando el registrationMoment y el draftMode mediante inspeccionar elemento. Luego visualizamos el progress log creado y comprobamos que esos campos no se han modificado por el hack ya que no deben de poder ser modificables.

update.safe

Con el cliente1 se comprueba exactamente lo mismo que en create.safe editando el progress log PG-BP-0000 (id 800), aunque en este caso no hay que comprobar recordId porque no puede modificarse una vez asignado.

update.hack

Sin iniciar sesión entrar en localhost:8082/Acme-SF-D04/client/progress-log/update para ver que no estamos autorizados.

Cin client2 intentamos actualizar un progress log alterando la id a 800 con inspeccionar elemento esperando no ser autorizados **(por progress log de contrato de cliente1 sin publicar)**. Repetimos para id 776 (por progress log de contrato de client1 publicado) y para id 2000 **(por progress log inexistente)**.

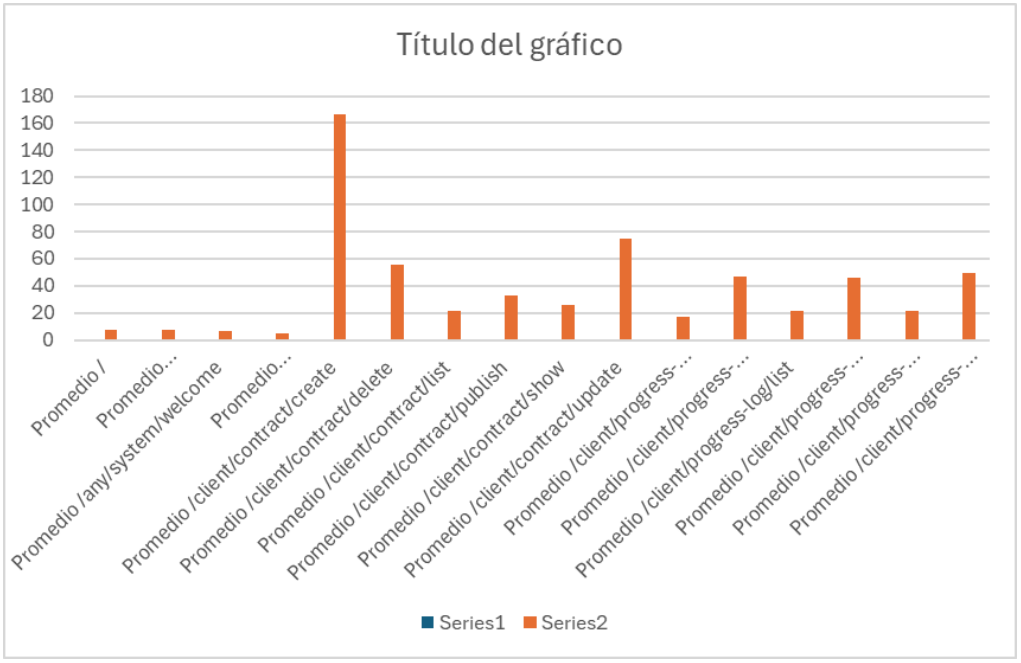
Con client1 intentamos actualizar un progress log alterando la id a 776 con inspeccionar elemento esperando no ser autorizados **(por contrato de client1 publicado)**.

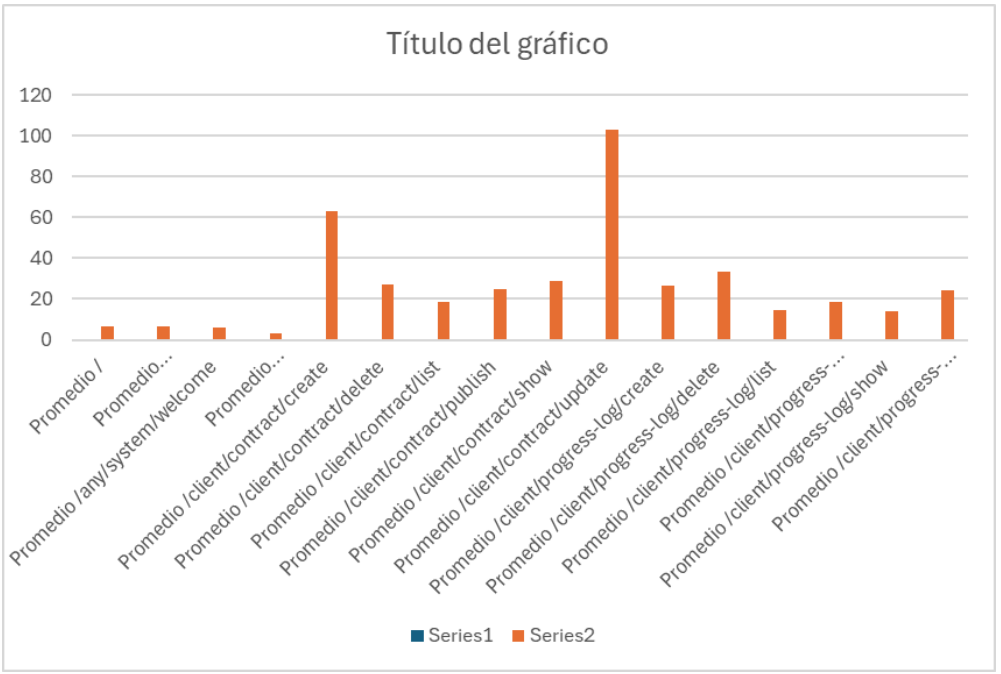
Con client1 intentamos actualizar cualquier progress log en borrador alterando el registrationMoment y el draftMode mediante inspeccionar elemento. Luego visualizamos el progress log actualizado y comprobamos que esos campos no se han modificado por el hack.

