**Arquitectura de Software**

**Experimento 1**



**Realizado por:**

|  |  |  |
| --- | --- | --- |
| **Persona** | **Rol** | **CódigoUniandes** |
| Carlos Ernesto González Vargas | Ingeniero de Requerimientos | 200819123 |
| Sandra Milena Gómez Ríos | Ingeniero de Producción | 201110951 |
| Andrés Mauricio Erazo Benavides | Ingeniero de Soporte | 201110949 |
| David Pérez Chibuque | Ingeniero de Calidad | 201117818 |
| Willian Alejandro Idrobo Luna | Arquitecto Jefe | 201110544 |
| Erik Fernando Arcos Franco | Ingeniero de Administración | 201110856 |

**Contenido**

**Pag.**

[**1.** **Definición del Sistema** 3](#_Toc296894778)

[**1.1.** **Identificación de patrones** 4](#_Toc296894779)

[**1.2.** **Punto de vista funcional** 5](#_Toc296894780)

[**1.2.1.** **Diagrama de descomposición** 5](#_Toc296894781)

[**1.2.2.** **Diagrama de despliegue** 7](#_Toc296894782)

[**2.** **Experimento A – Latencia** 9](#_Toc296894783)

[**2.1.** **A1 – Registro de Alarma** 9](#_Toc296894784)

[**2.2.** **A2 – Consulta de Información Consolidada** 11](#_Toc296894785)

[**3.** **Experimento B – Escalabilidad** 14](#_Toc296894786)

[**3.1.** **B1 – Escalabilidad del Sistema** 14](#_Toc296894787)

**Arquitectura de Software**

**Experimento 1**

1. **Definición del Sistema**

La empresa **Alpes Smart Home (ASH)** desea entrar a participar en el mercado de las casas inteligentes dado el considerable avance de la tecnología en dispositivos móviles, sensores inalámbricos y sistemas embebidos que ha habido durante los últimos años. Por esta razón la empresa **ASH** plantea el proyecto ***Casa Inteligente***, el cual surge para dar respuesta al diseño de arquitectura de solución de un sistema que permita la administración de viviendas inteligentes.

El sistema se debe cumplir con las necesidades iniciales del sistema, descritas a continuación:

* Analizar la información enviada por los sensores de humo, ventanas y puertas de las oficinas y viviendas monitoreadas.
* Una vez se detecta humo en una posición específica de la vivienda u oficina, se procede a notificar a los servicios de control de emergencias. Si se detecta que la vivienda está habitada, se informa a los bomberos y servicios de ambulancias, el número estimado de personas que están en ese momento en el lugar, así como las posibles ubicaciones.
* Si se trata de un sensor de ventana o puerta abierta sin autorización, se debe notificar a la policía sobre el evento, enviando como información adicional, la localización de la ventana dentro del inmueble y el número estimado de personas que están en la vivienda o los sectores de la vivienda en los que se ha detectado movimiento.
* Generar un reporte consolidad mostrando todos los eventos registrados (ordenadas por tipo de evento) de forma cronológica (hora, día, mes, año).
* Por cada inmueble u oficina se tienen en promedio 50 tags RFID enviando información cada segundo a cuatro antenas ubicadas en diferentes posiciones de la casa u oficina. Estas antenas a su vez consolidan la información y la envían a la central.
* Cuando se detecta que un tag RFID ha violado una de las reglas definidas para su comportamiento, se debe generar una alarma indicando a todas las porterías y a los responsables de la vivienda u oficina lo ocurrido.

