



PERFORMANCE REPORT

18 DE ABRIL DE 2022

E2.07

<https://github.com/antcamgil/Acme-Toolkits>

Jaime Borrego Conde: jaiborcon@alum.us.es

Antonio Campos Gil: antcamgil@alum.us.es

Ana Conde Marrón: anaconmar@alum.us.es

Gonzalo Martínez Fernández: gonmarfer2@alum.us.es

Jaime Moscoso Bernal: jaimosber@alum.us.es

Enrique Muñoz Pérez: enrmunper@alum.us.es

Tabla de contenido

Resumen ejecutivo	2
Tabla de revisión	3
Introducción	4
Primer Análisis.....	5
Segundo Análisis.....	9
Hipótesis de contraste	12
Conclusiones	13
Bibliografía	13

Resumen ejecutivo

El informe de rendimiento es un documento que recopila métricas relacionadas con este atributo de calidad para el proyecto *Acme Toolkits*. Estas métricas incluyen dos análisis, realizados en ordenadores diferentes, sobre el intervalo de confianza de tiempo real (*wall time*) que abarcan las peticiones, así como un contraste de hipótesis que clarifica qué ordenador es más eficiente dentro de dicho intervalo.

Tabla de revisión

<i>Número de revisión</i>	<i>Fecha</i>	<i>Descripción</i>
1	24/03/2022	Desarrollo del documento

Introducción

Los requisitos de rendimiento son requisitos no funcionales que establece el cliente para el funcionamiento de la aplicación, normalmente referidos al tiempo que tarda en ejecutar una petición, en mostrar vistas o en responder a las interacciones del usuario.

En el caso de *Acme-Toolkits*, deben realizarse dos análisis para el tiempo real medio de las peticiones del sistema, cada uno en un ordenador diferente, de manera que se clarifique cuál es el ordenador más eficiente dentro del intervalo de confianza del 95%. Para ello, se presenta el siguiente informe en el que se detalla el procedimiento seguido para obtener los resultados.

En primer lugar, se muestra el apartado referido a las pruebas de rendimiento del primer ordenador, es decir, la documentación del análisis estadístico llevado a cabo para reconocer el tiempo que tarda el sistema en servir las peticiones, así como el porqué de ese tiempo.

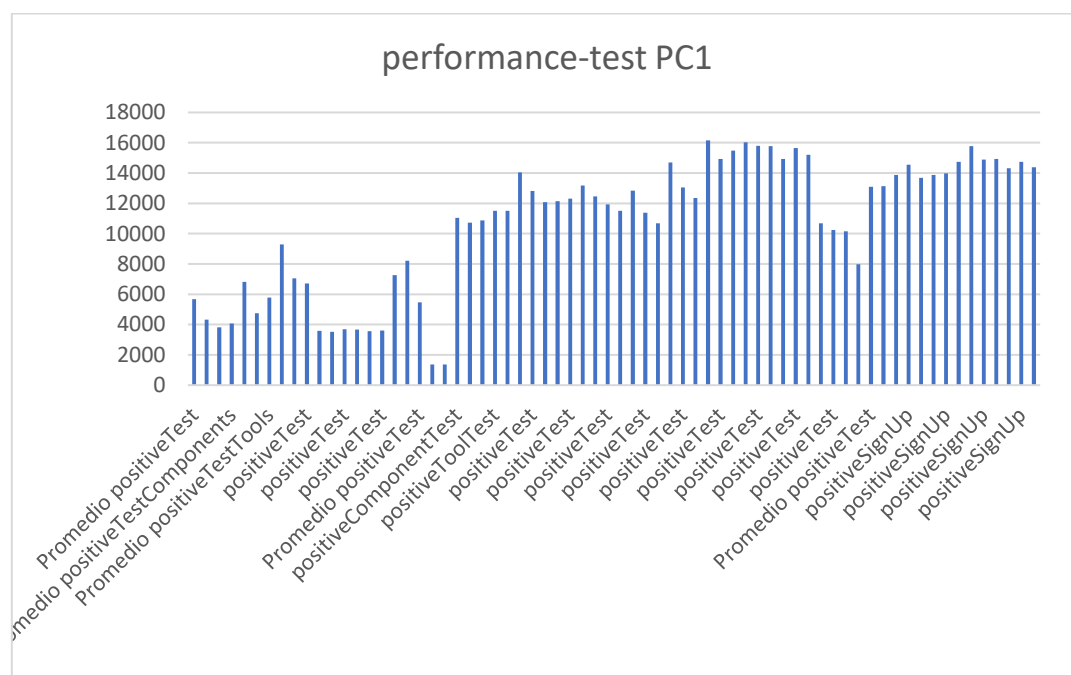
Posteriormente, se incluye el apartado del segundo análisis, que se emplea para realizar una hipótesis de contraste con el primer ordenador.

