UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO

FACULTAD DE INGENIERÍA

Desarrollo de un sistema de comunicación interna SCI

INFORME DE ACTIVIDADES PROFESIONALES

Que para obtener el título de

Ingeniero en computación

P R E S E N T A

Guillermo Vázquez Alcaraz

ASESOR DE INFORME

Ing. Marco Antonio Martínez Quintana

Ciudad Universitaria, Cd. Mx., 2018

Índice

Contenido

Introducción y objetivo

Una problemática que se ha presentado dentro del hospital general de México “Dr. Eduardo Liceaga” es que la comunicación entre usuarios y personal del área de sistemas puede llegar a ser muy poco eficiente por diferentes factores como puede ser la falta de extensiones telefónicas para comunicarse a dicha área, poco personal de sistemas orientado a la solución de problemas directos con el usuario o la alta demanda de asistencia en sitio para solucionar problemas. Por lo cual se busca implementar un sistema desarrollado en un lenguaje de programación que ayude a solucionar el problema de comunicación entre usuario y personal.

El siguiente reporte contiene el desarrollo detallado de un sistema de comunicación para una mesa de servicios o levantamiento de reportes solicitados por los usuarios para el hospital general de México.

Para la realización del sistema de comunicación interna (SCI), es necesario mencionar que se requerirá el desarrollo de una arquitectura cliente servidor y la programación orientada a objetos que le permitirá a los usuarios interactuar directamente con los trabajadores del área de sistemas, así como con el personal de soporte técnico y redes mediante la propia red interna del hospital.

El objetivo general de la aplicación es poder poner en comunicación las áreas médicas y administrativas del hospital con el personal de las áreas de sistemas y soporte técnico y redes.

Historia del hospital general de México Dr. Eduardo Liceaga

El Hospital General de México fue inaugurado el 5 de febrero de 1905 por el presidente Porfirio Díaz, con la presencia del Dr. Liceaga y su primer director, el Dr. Fernando López, todo el personal que integraba este nuevo Hospital contaba con nombramiento firmado por el mismo presidente Díaz, así mismo cabe resaltar también que los gastos de los servicios públicos de salud comenzaron a tomar en cuenta en el presupuesto de egresos de la federación.

La Institución desde sus inicios, funcionó como establecimiento de beneficencia a cargo del Poder Ejecutivo de la Secretaría de Estado y Gobernación para la asistencia gratuita de enfermos indigentes sin importar edad, sexo, raza, nacionalidad ni creencias religiosas, principios que a la fecha perduran y motivan el trabajo que se desarrolla en el Hospital.

En 1906, el Hospital comienza su historia como parte fundamental en la formación de profesionales en la salud en México al establecer la primer Escuela de Enfermería del país, inaugurada formalmente el 3 de octubre de 1906.

La primer Sociedad Médica del HGM, fundada el 7 de febrero de 1908, nace de la necesidad de agruparse para intercambiar experiencias ante el constante desarrollo del trabajo que se realizaba en el Hospital, la mesa directiva se conformó por los doctores Eduardo Liceaga y Fernando López como presidentes honorarios; el Doctor Julián Villareal, presidente; Doctor Eduardo Lamic, vicepresidente; Doctor Luis Troconis Alcalá, secretario perpetuo, y como segundo secretario, el Doctor Miguel Mendizábal.

Misión

Hospital Regional de la zona centro del país que proporciona servicios de salud de alta especialidad con gran calidad y calidez, en las especialidades médicas, quirúrgicas y de apoyo al diagnóstico y tratamiento, por lo que tiene el reconocimiento de la sociedad mexicana. Hospital formador de recursos humanos de excelencia para la salud del país y a nivel internacional. Realiza investigación de alto nivel cuyos resultados se difunden en publicaciones científicas de impacto internacional, se preocupa por sus recursos humanos capacitándolos formando equipos como una necesidad de las actividades humanas.

Visión

Ser un centro hospitalario con reconocimiento nacional y de referencia internacional, generador de modelos de atención en las especialidades médicas, en la enseñanza de la medicina y en proyectos de investigación con alto rigor científico. Participante en las políticas sectoriales, principalmente en el Seguro Popular y del Fondo Directo de Gastos Catastróficos en Salud y apoyando la formación de recursos humanos de alta calidad y de modelos de atención a la salud, que impacte en los indicadores básicos de salud, con aportaciones para la disminución de los problemas relacionados con el rezago social y que propicie el ataque oportuno a los factores casuales de los problemas emergentes, debiéndose mantener a la vanguardia.

Descripción del puesto de trabajo

Anteriormente se agrega una breve historia del hospital general de México así como también la misión y la visión del mismo.

Ahora se describirá el área específica de trabajo en la cual desarrollo mis actividades diarias, indicando la misión y visión de la misma aunque he participado en diferentes subáreas de la dirección en la que me encuentro.

Subdirección de sistemas administrativos

Misión

Fomentar y desarrollar la mejora de los procesos de planeación, sistematización, evaluación y registro de la información de los servicios y unidades del hospital, en apego a los planes y programas institucionales que redunden en el mejoramiento continuo de la calidad de atención de los pacientes del hospital.

Visión

Contar con los recursos humanos, técnicos, financieros y materiales de alto nivel para mejorar la planeación y sistematización de la información de los servicios y unidades del hospital, que conlleven al mejoramiento de la calidad en la atención a los usuarios

Objetivo

Proporcionar servicios informáticos de implementación y soporte, a través de soluciones en materia de Tecnologías de la Información y Comunicaciones (TIC), para contribuir a mejorar las actividades asistenciales, administrativas, de educación e investigación en alineación con los programas y objetivos fijados por el organismo con un enfoque centrado en el paciente.

Funciones

1. Evaluar la información sobre las necesidades de sistemas informáticos de las áreas médicas y administrativas, para desarrollar módulos informáticos o gestionar la provisión de software.

2. Diseñar, desarrollar e implementar los sistemas automatizados, sus requerimientos de interconexión que satisfagan con eficiencia, eficacia y

economía, las necesidades de las unidades médicas y administrativas, para la toma de decisiones de cada área.

3. Proporcionar la capacitación en el uso de los módulos informáticos desarrollados al personal de las distintas áreas donde son implementados para facilitar su puesta en marcha.

4. Promover la provisión de bienes informáticos para las unidades médicas y administrativas del hospital, requeridos para satisfacer las necesidades de equipamiento informático y la correcta implementación de proyectos especiales.

5. Formular los reportes estadísticos, informes de gestión e informes de avances de las diferentes áreas del hospital, para conformar el sistema interno de información

6. Supervisar el desarrollo de módulos informáticos para fortalecer el sistema integral de administración hospitalaria que permita solventar en tiempo y forma los requerimientos institucionales.

7. Supervisar la publicación de información proporcionada por las áreas médicas y administrativas del hospital en el portal Web institucional, para poner a disposición de los pacientes información de interés público.

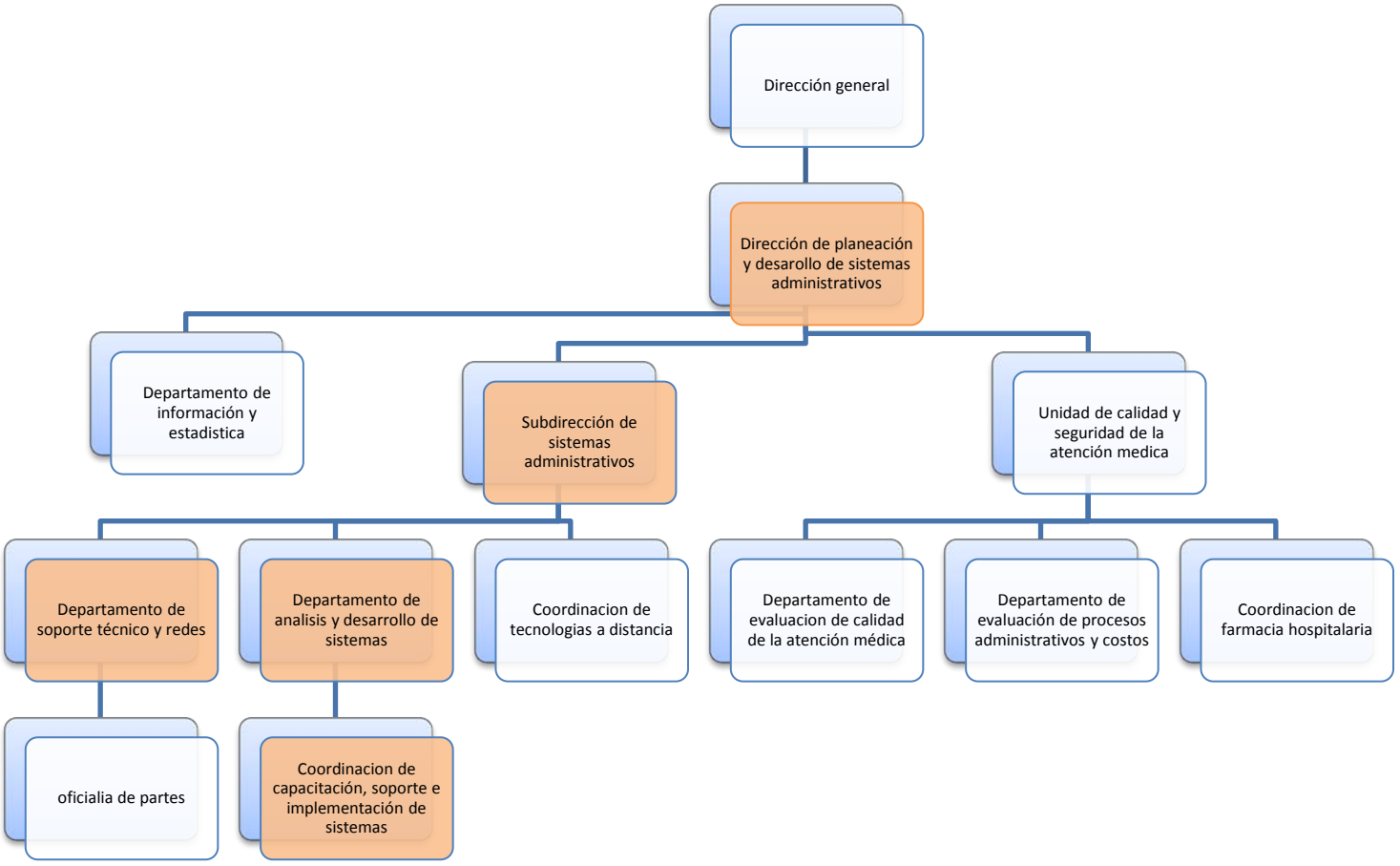
8. Supervisar los servicios de mantenimiento a los bienes informáticos donde se procesa información médica y administrativa, para vigilar la adecuada prestación de servicios por parte del proveedor.

9. Proporcionar apoyo y asesoría técnica a las diversas áreas médicas y administrativas del hospital para la provisión de bienes e insumos informáticos que resulten más convenientes para la operación de sus procesos.

10. Supervisar los documentos que se reciben a través del Sistema de Administración de Correspondencia (SAC) provisto por la DGTI, verificando que los usuarios responsables utilicen el sistema y atiendan los oficios, para desahogar las peticiones solicitadas del hospital.

Organigrama

A continuación, se muestra el organigrama resaltando las áreas en las cuales he realizado actividades.



Antecedentes

Para comenzar con este punto es necesario definir las bases teóricas que me ayudaron a la realización del proyecto y poder responder la siguiente pregunta:

¿Qué es la arquitectura cliente servidor y cómo implementarla?

Primeramente, hay que definir un poco de historia sobre la computación orientada a redes.

Historia de la computación orientada a redes

A principio de los años 60 solamente existían unas cuantas computadoras aisladas, el usuario tenía que estar cerca de la computadora porque los terminales, los únicos mecanismos de acceso al computador, estaban conectados al computador mediante un cable. La única posibilidad de acceso remoto era mediante el uso de una línea telefónica local.

El auténtico origen de Internet lo tenemos con ARPANet (Advanced Research Projects Agency Network o Red de la Agencia para los Proyectos de Investigación Avanzada de los Estados Unidos). En el año de su nacimiento, 1969, cuando nace ARPANET, aparece la primera conexión entre los ordenadores de Standford y UCLA.

Dos años más tarde, el primer envío de email lo protagonizó Ray Tomlinson. En ese año también aparece el primer virus Creeper.

A principios de los 70 nace la palabra Internet, que se aplicó al sistema de redes interconectadas mediante los protocolos TCP e IP (Protocolo de Control de Transmisión/Protocolo de Internet) en el que basaban los servicios de Internet y los mensajes de correo electrónico. A principios de los 80 se expanden los emoticonos, que a día de hoy siguen utilizándose.

No es hasta 1990 cuando Tim Berners-Lee, conocido como el padre de la web, crea el lenguaje HTML. El equipo del científico creó el primer cliente web “WorldWideWeb” (www).

Los nodos eran minicomputadoras Hopewell DDP-516 con 12K en memoria con líneas telefónicas de 50 kbps que se encontraban distribuidos de la siguiente manera:

Nodo 1: Establecido en UCLA (Septiembre).

Nodo 2: Establecido en Stanford Research Institute (SRI) (Octubre).

Nodo 3: Establecido en University of California Santa Barbaran (UCSB) (November).

Nodo 4: Establecido en University of Utah (December).

En 1983 dca (defensa comunication agency) y darpa establecen el transmission control protocol (tcp) e internet protocol (ip) y el conjunto de protocolos conocidos como tcp/ip. En 1983 arpanet se divide en arpanet y milnet. the military network, milnet. 68 nodos de los 113 fueron mudados a milnet.

A partir de su lanzamiento y el gran alcance de usuarios en Internet, la red informática mundial nos ha dado plataformas que todavía siguen ayudando a todos sus usuarios tanto a nivel personal, profesional y educativo.

Yahoo aparece en 1994, que un año más tarde se convertiría en empresa con el fin de “ser el servicio global de Internet más esencial para consumidores y negocios”.

Netscape anunció el 13 de octubre de 1994 que lanzaría un navegador disponible de forma gratuita para todos sus usuarios no comerciales, y que las versiones beta 1.0 y beta 1.1 se podrán descargar en noviembre de 1994 y marzo de 1995. La versión 1.0 final estuvo disponible en diciembre de 1994. Netscape hizo gratuita la disponibilidad de su software porque tenía en sus políticas la noción de que el software para Internet no debía tener costo. En 1997, el Netscape Navigator 2.0 fue el primer navegador en incluir un lenguaje de script en las páginas web, al introducir JavaScript en su versión 2.

Internet Explorer nace gracias a Microsoft para el sistema operativo Windows 95 y, en 2015, se anunciaba que a partir de Windows 10, se sustituía por Microsoft Edge.

A finales de los 90, dos creaciones cerraron el siglo: el término weblog otro de los términos informáticos de finales de los 90 es “weblog” y el nacimiento de Google , que provocó el crecimiento de usuarios en la red.

En la actualidad las redes evolucionan a una velocidad significativa. Constantemente aparecen nuevos protocolos, aplicaciones y dispositivos que mejoran las comunicaciones en diferentes niveles.

Conceptos básicos

Es necesario conocer los conceptos básicos sobre redes, ya que la arquitectura cliente servidor está basada en todos aquellos conceptos tales como:

¿Qué es un dato?

Sabemos que es la unidad mínima de información haciendo referencia a que con esto se aclara que es un tipo de información que no tiene contexto y no está procesada aun, dando apertura al concepto de información, que es un conjunto de datos que tiene un contexto y se encuentran ya procesados.

Analizando lo anterior se puede comenzar a juntar el concepto de redes y arquitecturas cliente servidor ya que es importante definir lo que es un tráfico de datos para poder comprender como se estructura el proyecto desarrollado.

Tráfico de datos se refiere a la transferencia de datos a través de un medio, por lo tanto, en redes hace referencia al tráfico de datos a través de una red.

Una vez aclarado lo anterior es importante mencionar los diferentes tipos de flujos por los cuales puede viajar la información.

Retomando con la explicación de los diferentes tipos de flujos de comunicación podemos listar los siguientes:

- Simplex: Envío de datos en una sola dirección.
- Half Duplex: Envío de datos en ambas direcciones pero utilizando un canal a la vez.
- Full duplex: Envío de datos en ambas direcciones al mismo tiempo.

Donde basado en que ya estamos formando un envío y recepción de información entre 2 o más direcciones es necesario definir lo que es un emisor, receptor, mensaje y un medio, ya que estos son los principales conceptos para que pueda realizarse una comunicación entre varias terminales.

- Emisor: Terminal, persona u objeto que envía la información a través de un medio.
- Receptor: terminal, persona u objeto que recibe la información a través de un medio.
- Mensaje: Conjunto de datos con o sin formato que viaja en un medio.
- Medio: Dispositivo o medio físico por el cual viaja la información o mensaje.

Pasando al análisis de conceptos de redes de datos, primordialmente es necesario destacar lo que es una red, sus tipos, las topologías que se emplean para poder tener una red de manera óptima, así como las organizaciones principales que rigen las redes de datos para una mejor implementación y poder comenzar a definir lo que es el modelo OSI.

Comenzare con lo siguiente:

¿Qué es un socket?

Un socket es un método para la comunicación combinando una dirección IP (la dirección numérica irrepetible en una red local de cuatro partes que identifica a un ordenador particular) y un número de puerto (Un número que especifica por donde se va a enviar y recibir la información).

Existen 2 tipos de sockets que se utilizan en TCP y en UDP manejando en cada uno de ellos orientado a conexión y no orientado a conexión respectivamente.

- Orientados a conexión:

Establece un camino virtual entre servidor y cliente, fiable, sin pérdidas de información ni duplicados, la información llega en el mismo orden que se envía.

El cliente abre una sesión en el servidor y este guarda un estado del cliente.

  - El cliente utiliza la clase Socket
  - El servidor utiliza la clase ServerSocket
  - Estas clases se encuentran definidas en el paquete java.net

- No orientados a conexión:

Envío de datagramas de tamaño fijo. No es fiable, puede haber pérdidas de información y duplicados, y la información puede llegar en distinto orden del que se envía.

No se guarda ningún estado del cliente en el servidor, por ello, es más tolerante a fallos del sistema.

Tanto el cliente como el servidor utilizan la clase DatagramSocket en el caso de programación con java y se encuentran en el paquete java.net.

Previamente a la explicación de los diferentes tipos de flujos de comunicación cabe resaltar un concepto básico en redes de datos que es el de “protocolo”, ya que este lleva las reglas que manejan la comunicación de los datos entre diferentes terminales.

Un protocolo de red, es un término utilizado en el mundo de la informática, para dar nombre a una serie de normas y criterios, los cuales, son utilizados para mantener una comunicación entre los ordenadores que forman parte de una red informática, es decir, entre los ordenadores que se encuentran conectados entre sí por cualquier sistema de comunicación

Una vez definido lo anterior es necesario describir las diferentes topologías con las que trabaja una red, debido a que el hospital cuenta con una estructura que vale tomar en cuenta para la realización del proyecto sin afectar las áreas o la misma estructura de la red.

Para proceder con las bases teóricas es importante definir lo siguiente:

¿Qué es una red?

El concepto más simple que se le puede dar es:

Un conjunto de dispositivos interconectados que envían y reciben información a través de un medio y que se pueden emplear utilizando diferentes topologías como:

TOPOLOGÍA EN ANILLO

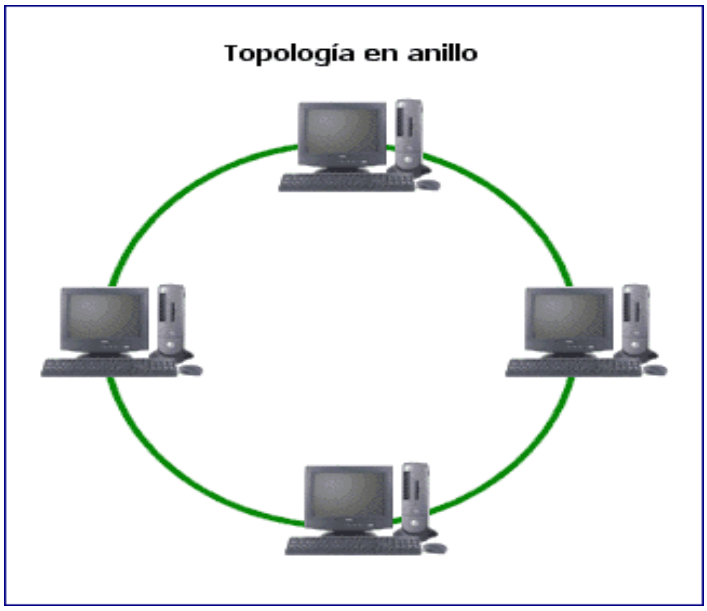


Imagen 1 Topología anillo

Los nodos de la red se disponen en un anillo cerrado conectados a él mediante enlaces punto a punto.

La información describe una trayectoria circular en una única dirección y el nodo principal es quien gestiona conflictos entre nodos al evitar la colisión de tramas de información.

En este tipo de topología, un fallo en un nodo afecta a toda la red, aunque actualmente hay tecnologías que permiten mediante unos conectores especiales, la desconexión del nodo averiado para que el sistema pueda seguir funcionando.

La topología de anillo está diseñada como una arquitectura circular, con cada nodo conectado directamente a otros dos nodos.

TOPOLOGÍA BUS

En la topología de bus todos los nodos (computadoras) están conectados a un circuito común (bus). La información que se envía de una computadora a otra viaja directamente o indirectamente, si existe un controlador que enruta los datos al destino correcto.

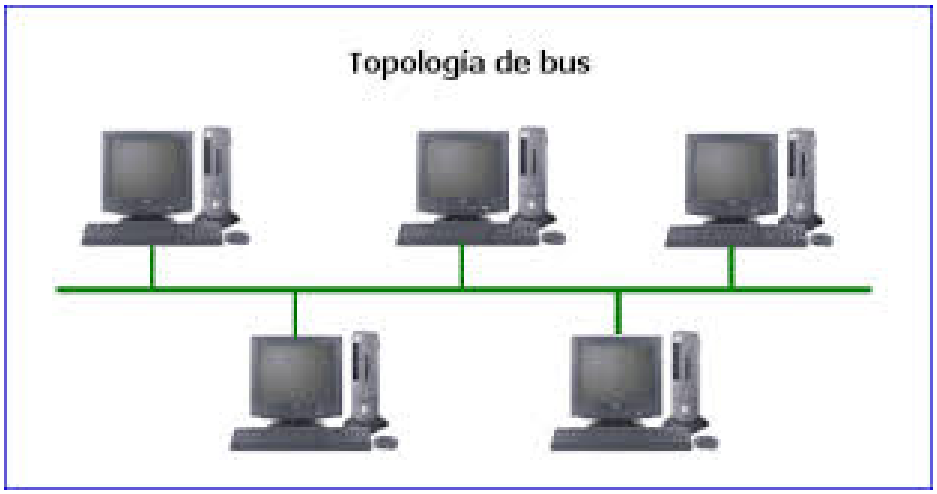


Imagen 1.1 Topología bus.

TOPOLOGÍA ESTRELLA

En la topología estrella todas las computadoras están conectadas a un concentrador central desde el cual se redireccionan los datos al computador adecuado. En este caso es una topología estrella pasiva, pues el concentrador central es solo un dispositivo con muchos puertos.
Si la función del concentrador central lo realiza una computadora es una topología estrella activa. En este caso la computadora regenera la señal y la envía a su destino. Estas computadoras muchas veces funcionan como servidores y realizan labores estadísticas.

La ventaja de la topología estrella es que, si una computadora o nodo falla, esta no afecta el funcionamiento del resto de la red, pero si el concentrador central o la computadora que hace la función de concentrador falla, falla toda la red.