Para el experimento descrito a continuación se tiene el siguiente alcance:

* Diseñar la arquitectura necesaria para recibir la información de los sensores de humo, ventas, puertas y RFIDs con la cual se logrará un segundo en el procesamiento del mensaje. Segundo que será contado a partir de la llegada del mensaje (trama) al sistema central hasta el envío de la notificación a su correspondiente parte.
* Diseñar la arquitectura necesaria para generar un reporte consolidado de los eventos generados por las viviendas u oficinas. Este reporte debe demorarse máximo dos segundos en ser presentado al usuario. Es importante anotar que usuario que generará el reporte será el usuario administrador y podrá ver el consolidado de todos los eventos de todas las viviendas u oficinas.
* Determinar la escalabilidad del sistema a través de unas pruebas de carga, generadas en el sistema central. Para determinar la escalabilidad se tendrá en cuenta que se debe mantener un segundo de respuesta en el procesamiento del evento, como se mencionó anteriormente.
  1. **Identificación de patrones**

Para el desarrollo de este sistema se tuvieron en cuenta los siguientes patrones:

***Acceso a recursos – temprana***

Se aplica en la parte de obtención de la conexión a la base de datos para la generación de reportes.

***Acceso a recursos – caché***

Es aplicada para el manejo de y la información de los usuarios y las reglas que son configuradas para cada casa u oficina.

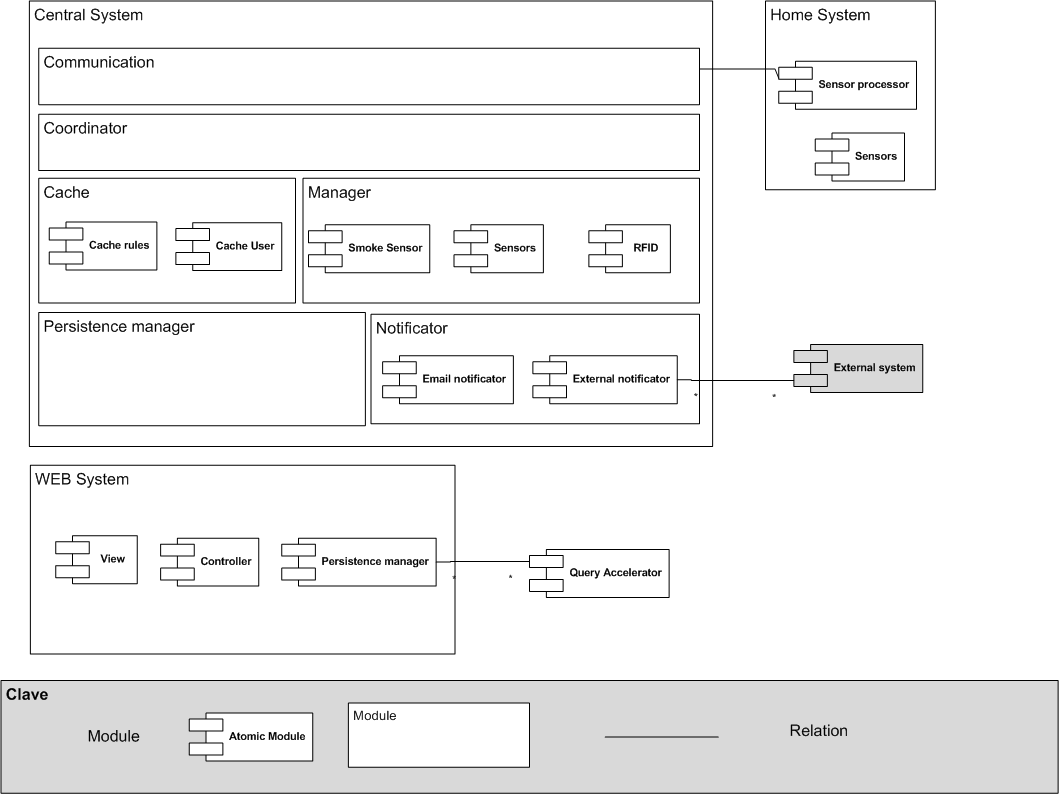
***Manejo de eventos – respuesta reactiva a eventos***

Al llegar un evento al sistema se delega la responsabilidad de manejar el evento a un hilo que se encargue de procesarlo de acuerdo a las reglas configuradas.