Finalmente, se ofrecen las conclusiones extraídas del informe que aportan el punto de cierre del documento.

Primer Análisis

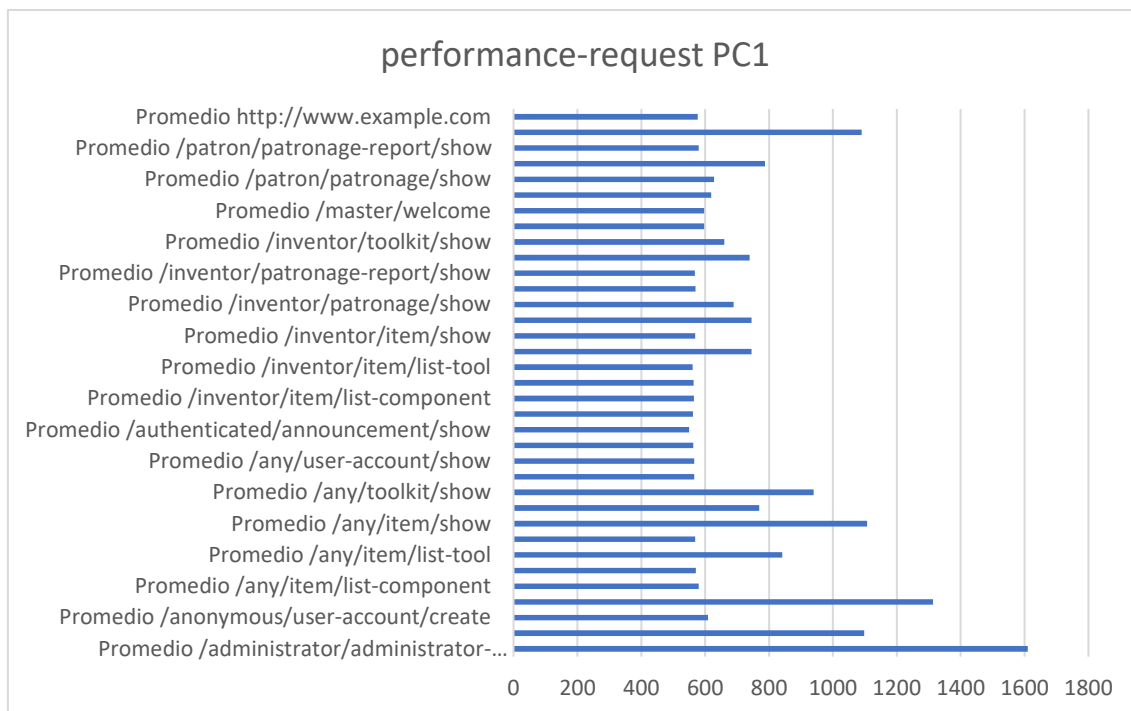
A la hora de evaluar el rendimiento de las solicitudes a la aplicación, la primera vez que se hizo el análisis fue en un equipo con un procesador Intel(R) Core(TM) i5-8265U CPU, 8 GB RAM y Windows 10 Home. Se pretendía hacer un primer análisis, tras este mejorar lo que permitiese llegar al nivel del 95% de confianza y volver a hacer un análisis. Pero, tras los siguientes resultados obtenidos se decidió que no era necesario refactorizar más la aplicación pues estaba dentro de los intervalos estipulados de rendimiento.