Resultados de pruebas de rendimiento

Para probar el rendimiento del sistema respecto a las pruebas se ha lanzado el replayer en 2 ordenadores personales distintos, aplicando en el segundo caso una pequeña optimización añadiendo índices.

Los gráficos de tiempo obtenidos en el primer y segundo ordenador son los siguientes:





A simple vista las gráficas parecen mostrar que hay una ligera mejora en la ejecución, pero aún queda mucho que probar. Observemos los intervalos de confianza del 95% para el tiempo de respuesta de los ordenadores:

Before				*After*		
Media	30,7129838			Media	21,8232832	
Error típico	6,23451242			Error típico	4,22984838	
Mediana	12,95495			Mediana	8,97815	
Moda	5,9326			Moda	3,7709	
Desviación estándar	145,412568			Desviación estándar	89,3289501	
Varianza de la muestra	21144,8149			Varianza de la muestra	7979,66132	
Curtosis	307,949726			Curtosis	386,475203	
Coeficiente de asimetría	16,6720127			Coeficiente de asimetría	19,0255891	
Rango	2920,229			Rango	1836,2601	
Mínimo	1,7495			Mínimo	2,0313	
Máximo	2921,9785			Máximo	1838,2914	
Suma	16707,8632			Suma	9733,1843	
Cuenta	544			Cuenta	446	
Nivel de confianza(95,0%)	12,2467171			Nivel de confianza(95,0%)	8,31295999	
Interval (ms)	18,4662668	42,9597009		Interval (ms)	13,5103232	30,1362432
Interval (s)	0,01846627	0,0429597		Interval (s)	0,01351032	0,03013624

Sigue pareciendo haber una mejora, dando la segunda ejecución una mejoría en el intervalo, así que por último observemos el contraste de hipótesis con un 95% de confianza para obtener una mejor comparación:

Prueba z para medias de dos muestras

	BEFORE	*AFTER*
Media	33,281987	21,8232832
Varianza (conocida)	21144,8149	7979,66132
Observaciones	446	446
Diferencia hipotética de las medias	0	
z	1,41799226	
P(Z<=z) una cola	0,07809651	
Valor crítico de z (una cola)	1,64485363	
Valor crítico de z (dos colas)	0,15619302	
Valor crítico de z (dos colas)	1,95996398	

Por desgracia, observando que el p-value (0.156) es superior a alfa (0.05), podemos deducir que no ha habido realmente una mejora tan significativa en la ejecución de las pruebas. Por ello el uso de los índices no parece haber sido suficiente.

Conclusión

En cuanto a los tests puedo decir que he intentado hacerlos los más rigurosos posibles alcanzando un coverage muy satisfactorio, aunque aún es posible que se me haya escapado algún que otro caso de uso o hackeo por comprobar.

En cuanto a la comparación se ha visto que los índices mejoraron un poco la ejecución pero no lo suficiente. Analizando bien los datos y los servicios he llegado a la conclusión que el mayor problema de rendimiento de estas pruebas es el constante manejo de tipo Money y el hecho de que haya que cambiarlo a la divisa del sistema cada vez que se sumen o comparen. Los tests se ejecutan con la caché de ratios de monedas vacía, así que esas llamadas a api externa serán responsables de ralentizar mucho el rendimiento y dar tests irregulares.

Una posible mejora para los tests sería usar emplear un mock para el cambio de divisa, ya que no nos habíamos preocupado de ello pues en los tests simplemente hemos usado otras divisas con cantidad 0 para que la varianza de ratios en función del día no afectase a la reproducción del test en cualquier otro momento.

Bibliografía

No aplica.