***Concurrencia y sincronización – coordinador***

Existe un componente receptor de todos los eventos que se encarga de crear una instancia de un manejador que procesa el evento de acuerdo a su tipo.

* 1. **Punto de vista funcional**
     1. **Diagrama de descomposición**



**Home system**

El sistema que controla el conjunto de sensores que capturan los eventos que hay en la casa, su responsabilidad es recibir las mediciones de los sensores, realizar la trama y enviar estos datos para que se evalúen las reglas de negocio en el sistema central.

**Sensor processor**

Se encarga de construir la trama que se envía periódicamente al sistema central.

**Sensors**

Miden el mundo real en las casas u oficinas del conjunto ASH.

**External System**

Son los sistemas externos a quienes se les notifican las alertas de emergencia.

**Central system**

El sistema central reúne las funcionalidades para manejar las tramas que envían los sensores, generar los eventos y enviar las notificaciones a los servicios de emergencias y seguridad, haciendo la integración de sus diferentes capas.

**Communication**

Escucha los sistemas de cada una de las casas y oficinas recibiendo las tramas que le envían para pasarlas a la capa coordinador, quien se encargara de su manejo.

**Coordinator**

Crea un hilo nuevo que se encargara del manejo de procesar la petición y realiza el llamado para el procesamiento de un mensaje de un sensor en particular.

**Cache**

Presenta el patrón cache para mantener cargados los datos relacionados a las reglas CACHE RULES y para enviar las notificaciones CACHE USER, evitando que deba acceder a la base de datos cada vez que necesita los datos.

**Manager**

La responsabilidad principal de manager es comparar las tramas a nivel de bytes contra las reglas almacenadas en el cache, definiendo si se debe generar una alerta y enviar una notificación, para los sensores de humo SMOKE SENSOR, los sensores de las puertas y ventanas SENSORS y las antenas de los RFID.

**Persistence manager**

Se encarga de manejar la persistencia de los datos.

**Notificator**

Su responsabilidad principal es realizar el envió de la notificación de alerta a los usuarios y los sistemas de emergencia, a través de los servicios de los sistemas externos EXTERNAL NOTIFICATOR y a través de correo electrónico a los usuarios EMAIL NOTIFICATOR.

