Universidad EAFIT

Departamento de informática y sistemas

Tópicos Especiales en Telemática

Reto No. 1

Análisis, Diseño e Implementación de un chat sencillo utilizando Sockets

Desarrollado por:

José Mateo Vidal Barrera

Rubén Darío Espinosa Roldán

Medellín - Colombia

Domingo, 28 de Agosto del 2011

**Tabla de contenido**

**Tema Página**

1. Introducción………………………………………………….3
2. Análisis y diseño……………………………………………..4
   1. Diagramas………………………………………………...6
3. Instalación y requerimientos de software………………….9
4. Requerimientos desarrollados…………………………..…10
5. Referencias……………………………...............................11

**Introducción**

Se desarrolla el análisis, diseño e implementación de un chat sencillo mediante sockets, en el cual existe una sala donde pueden haber una cantidad n de usuarios conectados con opción de enviar mensajes públicos (todos los usuarios que estén conectados pueden ver lo que escriba) o mensajes privados (donde se especifica a que usuario conectado se le va a mandar un mensaje); los mensajes son cortos, máximo 160 caracteres, tipo twittero SMS.

En este documento está desarrollado todo el trabajo de análisis, diseño, instalación y requerimientos de software, implementación, referencias y conclusiones; todo esto con el fin de tener la mejor comprensión del problema y las mejores retroalimentaciones posibles. Se observan todas las especificaciones de la arquitectura cliente/servidor, para los diferentes modelos que se podrían desarrollar se tienen una seria de especificaciones, como lo son: interacción asincrónica y sincrónica, interacción simétrica y asimétrica, manejo de sesión, etc.

**Análisis y Diseño**

La aplicación se debe implementar bajo la arquitectura cliente/servidor, en el cual existe un servidor y pueden haber n clientes conectados a este. Un cliente envía un mensaje, si es público, el servidor lo recibe y lo reenvía hacía todos los usuarios que estén conectados en el momento, si el mensaje es privado, el servidor lo recibe y lo reenvía hacia el destinatario correspondiente (mas adelante se verá dicho protocolo para enviar dichos tipos de mensajes). El mecanismo de comunicación entre el/los cliente(s) y el servidor es implementado mediante Sockets. El proceso de existencia de varios usuarios conectados a la misma vez es implementado por medio de hilos.

Toda la aplicacion (codigo fuente y documentación) esta alojada en un repositorio, el cual brinda un manejador de versiones distribuido, la direcciòn es: [www.github.com/rderoldan1/ruby\_chat](http://www.github.com/rderoldan1/ruby_chat)  
  
La implementación de la aplicación (chat) está codificada en el lenguaje de programación Ruby, se tiene un servicio orientado a la conexión (TCP), y usando protocolos de aplicación como modelo/middleware del sistema distribuido:  
  
Protocolo cliente-servidor

* Definición del servicio:

Implementar el proceso de envío y recepción de mensajes por parte de los clientes conectados a la aplicación y un determinado servidor, el cliente puede enviar mensajes por defecto que son públicos, es decir, todos los usuarios que estén conectados pueden ver el mensaje, y mensajes a un destinatario en específico. El servidor se ejecuta y queda en espera de una conexión con algún cliente, cuando un cliente se ejecuta, se solicita la dirección ip del servidor y el determinado puerto. Un usuario conectado tiene como opciones pedir la lista de todos los usuarios que están conectados en el momento, pedir ayuda acerca de los diferentes comandos que puedan existir en el chat, también el servidor informa al usuario si está solo en el chat y puede saber cuando un usuario se conecta o se desconecta.

* Suposiciones del entorno:

- Se maneja un modelo cliente/servidor.

- Un único servidor y un número inconstante de usuarios en el sistema.

- Un usuario manda un mensaje al servidor, y este dependiendo del tipo de mensaje realiza la respectiva decisión.

- Los usuarios están conectados al mismo servidor.

- En la ejecución de un usuario se solicita la dirección ip del servidor y el puerto.

* Vocabulario de mensajes:

- help: dar al usuario la lista de los posibles comandos a usar en el chat.

- list: mostrar al usuario la lista de todos los usuarios que están conectados.