Imagen 1.2 Topología estrella.

TOPOLOGÍA EN ÁRBOL

La topología en árbol es una variante de la de estrella.

Como en la estrella, los nodos del árbol están conectados a un concentrador central que controla el tráfico de la red. Sin embargo, no todos los dispositivos se conectan directamente al concentrador central.

La mayoría de los dispositivos se conectan a un concentrador secundario que, a su vez, se conecta al concentrador central.

Imagen 1.3 Topología en árbol.

TOPOLOGÍA EN MALLA

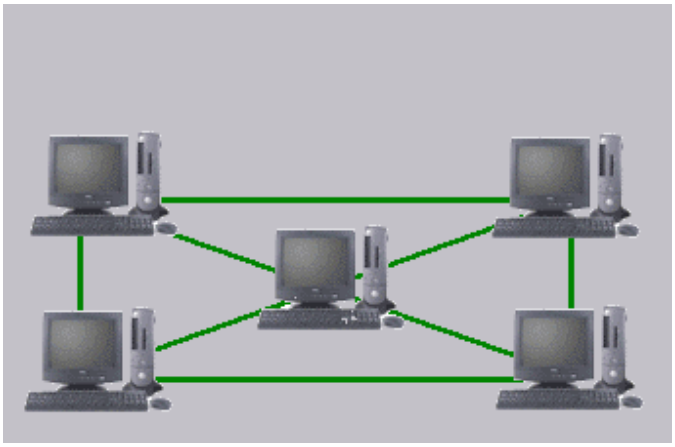


Imagen 1.4 Topología en malla

La topología en malla es una topología de red en la que cada nodo está conectado a ¦todos los nodos. De esta manera es posible llevar los mensajes de un nodo a otro por diferentes caminos.

Si la red de malla está completamente conectada, no puede existir absolutamente ninguna interrupción en las comunicaciones.

Cada servidor tiene sus propias conexiones con todos los demás servidores.

Una vez definidas las topologías, vale la pena conocer algunas de las organizaciones que estandarizan y procuran que las redes de datos sean implementadas de la mejor manera y también las capas del modelo OSI que rigen las redes de datos.

ESTÁNDARES

ISO (International Organization for Standarization)

La Organización Internacional para la Estandarización (ISO) es una federación de alcance mundial integrada por cuerpos de estandarización nacionales de 153 países, uno por cada país.

La ISO es una organización no gubernamental establecida en 1947. La misión de la ISO es promover el desarrollo de la estandarización y las actividades con ella relacionada en el mundo con la mira en facilitar el intercambio de servicios y bienes, y para promover la cooperación en la esfera de lo intelectual, científico, tecnológico y económico.

Imagen 1.5 Estandar ISO

IEEE (Institute of Electrical & Electronics Engineers)

Fue formado en 1963 por la fusión del IRE (Institute of Radio Engineers), fundado en 1912 y el AIEE (The American Institute of Electrical Engineers), fundado en 1884.

IEEE(The Institute of Electrical and Electronics Engineers), es un instituto internacional sin fines de lucro dedicado a promover la innovación y la excelencia tecnológica en beneficio de la humanidad.

Imagen 1.6 Estandar IEEE

ITU (International Telecomm Union)

La UIT fue fundada en París en 1865 con el nombre de Unión Telegráfica Internacional. En 1932 adoptó su nombre actual, y en 1947 se convirtió en organismo especializado de las Naciones Unidas. Su primer ámbito de especialización fue el telégrafo, pero hoy la

UIT abarca todo el sector de las TIC, desde la radiodifusión digital a Internet, y de las tecnologías móviles a la TV 3D.

Imagen 1.7 Estandar ITU

ANSI (American National Standard Institute)

ANSI, fundada en el año 1969 es una organización encargada de supervisar el desarrollo de normas para los servicios, productos, procesos y sistemas en los Estados Unidos. El ANSI forma parte de la Organización Internacional para la Estandarización (ISO) y de la Comisión Electrotécnica Internacional (IEC).

Imagen 1.8 Estandar ANSI

EIA/TIA (Electrical Industry Association / Telecomm Industries Association)

El estándar de cableado estructurado TIA / EIA definen la forma de diseñar, construir y administrar un sistema de cableado que es estructurado, lo que significa que el sistema está diseñado en bloques que tienen características de rendimiento muy específicos.

Imagen 1.9 Estándares EIA/TIA

IETF(Internet Engineering Task Force) generan los draft y RFC's

Comunidad internacional abierta de diseñadores de redes, operadores, vendedores e investigadores preocupados por la evolución de la arquitectura de Internet y el buen funcionamiento de Internet. Está abierta a cualquier persona interesada. La misión de la IETF está documentada en el RFC 3935.

Imagen 2.0 Estandar IETF

Una vez conocidas las instituciones que estandarizan y aplican normas que aplican para este proyecto es importante definir: ¿Cuáles son las capas del modelo OSI?

Aplicación - Interacción entre el usuario. (Datos)

Dos ordenadores se intercomunican a través de procesos, correspondiente a unas determinadas aplicaciones.

Presentación - La parte de presentación de los datos. (Datos)

Trata de homogeneizar los formatos de representación de los datos entre equipos de la red.

Para homogeneizar la representación de datos (Textos, Sonidos, imágenes, valores numéricos, instrucciones), la capa de presentación interpreta las estructuras de las informaciones intercambiadas por los procesos de la aplicación y las transforma convenientemente.

Sesión - Cuando ya se establece la conexión entre dos terminales. (Datos)

Cuando se realiza una transferencia entre dos ordenadores se establece una sesión de comunicaciones entre ambos. La capa de sesión es responsable de:

- Actuar de interfaz entre el usuario y la red, gestionando el establecimiento de la conexión entre procesos remotos.
- Establecer un dialogo entre dos equipos remotos para controlar la forma en que se intercambian los datos.
- Identificar los usuarios de procesos remotos
- Cuando se corta la conexión de forma anormal, en la capa de transporte o en inferiores, la capa de sesión puede encargarse de restablecer la sesión de forma transparente al usuario.

Su función es aumentar la fiabilidad de la comunicación obtenible por las capas inferiores, proporcionando el control de la comunicación entre aplicaciones al establecer, gestionar y cerrar sesiones o conexiones entre las aplicaciones que se comunican.

Transporte - (TCP o UDP). (segmentos)

Se encarga del transporte de la información, desde la fuente al destino, a través de la red.

Los accesos a la capa de transporte se efectúan a través de puertos (sockets).

Red - Direcciones IP. (paquetes)

Se encarga de Fragmentar los segmentos que se transmiten entre dos equipos de datos en unidades denominadas paquetes. En el ordenador receptor se efectúa el proceso inverso: los paquetes se ensamblan en segmentos.

También se encarga del encaminamiento de los paquetes de la red para elegir las rutas más óptimas por las cuales viajara la información.

Enlace de datos - Transformación de señales físicas en señales digitales. (frames)

Descompone los mensajes que recibe del nivel superior en tramas o bloques de información, en las que añade una cabecera (DH) e información redundante para control de errores. La cabecera suele contener información de direcciones de origen y destino, ruta que va a seguir la trama, etc.

Física - Transferencia de bits (interfaces). (bits)

Es donde se especifican los parámetros mecánicos (grosor de los cables, tipo de conectores), eléctricos (temporizador de las señales, niveles de tensión) de las conexiones físicas.

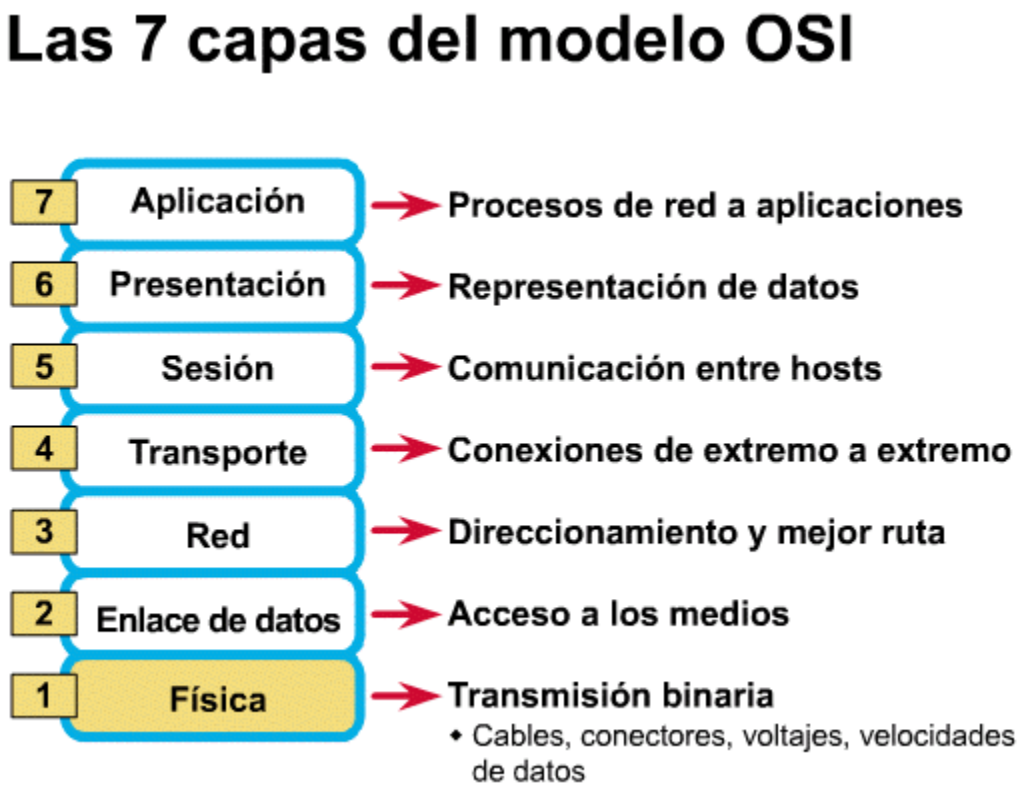


Imagen 2.1 Capas del modelo OSI

Una vez definidos los conceptos básicos definiremos el concepto de arquitectura cliente servidor que es la idea con la que se basa el proyecto principal.

¿Qué es una arquitectura cliente/servidor?

La comunicación de dos hosts se realiza generalmente mediante la filosofía Cliente/Servidor.

El usuario cliente obtiene servicios de la máquina remota proveedora de un servicio (servidor) y el servidor proporciona un puerto de comunicación por el cual se deben de conectar todos los clientes que deseen obtener dicho servicio, después se establece un socket en la máquina local (cliente) y otro en la máquina remota (servidor), y se comunican entre sí por el puerto proporcionado, así se establece la vía de comunicación entre dos hosts interconectados por una red.

Después de este proceso cabe resalta y describir individualmente el proceso que realiza el cliente y el servidor.

PROCESO CLIENTE.

- Abre el canal de comunicaciones para conectarse a la dirección de red atendida por el servidor.
- Enviar al servidor un mensaje de petición de servicio y esperar hasta recibir respuesta.
- Cerrar el canal de comunicación y terminar la ejecución del proceso.

PROCESO SERVIDOR.

- Abre el canal de comunicación e informa a la red de la dirección a la que responderá como de la disposición para aceptar peticiones de servicio.
- Espera a que el Cliente realice una petición de servicio en la dirección que él tiene declarada.
- Cuando recibe una petición de servicio, atiende al Cliente.
- La conexión es cerrada.

Los conceptos básicos anteriormente mencionados ayudan a la comprensión del proyecto cubriendo la parte del área de redes, a continuación, es conveniente mencionar los principales conceptos sobre la Programación Orientada a Objetos (POO) ya que el proyecto ha sido desarrollado en lenguaje java.

PERSPECTIVA ORIENTADA A OBJETOS.

La visión actual del desarrollo de software toma una perspectiva orientada a objetos.

En este enfoque, el principal bloque de construcción de todos los sistemas software es el objeto y la clase, en síntesis, la programación orientada a objetos puede ser resumida de la siguiente manera:

Objetos + Flujo de mensajes = Programas

La programación cuenta con 4 pilares fundamentales que son:

- Abstracción
- Encapsulamiento
- Herencia
- Polimorfismo

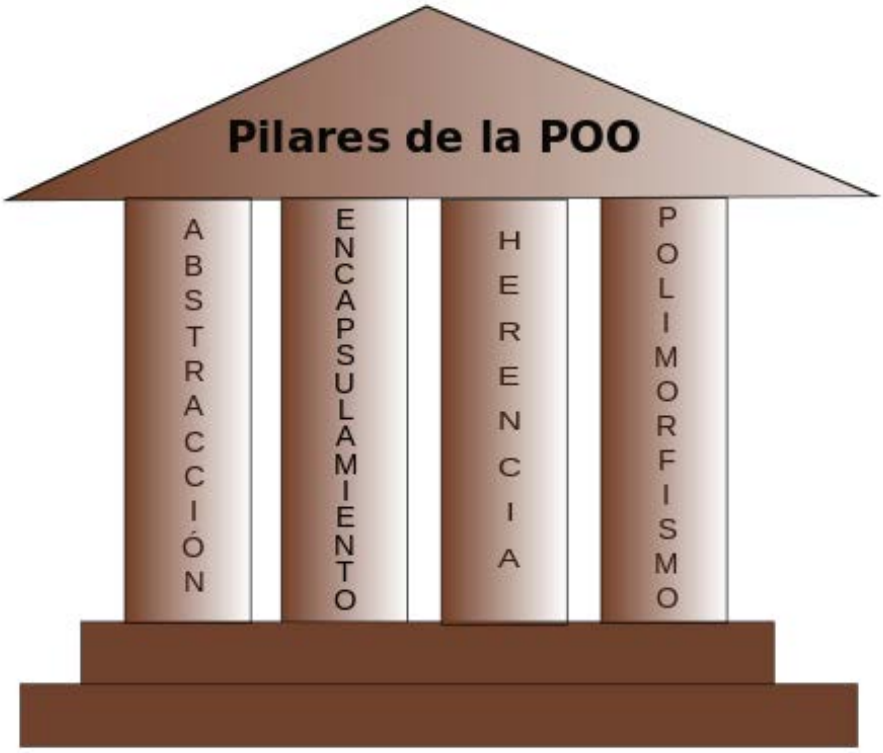


Imagen 2.2 Pilares de la POO

Pero antes de definir los pilares de la programación orientada a objetos, se debe definir completamente como trabaja la POO comenzando con el siguiente concepto:

ADOO (Análisis y Diseño Orientado a Objetos).

El Análisis y Diseño Orientado a Objetos (ADOO) es una técnica para modelar sistemas que se encargan de describir y modelar el sistema como si fuera un conjunto de objetos relacionados que interactúan entre sí.

Los métodos orientados a objetos centran su atención en objetos y clases de la siguiente manera:

*Un **objeto** es "una **cosa**".*

*Una **clase** es "una **plantilla** o definición de cosas.*

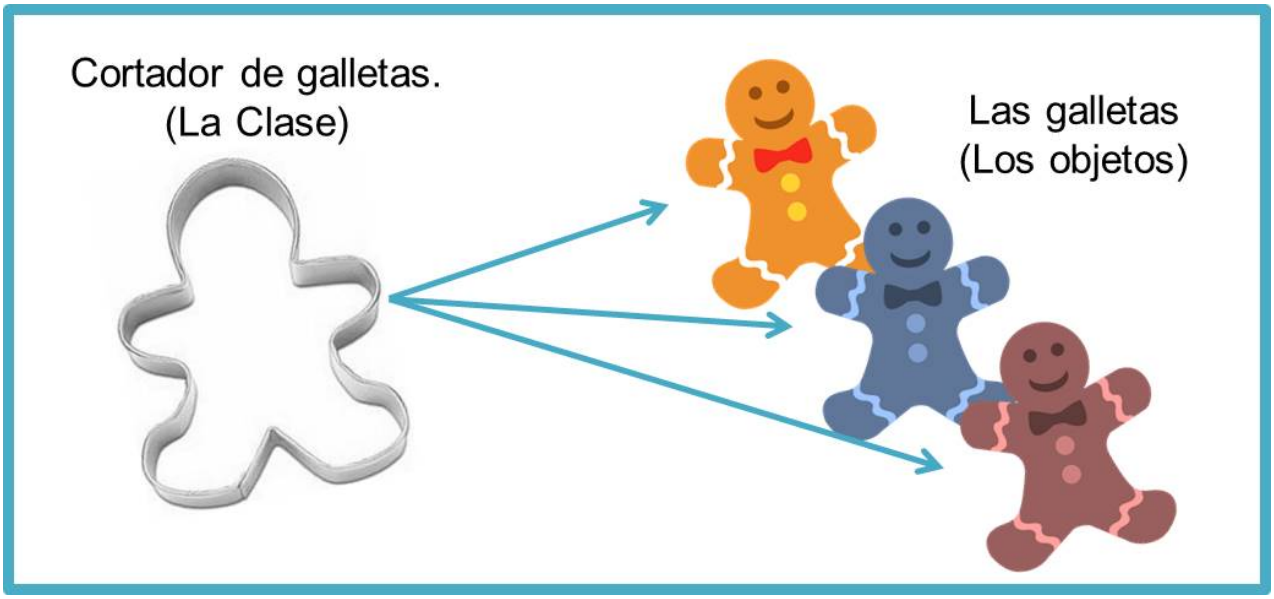


Imagen 2.3 Ejemplo de clase y objeto

Los métodos orientados a objetos crean modelos con los cuales:

- Se pueden crear clases y objetos.
- Definen su estructura, comportamiento y propósito.
- Definen la relación entre clases.
- Definen la relación entre objetos.

Ventajas de la POO.

- Los objetos modelan el comportamiento del mundo real y son fáciles de entender
- Los componentes pueden ser reutilizados a través de interfaces bien definidas.
- El cambio de requerimientos es fácil de soportar
- Los costos de mantenimiento son reducidos considerablemente.

Una vez definido lo anterior es importante mencionar las partes de la POO con conceptos como:

OBJETOS

Se define a un objeto como "una entidad tangible que muestra alguna conducta bien definida".

Un objeto “es cualquier cosa, real o abstracta, acerca de la cual almacenamos datos y hay métodos que hacen uso de dichos datos”.

Existen propiedades de los objetos como:

- Los atributos

Son las características que definen a un objeto como único.

- Método

Es la acción de un objeto, representado generalmente por verbos, el nombre de un método debe indicar idealmente la función de un objeto.

- Mensaje

Los mensajes son llamadas a los métodos del objeto con el que se quiere comunicar para que haga una o varias cosas.

CLASES

Una clase es una plantilla para crear objetos con características similares. Las clases comprenden todas las características de un conjunto particular de objetos.

Una instancia de una clase es otro termino para un objeto real, si la clase es la representación general de un objeto, una instancia es su representación concreta.

En los lenguajes orientados a objetos cada clase está compuesta de dos cualidades:

- Atributos (estado).
- Métodos (comportamiento).

Una manera de definir el comportamiento de un objeto, se crean métodos, los cuales tienen una apariencia y un comportamiento igual al de las funciones en otros lenguajes de programación.

Los métodos no siempre afectan a un solo objeto; los objetos también se comunican entre si mediante el uso de métodos.

Una parte importante que cabe mencionar es que una clase u objeto puede llamar métodos en otra clase u objeto para avisar sobre los cambios en el ambiente o para solicitarle a ese objeto que cambie su estado.

Luego de definir los principales conceptos de la POO es momento de comenzar a definir los pilares de la misma.

ABSTRACCIÓN

Este concepto se refiere básicamente a describir una entidad en forma simplificada, a través de sus propiedades principales, omitiendo los detalles.

La abstracción consiste en ignorar detalles y encontrar las principales características, de esta manera se simplifica el cómo los usuarios interactúan con los objetos.

Existen varios tipos de abstracción como lo son:

- Abstracción de datos

Son las características y las propiedades de un objeto.

- Abstracción funcional

Es lo que hace el objeto (acciones y/o eventos).

ENCAPSULACIÓN

Esta característica se refiere a ocultar la información de los detalles de la implementación.

El encapsulamiento es guardar o poner algo dentro de un objeto, de esta manera se separan los aspectos externos de implementación que se ocultan en los objetos.

La encapsulación está formada por los siguientes puntos:

- Interfaz pública: Son las operaciones que podemos hacer con un objeto para interactuar con él.

- Implementación: Son las operaciones internas que realiza un objeto para lograr su propósito.

- Información interna: Son los datos que ocupa el objeto de manera interna para complementar su función.

Los atributos y operaciones son miembros del objeto y pueden ser de tipo público o privado, si el miembro es público, forma parte de la interfaz pública, de lo contrario forma parte de la implementación.

En sistemas orientados a objetos todos los atributos son privados y pueden ser accedidos con operaciones públicas.

HERENCIA

Se puede crear una nueva clase u objeto heredando los atributos y métodos de una o varias clases padre (herencia simple y múltiple respectivamente).

Las nuevas clases que se van creando por herencia, crean lo que se denomina como jerarquía de clases, en este contexto se utiliza el termino de superclase para referir cualquiera de las clases de orden superior en una misma jerarquía.

Ejemplo:

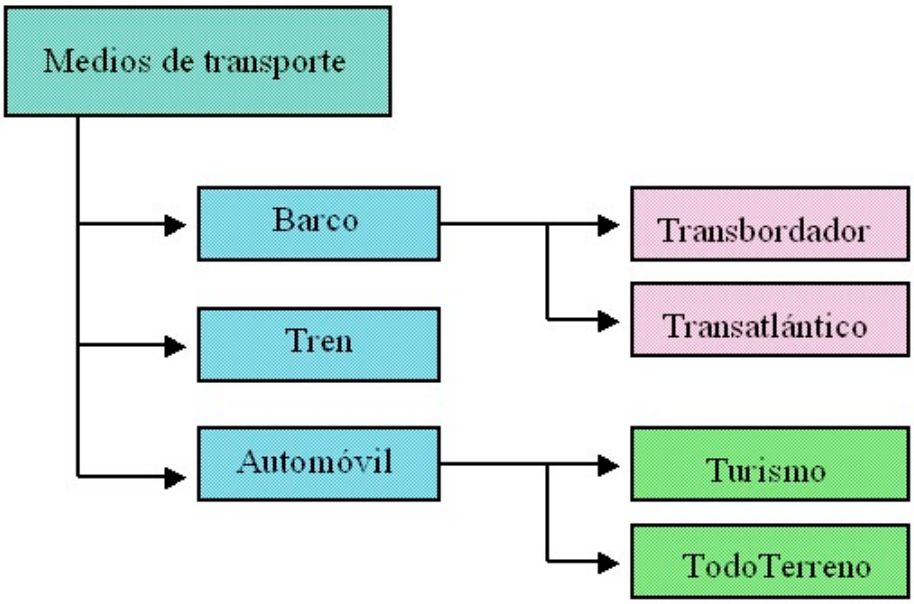


Imagen 2.4 Ejemplo de herencia

Ventajas de la herencia:

- Las subclases proveen conductas especializadas sobre la base de elementos comunes provistos por la superclase. A través del uso de herencia, los programadores pueden reutilizar el código de la superclase muchas veces.

- Los programadores pueden implementar superclases llamadas clases abstractas que definen conductas “genéricas”.

POLIMORFISMO

El polimorfismo permite implementar una operación de herencia en una subclase (operaciones heredadas de una superclase).

El polimorfismo significa que la operación existe en muchas clases, la operación tiene el mismo significado, pero casa clase personaliza la operación.