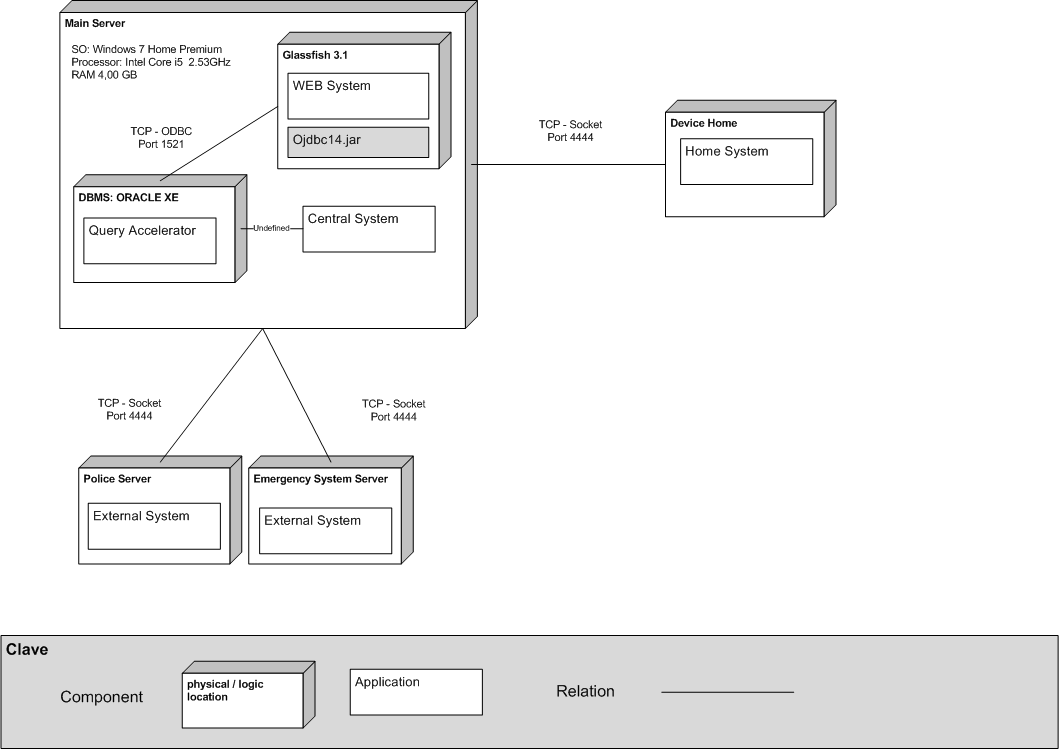
**Web system**

Es el aplicativo que va a interactuar con el cliente para presentar los informes de los eventos generados por el sistema. Se presentan a través de una página WEB por medio del módulo VIEW, se realiza el control de esta vista por medio del módulo CONTROLLER, finalmente se realiza la relación con una base de datos que almacena la información de los eventos del sistema por medio del módulo PERSISTENCE MANEGER.

**Query accelerator**

Es un conjunto de procedimientos almacenados en la base de datos que permiten acelerar la consulta de eventos que se presenta en los informes a los usuarios.

* + 1. **Diagrama de despliegue**



**Device home**

Es el dispositivo de hardware que concentra todas las señales de los diferentes sensores, dentro de los cuales se encuentran, el sensor de humo, los de puertas y ventanas, y las antenas de los RFID, Dentro de este dispositivo se encuentra el componente HOME SYSTEM.

**Main server**

El servidor principal donde viven los principales componentes de la arquitectura, además de la información de la base de datos que almacena los eventos, Aquí encontramos el servidor de aplicaciones donde vive el SYSTEM WEB (quien depende de Ojdbc1.4), el componente principal CENTRAL SYSTEM, quien reúne las funcionalidades clave de la arquitectura, y finalmente el manejador de la base de datos en donde se encuentran los procedimientos que mejoran las consultas QUERY ECCELERATOR.

**Police server**

El servidor de aplicaciones de la policía, donde se encuentra el sistema externo (EXTERNAL SYSTEM) que administra los eventos enviados por el sistema central. Esta comunicación se realiza a través de sockets.

**Emergency system service**

El servidor de aplicaciones donde se encuentra el sistema externo (EXTERNAL SYSTEM) que administra los eventos relacionados a las emergencias (EMERGENCY SYSTEM SERVER), enviados por el sistema central. Esta comunicación se realiza a través de sockets.

1. **Experimento A – Latencia**

El objetivo de este experimento es determinar si las decisiones de diseño satisfacen los escenarios de calidad relacionados con la latencia de la aplicación.