Estos fueron los datos arrojados por el primer análisis:



Gráfica 1. Performance-Test del PC1

Quizá por un mal nombramiento de los test la tabla no da valores concretos debido a que las etiquetas del eje X se repiten bastante, no obstante, el rendimiento obtenido en este análisis es correcto porque se adapta a los requisitos del cliente. Por tanto, la mayoría de los test tardarán entre 6 y 16 segundos.



Gráfica 2. Performance-Request del PC1

Tras la ejecución de las pruebas, se ha alcanzado un 81,9 % de cobertura.

Element	Covera...	Covered Ins...	Missed Instr...	Total Instruc...
> Acme-Toolkits	81,9 %	5.192	1.146	6.338

Imagen 1. Nivel De Cobertura

La mayoría de las peticiones se mantienen estables exceptuando algunos casos como los paneles (*DashBoard*), que necesitan un mayor tiempo de ejecución al necesitar realizar cálculos sobre varias entidades. También se ha encontrado un pico de tiempo en *Chirp*. Estos resultados no eran los esperados pues se trata de una de las clases más simples, aunque podría ser fruto de la ejecución en paralelo junto a otra tarea.

time	
Media	632,478892
Error típico	15,76279
Mediana	571
Moda	564
Desviación estándar	346,064149
Varianza de la muestra	119760,395
Curtosis	173,603216
Coeficiente de asimetría	11,4023472
Rango	6113
Mínimo	345
Máximo	6458
Suma	304854,826
Cuenta	482
Nivel de confianza(95,0%)	30,9724348
Intervalo	601,506457 663,451327

Tabla 1 Estadística descriptiva PC1

Como puede observarse en la tabla anterior, el nivel de confianza ha sido de 30,97 ms y el intervalo de confianza oscila entre 601.50 y 663.45, que está por debajo de los 1000 milisegundos requeridos. Por lo tanto, el resultado ha sido positivo y aceptable.

Así mismo, se ha realizado el *profiling* tanto del proyecto como del ordenador para encontrar cuellos de botella. Se han obtenido los siguientes resultados tras los análisis.