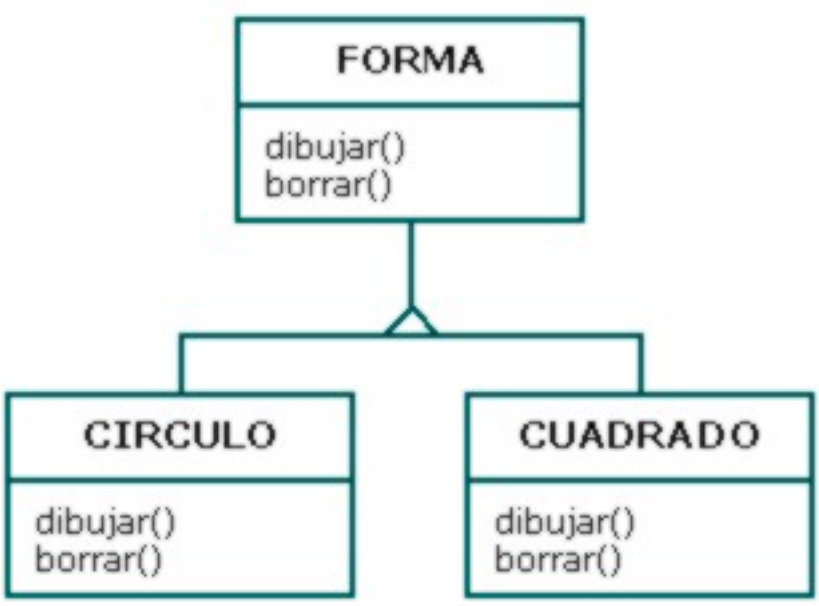


Imagen 2.5 Ejemplo del polimorfismo.

Un último concepto que es necesario definir es:

“¿Qué es un hilo?”,

Debido a que la aplicación maneja este concepto para poder realizar su mayor funcionamiento.

Un hilo es un flujo secuencial de control dentro de un programa. Un hilo o contexto de ejecución es una secuencia de instrucciones en ejecución, es decir, un hilo es un proceso ligero (más rápido y económico en cuanto a recursos informaticos), donde, generalmente un proceso padre genera diversos procesos-hijo, formando así un árbol de procesos.

También existen los programas multihilo pueden tener varios flujos de control en diferentes partes del código ejecutándose "simultáneamente".

Sabiendo esto la aplicación de SCI cuenta con una estructura multihilo.

Una vez definido todo lo necesario se muestra a continuación la problemática y metodología de los proyectos.

Definición del problema

Problemática para el proyecto principal.

Nombre de proyecto principal: Sistema de comunicación interna (SCI).

Actualmente el hospital general de México se encuentra dividido por unidades dependiendo de cada especialidad médica, por ejemplo, existe la unidad de oftalmología, otorrinolaringología, medicina interna, ginecología, etc. y a su vez se les asocia un número para identificarlas, por ejemplo, oncología es la unidad 111, torre quirúrgica es la unidad 310 y así respectivamente.

Cada unidad tiene sus respectivas áreas ya sean médicas y administrativas que cuentan con bienes informáticos para desempeñar sus actividades.

El problema comienza cuando algún sistema o ya sea el bien informático comienzan a tener problemas imposibles de solucionar por el usuario o ajenos al usuario, lo cual conlleva a solicitar asistencia técnica en sitio.

El método más usado y en este caso descrito idealmente en el hospital para solicitar la asistencia es hacer una llamada a la extensión telefónica del área de desarrollo de sistemas o al área de soporte técnico y redes dependiendo del tipo de problema presentado.

Al hacer la llamada, un personal del área tratará de dar asistencia telefónica para solucionar el problema y en caso de no poder brindar las indicaciones necesarias o que el problema requiera asistencia en sitio se tomarán los datos correspondientes del usuario como su nombre, área de la que llama, extensión y problema presentado finalizando la llamada.

El proceso anterior descrito se le conoce como “levantar un reporte en la mesa de servicios” que permite manejar una herramienta llamada mesa de servicios (por parte del personal del área de sistemas o soporte) para poder tener en una base de datos los reportes que han existido mediante un entorno web y gráfico, turnarlos al personal indicado para la realización del problema y poder cerrarlos cuando la actividad técnica llega a su fin.

Todo lo anterior descrito sobre la solicitud de asistencia por los usuarios fue en un caso ideal, ahora se describe el caso común por el cual se ha desarrollado la aplicación de la que trata este escrito.

Uno de los factores que más influyen a la hora de solicitar asistencia es que no todas las áreas dentro de las unidades cuentan con extensión propia ni cercana al bien informático, lo que conlleva al usuario a solicitar la asistencia por alguna de las siguientes opciones:

- Pedir a alguien realizar la llamada para solicitar la asistencia comentando el problema presentado y creando un “teléfono descompuesto” entre el usuario con el problema, el personal de sistemas y el tercero.

- Realizar la llamada lejos del bien informático sin poder ser sujeto a asistencia telefónica

- Desplazarse hasta el área de sistemas a solicitar asistencia, volviendo poco eficaz y eficiente la solución de problema

Todos los problemas anteriormente mencionados me llevaron a la elaboración del sistema de comunicación interna.

Actividad complementaria de impacto al hospital y relación con SCI.

Nombre del proyecto: Desarrollo de un helpdesk o mesa de servicios

Una de las problemáticas presentadas a la hora de manejar la herramienta de “mesa de servicios” (desarrollada en el hospital general por parte del área de sistemas) es que debido al tiempo que lleva de uso y la cantidad de reportes existentes se han presentado varias fallas o “bugs” que dificultan el uso de la mesa de servicios como:

- Cierre de sesión automático después de cierto tiempo de no usar la mesa marcando errores.
- Duplicación de reportes de fechas anteriores con los tomados en fechas recientes.
- No hay compatibilidad con navegadores, solo se puede operar en navegadores Mozilla Firefox.
- No hay portabilidad en dispositivos móviles y tablets.
- Problemas a la hora de imprimir los reportes levantados.
- Problemas a la hora de ingresar los datos del usuario que reporta algún problema.

Lo cual me llevó a realizar junto con un grupo de trabajo una mesa de servicios nueva y totalmente optimizada para el hospital, cumpliendo las funciones necesarias y arreglando los problemas anteriormente presentados para un mejor desempeño para el usuario, un mejor control de los reportes y mejor atención técnica.

La realización de la mesa de servicios complementa el proyecto de comunicación interna para una mejor toma de reportes y tener al usuario que solicita la ayuda técnica el menor tiempo posible en el chat o teléfono y también para que el personal que está tomando el reporte le sea más fácil la toma de datos, no lidie con los problemas actuales de la mesa y el trabajo sea más eficiente.

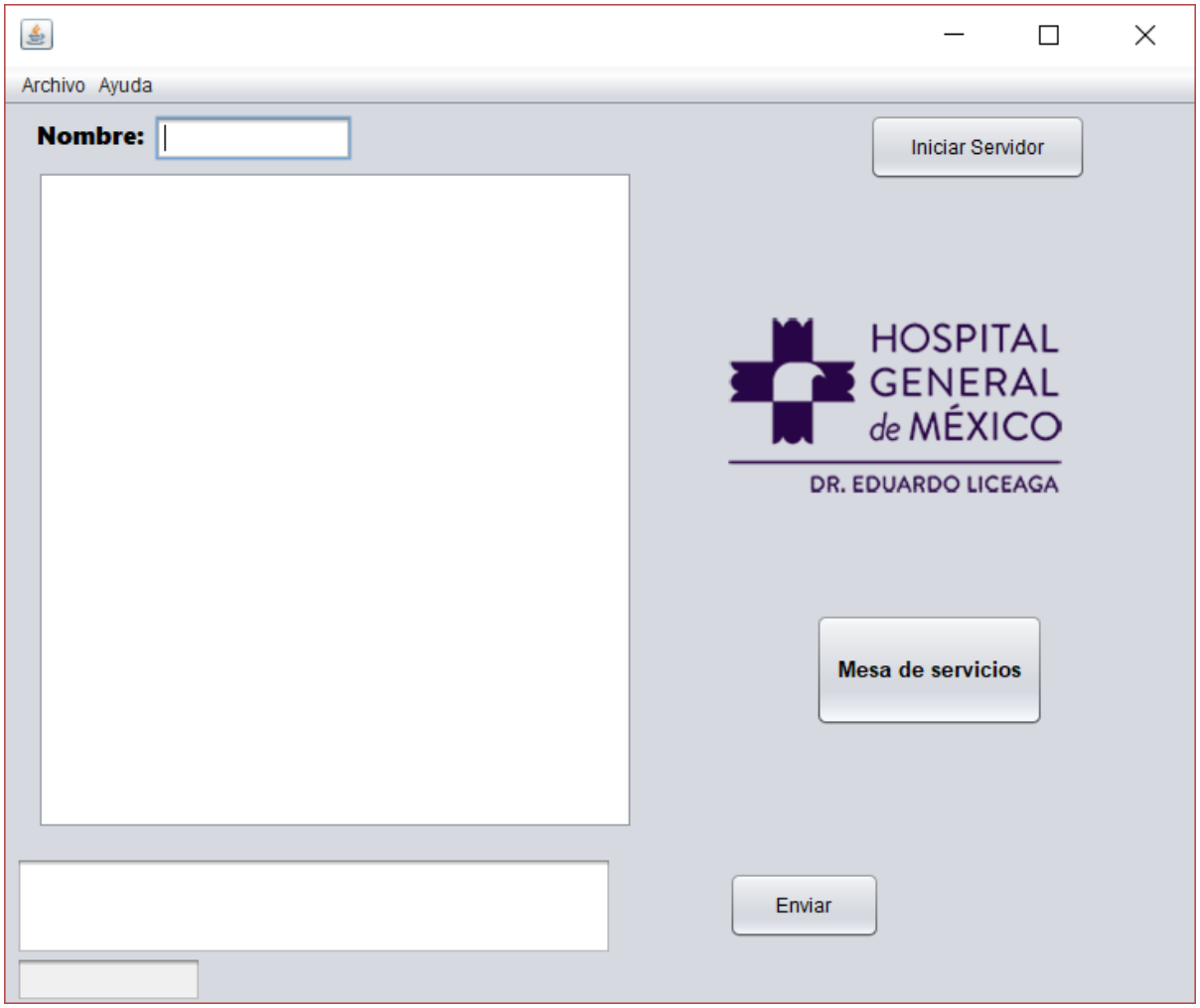
Desarrollo del proyecto.

Sistema de comunicación interna (SCI)

El principio por el cual funciona esta aplicación es una arquitectura cliente servidor donde básicamente se cumple el proceso de un cliente y un servidor previamente descritos.

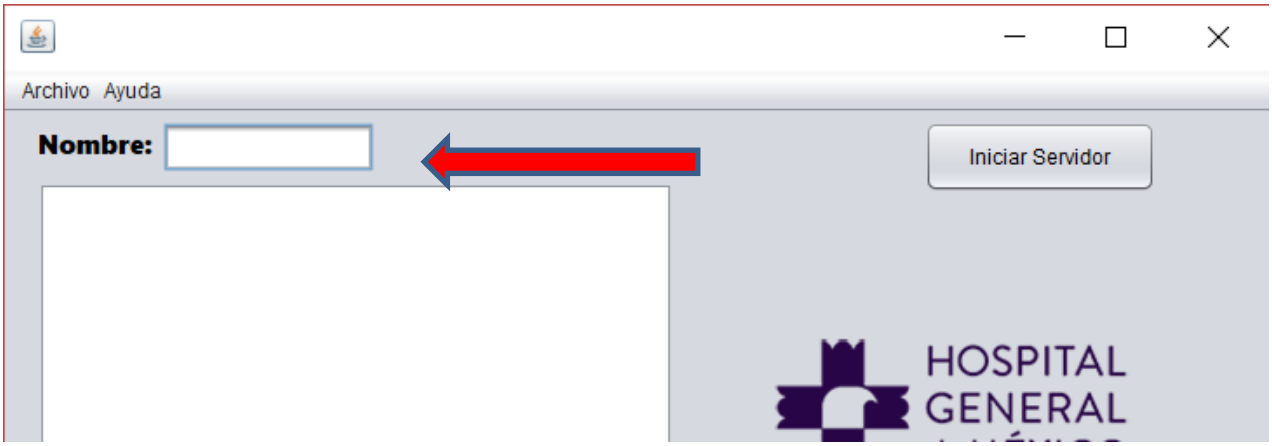
A continuación, se describirá el proceso de la aplicación primeramente por parte del **servidor**:

Después de la pantalla de carga, se muestra la ventana principal, que a simple vista puede notarse que es muy intuitiva.

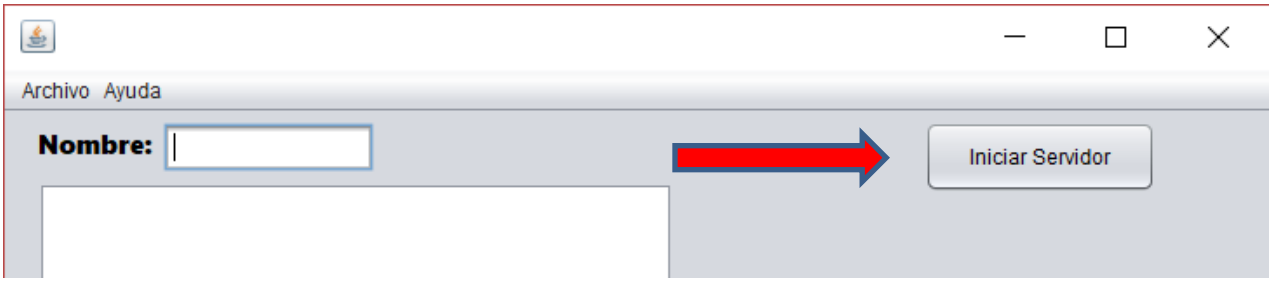


Esta ventana, tomando en cuenta que es la aplicación que utilizara el personal del área de sistemas o de soporte técnico y redes cuenta con los siguientes campos:

- Campo para ingresar el nombre del personal del área de sistemas o soporte técnico y redes. (se mostrará en el campo de chat tanto para cliente como para el servidor).

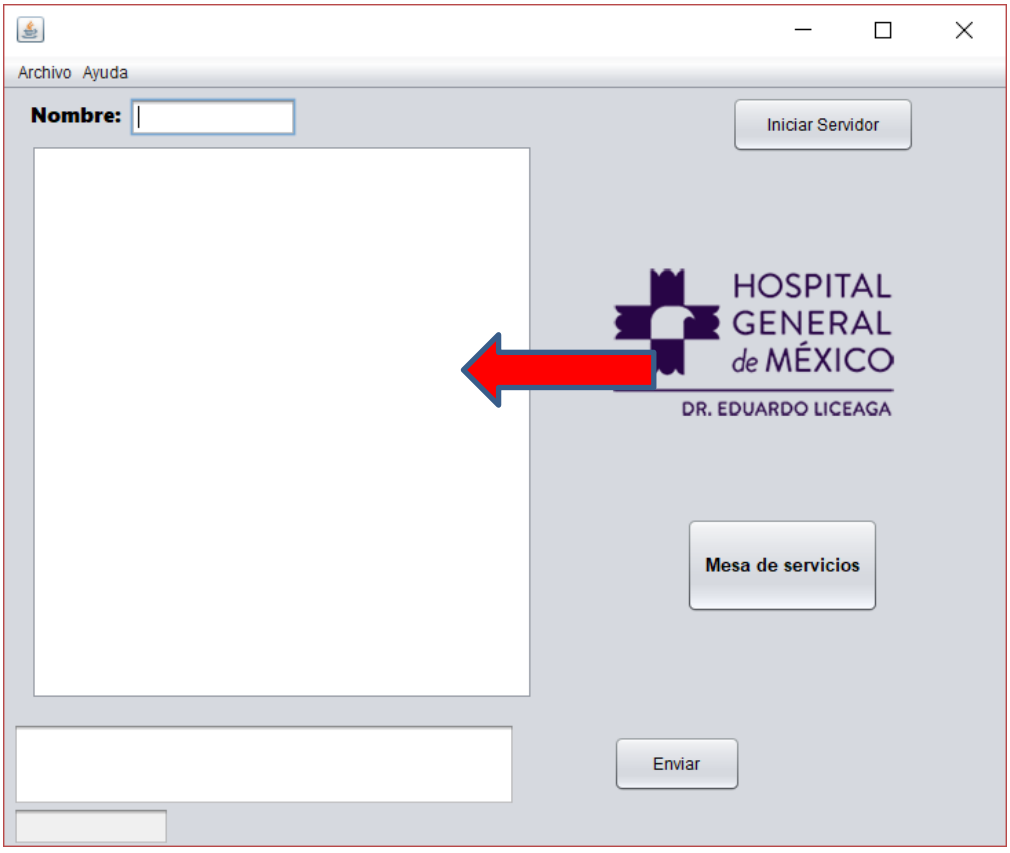


- Botón para iniciar la escucha o la espera de conexión de los clientes.

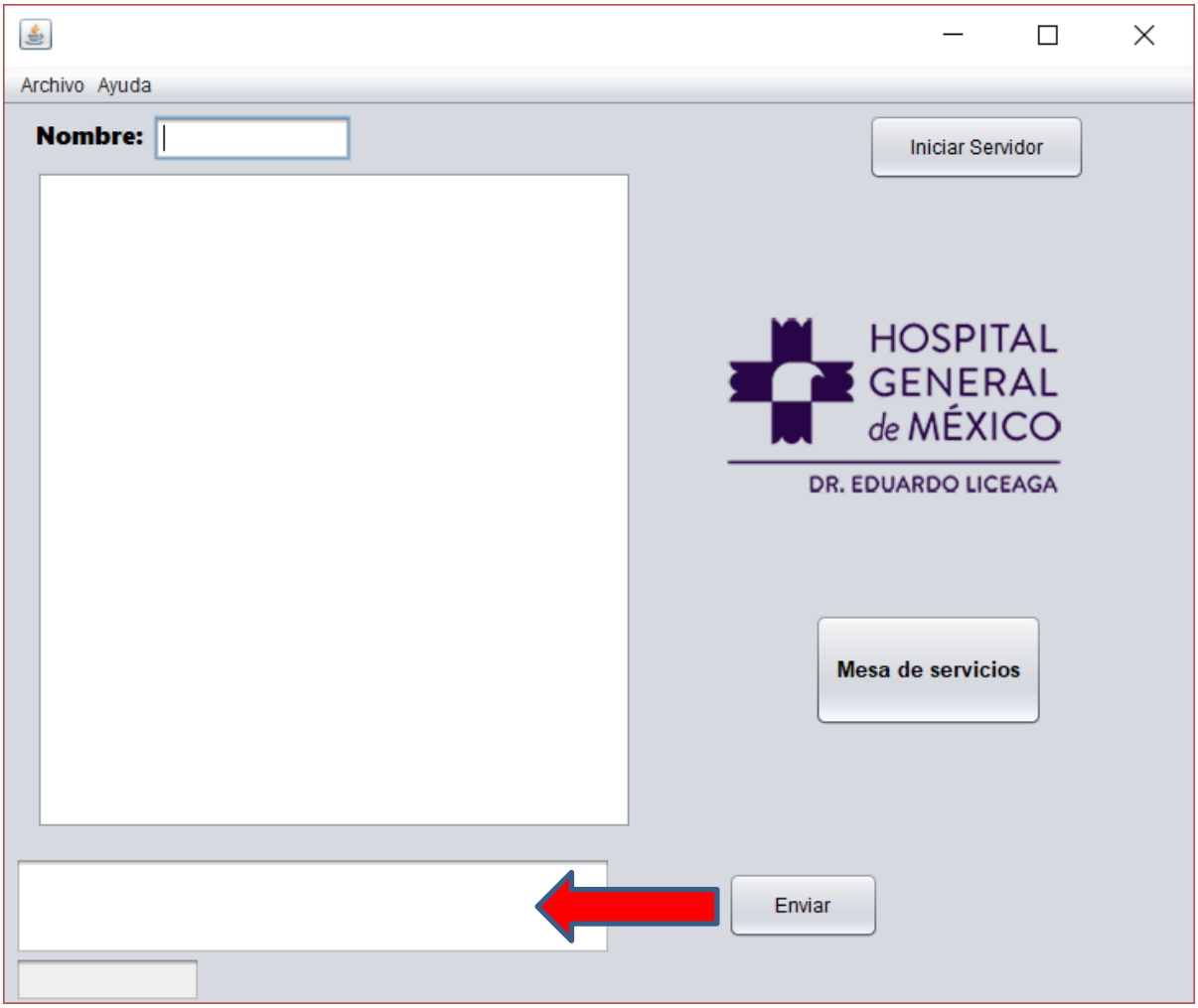


- Botón para facilitar el acceso a la mesa de servicios y botón de enviar mensaje.