* 1. **A1 – Registro de Alarma**

Tabla 1. Descripción del Experimento A1

|  |  |
| --- | --- |
| **Descripción del Experimento** | |
| ***Título*:** Registro de Alarma | ***ID*:** A1 |
| ***Descripción*:** Este experimento busca verificar la latencia del sistema | ***Responsable*:**Ingenium |
| ***Propósito:***  ( ) Reparación, actualizar, clarificar  ( X ) Obtener Información técnica  ( ) Obtener información de negocio  ( ) Otros: | |
| ***Propósito:***  Verificar las decisiones de diseño asociadas a la latencia del sistema. | |
| ***Descripción del experimento:***  El sistema debe estar en capacidad de analizar la información enviada por los sensores de humo, ventanas y puertas de las oficinas y viviendas monitoreadas.  Se debe tener en cuenta lo siguiente:   * Una vez se detecta humo en una posición específica de la vivienda u oficina, se procede a notificar a los servicios de control de emergencias. * Si se trata de un sensor de ventana o puerta abierta sin autorización, se debe notificar a la policía sobre el evento, enviando como información adicional, la localización de la ventana dentro del inmueble y los sectores de la vivienda en los que se ha detectado movimiento. | |
| ***Artefactos Creados:***  ASHCentralSystem ->Es el proyecto principal donde se encuentra toda la lógica de negocio para procesar los eventos recibidos por cada casa u oficina.  ASHCommunication -> Librería que administra la comunicación entre el sistema central y el concentrador ubicado en cada casa u oficina.  ASHHomeModule ->Es el proyecto con la lógica para la simulación de los eventos generados por la casa u oficina.  ASHExternalSistem -> Sistema que emula la policía y los sistemas de emergencia.  Para la ejecución de esta parte del experimento se empleo como mensaje una trama definida como se muestra en la siguiente gráfica:  **Trama**  **Evento**  La trama transmite la información relacionada de los eventos detectados en cada vivienda. A su vez el evento contiene la información del tipo, código y estado del sensor de la vivienda. | |
| ***Criterio de terminación:***  El tiempo para la notificación de la información, una vez recibida la alarma, no debe ser superior a 1 segundo. | |
| ***Recursos Requeridos:***  Recurso humano -> Se necesita el trabajo de 4 personas con conocimientos en java, sockets, estilos y tácticas de arquitectura.  Recursos técnicos ->NetBeans IDE 7.0, Microsoft Visio, JUnit, JUnitPerf | |
| ***Duración estimada:***  32 horas | |

Tabla 2. Resultados del Experimento A1

|  |
| --- |
| **Resultados del Experimento** |
| ***Resumen de los resultados:***  Se implemento la arquitectura descrita anteriormente y se ejecuto el test de envio de informacion desde una casa.  Se monto un test que permite emular el envío del estado de los 50 sensores de una casa (Humo, Puertas, RFID), el estado de todos estos sensores es enviado periódicamente cada segundo.  El test se ejecuto 10 veces dando como resultado un promedio de 436ms desde el momento en que el sistema central recibe el mensaje y es interpretado para notificar a las autoridades correspondientes. |
| ***Duración Real:***  50 horas |
| ***Recursos Reales:***  Recurso humano -> Se necesita el trabajo de 3 personas con conocimientos en java, estilos y tácticas de arquitectura, bases de datos y diseño Web.  Recursos técnicos -> NetBeans IDE 7.0, Microsoft Visio |
| ***Recomendaciones:***  Se había implementado inicialmente una notificación por medio de e-mail al usuario, únicamente el envio del correo ya consumía alrededor de 10s, por lo tanto esta notificación fue eliminada y queda pendiente de análisis para ser reemplazada.  La comunicación con las autoridades se realiza por medio de sockets directamente a sus centrales correspondientes. |

* 1. **A2 – Consulta de Información Consolidada**

Tabla 3. Descripción del Experimento A2-1

|  |  |
| --- | --- |
| **Descripción del Experimento** | |
| ***Título*:**Latencia | ***ID*:**A2-1 |
| ***Descripción*:** Este experimento busca verificar el desempeño del sistema en el generación de reporte consolidado de eventos generados. | ***Responsable*:**Ingenium |
| ***Propósito:***  ( ) Reparación, actualizar, clarificar  ( X ) Obtener Información técnica  ( ) Obtener información de negocio  ( ) Otros: | |
| ***Propósito:***  Verificar las decisiones de diseño asociada a la latencia del sistema. | |
| ***Descripción del experimento:***  Verificar el tiempo necesario para procesar la solicitud de un reporte consolidado de los 400 eventos registrados (ordenadas por tipo de evento) de forma cronológica (hora, día, mes, año). | |
| ***Artefactos Creados:***  ASHWebModule ->Sistema Web que permite el procesamiento para generar el reporte consolidado de eventos presentados ante el sistema central. Posee una conexión con la base de datos de donde obtiene la información de los eventos. | |
| ***Criterio de terminación:***  Este reporte no debe tomar más de 2 segundos en ser presentado al usuario. | |
| ***Recursos Requeridos:***  Recurso humano -> Se necesita el trabajo de 2 personas con conocimientos en java, estilos y tácticas de arquitectura, bases de datos y diseño Web.  Recursos técnicos -> Base de datos Oracle XE, NetBeans IDE 7.0, Microsoft Visio, Navegador, editor SQL Depeloper | |
| ***Duración estimada:***  16 horas | |