CPU samples Thread CPU time			
Results: View: Collected data: Thread Dump			
Name	Total Time	Total Time (CPU)	
main	754.835 ms (100%)	525.154 ms (100%)	
Reference Handler	754.835 ms (100%)	0,0 ms (-%)	
Finalizer	754.835 ms (100%)	7,84 ms (100%)	
mysql-cj-abandoned-connection-cleanup	754.835 ms (100%)	281 ms (100%)	
Exec Default Executor	754.835 ms (100%)	0,0 ms (-%)	
Exec Stream Pumper	754.835 ms (100%)	754.835 ms (100%)	
Exec Stream Pumper	754.835 ms (100%)	754.835 ms (100%)	
OkHttp ConnectionPool	754.835 ms (100%)	0,0 ms (-%)	
RMI TCP Accept-0	754.835 ms (100%)	0,0 ms (-%)	
RMI TCP Connection(1)-192.168.56.1	754.835 ms (100%)	754.835 ms (100%)	
RMI Scheduler(0)	754.835 ms (100%)	0,0 ms (-%)	
JMX server connection timeout 47	754.835 ms (100%)	0,0 ms (-%)	
RMI TCP Connection(2)-192.168.56.1	754.835 ms (100%)	754.835 ms (100%)	
Okio Watchdog	754.742 ms (100%)	0,0 ms (-%)	
Catalina-utility-1	751.533 ms (100%)	0,0 ms (-%)	
Catalina-utility-2	751.533 ms (100%)	343 ms (100%)	
Name	Self Time (CPU)	Total Time (CPU)	
java.lang.Thread.run ()	0,0 ms (0%)	3.247.784 ms (4,9%)	
java.net.SocketInputStream.read ()	0,0 ms (0%)	2.019.392 ms (3%)	
java.net.SocketInputStream.socketRead ()	0,0 ms (0%)	2.019.392 ms (3%)	
java.net.SocketInputStream.socketRead[native] ()	2.019.392 ms (44,8%)	2.019.392 ms (3%)	
java.util.concurrent.ThreadPoolExecutor\$Worker.run ()	0,0 ms (0%)	1.733.962 ms (2,6%)	
java.util.concurrent.ThreadPoolExecutor.runWorker ()	0,0 ms (0%)	1.733.962 ms (2,6%)	
java.security.AccessController.doPrivileged[native] ()	2.582 ms (0,1%)	1.630.978 ms (2,4%)	
java.io.FilterInputStream.read ()	0,0 ms (0%)	1.519.939 ms (2,3%)	
sun.rmi.transport.tcp.TCPTransport\$ConnectionHandler.run ()	0,0 ms (0%)	1.509.670 ms (2,3%)	
sun.rmi.transport.tcp.TCPTransport\$ConnectionHandler\$\$lambda\$1437.650774690.run ()	0,0 ms (0%)	1.509.670 ms (2,3%)	
sun.rmi.transport.tcp.TCPTransport\$ConnectionHandler.lambda\$run\$0 ()	0,0 ms (0%)	1.509.670 ms (2,3%)	
sun.rmi.transport.tcp.TCPTransport\$ConnectionHandler.run0 ()	0,0 ms (0%)	1.509.670 ms (2,3%)	
sun.rmi.transport.tcp.TCPTransport.handleMessages ()	0,0 ms (0%)	1.509.670 ms (2,3%)	
java.io.FileInputStream.read ()	0,0 ms (0%)	1.507.413 ms (2,3%)	
java.io.FileInputStream.readBytes[native] ()	1.507.413 ms (33,4%)	1.507.413 ms (2,3%)	
java.io.BufferedInputStream.read ()	0,0 ms (0%)	1.506.377 ms (2,3%)	
java.io.BufferedInputStream.fill ()	0,0 ms (0%)	1.506.292 ms (2,3%)	
java.lang.reflect.Method.invoke ()	0,0 ms (0%)	1.506.292 ms (2,3%)	

