John Ardila
Rogelio García
Felipe Plazas
Diana Solano

Experimento I

Grupo St. ValenTeam

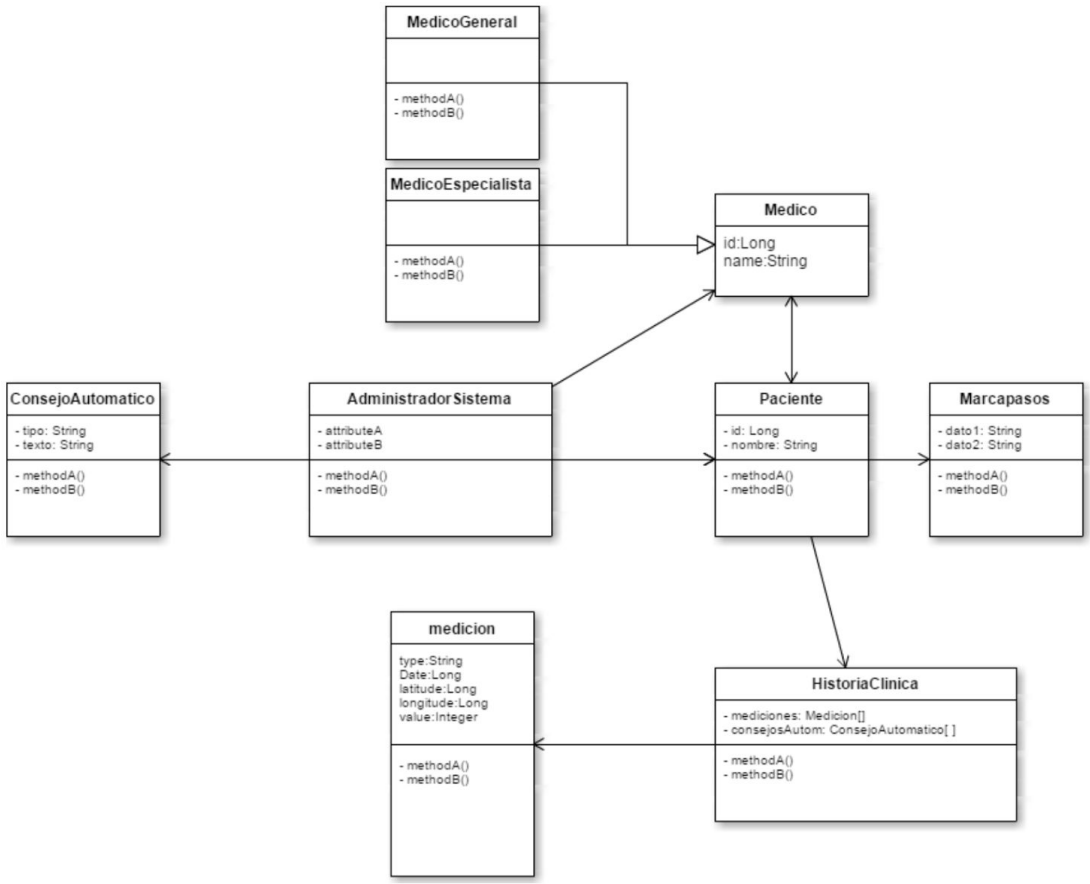
Pre-experimentación

En esta fase se realiza la planeación del experimento. Es imperativo tener claro lo siguiente:

1. Hipótesis: La arquitectura por actores propuesta y desarrollada en Play satisface los atributos de calidad de desempeño y escalabilidad. El sistema soporta la recepción de 3.000 solicitudes concurrentemente y no presenta errores para dichas solicitudes.
2. Objetivo del experimento: Desarrollar la aplicación requerida por el Hospital Santa Fe a partir de una arquitectura por actores y realizar unas pruebas de carga en el servicio en la nube Loader.io incrementando el número de threads con el propósito de evaluar el desempeño y la escalabilidad de la misma.
3. Descripción del experimento: Desarrollar la aplicación requerida por el Hospital Sante Fe, haciendo uso de la arquitectura por actores de Play y de la base de datos Mongo y desarrollar las pruebas de carga en el servicio en la nube Loader.io el cual garantiza la creación de threads y envío en la ventana de tiempo establecido. Se toman los datos como resultado de la prueba de carga y se analizan para verificar el cumplimiento de los requerimientos de negocio y de los atributos de calidad.
4. Artefactos a construir:
    - Arquitectura por actores (Akka, Play! Framework, Scala)
        - Patrón MVC
        - Servicios REST
    - Base de datos Mongo (Nosql)
    - Asíncrono

    Se desarrollará la aplicación a partir del siguiente modelo de dominio.