- private: con esta linea el usuario le dice al servidor que desea enviar un mensaje privado, el servidor le preguntara un nombre de usuario para realizar el mensaje privado, en caso de que el usuario no exista el servidor le informara.

- quit: el usuario debe escribir este comando para salir del chat, este comando informara a todos los clientes cuando un usuario salga.

* Codificación de mensajes:

Mensaje = { tipo de mensaje }

kindMessage

→ kindMessage =~ /list/  
→ kindMessage =~ /private/

→ kindMessage =~ /help/

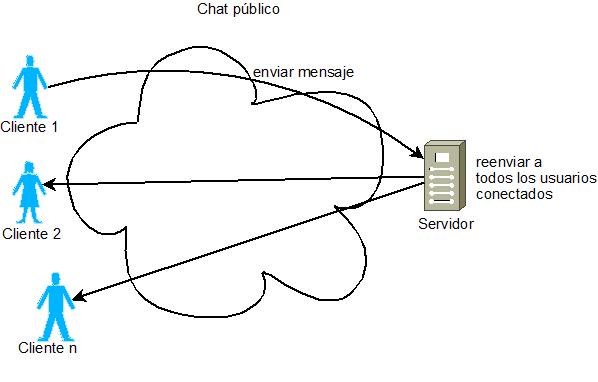
→ kindMessage =~ /quit/

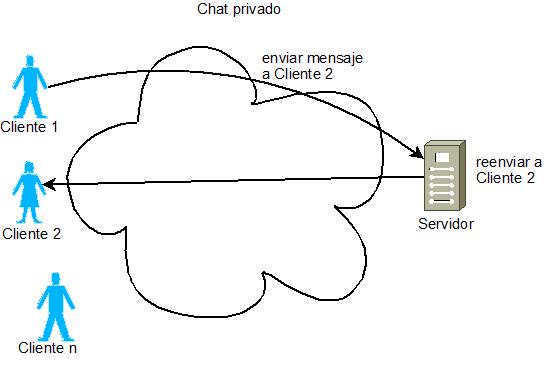
* Reglas de procedimiento:

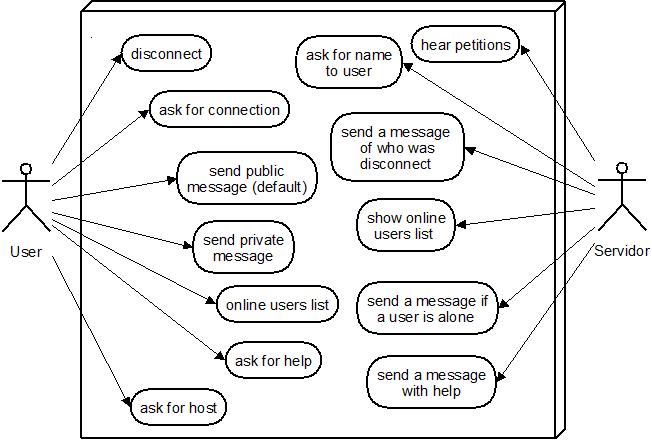
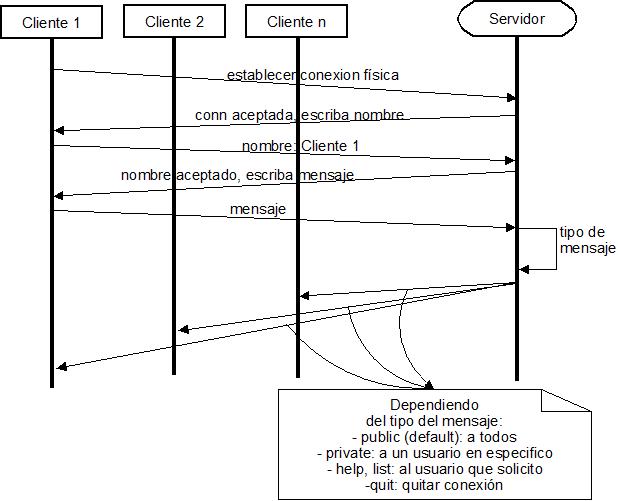
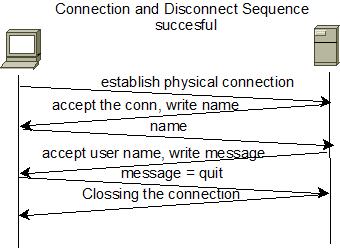
1. El usuario debe ingresar la ip del servidor y el host, si por algún motivo esto no esta correcto, el usuario no podrá establecer la conexión con el servidor.