- Campo de texto del chat



- Campo para escribir el mensaje



Al momento de presionar el botón de mesa de servicios, brinda un acceso a la mesa de servicios actual. (Se busca implementar la nueva mesa de servicios en este botón)

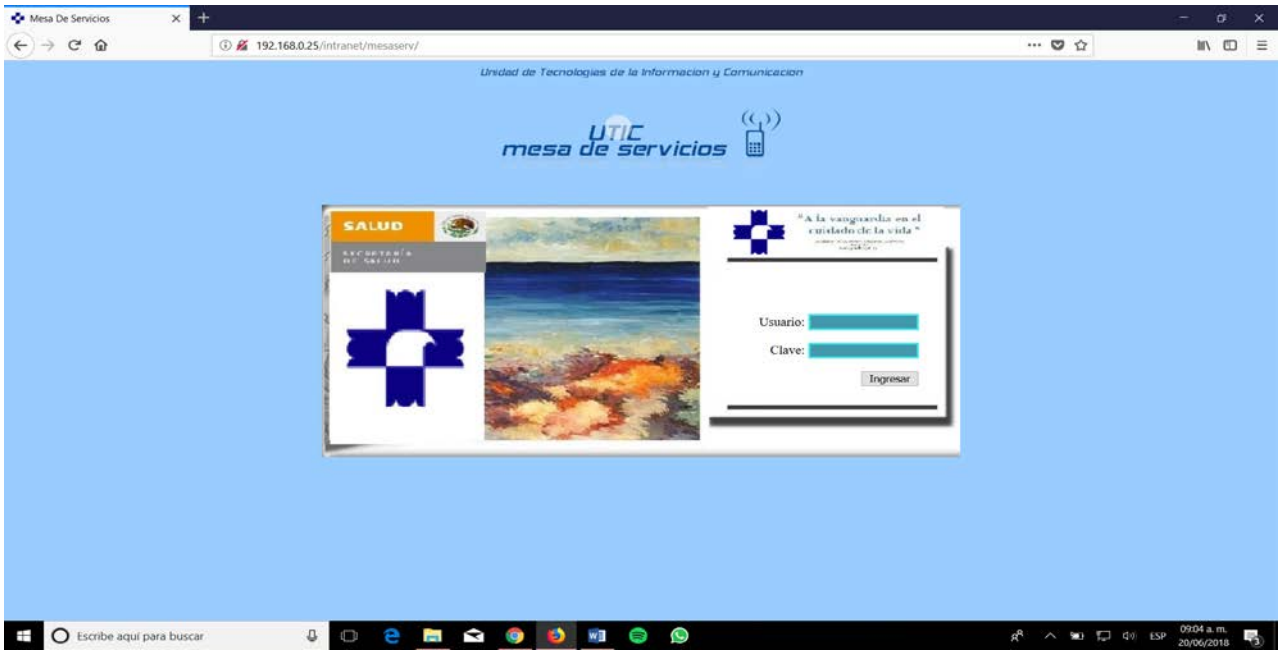
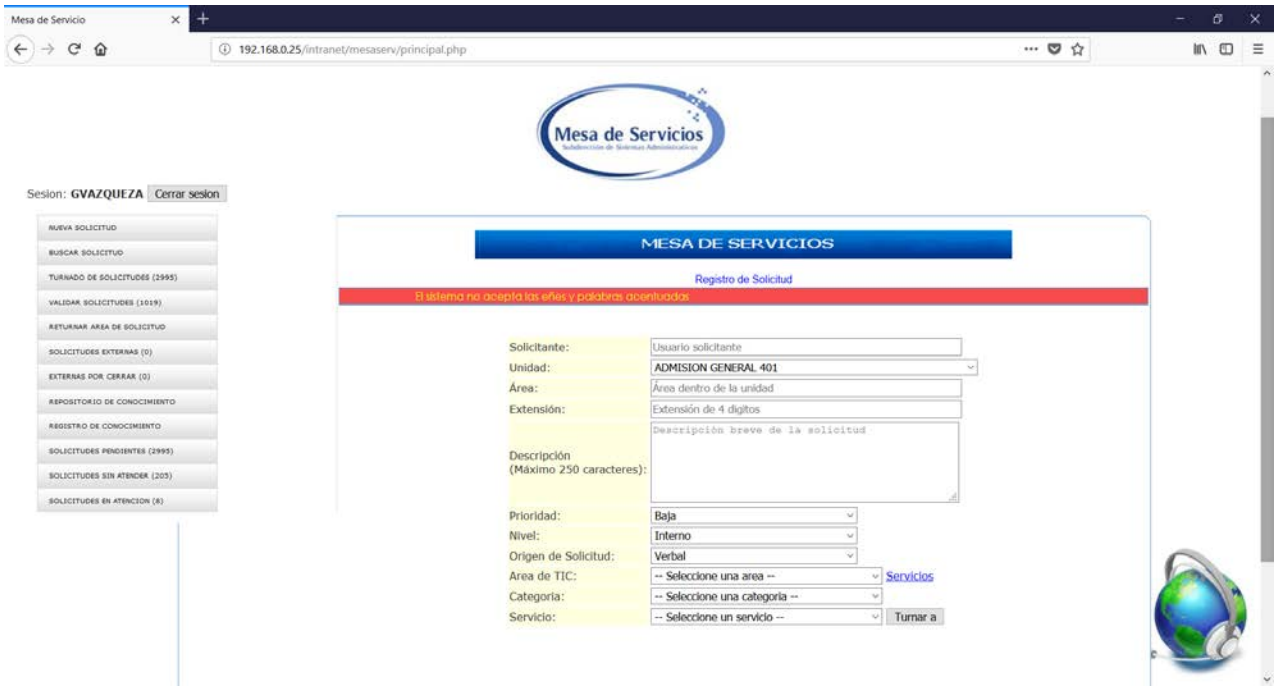


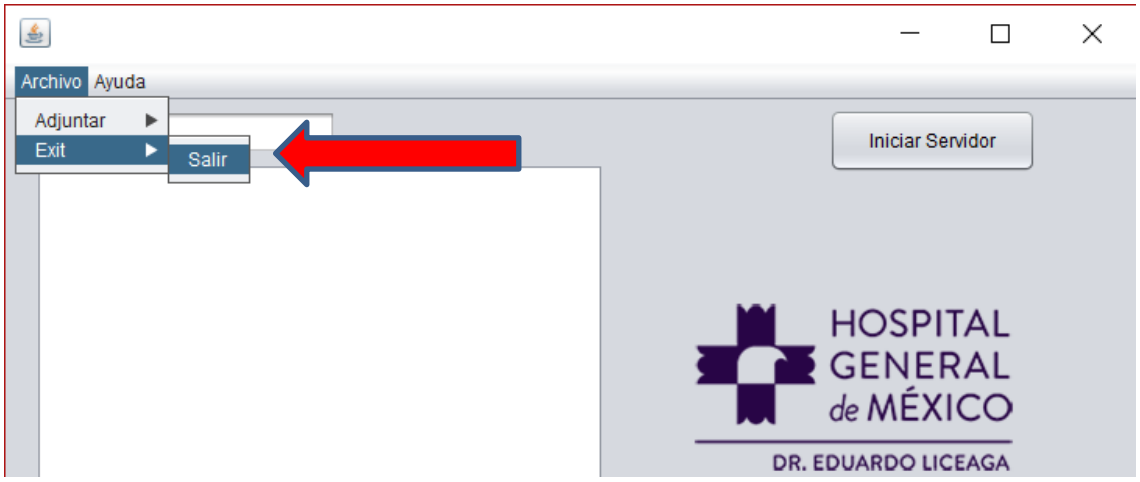
Imagen de ingreso a la mesa de servicios.



La imagen del ingreso a la mesa de servicios sirve para mencionar los datos que se le solicitan al usuario al momento que solicita alguna ayuda por el chat, que son:

- Solicitante (Responsable que solicita el servicio).
- Unidad (Área general de donde se solicita la ayuda).
- Área (Lugar específico dentro del área general).
- Extensión (Extensión telefónica de donde llama).
- Descripción (Breve descripción del problema reportado).
- Prioridad (Que tanta urgencia se le da al reporte).
- Nivel (Si es un problema interno o externo al hospital como empresas externas).
- Origen de la solicitud (Aquí se especifica si es escrito, verbal y se busca agregar por chat).
- Área de TIC (El área general a la que se turna el servicio).
- Categoría (Aquí se especifica si es problema de servidores, computadoras, consumibles, etc.).
- Servicio (Área específica a la que se turna el servicio).
- Botón turnar (Muestra el nombre del personal basado en el área seleccionada anteriormente para turnar el reporte).

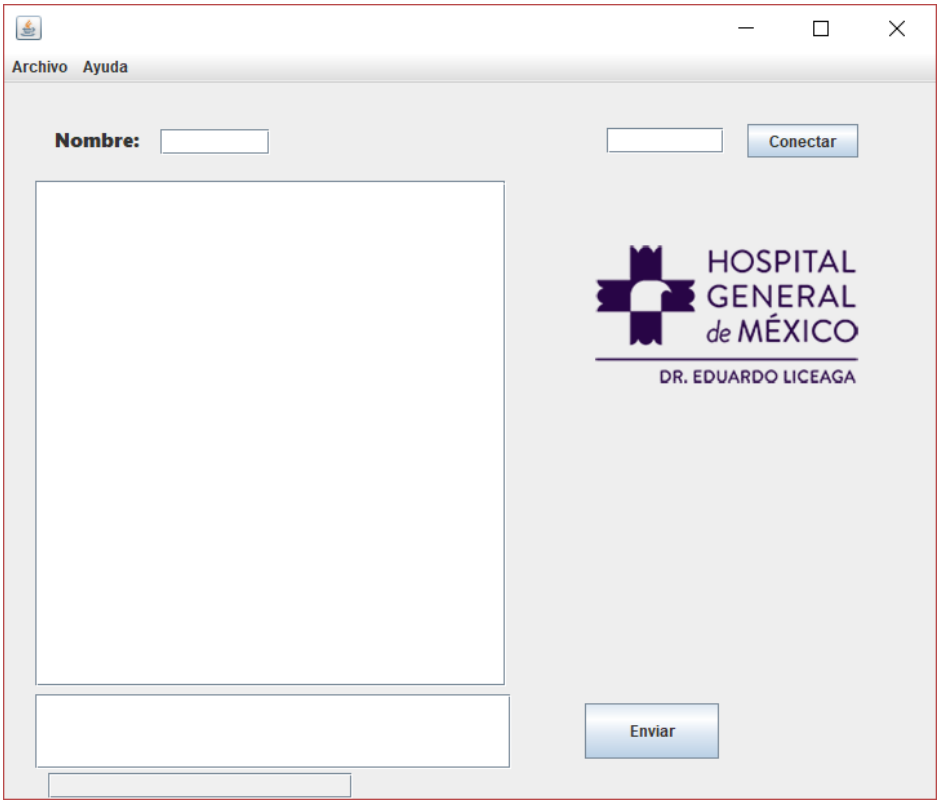
- Menú desplegable para salir de la aplicación.



A continuación, se describe la ventana de la aplicación cliente (la que usará el usuario solicitante).

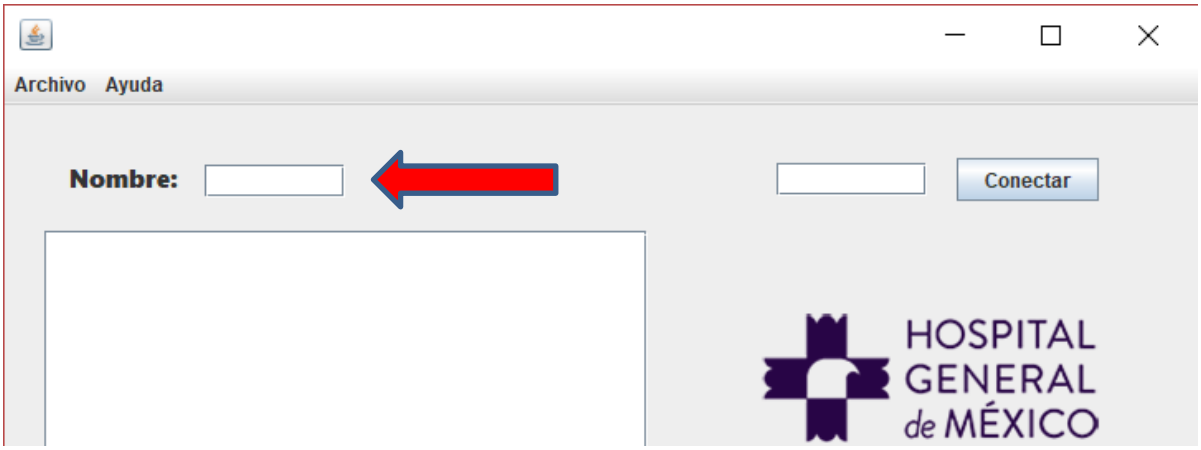
(Pantalla de carga de la aplicación cliente).

La siguiente imagen es la ventana principal de la aplicación del cliente, que al igual que la del servidor se muestra muy intuitiva y de fácil entendimiento para el usuario, recordando que, a la hora de realizar una aplicación, se debe considerar que el usuario no tiene ningún conocimiento sobre la aplicación que utilizará.

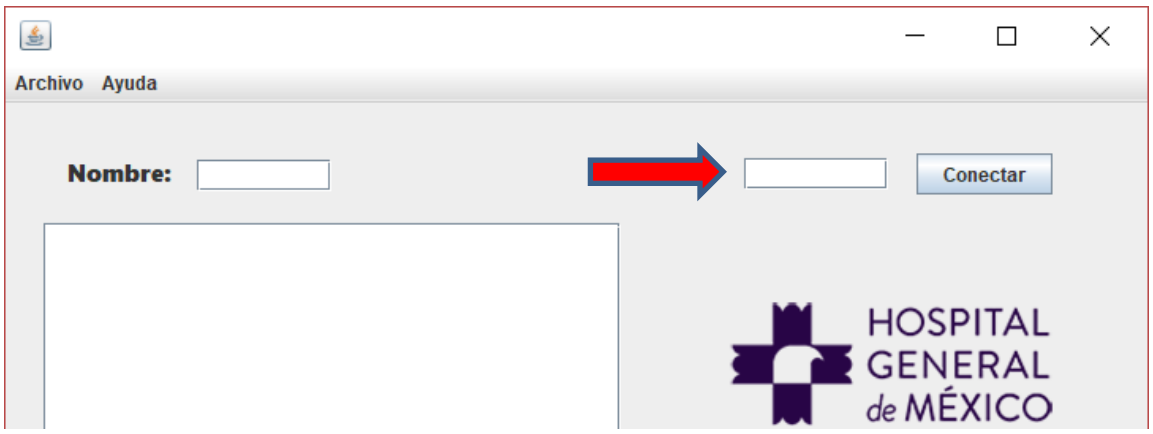


Al igual que la ventana de la aplicación servidor, la aplicación cliente cuenta con campos que se mencionan y describen a continuación:

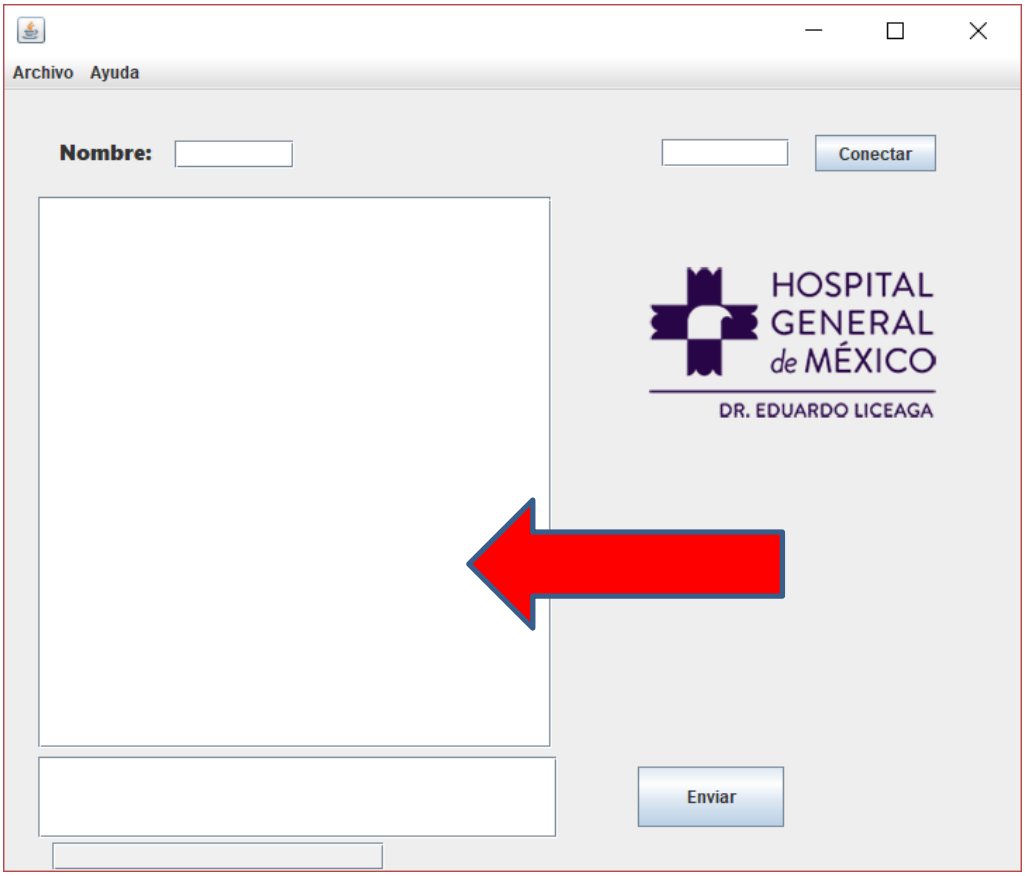
- Campo donde el usuario solicitante o cliente ingresa su nombre. (Se mostrará en el campo de chat de la ventana Servidor y Cliente)



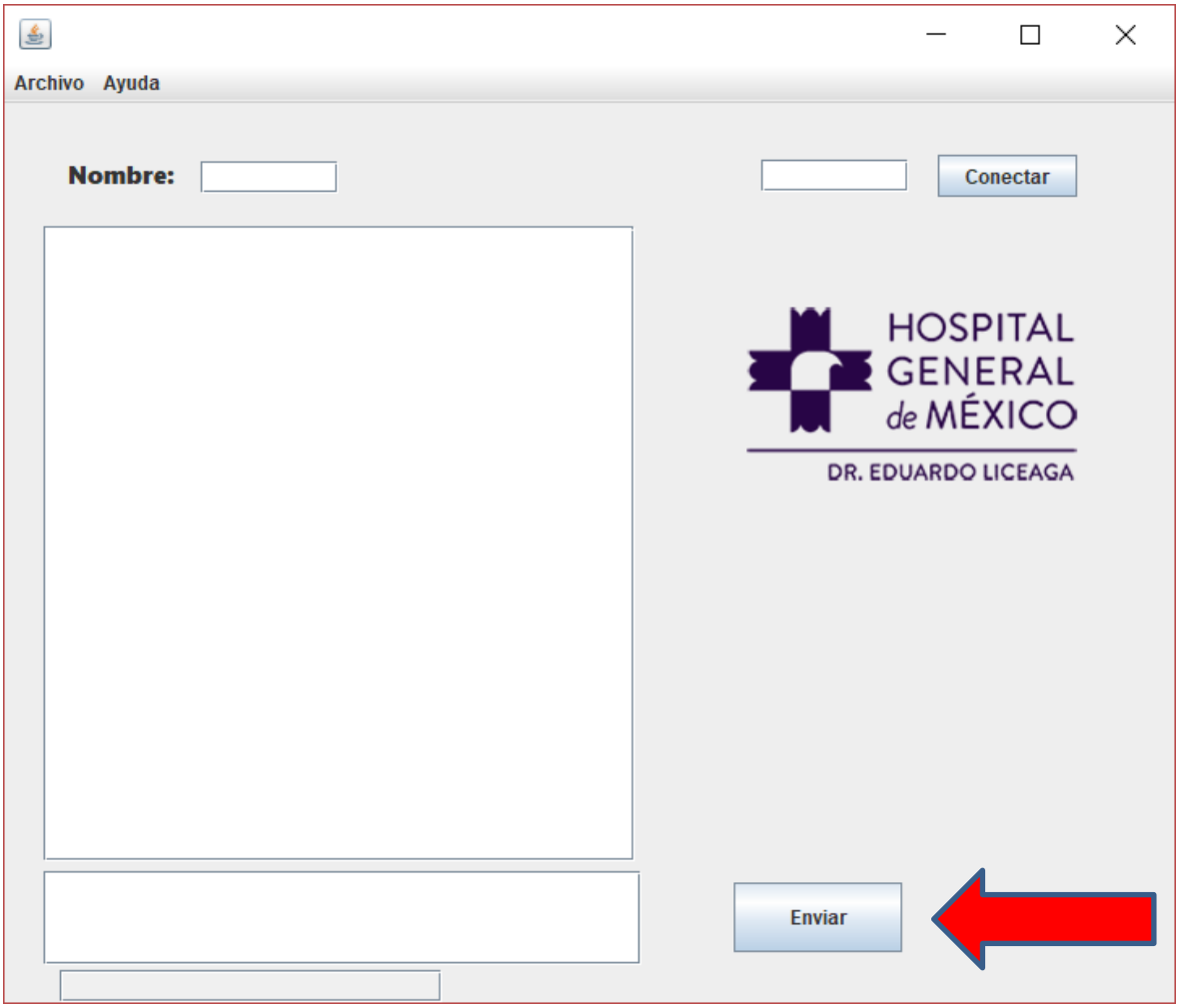
- Campo donde se ingresa la dirección del servidor a donde quiere conectarse o con que personal del área de sistemas quiere contactar y su respectivo botón para conectar.



- Campo de texto del chat.



- Campo de texto donde el usuario escribe con su respectivo botón para enviar el mensaje.



Funcionamiento del SCI.

El sistema de comunicación interna cuenta con dos aplicaciones como se ha mencionado anteriormente llamadas Ms–Cliente y Ms–Servidor haciendo referencia a la mesa de servicios, el nombre cliente y servidor es para identifica que aplicación usará el personal de sistemas y cual los usuarios solicitantes.

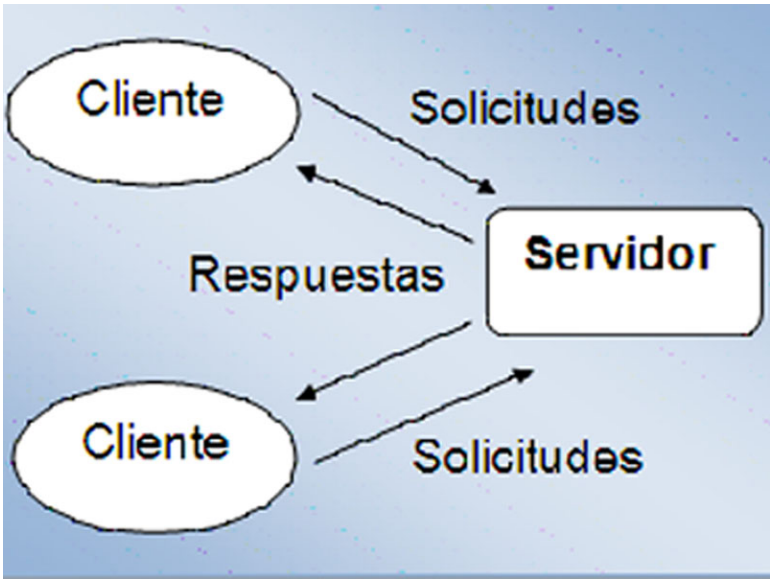
Cada uno cumple con su respectivo proceso, es decir, la aplicación Ms-Servidor cumple con el proceso de una arquitectura servidor y Ms-Cliente cumple con el proceso de una arquitectura Cliente. (A continuación, mencionadas como recordatorio)

PROCESO CLIENTE

- Abre el canal de comunicaciones para conectarse a la dirección de red atendida por el servidor.
- Enviar al servidor un mensaje de petición de servicio y esperar hasta recibir respuesta.
- Cerrar el canal de comunicación y terminar la ejecución del proceso.

PROCESO SERVIDOR

- Abre el canal de comunicación e informa a la red de la dirección a la que responderá como de la disposición para aceptar peticiones de servicio.
- Espera a que el Cliente realice una petición de servicio en la dirección que él tiene declarada.
- Cuando recibe una petición de servicio, atiende al Cliente.
- La conexión es cerrada.



El desarrollo del proyecto fue mediante el lenguaje de programación en java así que cabe mencionar algunas partes de código que son estándar a la hora de programar un cliente y un servidor y la utilización de socket TCP orientado a conexion en este lenguaje.

En la clase socket (aplicación del cliente) se muestra el estándar para la construcción.

Constructores:

```
public Socket ()
public Socket (InetAddress address, int port)
public Socket (String host, int port)
public Socket (InetAddress address, int port, InetAddress localAddr, int
    localPort)
public Socket (String host, int portt, InetAddress localAddr, int localPort)
```

address / localAddr	**Dirección IP de la máquina remota /local.**
port / localPort	**Puerto de la máquina remota / local.**
host	**Nombre de la máquina remota**

En la clase ServerSocket (aplicación del Servidor) se muestra el estándar para la construcción.

Constructores:

```
public ServerSocket (int port)
public ServerSocket (int port, int backlog)
public ServerSocket (int port, int backlog, InetAddress bindAddr)
```

port	puerto de escucha de la máquina servidora.
backlog	tamaño de la cola de espera, en el primero es 50.
bindAddr	dirección IP local que se hará pública mediante el bind.

Una vez mostrado el proceso que ejecuta la aplicación cliente y servidor mediante código y como se estructura, es conveniente mostrar la manera en la que interacciona el ingeniero (encargado de atender la falla) y el usuario (quien reporta la falla) mediante un diagrama para su fácil comprensión.