John Ardila
Rogelio García
Felipe Plazas
Diana Solano



Se van a desarrollar las entidades virtuales correspondientes a este modelo; la interacción con los sensores no se realiza todavía, por lo que las entidades virtuales de dichos sensores no se realizarán. Sin embargo, la recepción de solicitudes por medio de los servicios REST expuestos, y la creación de elementos como los consejos y el historial médico también se modelarán.

5. Recursos de la experimentación: Especificaciones, de hardware y software, con las que se va a desarrollar el experimento.
   - Una máquina AWS
   - Máquinas virtuales (windows 7)
   - Dos máquinas Unix basada en Mac OSX
   - Máquina Windows 10
   - Para las pruebas, se utilizó la aplicación web “loader.io”
6. Resultados esperados:

Se debe poder procesar y responder a todas las solicitudes que se espera que sean recibidas por el sistema. En el peor caso, se recibirían todas las solicitudes posibles en un mismo momento. Por lo anterior, la ventana de tiempo estipulada para recibirlas

se definió en un segundo; es decir, se debe poder procesar 3000 solicitudes en 1 segundo dado. Por otro lado, las solicitudes de emergencia también deben priorizar el desempeño, y deben ser respondidas, sin embargo, para estas solicitudes es vital que no se presenten errores asociados a la carga.

7. Duración y etapas:

Para el primer experimento se definen 3 etapas: diseño de la solución y elección de la arquitectura, desarrollo de la aplicación y pruebas de carga. Se estima que se dedicarán 3 horas a la etapa de diseño de la solución y elección de la arquitectura, 8 horas al desarrollo de la aplicación y 3 horas al desarrollo de pruebas de carga.

Post-experimentación

1. Duración real: El tiempo real invertido en el experimento fue 4 horas en el diseño de la solución y elección de la arquitectura, 15 hora en desarrollo de la aplicación (se debe tener en cuenta que se desarrollaron los servicios con persistencia) y se dedicaron 4,5 horas al desarrollo de pruebas de carga. Por lo tanto se dedicaron en total 23,5 horas al desarrollo del experimento 1.
2. Artefactos construidos: Se realizaron cambios en el modelo de dominio en el proceso de desarrollo, esto se debe a la necesidad de una clase consejo que permita enviar consejos automáticos y estructurados. No se implementó la clase médico general que heredaba de médico ya que no presentaba ninguna diferencia a la clase de la que heredaba. Y se modeló una clase emergencia, la que se encarga de recibir las solicitudes de emergencia del dispositivo móvil y procesarlas.

Se desarrollaron todos los servicio mencionados anteriormente y la capa de persistencia de la información haciendo uso de la base de datos Mongo. Los servicios se desarrollaron a partir de la arquitectura por actores desarrollada en Play.

Pruebas de carga

Se implementó la siguiente prueba de carga, basado en los requerimientos del Hospital Cardiológico de Santa Fe, 1000 pacientes de los cuales se reciben 3 mediciones por segundo, por ello es necesario soportar 3000 solicitudes por segundo, ya que todas las mediciones llegan a un solo endpoint en nuestro servidor:

- Número de threads: 3000

John Ardila
Rogelio García
Felipe Plazas
Diana Solano

- Ramp-up period: 1s
- Loop count: 1

La prueba se ejecutó 3 veces, y se obtuvieron los siguientes resultados:

- Primera ejecución:

Label	# Samples	Average	Min	Max	Std. Dev.	Error %	Throughput	Received KB/...	Sent KB/sec	Avg. Bytes
enviarMedic...	3000	98	1	1247	192.02	0.00%	1153.4/sec	84.48	0.00	75.0
TOTAL	3000	98	1	1247	192.02	0.00%	1153.4/sec	84.48	0.00	75.0

- Segunda ejecución:

Label	# Samples	Average	Min	Max	Std. Dev.	Error %	Throughput	Received KB/...	Sent KB/sec	Avg. Bytes
enviarMedic...	3000	14	0	166	25.45	0.00%	1284.8/sec	94.10	0.00	75.0
TOTAL	3000	14	0	166	25.45	0.00%	1284.8/sec	94.10	0.00	75.0

- Tercera ejecución

Label	# Samples	Average	Min	Max	Std. Dev.	Error %	Throughput	Received KB/...	Sent KB/sec	Avg. Bytes
enviarMedic...	3000	41	1	367	58.02	0.00%	1263.7/sec	92.56	0.00	75.0
TOTAL	3000	41	1	367	58.02	0.00%	1263.7/sec	92.56	0.00	75.0

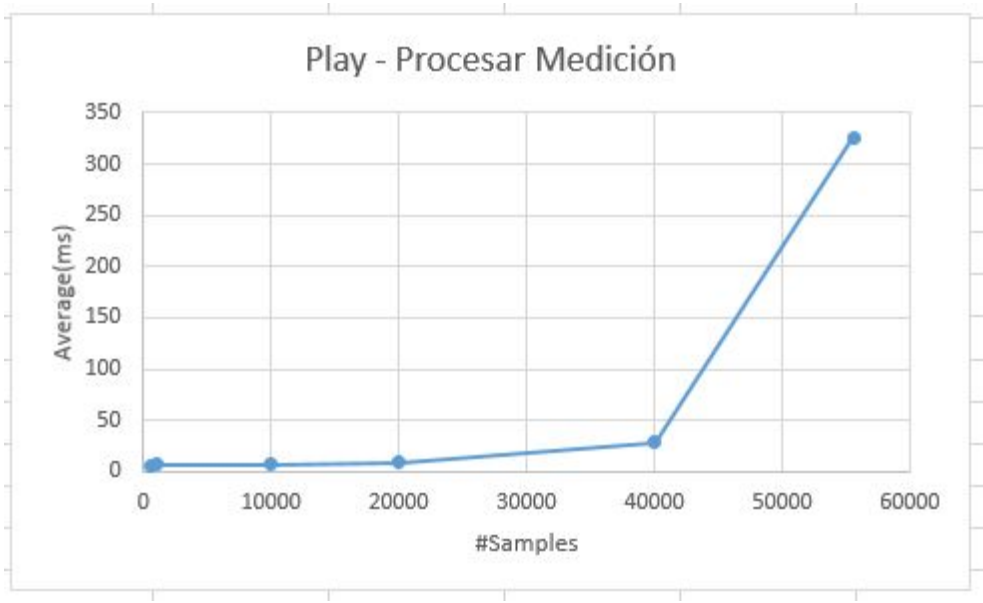
Se puede evidenciar que los tiempos promedio de respuesta rondan entre los 14 y los 98 milisegundos, con un porcentaje de error de 0%. Esto nos indica que el sistema cumple con el requerimiento de carga solicitado.

Pruebas de carga:

- **Prueba 1: Procesar Medición (POST)**

Play - Procesar Medición							
#Samples	Average(ms)	Min(ms)	Max(ms)	Error (%)	Throughput	Received	Sent KB
540	6	4	37	0,0	9	56,43KB	182,99KB
1020	7	4	149	0,0	17	106,58KB	345,64KB
10020	7	4	247	0,0	167	1,02MB	3,32MB
20038	9	6	1169	0,0	334	2,04MB	6,63MB
40019	28	5	3116	0,0	667	4,08MB	13,24MB
55588	326	3	20027	0,4	926	6,68MB	19,33MB