Imagen 2. Project Profiling PC1

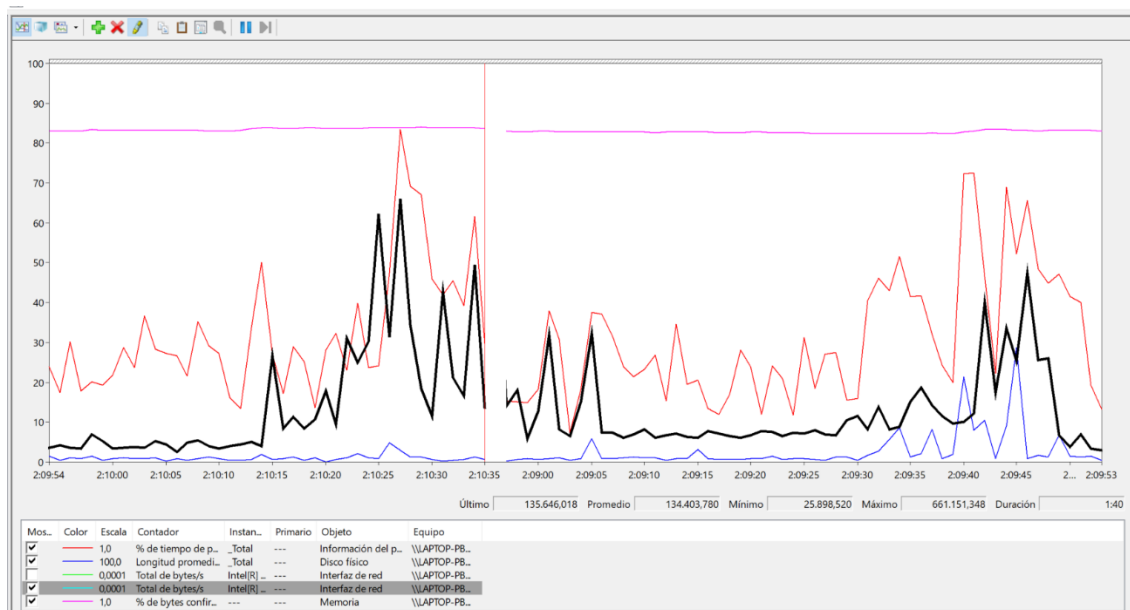
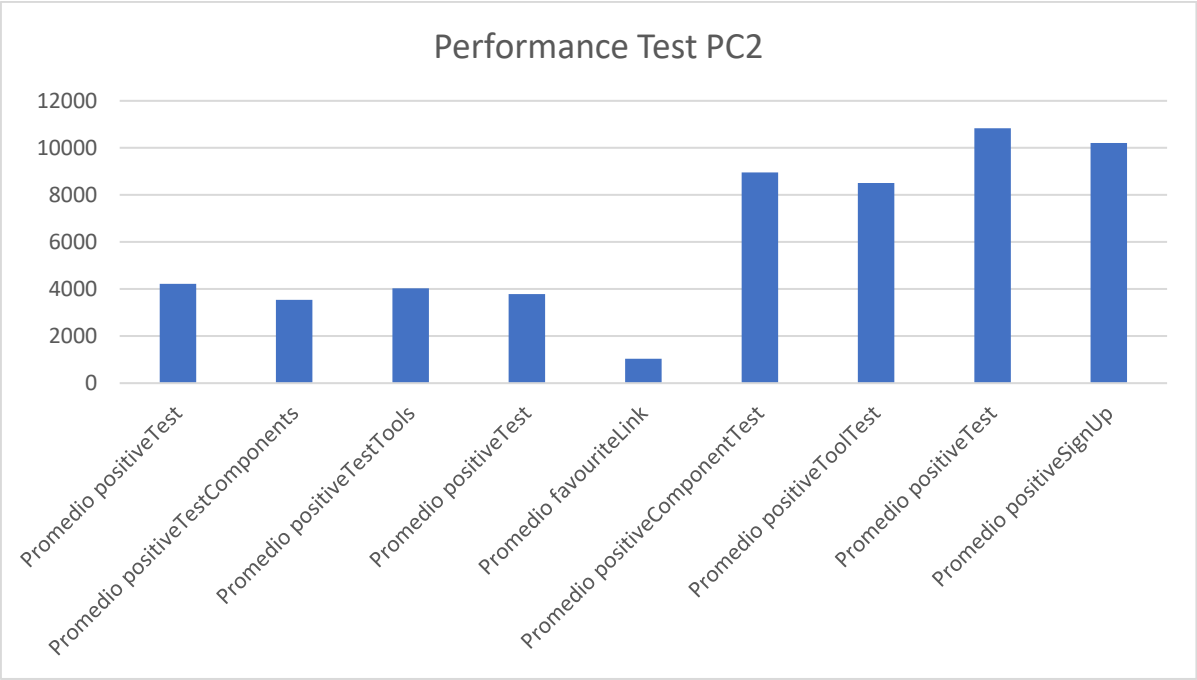


Imagen 3. Computer Profiling PC1

El consumo de recursos del sistema ha sido medio. No obstante, a pesar de observar la presencia de picos de consumo, se considera aceptable, pues estos han sido casos puntuales y no alargados en el tiempo.

