EXPERIMENTO I – ENTERGA FINAL  
ISIS2503 – ARQUITECTURA Y DISEÑO DE SOFTWARE  
Christian Chavarro  
Federico Posada  
Santiago Sáenz  
Juan Pablo Sanmiguel

En el siguiente documento se describirán las soluciones de TI que permitirán la innovación de la actual empresa Yale (empresa experta en soluciones de seguridad).

En nuestro caso se buscó tener los siguientes servicios para esta entrega del proyecto:

* 1. La batería puede tener dos estados: nivel apropiado y batería baja.
  2. Se debe poder variar el nivel de voltaje censado con el potenciómetro.
  3. El LED indicador del nivel de batería se debe encender cuando se detecte que la carga de la batería sea menor o igual a la definida (1.2V).
  4. El buzzer cuenta con dos estados básicos: sonido como indicador de clave fallida o puerta abierta, y standby para los demás estados.
  5. Debe sonar cada vez que el indicador RGB se encuentra en rojo, es decir cada vez que se detecta un intento fallido o una alerta.
  6. Debe sonar cuando se tiene batería baja, durante 2 segundos con una frecuencia de 30 segundos.
  7. Debe permitir la publicación de mensajes de alerta en uno o varios tópicos del servidor MQTT.
  8. Debe permitir la recepción de mensajes mediante la suscripción en uno o varios tópicos del servidor MQTT.

Restricciones

|  |  |  |
| --- | --- | --- |
| Identificador | Tipo | Descripción |
| RE-01 | Tecnología | El servicio central del servicio debe estar implementado en JEE (Java Platform, Enterprise Edition) |
| RE-04 | Negocio | El experimento debe ser desarrollado antes del 25/03/2018 |

**Diagrama de despliegue,** indicando la capacidad de cada nodo.

A continuación, se muestra el diagrama de despliegue para esta entrega.

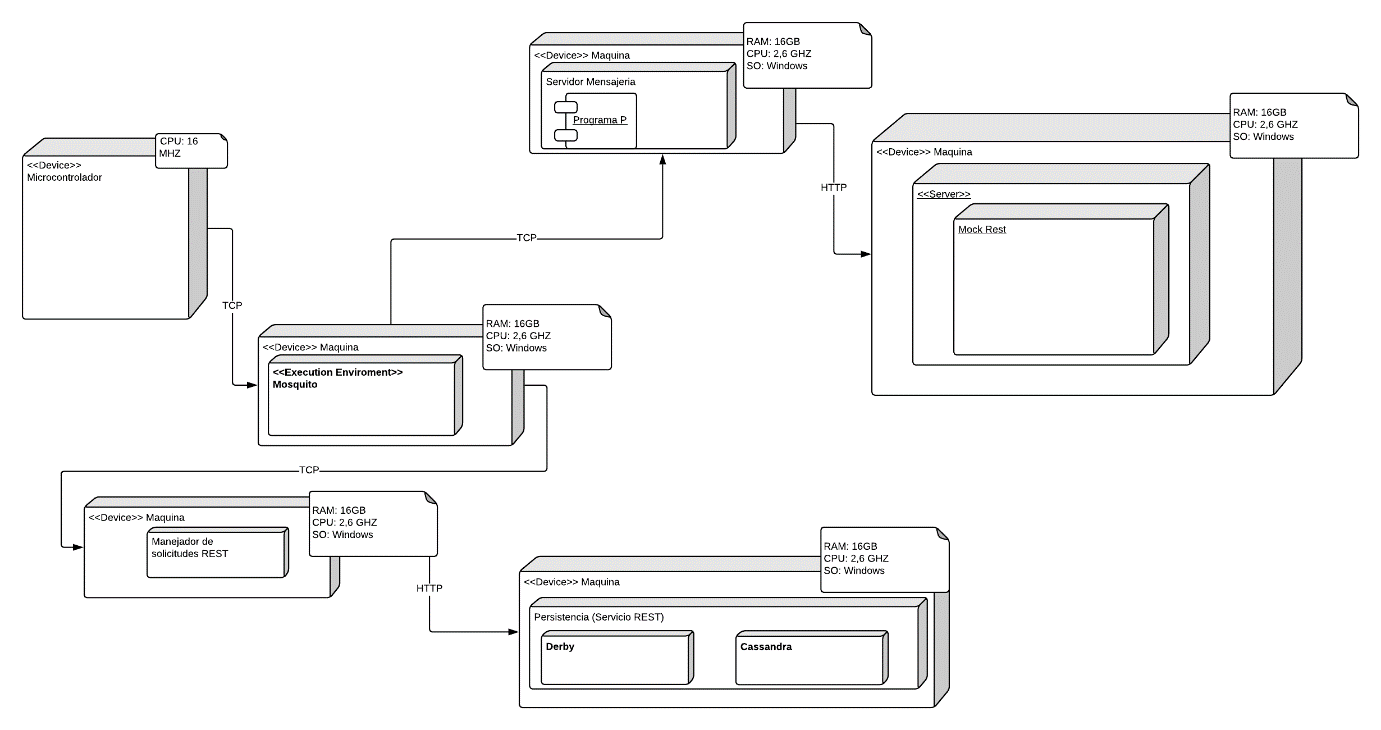


Imagen 1: Diagrama de despliegue para este experimento.