Tabla 4. Resultados del Experimento A2-1

|  |
| --- |
| **Resultados del Experimento** |
| ***Resumen de los resultados:***  Se realizaron pruebas por medio de la Suite de pruebas de Microsoft Visual Studio, generando el reporte con 400 eventos registrados en base de datos, en un tiempo de 0,442 segundos. |
| ***Duración Real:***  19 horas |
| ***Recursos Reales:***  Recurso humano -> Se necesita el trabajo de 3 personas con conocimientos en java, estilos y tácticas de arquitectura, bases de datos y diseño Web.  Recursos técnicos -> Base de datos Oracle XE, NetBeans IDE 7.0, Microsoft Visio, Navegador, editor SQL Depeloper |
| ***Recomendaciones:***  Verificar el experimento A2-2 |

Tabla 5. Descripción del Experimento A2-2

|  |  |
| --- | --- |
| **Descripción del Experimento** | |
| ***Título*:** Latencia | ***ID*:** A2-2 |
| ***Descripción*:** Este experimento busca verificar el desempeño del sistema en el generación de reporte consolidado de eventos generados. | ***Responsable*:**Ingenium |
| ***Propósito:***  ( ) Reparación, actualizar, clarificar  ( X ) Obtener Información técnica  ( ) Obtener información de negocio  ( ) Otros: | |
| ***Propósito:***  Verificar las decisiones de diseño asociada a la latencia del sistema. | |
| ***Descripción del experimento:***  Verificar el tiempo necesario para procesar la solicitud de un reporte consolidado de los 4800 eventos registrados (ordenadas por tipo de evento) de forma cronológica (hora, día, mes, año). | |
| ***Artefactos Creados:***  ASHWebModule -> Sistema Web que permite el procesamiento para generar el reporte consolidado de eventos presentados ante el sistema central. Posee una conexión con la base de datos de donde obtiene la información de los eventos. | |
| ***Criterio de terminación:***  Este reporte no debe tomar más de 2 segundos en ser presentado al usuario. | |
| ***Recursos Requeridos:***  Recurso humano -> Se necesita el trabajo de 2 personas con conocimientos en java, estilos y tácticas de arquitectura, bases de datos y diseño Web.  Recursos técnicos -> Base de datos Oracle XE, NetBeans IDE 7.0, Microsoft Visio, Navegador, editor SQL Depeloper | |
| ***Duración estimada:***  2 horas | |

Tabla 6. Resultados del Experimento A2-2

|  |
| --- |
| **Resultados del Experimento** |
| ***Resumen de los resultados:***  Se realizaron pruebas por medio de la Suite de pruebas de Microsoft Visual Studio, generando el reporte con 4800 eventos registrados en base de datos, en un tiempo de 4,165 segundos (Ver Recomendaciones).  Es importante anotar que el tiempo se aumentó con relación al experimento A2-1 debido al tiempo adicional tomado por el navegador en realizar el renderizado de todos los datos La información sale de la base de datos en 0,6 segundos. |
| ***Duración Real:***  3 horas |
| ***Recursos Reales:***  Recurso humano -> Se necesita el trabajo de 3 personas con conocimientos en java, estilos y tácticas de arquitectura, bases de datos y diseño Web.  Recursos técnicos -> Base de datos Oracle XE, NetBeans IDE 7.0, Microsoft Visio, Navegador, editor SQL Depeloper |
| ***Recomendaciones:***  Para mejorar el rendimiento del tiempo en la generación del reporte es recomendable realizar una paginación en presentación y realizar paginación por base de datos, es decir, utilizar el patrón Acceso a recursos temprana (LazyAcquisition). Esta recomendación puede tomar valor en el crecimiento de registros en base de datos de los eventos generados. |

1. **Experimento B – Escalabilidad**

El objetivo de este experimento es determinar la escalabilidad del sistema. Para ello usted deberá utilizar una herramienta para análisis de desempeño y escalabilidad como JUnitPerf, para simular la ocurrencia de múltiples eventos concurrentemente.