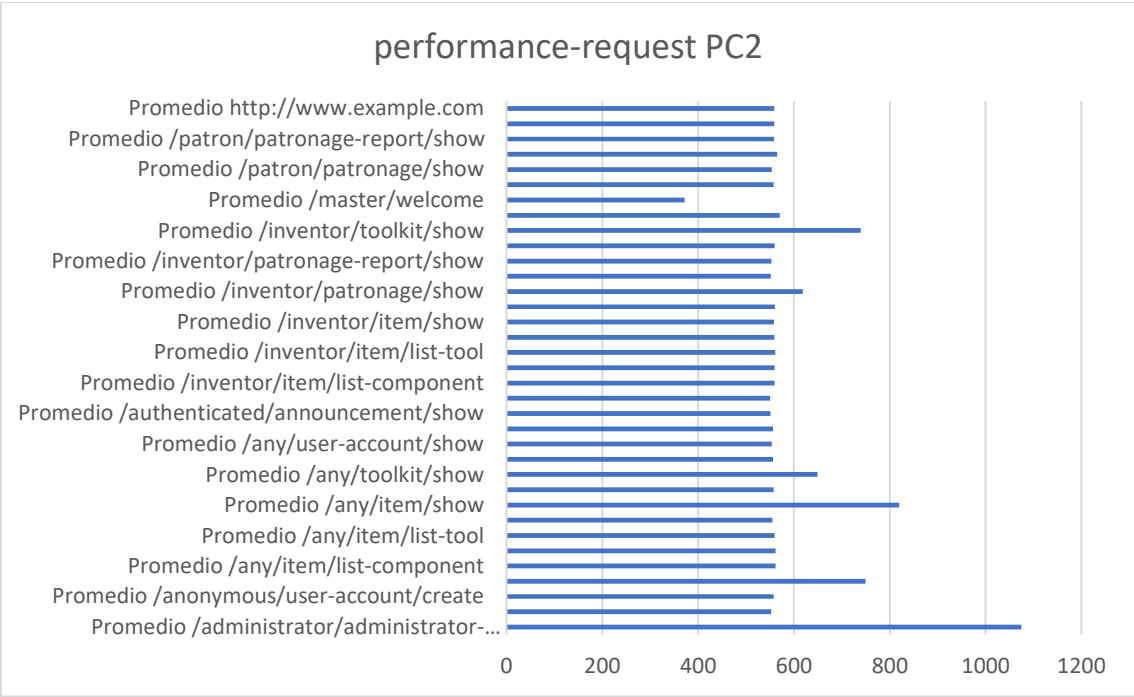
Segundo Análisis

El segundo análisis se realizó sobre un equipo con un procesador AMD Ryzen 7 4800HS, 16 GB RAM y Windows 11 Home. A continuación, se muestran los resultados obtenidos:



Gráfica 3. Performance Test PC2

Como se puede observar, este ordenador era más potente y se ve reflejado en la ejecución de los test, la ejecución iba de 4 a 11 segundos, mejorando así los tiempos de los tests.



Gráfica 4. Performance Request PC2

Tras la ejecución de los test, se ha alcanzado también un 81,9 % de cobertura.


Element	Covera...	Covered Ins...	Missed Instr...	Total Instruc...
>  Acme-Toolkits	81,9 %	5.192	1.146	6.338

Imagen 4. Nivel de Cobertura

La mayoría de las peticiones se mantienen estables, de manera similar al primer análisis, exceptuando algunos casos como el *DashBoard* de administrador que necesita un mayor tiempo de ejecución debido a los cálculos necesarios. Sorprendentemente, en este PC el *DashBoard* de patrocinador obtiene un valor similar al promedio. En la clase *Chirp* se vuelven a obtener valores muy altos, por lo que se descarta que sea debido a la ejecución en paralelo. Por ello se concluye que es probable que este retardo provenga de la comprobación de que los *Chirp* tengan menos de un mes de duración.