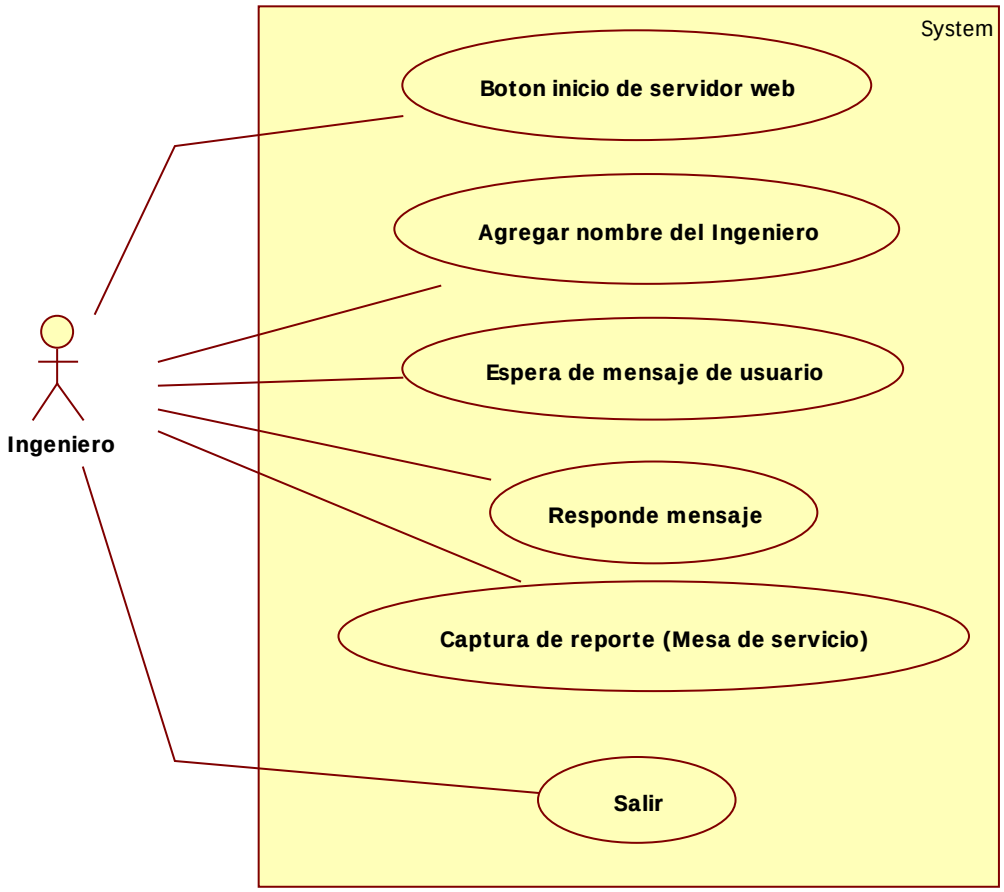


Imagen 2.6 Diagrama de casos de uso de aplicación servidor.

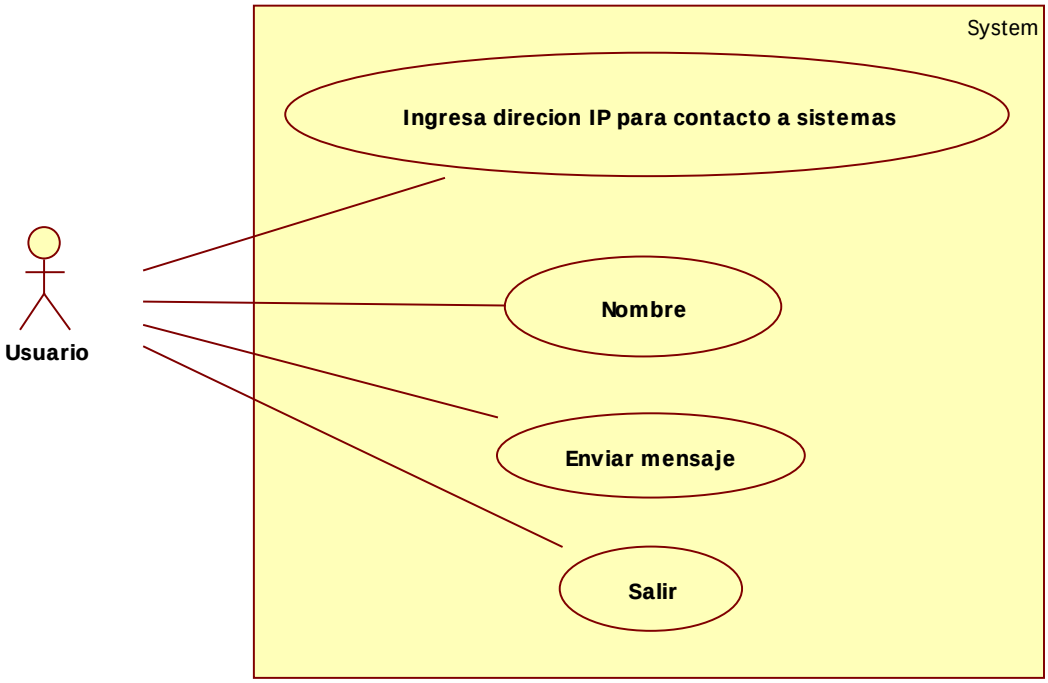


Imagen 2.7 Diagrama de casos de uso de aplicación cliente.

Para finalizar la manera en la que funciona la aplicación cabe mencionar que en todo momento, el ingeniero tiene en escucha la aplicación servidor esperando a que un usuario se conecte y esto es gracias a la tecnología implementada con arquitecturas cliente/servidor que permite estar esperando una conexión como se ha mostrado en los diagramas anteriores y funcionamiento del aplicativo.

Resultados

Como resultado general, se verifica tanto por mi jefe inmediato, usuarios y yo, que esta aplicación es de gran ayuda para que el personal del hospital cumpla de manera más eficiente sus actividades, por lo cual la atención al paciente es más rápida y la comunicación del personal médico y administrativo es más directo con el personal de sistemas, soporte, redes y mantenimiento.

Uno de los resultados más inesperados fue que debido a que el hospital cuenta con empresas externas que realizan actividades que el hospital no puede desarrollar como por ejemplo: auditorias, mantenimientos preventivos tanto a bienes informáticos o médicos, inventarios, etc. Una de esas empresas que desarrolla el mantenimiento de bienes informáticos también está utilizando la herramienta para comunicación con el jefe del departamento de soporte técnico y redes para cuestiones laborales e incluso también los usuarios pueden ponerse en contacto con ellos cuando alguno de sus bienes informáticos se les dictamina daño permanente o falla en alguna de las piezas de hardware, por lo tanto la aplicación comienza a cubrir más áreas en el hospital para mantener comunicación entre ellas.

Cabe mencionar que gracias a los conocimientos sobre programación orientada a objetos brindada en la facultad de ingeniería pude desarrollar una aplicación en lenguaje java que ha resultado altamente funcional, eficiente y adaptable a futuras actualizaciones que se adapten a nuevos requerimientos del hospital.

Una parte importante es la aportación que he brindado al hospital, debido a que el haber realizado la aplicación, el equipo de desarrollo de sistemas ha decidido apoyar la aplicación y estar a la espera de nuevos requerimientos para trabajar y poder aplicarlos de una manera más rápida.

Por último, me gustaría comentar que, gracias a la unión de la mesa de servicios actual del hospital, se ha detectado que puede optimizarse para cumplir mejor sus actividades, ahorrar tiempo y gracias a mis conocimientos como ingeniero en computación puedo trabajar en una actualización de mejora.

Conclusiones

A lo largo de estar trabajando en este proyecto me he dado cuenta que estoy cumpliendo la misión de un ingeniero, la cual es aplicar los conocimientos que he adquirido a lo largo de la carrera para resolver problemas de la sociedad y sobre todo comprender que el campo de un ingeniero en computación no siempre será trabajar con otros ingenieros en computación, sino que también trabajamos en conjunto con otras especialidades.

Cumplir el objetivo planteado en este proyecto fue algo que se cumplió incluso con mayor alcance, debido a que el objetivo general era crear la aplicación y que lograra mantener en comunicación a las áreas administrativas y medicas con sistemas y soporte técnico, pero también se ha logrado poner en comunicación incluso áreas externas y puede llegar a crecer hasta donde se necesite creando actualizaciones que permitan la unión de todos.

Otra conclusión que es importante recalcar es que como ingenieros en computación debemos comenzar a hacernos la idea de que en el ámbito laboral podemos encontrarnos con cualquier persona de cualquier carrera diferente a la nuestra y tenemos que aprender cómo lidiar con esas áreas en las que no estamos tan enfocados y seguir aplicando las matemáticas, la tecnología y el ingenio.

Hay que comprender que otro punto importante es el avance tecnológico, cada día hay avances en este ramo y debemos estar al día de que lenguajes se utilizan, que hardware podemos utilizar, que sistemas operativos son compatibles con nuestros desarrollos, etc y poder ejercer de la mejor manera la ingeniería.

Bibliografía

- Historia de la computación y redes.
  https://es.slideshare.net/Micheel_Flores/la-historia-de-las-redes-de-computadoras-48332711
  https://www.iebschool.com/blog/historia-de-internet-innovacion/
  Consultado el 11 de julio del 2018

- Topologías
  http://jorge-star.galeon.com/ANILLO.html
  https://redesinalambricasycableadas.wordpress.com/redes-cableadas/diferentes-topologias-de-red/topologia-de-estrella/
  http://jorge-star.galeon.com/MALLA.html
  https://unicrom.com/topologias-de-redes-estrella-malla/
  Consultado el 14 de julio del 2018

- socket
  http://www.masadelante.com/faqs/socket
  Consultado el 3 de julio del 2018

- Modelo OSI
  https://www.telepieza.com/wordpress/2008/04/28/los-7-niveles-o-capas-del-modelo-osi/
  Consultado el 5 de julio del 2018

- Apuntes de clase de arquitectura cliente/servidor
  México 2017

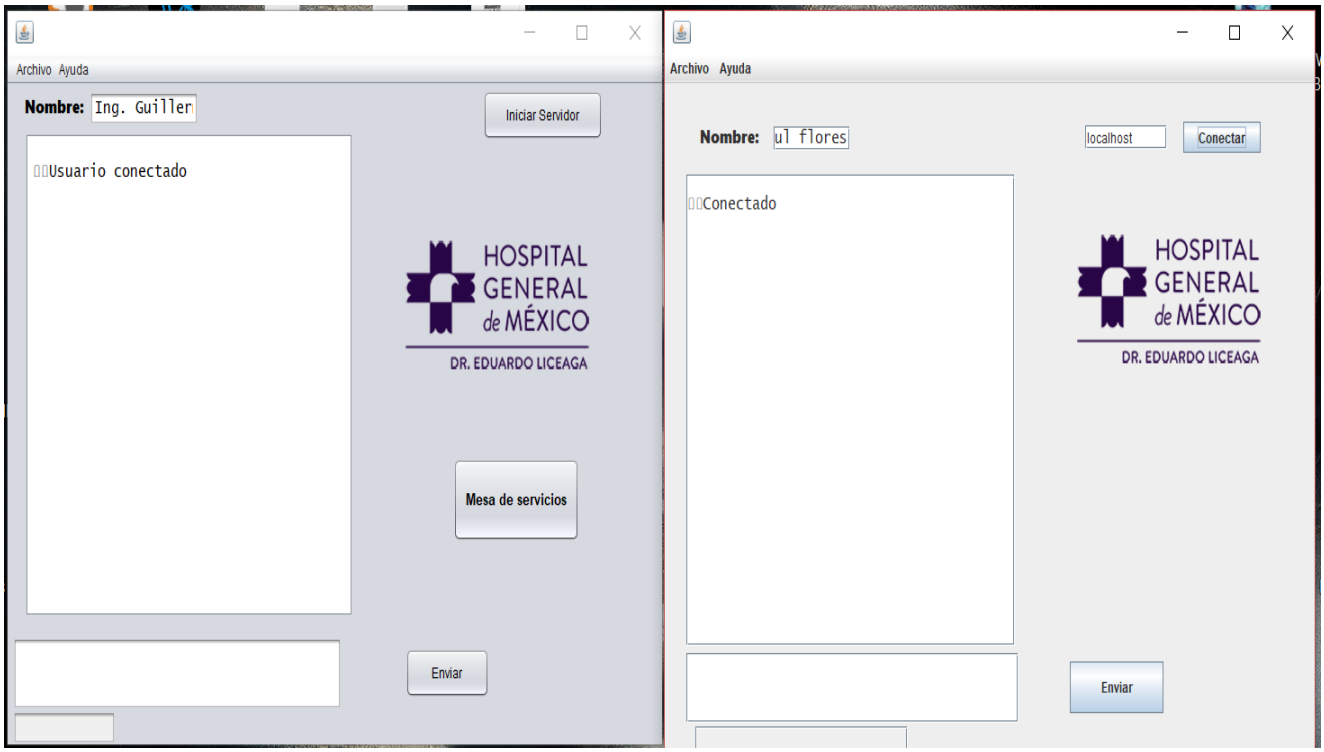
- Ceballos Sierra, Francisco Javier. (2014). Java 2: Curso de programación. (2da edición). México. Alfaomega Ra-Ma.

- Estándares

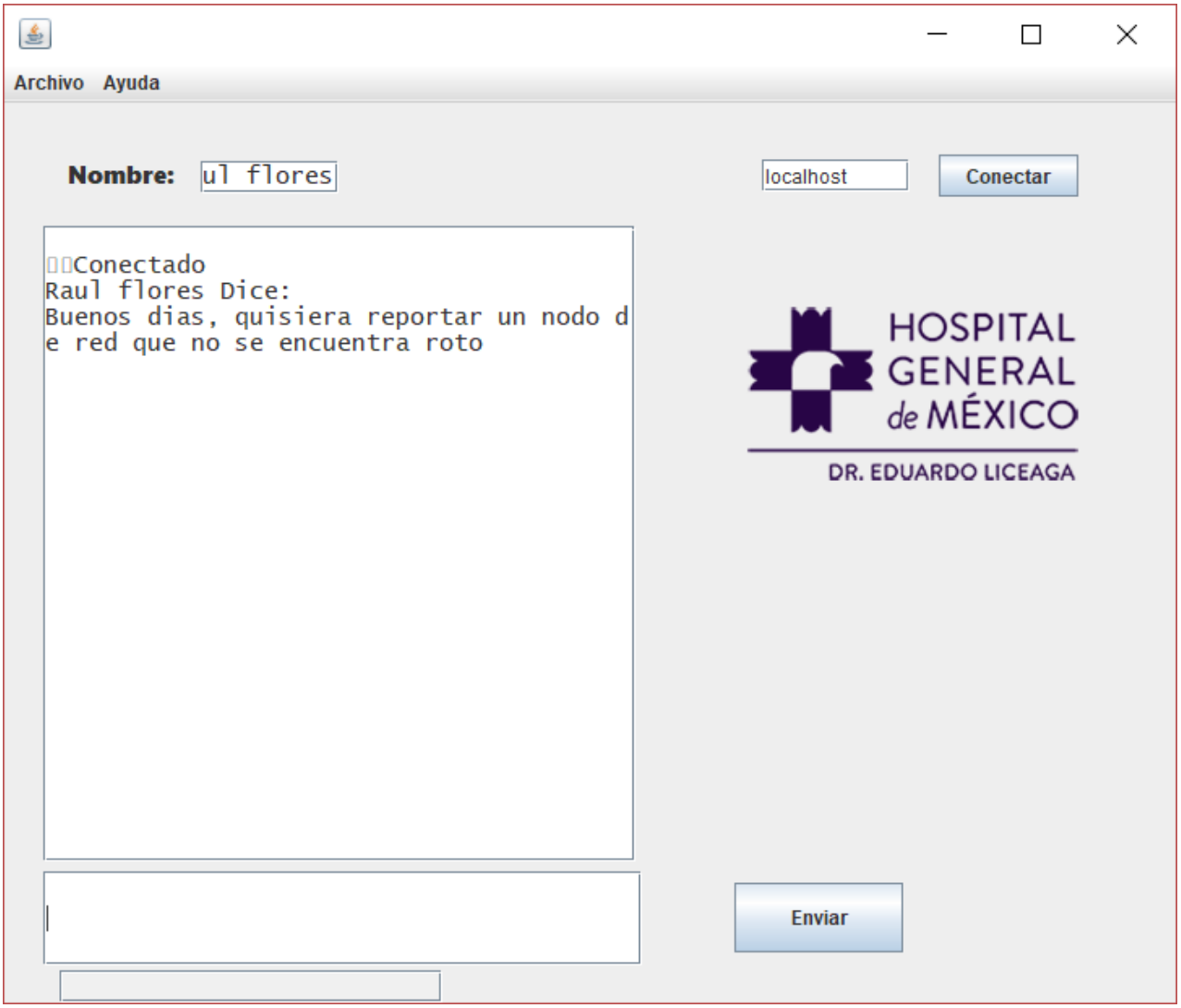
http://www.bajacalifornia.gob.mx/registrocivilbc/iso_informa2.htm
http://ewh.ieee.org/sb/el_salvador/uca/historia.html
https://www.itu.int/es/about/Pages/history.aspx
https://www.ecured.cu/ANSI
https://prezi.com/xmqnm0ja50kd/historia-eia-tia-caracteristicas-de-la-norma/
https://www.ecured.cu/IETF
Consultados el 6 de lulio del 2018

Anexos

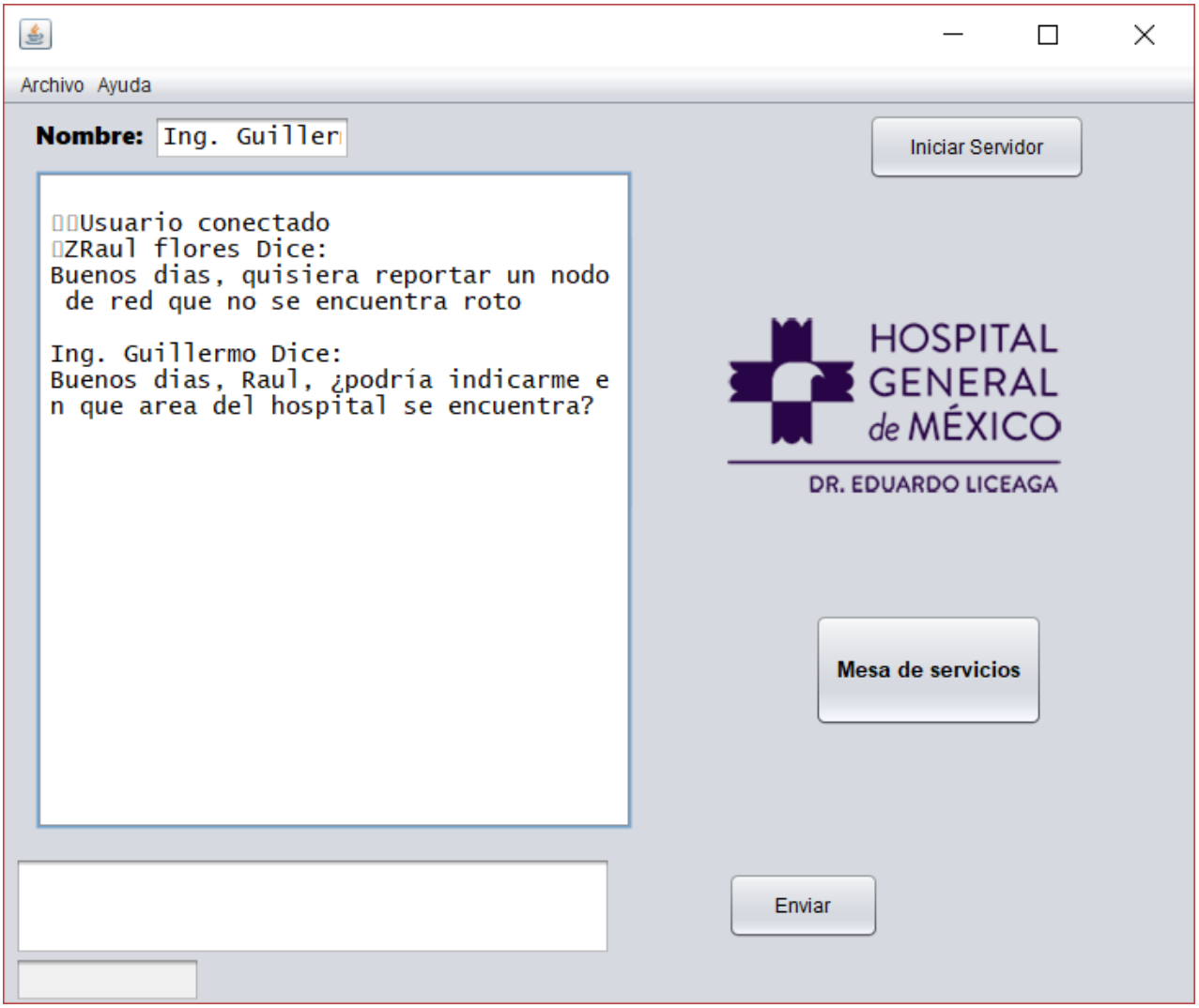
A. Pantallas del sistema simulando ejecución



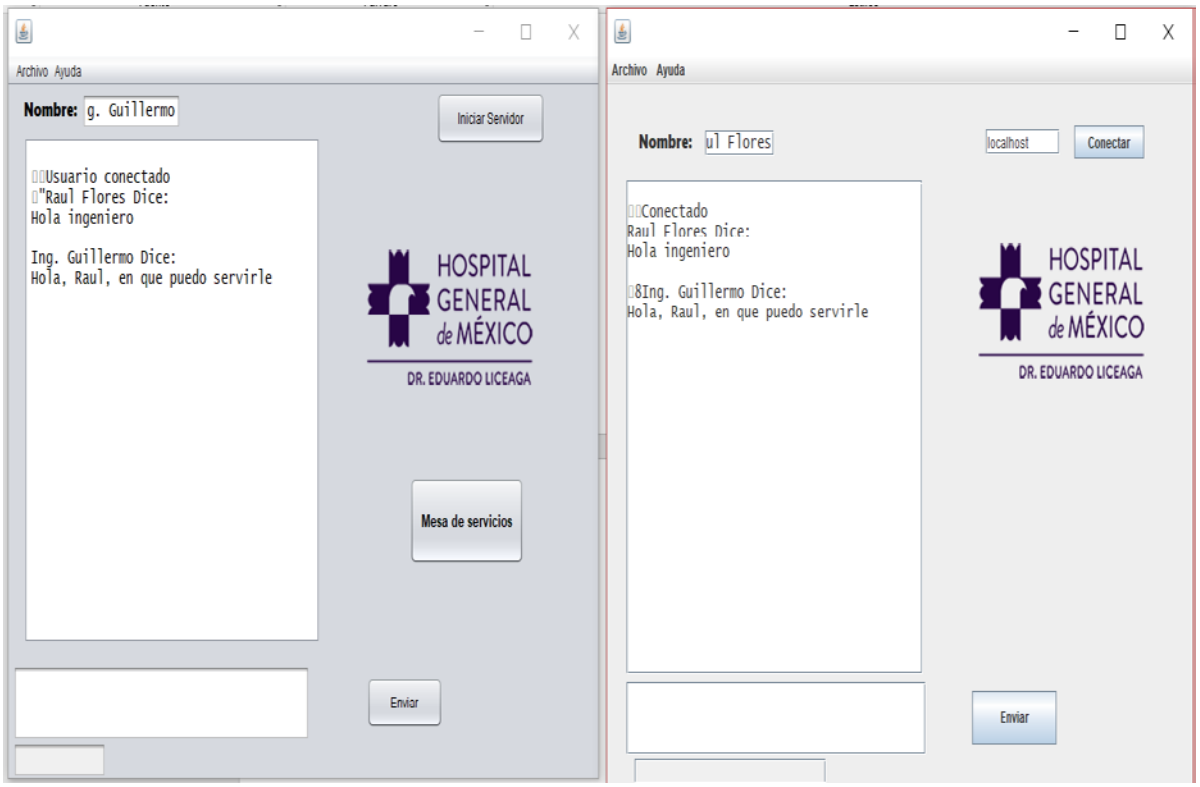
Recordar que se coloca localhost haciendo referencia que intentas conectarte a la misma computadora, por eso la ventana cliente dice localhost.



Cliente enviando mensaje de ayuda por un nodo en mal estado.