* 1. **B1 – Escalabilidad del Sistema**

Tabla 6. Descripción del Experimento B1

|  |  |
| --- | --- |
| **Descripción del Experimento** | |
| ***Título*:** Escalabilidad | ***ID*:** B1 |
| ***Descripción*:** Este experimento busca verificar la escalabilidad del sistema | ***Responsable*:**Ingenium |
| ***Propósito:***  ( ) Reparación, actualizar, clarificar  ( X ) Obtener Información técnica  ( ) Obtener información de negocio  ( ) Otros: | |
| ***Propósito:***  Verificar las decisiones de diseño asociadas a la escalabilidad del sistema. | |
| ***Descripción del experimento:***  En este experimento debe simular la recepción de información proveniente de los tags RFID que se encuentran en cada uno de los inmuebles. Suponga que por cada inmueble u oficina se tienen en promedio 50 tags RFID enviando información cada segundo a cuatro antenas ubicadas en diferentes posiciones de la casa u oficina. Estas antenas a su vez consolidan la información y la envían a la central. | |
| ***Artefactos Creados:***  ASHCentralSystem ->Es el proyecto principal donde se encuentra toda la lógica de negocio para procesar los eventos recibidos por cada casa u oficina.  ASHCommunication -> Librería que administra la comunicación entre el sistema central y el concentrador ubicado en cada casa u oficina.  ASHHomeModule ->Es el proyecto con la lógica para la simulación de los eventos generados por la casa u oficina.  ASHExternalSistem -> Sistema que emula la policía y los sistemas de emergencia. | |
| ***Criterio de terminación:***  Cuando se detecta que un tag RFID ha violado una de las reglas definidas para su comportamiento, se debe generar una alarma indicando a todas las porterías y a los responsables de la vivienda u oficina lo ocurrido. Esto debe suceder en un tiempo inferior a un segundo. | |
| ***Recursos Requeridos:***  Recurso humano -> Se necesita el trabajo de 3 personas con conocimientos en java, sockets, estilos y tácticas de arquitectura.  Recursos técnicos ->NetBeans IDE 7.0, Microsoft Visio, JUnit, JUnitPerf | |
| ***Duración estimada:***  8 horas | |

Tabla 7. Resultados del Experimento B1

|  |
| --- |
| **Resultados del Experimento** |
| ***Resumen de los resultados:***  A continuación se presenta la gráfica de tiempo contra cantidad de usuarios concurrentes. Se puede notar que a medida que se incrementa el número de usuarios concurrentes, el tiempo de respuesta aumenta cada vez más rápido.    En la siguiente gráfica se puede notar como el número de fallas (una falla equivale a que un evento se procesó con un tiempo superior a un segundo) no aumenta hasta llegar a un punto donde se dispara repentinamente (entre 2400 a 2600 usuarios concurrentes). Este punto es importante, porque es el que determina la capacidad que tiene el sistema de responder a un número de usuarios concurrentes manteniendo las reglas del negocio (un segundo). |
| ***Duración Real:***  12 horas |
| ***Recursos Reales:***  Recurso humano -> Se necesita el trabajo de 2 personas con conocimientos en java, sockets, estilos y tácticas de arquitectura.  Recursos técnicos ->NetBeans IDE 7.0, Microsoft Visio, JUnit, JUnitPerf |
| ***Recomendaciones:***  Cuando se requiera aumentar el número de usuarios concurrentes más allá de lo que puede responder el sistema, se puede hacer uso de tácticas de escalabilidad / desempeño como por ejemplo el balanceo de cargas. |