John Ardila
Rogelio García
Felipe Plazas
Diana Solano

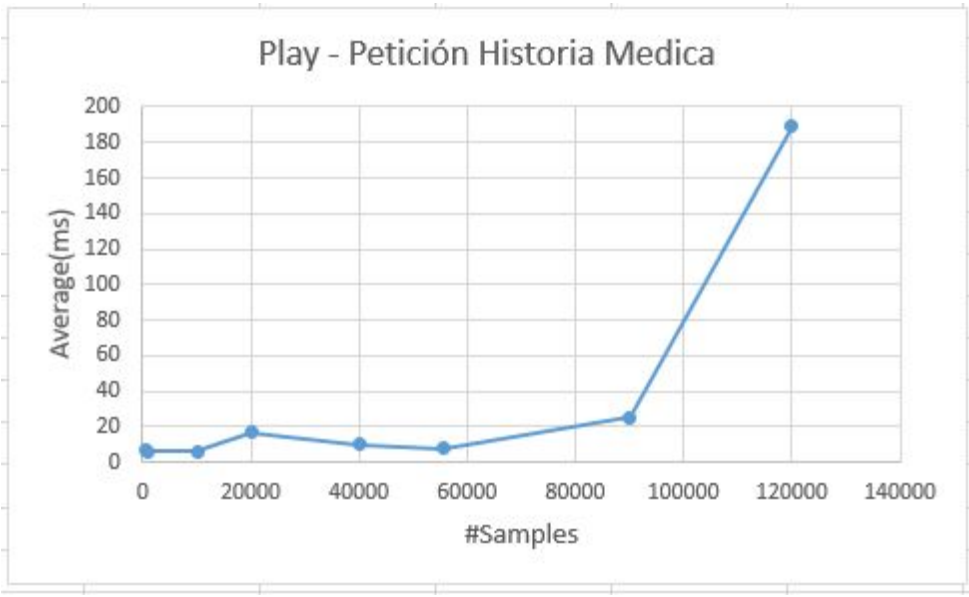


A partir de la gráfica anterior se evidencia que el desempeño de la aplicación comienza a deteriorarse a partir de 40.000 solicitudes. Dado que se espera una recepción de 3.000 solicitudes concurrentemente, ya que el hospital cuenta con 1.000 pacientes de los cuales se recibirán 3 datos por paciente, en 1 segundo. La arquitectura diseñada para la recepción de solicitudes responde correctamente al experimento con un 0% de error y en menos de 4 milisegundos.

- **Prueba 2: Petición Historia Médica (GET)**

Play - Petición Historia Medica							
#Samples	Average(ms)	Min(ms)	Max(ms)	Error (%)	Throughput	Received	Sent KB
540	7	5	64	0,0	9	90	130
1020	6	5	174	0,0	17	247,03KB	169,34KB
10020	6	4	88	0,0	167	2,34MB	1,61MB
20038	17	5	3011	0,0	334	4,71MB	3,25MB
40019	10	5	1040	0,0	667	9,47MB	6,49MB
55588	8	5	1044	0,0	926	13MB	9MB
90000	25	4	1271	0,0	1500	14MB	21MB
120000	189	3	10004	2,6	2000	19MB	29MB

John Ardila
Rogelio García
Felipe Plazas
Diana Solano



A partir de la gráfica anterior, se observa que el sistema presenta un incremento en el tiempo promedio de respuesta a la solicitud GET a partir de las 20.000 solicitudes. Dado que se esperan solo 3.000 solicitudes por minuto el sistema es capaz de responder en menos de 6 milisegundos y presenta un error del 0% para éste número de solicitudes.

Conclusiones

A partir del experimento se pudo observar que el desarrollo de la aplicación para el Hospital Santa Fe, haciendo uso de la arquitectura de actores por medio de Play y de la base de datos Mongo, satisface los requerimientos de calidad principales; desempeño y escalabilidad. Se evidenció que la aplicación soporta 3.000 peticiones con un tiempo de respuesta menor a 7 milisegundos y con un 0% de error.
Adicionalmente, se observa que la aplicación presenta un buen desempeño hasta 20.000 solicitudes con tiempos de respuesta promedio menores a 17 milisegundos y 0% de error.