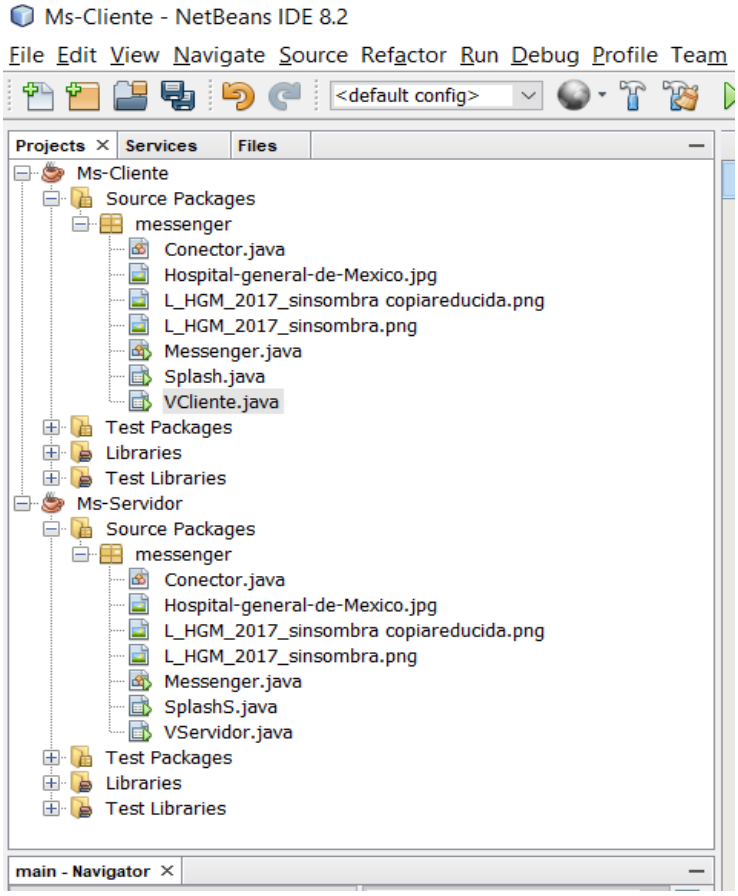
Ventana del servidor recibiendo el mensaje del cliente y respondiendo.



Simulación de conversación mostrando ambas ventanas.

B. Listado de Clases creadas en java (código).

Para facilitar la creación del proyecto decidí crear varias clases en proyectos separados (tanto el de cliente como el de servidor) para poder identificar el código más fácil y por comodidad mostrándose de la siguiente manera:



Como se puede apreciar, se tienen 7 clases por cada proyecto, tanto como el de Ms-Cliente y el Ms-Servidor, por lo tanto comenzare a escribir el código comenzando en orden descendente, es decir, la aplicación Ms-Cliente y con la clase Conector siguiendo por Messenger, splash y Vcliente.

CÓDIGO DE CLASE "CONECTOR" DEL CLIENTE

```
package messenger;
import java.net.*;
import java.io.*;
//comienza la clase conector utilizando hilos para realizar funciones multiples
//y se declaran las variables necesarias para el cliente como lo son sockets,
entradas y salidas.
public class Conector extends Thread{
    private Socket s;
    private ServerSocket ss;
    private InputStreamReader entradaSocket;
    private DataOutputStream salida;
    private BufferedReader entrada;
    final int puerto = 8015;
    byte[] arreglo;
    int in;
    BufferedInputStream bis;
    BufferedOutputStream bos;

    //función en la que se declara la arquitectura cliente servidor como un
cliente
    //*Notar que el socket solicita una ip y el puerto para poder proceder
    public Conector(String ip)
    {
        try{
            this.s = new Socket(ip, this.puerto);

            this.entradaSocket = new
InputStreamReader(this.s.getInputStream());
            this.entrada = new BufferedReader(this.entradaSocket);

            this.salida= new DataOutputStream(this.s.getOutputStream());
            this.salida.writeUTF("Usuario conectado \n");

        }catch (Exception e){};
    }

    //funcion en hilo que permite leer el mensaje e imprimirlo en la ventana de
texto del cliente
    public void run()
    {
        String texto;
        while(true)
        {
            try{
            texto = this.entrada.readLine();
            System.out.println(texto);
            VCliente.jTextArea1.setText(VCliente.jTextArea1.getText()+"\n"+
texto);
        }catch(IOException e){};
        }////

    }
    //funcion que permite enviar el mensaje para el servidor
```

```
    public void enviarMSG(String msg)
  {
    System.out.println("enviado");
    try
    {
      this.salida = new DataOutputStream(this.s.getOutputStream());
      this.salida.writeUTF(msg + " ");
    }
    catch (IOException e)
    {
      System.out.println("Problema al enviar");
    }
  }

    public String leerMSG()
    {
        try{
            return this.entrada.readLine();
        }catch(IOException e){};
        return null;
    }

}
```

CÓDIGO DE CLASE "MESSENGER" DEL CLIENTE

```
package messenger;

//funcion main donde hacemos visible la ventana cliente y se inicializa el hilo
public class Messenger {

    public static Conector servidor,cliente;
    public static void main (String[] args){

        new VCliente().setVisible(false);
        new Thread(new Splash()).start();

    }

    //inicializamos el cliente
    public static void initCliente(String ip)
    {
        cliente = new Conector(ip);
        cliente.start();
    }

}
```

CÓDIGO DE LA CLASE "SPLASH" DEL CLIENTE

```
package messenger;

import java.util.logging.Level;
import java.util.logging.Logger;

/**
 *
 * @author Guillermo
 */
//se inicializa la funcion splash utiliando hilos por el metodo runnable
public class Splash extends javax.swing.JFrame implements Runnable{
    VCliente vc = new VCliente();
    Thread hilo;



    //funcion que permite inicializar la ventana en la parte centrada de la
pantalla
    public Splash() {
        initComponents();
        setLocationRelativeTo(null);
    }


    @SuppressWarnings("unchecked")

    private void initComponents() {

        jLabel1 = new javax.swing.JLabel();
        Barra = new javax.swing.JProgressBar();
        jLabel2 = new javax.swing.JLabel();

        setDefaultCloseOperation(javax.swing.WindowConstants.EXIT_ON_CLOSE);
        setUndecorated(true);

        jLabel1.setHorizontalAlignment(javax.swing.SwingConstants.CENTER);
        jLabel1.setIcon(new
javax.swing.ImageIcon(getClass().getResource("/messenger/L_HGM_2017_sinsombra.p
ng")));

        Barra.setForeground(new java.awt.Color(255, 0, 0));
        Barra.setBorder(javax.swing.BorderFactory.createLineBorder(new
java.awt.Color(0, 0, 0)));

        jLabel2.setText("Cargando...");

        javax.swing.GroupLayout layout = new
javax.swing.GroupLayout(getContentPane());
        getContentPane().setLayout(layout);
        layout.setHorizontalGroup(

layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
            .addGroup(layout.createSequentialGroup()
```

```
                .addContainerGap()
                .addComponent(jLabel2)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
                .addComponent(jLabel1, javax.swing.GroupLayout.DEFAULT_SIZE,
782, Short.MAX_VALUE))
            .addComponent(Barra, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
        );
        layout.setVerticalGroup(

layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
            .addGroup(layout.createSequentialGroup()

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
                    .addComponent(jLabel1,
javax.swing.GroupLayout.DEFAULT_SIZE, 398, Short.MAX_VALUE)
                    .addGroup(javax.swing.GroupLayout.Alignment.TRAILING,
layout.createSequentialGroup()
                        .addGap(0, 0, Short.MAX_VALUE)
                        .addComponent(jLabel2)))
                .addGap(18, 18, 18)
                .addComponent(Barra, javax.swing.GroupLayout.PREFERRED_SIZE,
22, javax.swing.GroupLayout.PREFERRED_SIZE))
        );

        pack();
    }



    //funcion main o principal que permite correr el hilo y mantener visible la
pantalla
    public static void main(String args[]) {

        try {
            for (javax.swing.UIManager.LookAndFeelInfo info :
javax.swing.UIManager.getInstalledLookAndFeels()) {
                if ("Nimbus".equals(info.getName())) {
                    javax.swing.UIManager.setLookAndFeel(info.getClassName());
                    break;
                }
            }
        } catch (ClassNotFoundException ex) {

java.util.logging.Logger.getLogger(Splash.class.getName()).log(java.util.loggin
g.Level.SEVERE, null, ex);
        } catch (InstantiationException ex) {

java.util.logging.Logger.getLogger(Splash.class.getName()).log(java.util.loggin
g.Level.SEVERE, null, ex);
        } catch (IllegalAccessException ex) {

java.util.logging.Logger.getLogger(Splash.class.getName()).log(java.util.loggin
g.Level.SEVERE, null, ex);
```

```
        } catch (javax.swing.UnsupportedLookAndFeelException ex) {
java.util.logging.Logger.getLogger(Splash.class.getName()).log(java.util.loggin
g.Level.SEVERE, null, ex);
        }




        java.awt.EventQueue.invokeLater(new Runnable() {
            public void run() {
                new Splash().setVisible(true);
            }
        });
    }

    // Declaracion de variables
    private javax.swing.JProgressBar Barra;
    private javax.swing.JLabel jLabel1;
    private javax.swing.JLabel jLabel2;
    // Fin de la declaracion

    @Override
    //funcion donde se especifica la accion que hara el hilo y en este caso es
especificar
    //todo el tiempo de vida del hilo y como cargara la barra, ademas de
mantener inicializada la pantalla
    public void run() {

        this.setVisible(true);

        try {
            for(int i=0; i<100; i++){
                Barra.setValue(i);
                hilo.sleep(10);
            }
        } catch (InterruptedException ex) {
            Logger.getLogger(Splash.class.getName()).log(Level.SEVERE, null,
ex);
        }
        this.dispose();
        vc.setVisible(true);
    }
}
```

CÓDIGO DE LA CLASE "VCLIENTE"

```
package messenger;

import java.awt.BorderLayout;
import java.awt.Dimension;
import static java.awt.SystemColor.text;
import java.awt.TextArea;
import java.awt.event.KeyEvent;
import javax.swing.JFileChooser;
import javax.swing.JScrollPane;
import javax.swing.JTextArea;

/**
 *
 * @author Guillermo
 */
public class VCliente extends javax.swing.JFrame {


    //funcion que permite inicializar la ventana en la parte centrada de la
pantalla
    public VCliente() {
        initComponents();
        setLocationRelativeTo(null);
    }

    @SuppressWarnings("unchecked")

    private void initComponents() {

        jButton1 = new javax.swing.JButton();
        jButton2 = new javax.swing.JButton();
        TXTenviarMSG = new javax.swing.JTextField();
        TXTip = new javax.swing.JTextField();
        jScrollPane1 = new javax.swing.JScrollPane();
        jTextArea1 = new javax.swing.JTextArea();
        nickName = new javax.swing.JTextField();
        jLabel1 = new javax.swing.JLabel();
        jLabel2 = new javax.swing.JLabel();
        TxtRuta = new javax.swing.JTextField();
        jMenuBar1 = new javax.swing.JMenuBar();
        jMenu1 = new javax.swing.JMenu();
        jMenu3 = new javax.swing.JMenu();
        jMenuItem1 = new javax.swing.JMenuItem();
        jMenu4 = new javax.swing.JMenu();
        jMenuItem2 = new javax.swing.JMenuItem();
        jMenu2 = new javax.swing.JMenu();

        setDefaultCloseOperation(javax.swing.WindowConstants.EXIT_ON_CLOSE);

        jButton1.setText("Conectar");
        jButton1.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                jButton1ActionPerformed(evt);
            }
```

```
        });

        jButton2.setText("Enviar");
        jButton2.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                jButton2ActionPerformed(evt);
            }
        });

        TXTenviarMSG.setFont(new java.awt.Font("Lucida Console", 0, 15));

        TXTenviarMSG.addPropertyChangeListener(new
java.beans.PropertyChangeListener() {
            public void propertyChange(java.beans.PropertyChangeEvent evt) {
                TXTenviarMSGPropertyChange(evt);
            }
        });
        TXTenviarMSG.addKeyListener(new java.awt.event.KeyAdapter() {
            public void keyTyped(java.awt.event.KeyEvent evt) {
                TXTenviarMSGKeyTyped(evt);
            }
        });

        TXTip.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                TXTipActionPerformed(evt);
            }
        });
        TXTip.addKeyListener(new java.awt.event.KeyAdapter() {
            public void keyTyped(java.awt.event.KeyEvent evt) {
                TXTipKeyTyped(evt);
            }
        });

        jTextArea1.setColumns(20);
        jTextArea1.setFont(new java.awt.Font("Lucida Console", 0, 15));

        jTextArea1.setRows(5);
        jTextArea1.addPropertyChangeListener(new
java.beans.PropertyChangeListener() {
            public void propertyChange(java.beans.PropertyChangeEvent evt) {
                jTextArea1PropertyChange(evt);
            }
        });
        jScrollPane1.setViewportView(jTextArea1);

        nickName.setFont(new java.awt.Font("Lucida Console", 0, 15));
        nickName.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                nickNameActionPerformed(evt);
            }
        });
```

```
        jLabel1.setIcon(new
javax.swing.ImageIcon(getClass().getResource("/messenger/L_HGM_2017_sinsombra
copiareducida.png")));

        jLabel2.setFont(new java.awt.Font("Segoe UI Black", 0, 15));
jLabel2.setText("Nombre: ");

        TxtRuta.setBackground(java.awt.SystemColor.control);

        jMenu1.setText("Archivo");

        jMenu3.setText("Abrir");
        jMenu3.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                jMenu3ActionPerformed(evt);
            }
        });

        jMenuItem1.setText("Seleccionar archivo");
        jMenuItem1.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                jMenuItem1ActionPerformed(evt);
            }
        });
        jMenu3.add(jMenuItem1);

        jMenu1.add(jMenu3);

        jMenu4.setText("Exit");
        jMenu4.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                jMenu4ActionPerformed(evt);
            }
        });

        jMenuItem2.setText("Salir");
        jMenuItem2.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                jMenuItem2ActionPerformed(evt);
            }
        });
        jMenu4.add(jMenuItem2);

        jMenu1.add(jMenu4);

        jMenuBar1.add(jMenu1);

        jMenu2.setText("Ayuda");
        jMenuBar1.add(jMenu2);

        setJMenuBar(jMenuBar1);

        javax.swing.GroupLayout layout = new
javax.swing.GroupLayout(getContentPane());
        getContentPane().setLayout(layout);
```

```
        layout.setHorizontalGroup(

layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
            .addGroup(layout.createSequentialGroup()

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
                    .addGroup(layout.createSequentialGroup()
                        .addGap(39, 39, 39)
                        .addComponent(jLabel2)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
                        .addComponent(nickName,
javax.swing.GroupLayout.PREFERRED_SIZE, 85,
javax.swing.GroupLayout.PREFERRED_SIZE)
                        .addGap(261, 261, 261)
                        .addComponent(TXTip,
javax.swing.GroupLayout.PREFERRED_SIZE, 91,
javax.swing.GroupLayout.PREFERRED_SIZE)
                        .addGap(18, 18, 18)
                        .addComponent(jButton1))
                    .addGroup(layout.createSequentialGroup()
                        .addGap(24, 24, 24)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
                            .addComponent(jScrollPane1,
javax.swing.GroupLayout.PREFERRED_SIZE, 365,
javax.swing.GroupLayout.PREFERRED_SIZE)
                            .addComponent(TXTenviarMSG,
javax.swing.GroupLayout.PREFERRED_SIZE, 369,
javax.swing.GroupLayout.PREFERRED_SIZE)
                            .addGroup(layout.createSequentialGroup()
                                .addGap(10, 10, 10)
                                .addComponent(TxtRuta,
javax.swing.GroupLayout.PREFERRED_SIZE, 236,
javax.swing.GroupLayout.PREFERRED_SIZE)))
                        .addGap(57, 57, 57)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
                            .addComponent(jButton2,
javax.swing.GroupLayout.PREFERRED_SIZE, 104,
javax.swing.GroupLayout.PREFERRED_SIZE)
                            .addComponent(jLabel1))))
                .addContainerGap(52, Short.MAX_VALUE))
        );
        layout.setVerticalGroup(

layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
            .addGroup(layout.createSequentialGroup()
                .addGap(32, 32, 32)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE
)
                    .addComponent(jLabel2)
```

```
                    .addComponent(nickName,
javax.swing.GroupLayout.PREFERRED_SIZE, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
                    .addComponent(TXTip,
javax.swing.GroupLayout.PREFERRED_SIZE, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
                    .addComponent(jButton1))
                .addGap(18, 18, 18)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
                    .addComponent(jScrollPane1,
javax.swing.GroupLayout.PREFERRED_SIZE, 392,
javax.swing.GroupLayout.PREFERRED_SIZE)
                    .addComponent(jLabel1,
javax.swing.GroupLayout.PREFERRED_SIZE, 208,
javax.swing.GroupLayout.PREFERRED_SIZE))

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE
)
                    .addComponent(TXTenviarMSG,
javax.swing.GroupLayout.PREFERRED_SIZE, 58,
javax.swing.GroupLayout.PREFERRED_SIZE)
                    .addComponent(jButton2,
javax.swing.GroupLayout.PREFERRED_SIZE, 43,
javax.swing.GroupLayout.PREFERRED_SIZE))
                .addGap(3, 3, 3)
                .addComponent(TxtRuta, javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.GroupLayout.PREFERRED_SIZE))
        );

        pack();
    }



    private void jButton1ActionPerformed(java.awt.event.ActionEvent evt) {
         Messenger.initCliente(this.TXTip.getText()); //inicializa el cliente
obteniendo la direccion IP
    }

    //Se obtienen los textos, los nick names y las rutas de los archivos y se
imprimen en el text area principal
    private void jButton2ActionPerformed(java.awt.event.ActionEvent evt) {
        Messenger.cliente.enviarMSG(this.nickName.getText()+ " Dice: "+
         this.TXTenviarMSG.getText()+"\n"+this.TxtRuta.getText());
        this.jTextArea1.setText(this.jTextArea1.getText()+"\n"+
                this.nickName.getText()+" Dice: "+
                this.TXTenviarMSG.getText()+ "\n" +this.TxtRuta.getText());
        TXTenviarMSG.setText(null);
        TxtRuta.setText(null);

    }
```

```
    //Se crea la alternativa de que el boton de enviar detecte el enter como
click
    private void TXTenviarMSGKeyTyped(java.awt.event.KeyEvent evt) {

        char cTeclaPresionada = evt.getKeyChar();
        if(cTeclaPresionada == KeyEvent.VK_ENTER)
        {
            jButton2.doClick();
        }
    }

    //Se crea la alternativa de que el boton de IP o conectar detecte el enter
como click
    private void TXTipKeyTyped(java.awt.event.KeyEvent evt) {

        char cTeclaPresionada = evt.getKeyChar();
        if(cTeclaPresionada == KeyEvent.VK_ENTER)
        {
            jButton1.doClick();
        }
    }

    private void nickNameActionPerformed(java.awt.event.ActionEvent evt) {

    }

        //Cierra el programa cuando se pulsa exit en el menu superior
    private void jMenu4ActionPerformed(java.awt.event.ActionEvent evt) {

        System.exit(0);
    }

    private void jMenu3ActionPerformed(java.awt.event.ActionEvent evt) {


    }


    //se utiliza un filechooser para crear la ventana que da la accion de
seleccionar archivos y enviarlos
    private void jMenuItem1ActionPerformed(java.awt.event.ActionEvent evt) {

        JFileChooser filechoose = new JFileChooser();
        int opcion = filechoose.showOpenDialog(this);
        if(opcion==JFileChooser.APPROVE_OPTION)
        {
            String nombre_archivo = filechoose.getSelectedFile().getName();
           String ruta = filechoose.getSelectedFile().getPath();

            TxtRuta.setText(ruta);

        }
    }
```

```
    private void jMenuItem2ActionPerformed(java.awt.event.ActionEvent evt) {

        System.exit(0);
    }

    //funcion de seguridad que evita que el area de conversacion pueda ser
editada
    private void jTextArea1PropertyChange(java.beans.PropertyChangeEvent evt) {

        jTextArea1.setLineWrap(true);
        jTextArea1.setEditable(false);
    }

    //funcion que edita el texto y lo alinea a la izquierda
    private void TXTenviarMSGPropertyChange(java.beans.PropertyChangeEvent evt)
{

        TXTenviarMSG.setHorizontalAlignment(TXTenviarMSG.LEFT);

    }



    public static void main(String args[]) {

        try {
            for (javax.swing.UIManager.LookAndFeelInfo info :
javax.swing.UIManager.getInstalledLookAndFeels()) {
                if ("Nimbus".equals(info.getName())) {
                    javax.swing.UIManager.setLookAndFeel(info.getClassName());
                    break;
                }
            }
        } catch (ClassNotFoundException ex) {

java.util.logging.Logger.getLogger(VCliente.class.getName()).log(java.util.logg
ing.Level.SEVERE, null, ex);
        } catch (InstantiationException ex) {

java.util.logging.Logger.getLogger(VCliente.class.getName()).log(java.util.logg
ing.Level.SEVERE, null, ex);
        } catch (IllegalAccessException ex) {

java.util.logging.Logger.getLogger(VCliente.class.getName()).log(java.util.logg
ing.Level.SEVERE, null, ex);
        } catch (javax.swing.UnsupportedLookAndFeelException ex) {

java.util.logging.Logger.getLogger(VCliente.class.getName()).log(java.util.logg
ing.Level.SEVERE, null, ex);
        }


        //se inicia el hilo o el runnable
        java.awt.EventQueue.invokeLater(new Runnable() {
```

```
            public void run() {
                new VCliente().setVisible(true);

            }
        });
    }

    // Declaracion de variables
    private javax.swing.JTextField TXTenviarMSG;
    private javax.swing.JTextField TXTip;
    private javax.swing.JTextField TxtRuta;
    private javax.swing.JButton jButton1;
    private javax.swing.JButton jButton2;
    private javax.swing.JLabel jLabel1;
    private javax.swing.JLabel jLabel2;
    private javax.swing.JMenu jMenu1;
    private javax.swing.JMenu jMenu2;
    private javax.swing.JMenu jMenu3;
    private javax.swing.JMenu jMenu4;
    private javax.swing.JMenuBar jMenuBar1;
    private javax.swing.JMenuItem jMenuItem1;
    private javax.swing.JMenuItem jMenuItem2;
    private javax.swing.JScrollPane jScrollPane1;
    public static javax.swing.JTextArea jTextArea1;
    private javax.swing.JTextField nickName;
    //Fin de declaracion
}
```

Ahora se mostrará el código de la aplicación servidor igualmente por clases

CÓDIGO DE LA CLASE "CONECTOR" DEL SERVIDOR

```
//Declaracion de variables e importacion de librerias
package messenger;
import java.net.*;
import java.io.*;
import messenger.VServidor;
//comienza la clase conector utilizando hilos para realizar funciones multiples
//y se declaran las variables necesarias para el servidor como lo son
serversockets, entradas y salidas.
public class Conector extends Thread {
    private Socket s;
    private ServerSocket ss;
    private InputStreamReader entradaSocket;
    private DataOutputStream salida;
    private BufferedReader entrada;
    final int puerto = 8015;
    BufferedInputStream bis;
    BufferedOutputStream bos;
    byte[] recive;
    int in;
    String file;

    //funcion que ingresa el nombre en el campo de la ventana
    public Conector(String nombre)
    {
        super(nombre);
    }
    //se crea la funcion en la cual se manda como salida un mensaje utilizando
writeUTF
    public void enviarMSG(String msg)
    {
        try{
            this.salida.writeUTF(msg + " ");
        }catch (IOException e){};
    }

    //utilizando hilos, se crea ademas la estrucutra basica de una arquitectura
cliente servidor declarando el socket mandandole el puerto
    //y mandando mensajes de conectado, cabe mencionar que toda la informacion
se coloca
    //directamente en la ventana servidor utilizando el setText
    public void run()
    {
     String text;
     try{
         this.ss = new ServerSocket(puerto);
         this.s = ss.accept();

         this.entradaSocket = new InputStreamReader(this.s.getInputStream());
         this.entrada = new BufferedReader(this.entradaSocket);
```

```
            this.salida = new DataOutputStream(this.s.getOutputStream());
            this.salida.writeUTF("Conectado \n");
            while(true)
            {
                text = this.entrada.readLine();
                System.out.println(text);

VServidor.jTextArea1.setText(VServidor.jTextArea1.getText()+"\n"+text);

            }
            }catch (IOException e){
            System.out.println("Algun Tipo de error");
            };
        }

        //función que recibe un mensaje utilizando readLine
        public String leerMSG()
        {

           // return null;
            try{
                return this.entrada.readLine();
            }catch(IOException e){};
            return null;


        }
        //funcion que cierra la conexion del socket para no dejarla abierta
        public void desconectar()
        {
            try{
                s.close();
            }catch(IOException e){};
            try{
                ss.close();
            }catch(IOException e){};
        }
}
```