Como es mostrado en el diagrama, para esta entrega, no se utilizó un balanceador de carga, en remplazo, se tienen varias máquinas que escuchan diferentes tópicos y manejan las solicitudes. Dependiendo de la solicitud, esta va a un tópico y es atendida por quien lo escucha.

**Comparación de los resultados esperados versus los obtenidos con respecto a tiempo de respuesta y errores para 200 y 300000 (200 inmuebles \* 1500 unidades) alarmas de cualquier tipo.**

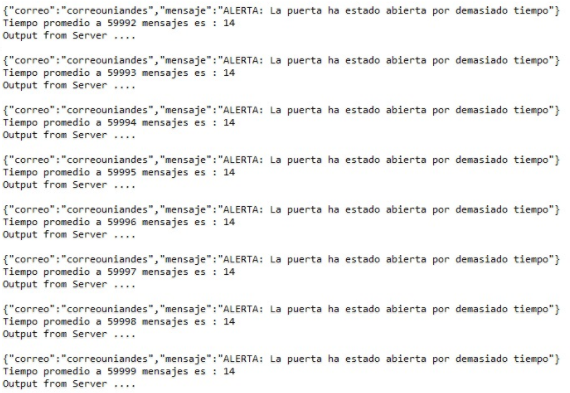
Se esperaba que la arquitectura soportara el procesamiento de 300000 solicitudes, provenientes de unidades residenciales e inmuebles. Esto, en una ventana de 1 minuto, con 0% en la medida de porcentaje de error y tiempos de respuesta inferiores a 1 segundo.

En las pruebas, se obtuvo que la arquitectura utilizada para el experimento soporta estas solicitudes con un error de **0%** en un minuto. Además, presenta un tiempo máximo de respuesta de 98 ms, **0.1 segundos.**

Al comparar los resultados esperados con los obtenidos, se aprecia que la arquitectura e implementación realizadas, cumplen el escenario critico de Yale.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Etiqueta | # muestras | Media | Min | Máx. | %Error | Rendimiento | kb/sec |
| ServMens1 | 60000 | 3 | 0 | 57 | 0% | 997,96751 | 19,4 |
| ServMens2 | 60000 | 4 | 0 | 48 | 0% | 997,96751 | 19,4 |
| ServMens3 | 60000 | 3 | 0 | 21 | 0% | 997,96751 | 19,4 |
| ServMens4 | 60000 | 17 | 0 | 98 | 0% | 997,96751 | 19,4 |
| ServMens5 | 60000 | 5 | 0 | 42 | 0% | 997,96751 | 19,4 |

En las pruebas de la arquitectura, se utilizaron 5 tópicos. Para cada tópico, se conectaba una máquina. Cada vez que se recibía un mensaje, lo enviaba al mock rest. Con JMeter, se generaron 60000 mensajes para cada tópico, 300000 mensajes en total. Para cada mensaje, se mostraba la información en consola, tal como se muestra en la imagen 2. Cada vez que se publicaba un mensaje, cada instancia del servidor revisaba el tiempo de respuesta promedio y lo imprimía.

  
Imagen 2: Mensajes impresos en la consola de una máquina que atiende las alarmas.

**Declaración del esfuerzo de cada miembro en Exp. 1 intermedio y final.**

Esta declaración se encuentra en un archivo adjunto a esta misma carpeta del repositorio. Acá se encuentra un resumen, en imagen, de esta declaración de esfuerzo.

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| ARTEFACTO INTEGRANTES | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | Esfuerzo % |
| Christian Chavarro |  | x |  | x | x | x | x | x | 22,5 |
| Federico Posada |  | x | x | x | x | x |  |  | 30 |
| Santiago Sáenz |  |  |  |  |  |  | x | x | 22,5 |
| Juan Sanmiguel | x |  |  |  |  |  |  | x | 25 |

|  |  |
| --- | --- |
| ID | ARTEFACTO |
| 1 | Programa de cerradura (Simulado a través de wiring) |
| 2 | Programa de hub (Simulado a través de nodered) |
| 3 | Servidores de mensajería (Por ejemplo, Mosquitto, Kafka) |
| 4 | Programa P |
| 5 | Código de notificación |
| 6 | Código de persistencia |
| 7 | Documentación |
| 8 | Montaje físico de cerradura |