<i>time</i>	
Media	464,5444598
Error típico	11,00282955
Mediana	553
Moda	552
Desviación estándar	241,5616039
Varianza de la muestra	58352,00846
Curtosis	74,35538803
Coefficiente de asimetría	5,625350813
Rango	3610
Mínimo	175
Máximo	3785
Suma	223910,4296
Cuenta	482
Nivel de confianza(95,0%)	21,61954962
Intervalo	442,9249102 486,164009

Tabla 2. Estadística Descriptiva PC2

En esta ocasión, el nivel de confianza es de 21.61 y el intervalo de confianza oscila entre 442.92 y 486.16 milisegundos, por lo que los resultados obtenidos son mejores y, por tanto, positivos.

También se ha vuelto a realizar el *profiling* del proyecto y del ordenador para encontrar cuellos de botella. Los resultados obtenidos de este proceso son los siguientes:

CPU samples			
Thread CPU time			
Results:	View:	Collected data:	Snapshot Thread Dump
Name	Total Time	Total Time (CPU)	
Forwarding findElement on session 71	640.998 ms (100%)	474.659 ms	(100%)
Reference Handler	640.998 ms (100%)	0,0 ms	(-%)
Finalizer	640.998 ms (100%)	0,0 ms	(-%)
mysql-cj-abandoned-connection-cle:	640.998 ms (100%)	0,0 ms	(-%)
Exec Default Executor	640.998 ms (100%)	0,0 ms	(-%)
Exec Stream Pumper	640.998 ms (100%)	640.998 ms	(100%)
Exec Stream Pumper	640.998 ms (100%)	640.998 ms	(100%)
Name	Self Time (CPU)	Total Time (CPU)	
java.lang.Thread.run ()	0,0 ms (0%)	2.658.241 ms	(4,8%)
java.net.SocketInputStream.read ()	0,0 ms (0%)	1.751.112 ms	(3,1%)
java.net.SocketInputStream.socketRead ()	0,0 ms (0%)	1.751.112 ms	(3,1%)
java.net.SocketInputStream.socketRead0[n	1.750.900 ms (46,5%)	1.751.112 ms	(3,1%)
java.util.concurrent.ThreadPoolExecutor\$Work	0,0 ms (0%)	1.375.139 ms	(2,5%)
java.util.concurrent.ThreadPoolExecutor.runW	0,0 ms (0%)	1.375.139 ms	(2,5%)
java.security.AccessController.doPrivileged[2.106 ms (0,1%)	1.331.464 ms	(2,4%)
java.io.FilterInputStream.read ()	0,0 ms (0%)	1.291.498 ms	(2,3%)