CÓDIGO DE LA CLASE "MESSENGER" DEL SERVIDOR

```
package messenger;

//funcion main donde hacemos visible la ventana servidor
public class Messenger {
    public static Conector servidor,cliente;
    public static void main (String[] args){
        VServidor server = new VServidor();
        server.show();



    }

    //inicializamos el servidor
    public static void initServer()
    {
        servidor = new Conector("hilos");
        servidor.start();
    }

}
```

CÓDIGO DE LA CLASE "SPLASH" DEL SERVIDOR

```
package messenger;

import java.util.logging.Level;
import java.util.logging.Logger;

/**
 *
 * @author Astic
 */
//clase splash declarada de manera que permite utilizar hilos por el metodo
runnable
public class SplashS extends javax.swing.JFrame implements Runnable{
    VServidor vss = new VServidor();

    Thread hilo;
    /**
     * Creates new form SplashS
     */
    public SplashS() {
        initComponents();
        setLocationRelativeTo(null);
    }

    @SuppressWarnings("unchecked")

    private void initComponents() {

        jLabel1 = new javax.swing.JLabel();
        Barra2 = new javax.swing.JProgressBar();
        jLabel2 = new javax.swing.JLabel();

        setDefaultCloseOperation(javax.swing.WindowConstants.EXIT_ON_CLOSE);
        setUndecorated(true);

        jLabel1.setHorizontalAlignment(javax.swing.SwingConstants.CENTER);
        jLabel1.setIcon(new
javax.swing.ImageIcon(getClass().getResource("/messenger/L_HGM_2017_sinsombra.p
ng")));

        jLabel2.setText("Cargando...");

        javax.swing.GroupLayout layout = new
javax.swing.GroupLayout(getContentPane());
        getContentPane().setLayout(layout);
        layout.setHorizontalGroup(

layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
            .addGroup(layout.createSequentialGroup()
                .addContainerGap()
                .addComponent(jLabel2)
```

```
.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
                .addComponent(jLabel1, javax.swing.GroupLayout.DEFAULT_SIZE,
783, Short.MAX_VALUE))
            .addComponent(Barra2, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
        );
        layout.setVerticalGroup(

layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
            .addGroup(layout.createSequentialGroup()

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
                    .addComponent(jLabel1,
javax.swing.GroupLayout.DEFAULT_SIZE, 408, Short.MAX_VALUE)
                    .addGroup(layout.createSequentialGroup()
                        .addGap(0, 0, Short.MAX_VALUE)
                        .addComponent(jLabel2)))

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
                .addComponent(Barra2, javax.swing.GroupLayout.PREFERRED_SIZE,
24, javax.swing.GroupLayout.PREFERRED_SIZE))
        );

        pack();
    }



    //main que permite visualizar la ventana splash
    public static void main(String args[]) {
        try {
            for (javax.swing.UIManager.LookAndFeelInfo info :
javax.swing.UIManager.getInstalledLookAndFeels()) {
                if ("Nimbus".equals(info.getName())) {
                    javax.swing.UIManager.setLookAndFeel(info.getClassName());
                    break;
                }
            }
        } catch (ClassNotFoundException ex) {

java.util.logging.Logger.getLogger(SplashS.class.getName()).log(java.util.loggi
ng.Level.SEVERE, null, ex);
        } catch (InstantiationException ex) {

java.util.logging.Logger.getLogger(SplashS.class.getName()).log(java.util.loggi
ng.Level.SEVERE, null, ex);
        } catch (IllegalAccessException ex) {

java.util.logging.Logger.getLogger(SplashS.class.getName()).log(java.util.loggi
ng.Level.SEVERE, null, ex);
        } catch (javax.swing.UnsupportedLookAndFeelException ex) {

java.util.logging.Logger.getLogger(SplashS.class.getName()).log(java.util.loggi
ng.Level.SEVERE, null, ex);
```

```
        }

        java.awt.EventQueue.invokeLater(new Runnable() {
            public void run() {
                new SplashS().setVisible(true);
            }
        });
    }

    // Variables declaration - do not modify
    private javax.swing.JProgressBar Barra2;
    private javax.swing.JLabel jLabel1;
    private javax.swing.JLabel jLabel2;
    // End of variables declaration

    @Override
    //funcion donde permite modificar el estado del hilo, la barra de carga y
ponerlos visibles
    public void run() {
        this.setVisible(true);

        try {
            for(int a=0; a<100; a++){
                Barra2.setValue(a);
                hilo.sleep(10);
            }

        } catch (InterruptedException ex) {
            Logger.getLogger(SplashS.class.getName()).log(Level.SEVERE, null,
ex);
        }
        this.dispose();
        vss.setVisible(true);
    }
}
```

CÓDIGO DE LA CLASE "VSERVIDOR"

```
package messenger;
import java.awt.Desktop;
import java.awt.event.KeyEvent;
import java.net.URI;
import javax.swing.JFileChooser;
import java.io.*;
/**
 *
 * @author Guillermo
 */
public class VServidor extends javax.swing.JFrame {


    public VServidor() {
        initComponents();
        setLocationRelativeTo(null); //inicializa la ventana principal en la
parte central de la pantalla
    }


    @SuppressWarnings("unchecked")
    private void initComponents() {

        jButton1 = new javax.swing.JButton();
        jButton2 = new javax.swing.JButton();
        jButton3 = new javax.swing.JButton();
        TXTmsgEnviar = new javax.swing.JTextField();
        jScrollPane1 = new javax.swing.JScrollPane();
        jTextArea1 = new javax.swing.JTextArea();
        nickServidor = new javax.swing.JTextField();
        jLabel1 = new javax.swing.JLabel();
        jLabel2 = new javax.swing.JLabel();
        TxtRutaS = new javax.swing.JTextField();
        jButton4 = new javax.swing.JButton();
        jMenuBar1 = new javax.swing.JMenuBar();
        jMenu1 = new javax.swing.JMenu();
        jMenu3 = new javax.swing.JMenu();
        jMenuItem1 = new javax.swing.JMenuItem();
        jMenu4 = new javax.swing.JMenu();
        jMenuItem2 = new javax.swing.JMenuItem();
        jMenu2 = new javax.swing.JMenu();

        jButton1.setText("jButton1");

        setDefaultCloseOperation(javax.swing.WindowConstants.EXIT_ON_CLOSE);

        jButton2.setText("Iniciar Servidor");
        jButton2.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                jButton2ActionPerformed(evt);
            }
```

```
        });

        jButton3.setText("Enviar");
        jButton3.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                jButton3ActionPerformed(evt);
            }
        });
        jButton3.addKeyListener(new java.awt.event.KeyAdapter() {
            public void keyTyped(java.awt.event.KeyEvent evt) {
                jButton3KeyTyped(evt);
            }
        });

        TXTmsgEnviar.setFont(new java.awt.Font("Lucida Console", 0, 15));

        TXTmsgEnviar.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                TXTmsgEnviarActionPerformed(evt);
            }
        });
        TXTmsgEnviar.addKeyListener(new java.awt.event.KeyAdapter() {
            public void keyTyped(java.awt.event.KeyEvent evt) {
                TXTmsgEnviarKeyTyped(evt);
            }
        });

        jTextArea1.setColumns(20);
        jTextArea1.setFont(new java.awt.Font("Lucida Console", 0, 15));

        jTextArea1.setRows(5);
        jTextArea1.addPropertyChangeListener(new
java.beans.PropertyChangeListener() {
            public void propertyChange(java.beans.PropertyChangeEvent evt) {
                jTextArea1PropertyChange(evt);
            }
        });
        jScrollPane1.setViewportView(jTextArea1);

        nickServidor.setFont(new java.awt.Font("Lucida Console", 0, 15));

        nickServidor.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                nickServidorActionPerformed(evt);
            }
        });

        jLabel1.setIcon(new
javax.swing.ImageIcon(getClass().getResource("/messenger/L_HGM_2017_sinsombra
copiareducida.png")));

        jLabel2.setFont(new java.awt.Font("Segoe UI Black", 0, 15)); // NOI18N
        jLabel2.setText("Nombre:");

        TxtRutaS.setBackground(java.awt.SystemColor.control);
```

```
        TxtRutaS.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                TxtRutaSActionPerformed(evt);
            }
        });
        TxtRutaS.addKeyListener(new java.awt.event.KeyAdapter() {
            public void keyTyped(java.awt.event.KeyEvent evt) {
                TxtRutaSKeyTyped(evt);
            }
        });

        jButton4.setFont(new java.awt.Font("Arial", 1, 13));
        jButton4.setText("Mesa de servicios");
        jButton4.addMouseListener(new java.awt.event.MouseAdapter() {
            public void mouseClicked(java.awt.event.MouseEvent evt) {
                jButton4MouseClicked(evt);
            }
        });

        jMenu1.setText("Archivo");

        jMenu3.setText("Adjuntar");

        jMenuItem1.setText("Seleccionar archivo");
        jMenuItem1.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                jMenuItem1ActionPerformed(evt);
            }
        });
        jMenu3.add(jMenuItem1);

        jMenu1.add(jMenu3);

        jMenu4.setText("Exit");

        jMenuItem2.setText("Salir");
        jMenuItem2.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                jMenuItem2ActionPerformed(evt);
            }
        });
        jMenu4.add(jMenuItem2);

        jMenu1.add(jMenu4);

        jMenuBar1.add(jMenu1);

        jMenu2.setText("Ayuda");
        jMenuBar1.add(jMenu2);

        setJMenuBar(jMenuBar1);

        javax.swing.GroupLayout layout = new
javax.swing.GroupLayout(getContentPane());
        getContentPane().setLayout(layout);
```

```
        layout.setHorizontalGroup(

layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
            .addGroup(layout.createSequentialGroup()

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
                    .addGroup(layout.createSequentialGroup()
                        .addContainerGap()
                        .addComponent(TxtRutaS,
javax.swing.GroupLayout.PREFERRED_SIZE, 114,
javax.swing.GroupLayout.PREFERRED_SIZE))
                    .addGroup(layout.createSequentialGroup()

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
                            .addGroup(layout.createSequentialGroup()
                                .addGap(19, 19, 19)
                                .addComponent(jLabel2)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
                                .addComponent(nickServidor,
javax.swing.GroupLayout.PREFERRED_SIZE, 121,
javax.swing.GroupLayout.PREFERRED_SIZE))
                            .addGroup(layout.createSequentialGroup()
                                .addContainerGap()
                                .addComponent(TXTmsgEnviar,
javax.swing.GroupLayout.PREFERRED_SIZE, 365,
javax.swing.GroupLayout.PREFERRED_SIZE))

.addGroup(javax.swing.GroupLayout.Alignment.TRAILING,
layout.createSequentialGroup()
                                .addGap(19, 19, 19)
                                .addComponent(jScrollPane1,
javax.swing.GroupLayout.PREFERRED_SIZE, 358,
javax.swing.GroupLayout.PREFERRED_SIZE)))
                        .addGap(11, 11, 11)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
                            .addGroup(layout.createSequentialGroup()
                                .addGap(46, 46, 46)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.TRAILING
)
                                    .addComponent(jLabel1,
javax.swing.GroupLayout.PREFERRED_SIZE, 227,
javax.swing.GroupLayout.PREFERRED_SIZE)
                                    .addComponent(jButton2,
javax.swing.GroupLayout.PREFERRED_SIZE, 133,
javax.swing.GroupLayout.PREFERRED_SIZE)))
                            .addGroup(layout.createSequentialGroup()
                                .addGap(54, 54, 54)
                                .addComponent(jButton3,
javax.swing.GroupLayout.PREFERRED_SIZE, 93,
javax.swing.GroupLayout.PREFERRED_SIZE))
                            .addGroup(layout.createSequentialGroup()
                                .addGap(107, 107, 107)
```

```
                                .addComponent(jButton4)))))
                    .addContainerGap(66, Short.MAX_VALUE))
        );
        layout.setVerticalGroup(

layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
            .addGroup(layout.createSequentialGroup()
                .addGap(6, 6, 6)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
                    .addGroup(layout.createSequentialGroup()
                        .addGap(1, 1, 1)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE
)
                            .addComponent(nickServidor,
javax.swing.GroupLayout.PREFERRED_SIZE, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
                            .addComponent(jLabel2))

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
                        .addComponent(jScrollPane1,
javax.swing.GroupLayout.PREFERRED_SIZE, 403,
javax.swing.GroupLayout.PREFERRED_SIZE)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED, 17,
Short.MAX_VALUE)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE
)
                            .addComponent(TXTmsgEnviar,
javax.swing.GroupLayout.PREFERRED_SIZE, 60,
javax.swing.GroupLayout.PREFERRED_SIZE)
                            .addComponent(jButton3,
javax.swing.GroupLayout.PREFERRED_SIZE, 41,
javax.swing.GroupLayout.PREFERRED_SIZE))
                        .addGap(1, 1, 1)
                        .addComponent(TxtRutaS,
javax.swing.GroupLayout.PREFERRED_SIZE, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE))
                    .addGroup(layout.createSequentialGroup()
                        .addComponent(jButton2,
javax.swing.GroupLayout.PREFERRED_SIZE, 41,
javax.swing.GroupLayout.PREFERRED_SIZE)
                        .addGap(18, 18, 18)
                        .addComponent(jLabel1,
javax.swing.GroupLayout.PREFERRED_SIZE, 241,
javax.swing.GroupLayout.PREFERRED_SIZE)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
                        .addComponent(jButton4,
javax.swing.GroupLayout.PREFERRED_SIZE, 69,
javax.swing.GroupLayout.PREFERRED_SIZE)
                        .addGap(0, 0, Short.MAX_VALUE))))
        );
```

```
        pack();
    }
    //funcion que inicializa el boton "iniciar servidor"
    private void jButton2ActionPerformed(java.awt.event.ActionEvent evt) {
       Messenger.initServer();
    }
    //en esta funcion se estipulan las acciones que realizara el boton enviar,
como enviar el mensaje que recibe del el area escribir mensaje
    private void jButton3ActionPerformed(java.awt.event.ActionEvent evt) {
        //Messenger.servidor.enviarMSG(this.TXTmsgEnviar.getText());
        Messenger.servidor.enviarMSG(this.nickServidor.getText()+ " Dice: "+
         this.TXTmsgEnviar.getText() + "\n" + this.TxtRutaS.getText());
     /*   this.jTextArea1.setText(this.jTextArea1.getText()+"\n"+
                this.nickServidor.getText()+" Dice:\n"+
                this.TXTmsgEnviar.getText()+"\n"+this.TxtRutaS.getText());
        TXTmsgEnviar.setText(null);
        TxtRutaS.setText(null);*/
     this.jTextArea1.setText(this.jTextArea1.getText()+"\n"+
                this.nickServidor.getText()+" Dice: "+
                this.TXTmsgEnviar.getText()+"\n"+this.TxtRutaS.getText());
        this.TXTmsgEnviar.setText(null);
        this.TxtRutaS.setText(null);


    }



    //funcion que permite al boton enviar detectar el enter como un click
    private void TXTmsgEnviarKeyTyped(java.awt.event.KeyEvent evt) {
        char cTeclaPresionada = evt.getKeyChar();
        if(cTeclaPresionada == KeyEvent.VK_ENTER)
        {
            jButton3.doClick();
        }
    }

    private void nickServidorActionPerformed(java.awt.event.ActionEvent evt) {
    }

    private void jMenuItem1ActionPerformed(java.awt.event.ActionEvent evt) {
        JFileChooser filechoose = new JFileChooser();
        int opcion = filechoose.showOpenDialog(this);
        if(opcion==JFileChooser.APPROVE_OPTION)
        {
            String nombre_archivo = filechoose.getSelectedFile().getName();
           String ruta = filechoose.getSelectedFile().getPath();

            this.TxtRutaS.setText(ruta);

        }
    }

    private void TxtRutaSKeyTyped(java.awt.event.KeyEvent evt) {
```

```
        char cTeclaPresionada2 = evt.getKeyChar();
        if(cTeclaPresionada2 == KeyEvent.VK_ENTER)
        {
            jButton3.doClick();
        }
    }

    private void jMenuItem2ActionPerformed(java.awt.event.ActionEvent evt) {
        System.exit(0);
    }

    //funcion que permite al usuario dar click en mesa de servicios y que abra
una pagina web
    private void jButton4MouseClicked(java.awt.event.MouseEvent evt) {
        try{

            if (Desktop.isDesktopSupported()){
                Desktop desktop = Desktop.getDesktop();
               if(desktop.isSupported(Desktop.Action.BROWSE)){
                    desktop.browse(new
URI("http://192.168.0.25/intranet/mesaserv/"));
                }
            }

        }catch(Exception e){
            e.printStackTrace();
        }
    }



    //funcion que evita modificar el area de texto por seguridad y no editar
conversaciones
    private void jTextArea1PropertyChange(java.beans.PropertyChangeEvent evt) {
        jTextArea1.setLineWrap(true);
        jTextArea1.setEditable(false);


//jTextArea1.setText(this.jTextArea1.getText()+"\n"+this.TxtRutaS.getText());
    }




    //funcion principal main que inicializa la ventana y el hilo para despues
del splash
    public static void main(String args[]) {

        try {
            for (javax.swing.UIManager.LookAndFeelInfo info :
javax.swing.UIManager.getInstalledLookAndFeels()) {
                if ("Nimbus".equals(info.getName())) {
                    javax.swing.UIManager.setLookAndFeel(info.getClassName());
                    break;
                }
```

```
            }
        } catch (ClassNotFoundException ex) {

java.util.logging.Logger.getLogger(VServidor.class.getName()).log(java.util.log
ging.Level.SEVERE, null, ex);
        } catch (InstantiationException ex) {

java.util.logging.Logger.getLogger(VServidor.class.getName()).log(java.util.log
ging.Level.SEVERE, null, ex);
        } catch (IllegalAccessException ex) {

java.util.logging.Logger.getLogger(VServidor.class.getName()).log(java.util.log
ging.Level.SEVERE, null, ex);
        } catch (javax.swing.UnsupportedLookAndFeelException ex) {

java.util.logging.Logger.getLogger(VServidor.class.getName()).log(java.util.log
ging.Level.SEVERE, null, ex);
        }


//hilo utilizado mediante runnable para el splash
        java.awt.EventQueue.invokeLater(new Runnable() {
            public void run() {
                new VServidor().setVisible(false);
                new Thread(new SplashS()).start();
            }
        });
    }

    // Declaracion de variables
    private javax.swing.JTextField TXTmsgEnviar;
    private javax.swing.JTextField TxtRutaS;
    private javax.swing.JButton jButton1;
    private javax.swing.JButton jButton2;
    public javax.swing.JButton jButton3;
    private javax.swing.JButton jButton4;
    private javax.swing.JLabel jLabel1;
    private javax.swing.JLabel jLabel2;
    private javax.swing.JMenu jMenu1;
    private javax.swing.JMenu jMenu2;
    private javax.swing.JMenu jMenu3;
    private javax.swing.JMenu jMenu4;
    private javax.swing.JMenuBar jMenuBar1;
    private javax.swing.JMenuItem jMenuItem1;
    private javax.swing.JMenuItem jMenuItem2;
    private javax.swing.JScrollPane jScrollPane1;
    public static javax.swing.JTextArea jTextArea1;
    public javax.swing.JTextField nickServidor;
    // Fin de la delcaracion
}
```

C. Recomendaciones e instalación de java

Al haber creado la aplicación en el lenguaje de programación java, es necesario tener instalado el software java para el correcto funcionamiento.

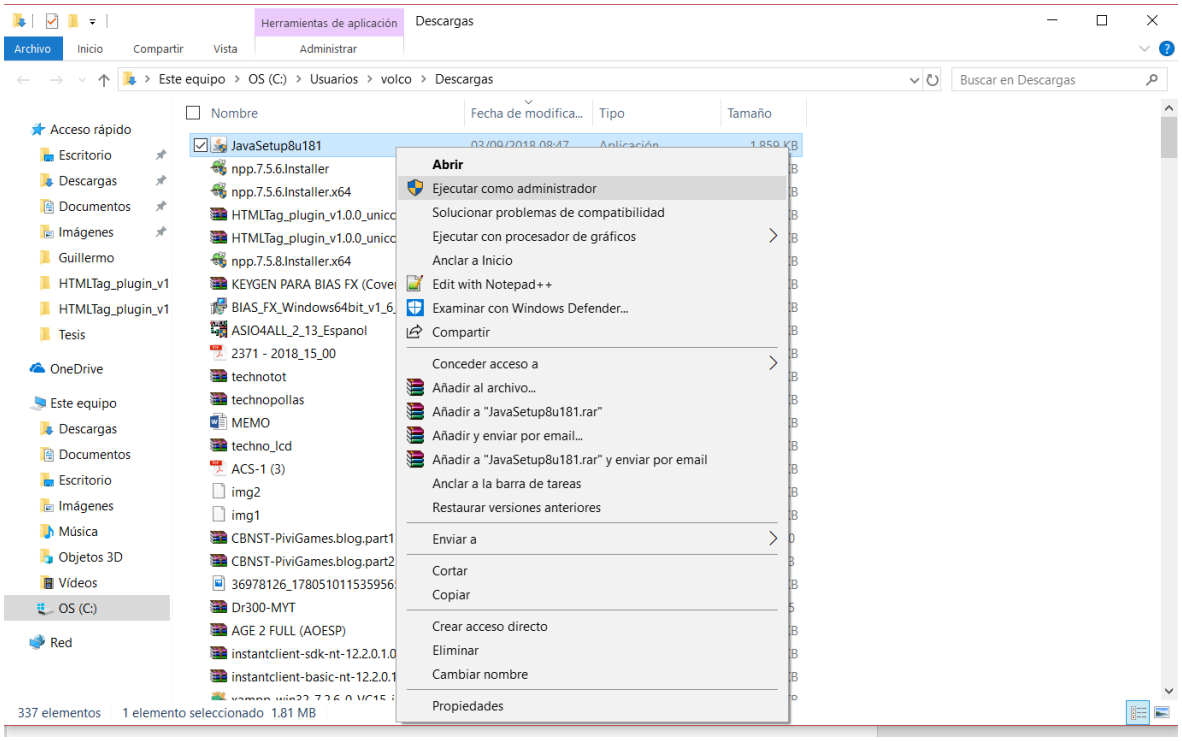
El software se puede descargar de la página oficial de Oracle:

https://www.java.com/es/download/

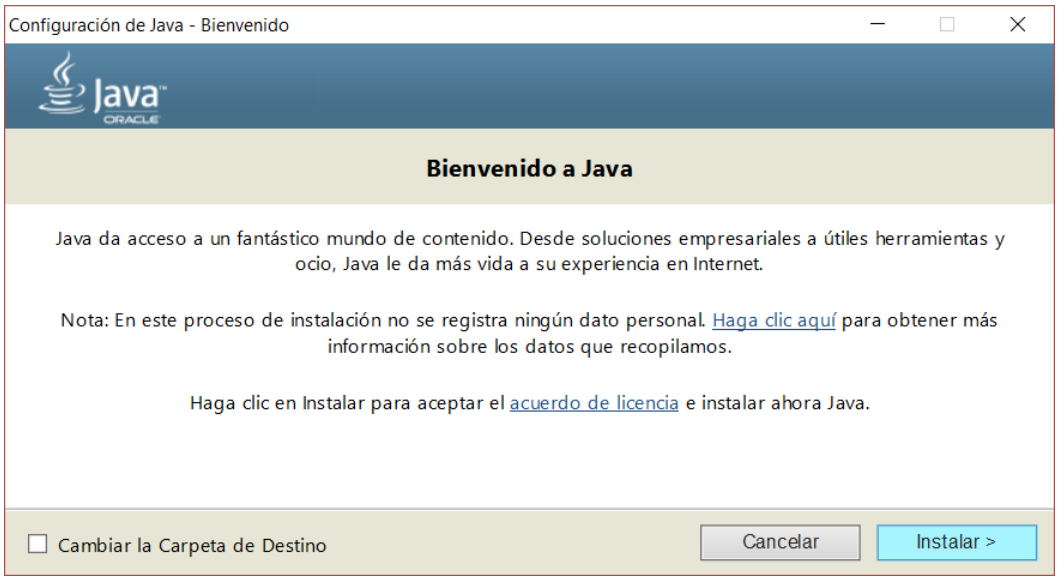
Al abrir la página, se nos mostrará la siguiente pantalla y daremos clic en descarga gratuita de java.