2. Si el usuario ingresa correctamente los comandos definidos en el vocabulario de mensajes, el servidor hará específicamente lo que tiene hacer con cada comando.

3. Si el usuario desea ingresar algún comando definido en el vocabulario de mensajes, y por alguna razón hubo un error, el servidor lo tomará como un mensaje por defecto a todos los usuarios conectados.

Diagramas





Instalación y requerimientos de software

La aplicación fue desarrollada en el ambiente Ubuntu, para correrlo basta con seguir estos pasos, teniendo en cuenta que se debe tener instalado ruby  versión 1.9.2, puedes mirar que versión tienes ejecutando el siguiente comando en una consola: ruby -v.  
  
Si no tienes instalado ruby, sigue los siguientes pasos:  
  
1. Abrimos la consola de ubuntu  
2. cd /usr/src/  
3. sudo wget ftp://ftp.ruby-lang.org//pub/ruby/1.9/ruby-1.9.2-p180.tar.gz  
4. sudo tar xvf ruby-1.9.2-p180.tar.gz  
5. cd ruby-1.9.2-p180  
6. sudo ./configure --enable-pthread  
7. sudo make  
8. sudo make install  
  
ahora que ya tenemos la última versión de ruby, vamos a actualizar las gemas GEM, que es algo parecido a apt-get:  
  
1. cd /usr/src/  
2. sudo wget http://production.cf.rubygems.org/rubygems/rubygems-1.7.2.tgz  
3. sudo tar xvf rubygems-1.7.2.tgz  
4. cd rubygems-1.7.2  
5. sudo ruby setup.rb/  
  
ya con el sistema listo procedemos a ejecutar ruby\_chat  
  
1. Abre la consola y abre la carpeta ruby\_chat:

cd /<nombre subdirectorio>.../ruby\_chat

2. Se debe correr primero el servidor, luego pedirá que ingrese un puerto, si escribe 'df' pondrá un puerto por defecto:  
 $ruby tcpserver2.rb → Ejecutar el servidor  
 $df                             → Puerto por defecto

3. Abre n consolas para ejecutar n clientes, luego pedirá ingresar la ip del sevidor, si estas corriendo el servidor en el mismo computador ingresa 'lh', luego pedirá un puerto, ingresa 'df' para entrar el puerto por defecto:  
 $ ruby tcpclient2.rb  → Ejecutar un cliente  
 $lh                              → “Localhost”, si se ejecuta el servidor en el mismo                                                                c                                                              computador  
 $df                              → Puerto por defecto

Requerimientos desarrollados

La siguiente lista son todos las funcionalidades que fueron desarrolladas en la aplicación:

* Se implementa bajo una arquitectura cliente/servidor.
* Ejecución de más de dos clientes y en diferentes máquinas.
* Implementación del mecanimos de comunicación mediante sockets.
* Los mensajes públicos están establecidos por defecto.
* Un usuario pueden mandar un mensaje privado a otro usuario conectado.
* Implementación bajo el servicio TCP y emulando protocolos de aplicación.
* Un usuario puede solicitar la lista de todos los usuarios conectados.
* Un usuario puede solicitar ayuda acerca de los diferentes comandos que puede usar.

**Referencias**

→ **Página oficial de ruby:**  
[www.ruby-lang.org](http://www.ruby-lang.org/)  
  
→ **Tutorial de ruby + sockets por IBM:**  
<https://www6.software.ibm.com/developerworks/education/l-rubysocks/l-rubysocks-a4.pdf>  
  
→ **Libro**  
Peter Cooper, Beggining Ruby -From novice to Professional- Second Edition editorial Apress  
  
→ **Tutorial para instalar ruby y las gemas**  
<http://www.taringa.net/posts/linux/11309756/ruby-1_9_2_-rails-3-ubuntu.html>