Imagen 5. Project Profiling PC2

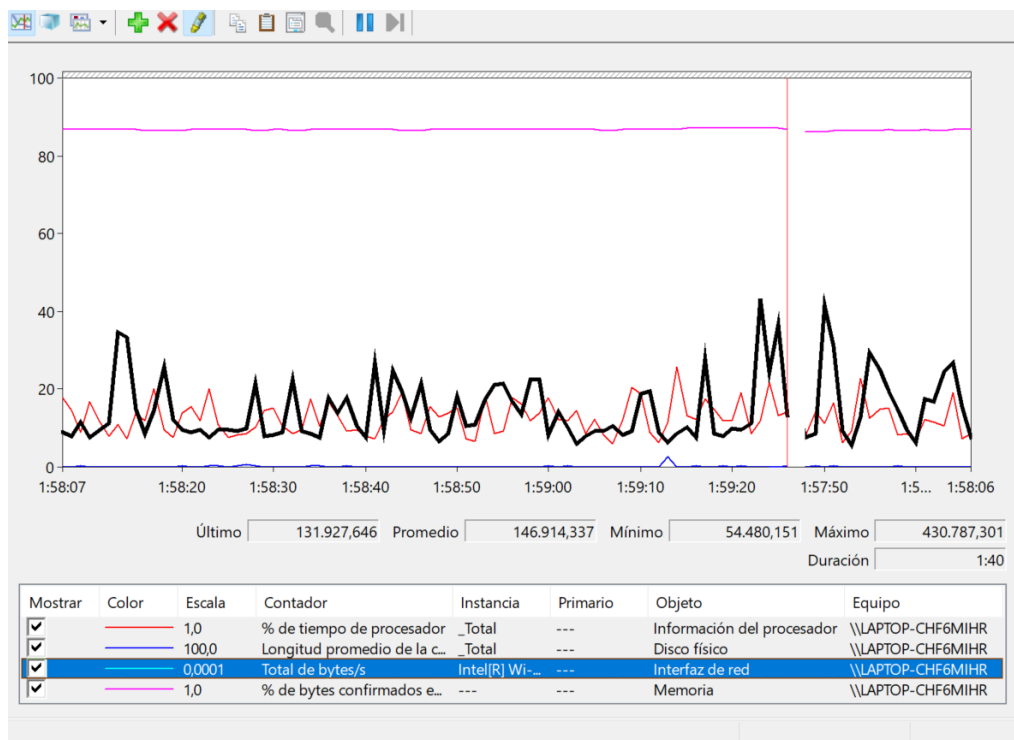


Imagen 6. Computer Profiling PC2

Cabe destacar la homogeneidad en la utilización de recursos, ya que no ha habido ningún aumento prolongado en el consumo de recursos del sistema.

Hipótesis de contraste

A continuación, se introduce la comparación de los resultados obtenidos entre los dos PC a través de un Z-Test.

Prueba z para medias de dos muestras		
	PC1	PC2
Media	632,4788917	464,54446
Varianza (conocida)	119760,3949	58352,0085
Observaciones	482	482
Diferencia hipotética de las medias	0	
z	8,736071635	
P(Z<=z) una cola	0	
Valor crítico de z (una cola)	1,644853627	
Valor crítico de z (dos colas)	0	
Valor crítico de z (dos colas)	1,959963985	

Tabla 3. Z-Test

Se puede observar que, pese a que los resultados obtenidos por ambos PC son satisfactorios, uno es mejor que otro. Se sospecha que el PC2 es más potente, por tanto, para cerciorarse de que es mejor, se realiza un Z-Test. El resultado de P(Z) es menor que alfa (0.05).

Conclusiones

El rendimiento probablemente sea de los atributos de calidad más importantes de un proyecto, pues si un sistema presenta grandes tiempos de espera u opera de manera muy lenta perjudica enormemente la experiencia del cliente con el mismo, así como el nombre de la empresa que crea la aplicación.

Por ello, se debe trabajar con sumo cuidado que el rendimiento del sistema sea óptimo. Este documento ha servido como apoyo para entender el estudio de esta característica inherente del sistema, pues gracias a todos los análisis, poco a poco se ha ido desgranando el funcionamiento interno del proyecto, mostrando todos los comportamientos que provocaban retardos en la ejecución, bien por un cálculo excesivo de datos o por una cantidad de bucles inusitada.

Otro de los aspectos que ha ayudado a comprender el funcionamiento del desarrollo del proyecto ha sido poder comparar los resultados entre dispositivos, lo que ha otorgado el conocimiento sobre la diferencia que podía existir al probar la aplicación en ordenadores de distintas gamas, marcas o componentes. De esta manera, si el dispositivo de gama baja sufría mucho en la ejecución de las pruebas, se debían implementar los cambios necesarios para aligerar el proceso y mejorar el rendimiento.

Este documento ha sido, por tanto, una grata experiencia que ha aportado enseñanzas valiosas a todos los miembros del grupo y que ha acercado al grupo al desarrollo de un proyecto profesional.

Bibliografía

Intencionalmente en blanco.