Después de aceptar los términos se comenzará la descarga del ejecutable de la aplicación java.

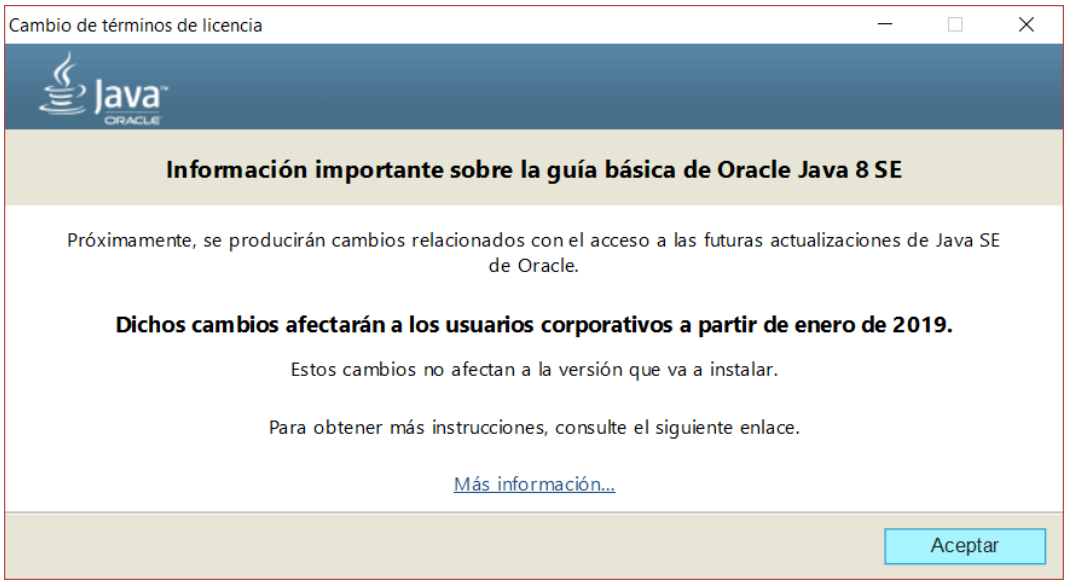
Una vez descargada, nos iremos a la carpeta de descargas o a la carpeta donde se haya descargado el ejecutable y lo ejecutamos como administrador.



AL ejecutar, nos abrirá el asistente de instalación y deberemos dar click en instalar.

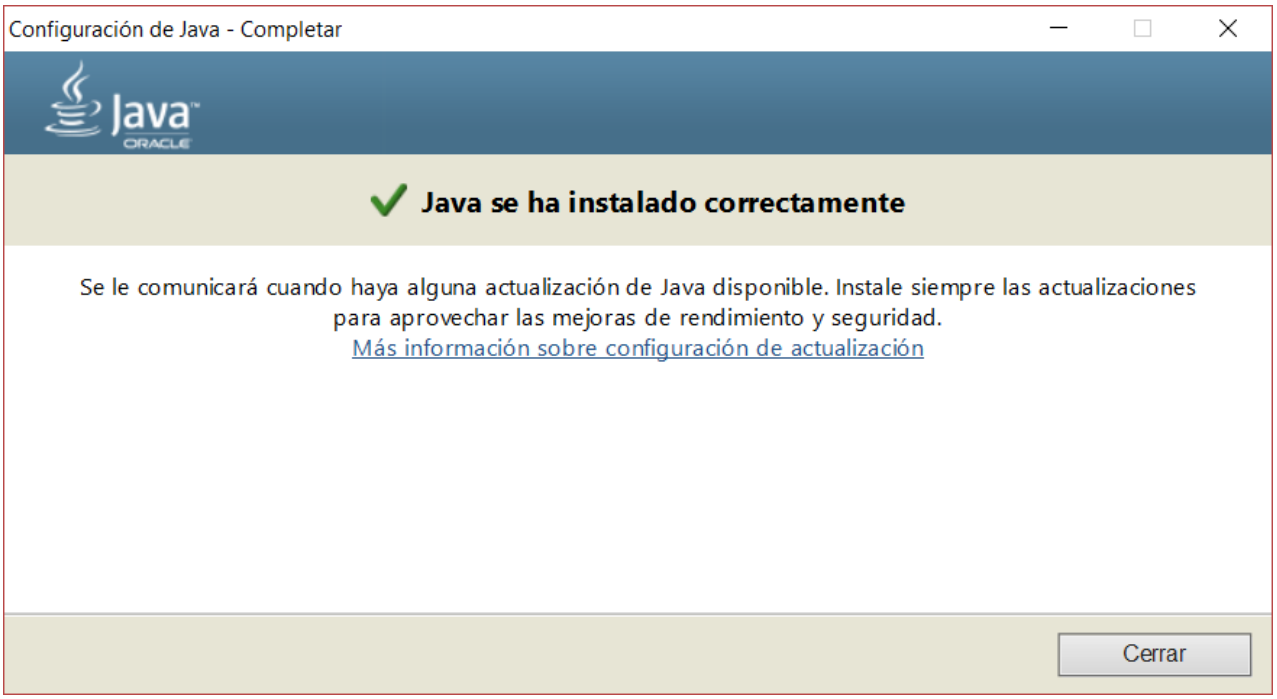


Después le damos click en aceptar.



Comenzará la instalación.

Una vez concluida la instalación, le daremos click en cerrar y con eso quedará instalado correctamente el software java para poder utilizar la aplicación SCI.



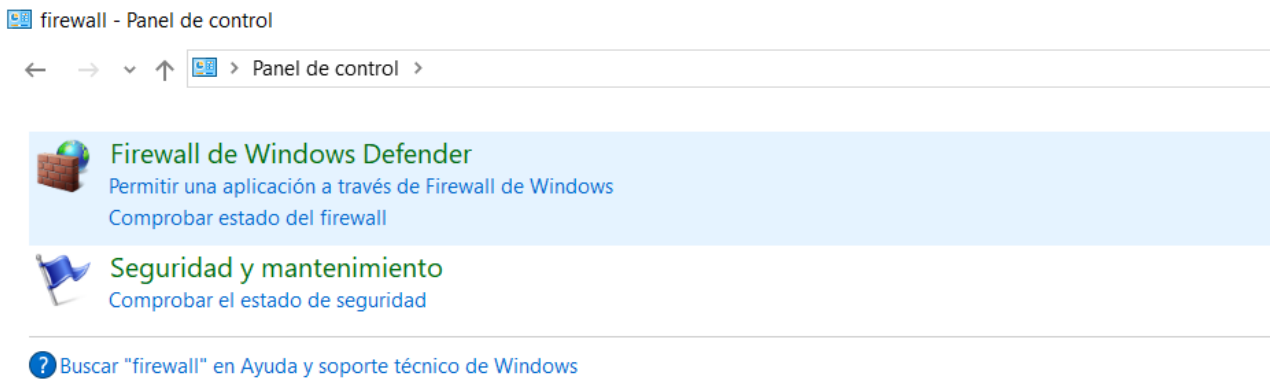
D. Corrección de errores comunes.

Al utilizar la aplicación SCI (sistema de comunicación interna) se presentó un error en Windows 7 y Windows 10 sobre comunicación entre el cliente y el servidor.

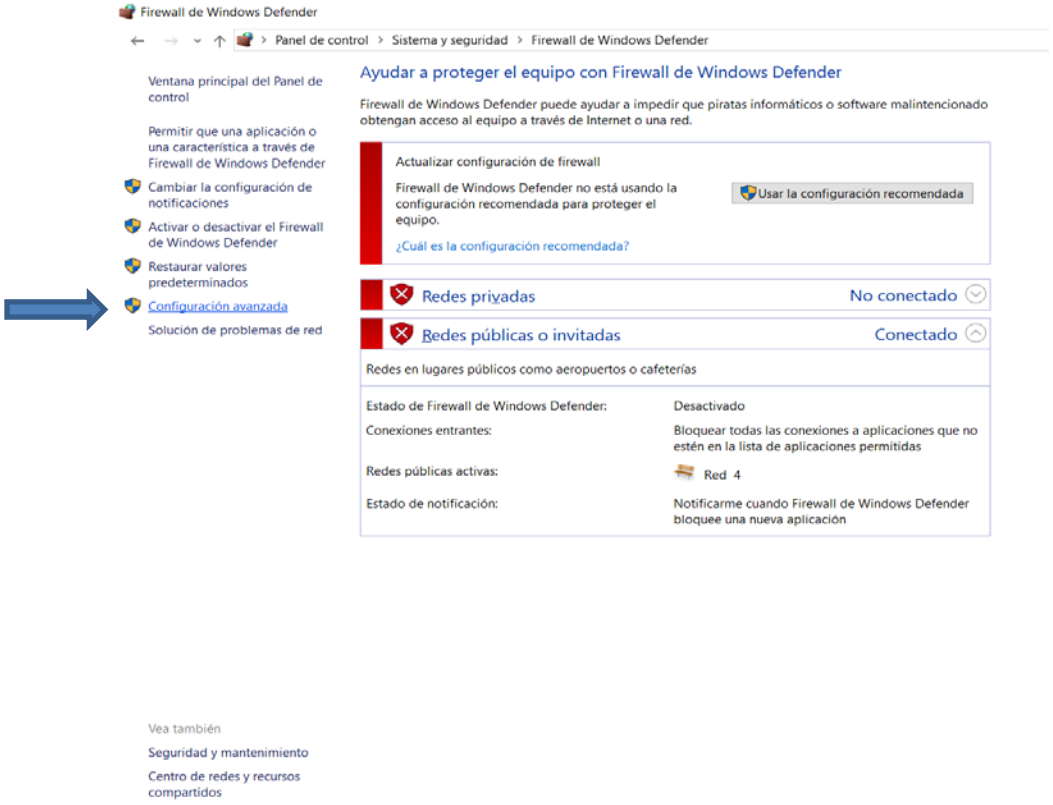
Comenzaré con el de Windows 10 al ser el sistema operativo mas actual y el más complicado de corregir.

Al ser un problema con comunicación entre red se recomienda pausar el firewall del antivirus que se tenga o el firewall de Windows, aunque con esto la seguridad del equipo queda expuesta, por lo cual se recomienda crear una regla de entrada y una regla de salida especificando el puerto utilizado por la aplicación SCI de la siguiente manera:

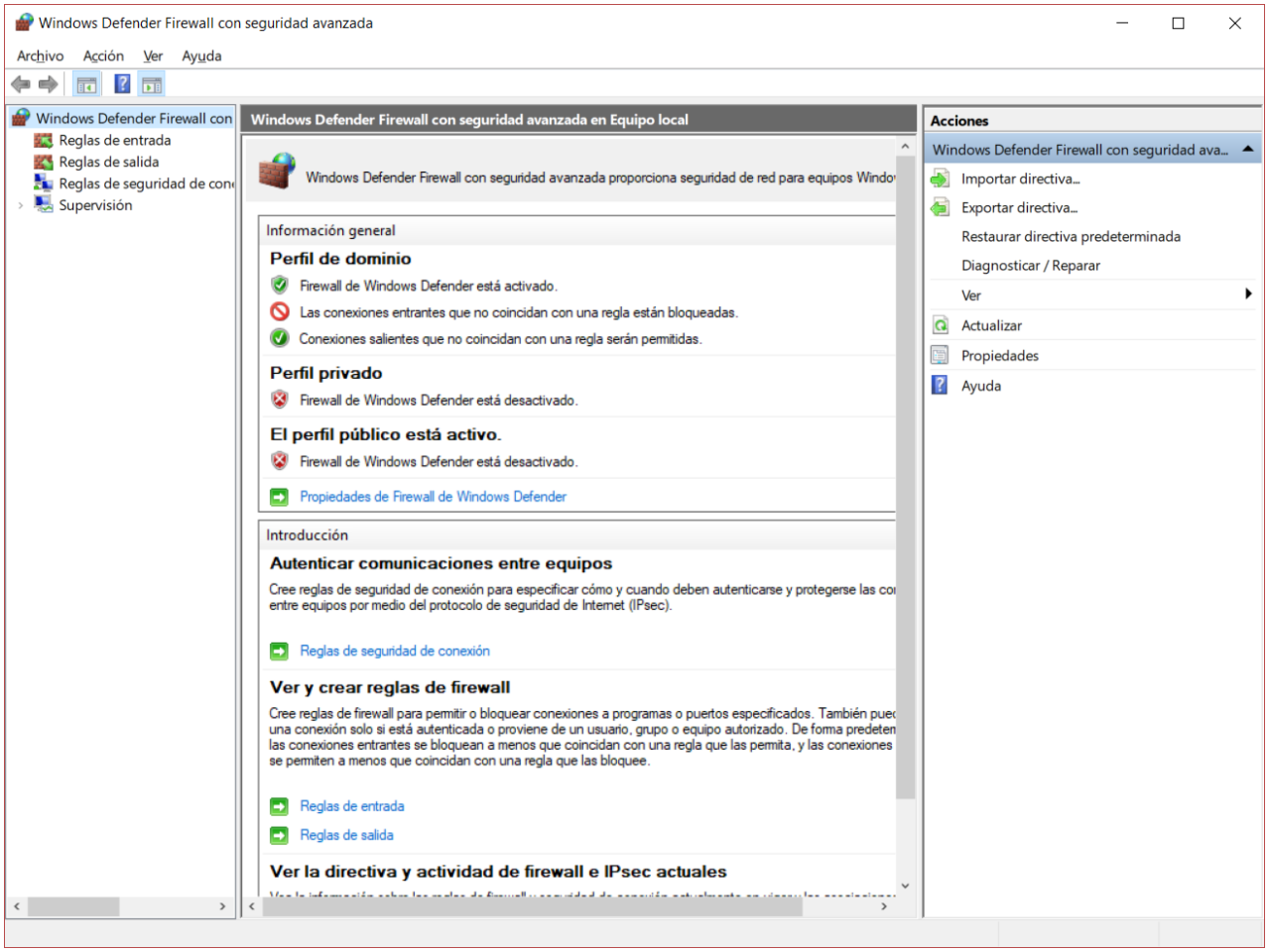
En el panel de control de Windows nos iremos la sección de firewall.



Una vez ahí se nos mostrara la siguiente ventana y daremos click en configuración avanzada.



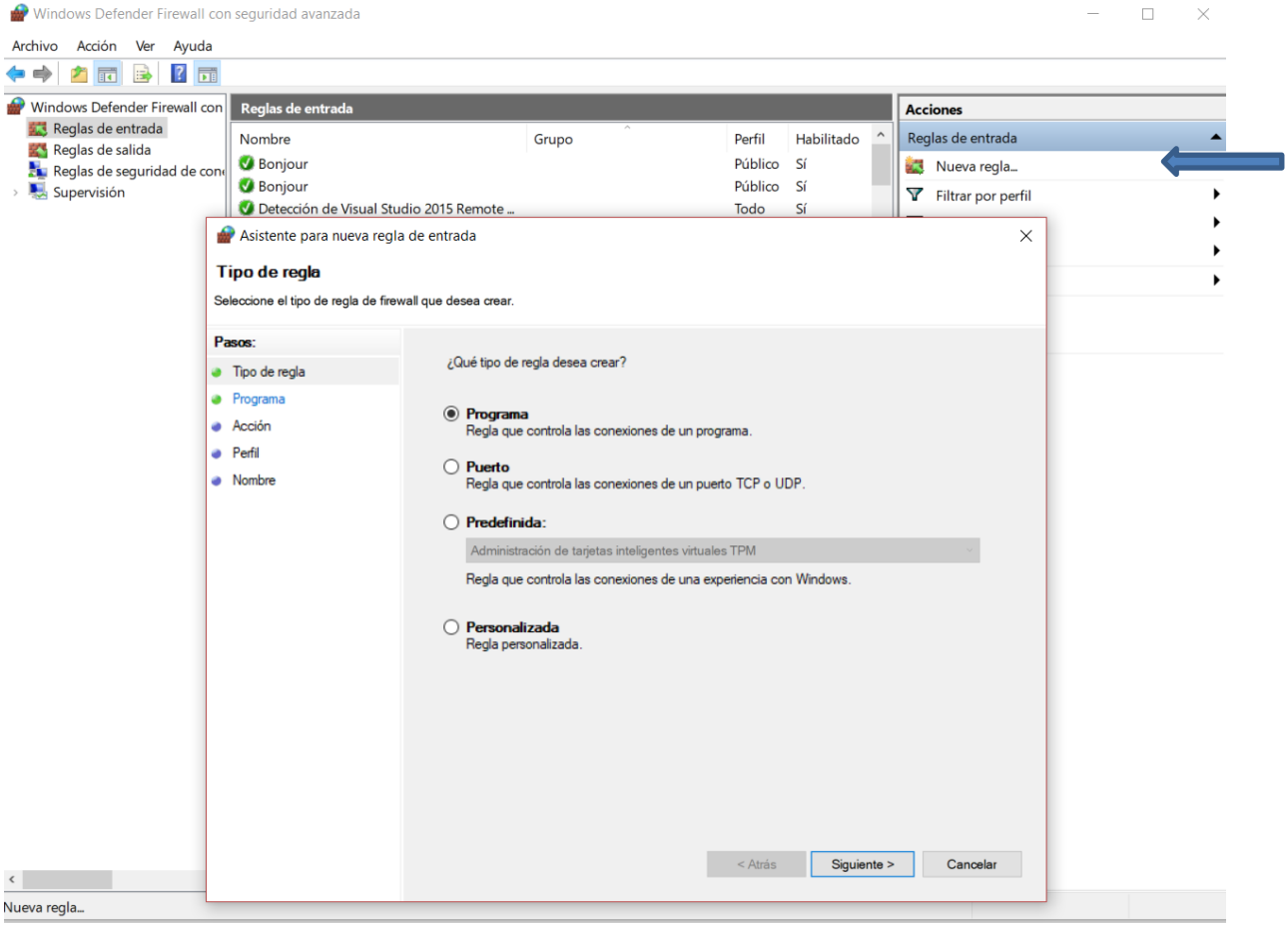
Se nos abrirá la siguiente ventana:



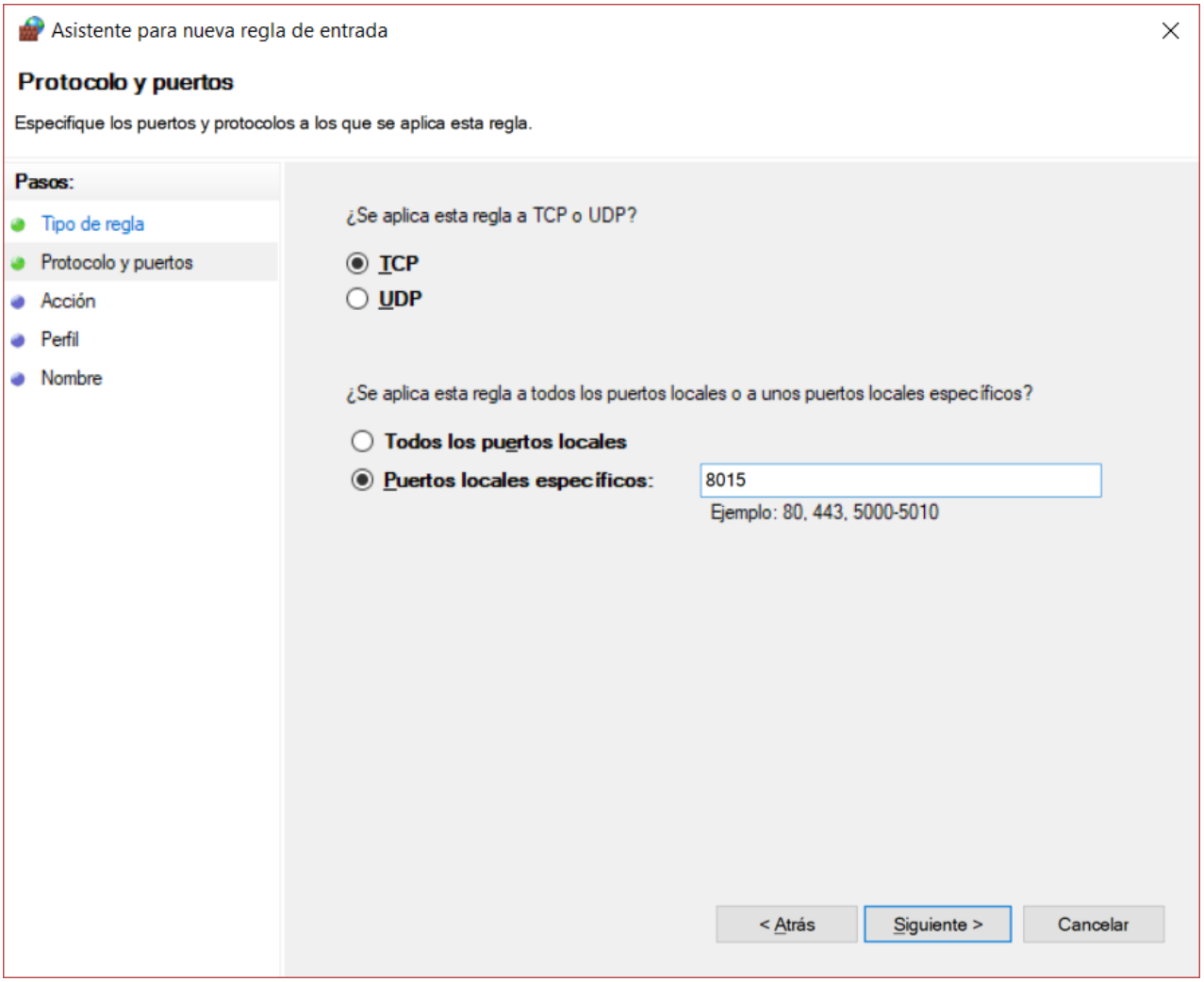
Es importante notar que aquí es donde crearemos las reglas de entrada y de salida utilizando el puerto de la aplicación que es el **8015** para poder permitir a través del firewall la correcta conexión de ese puerto.

Regla de entrada.

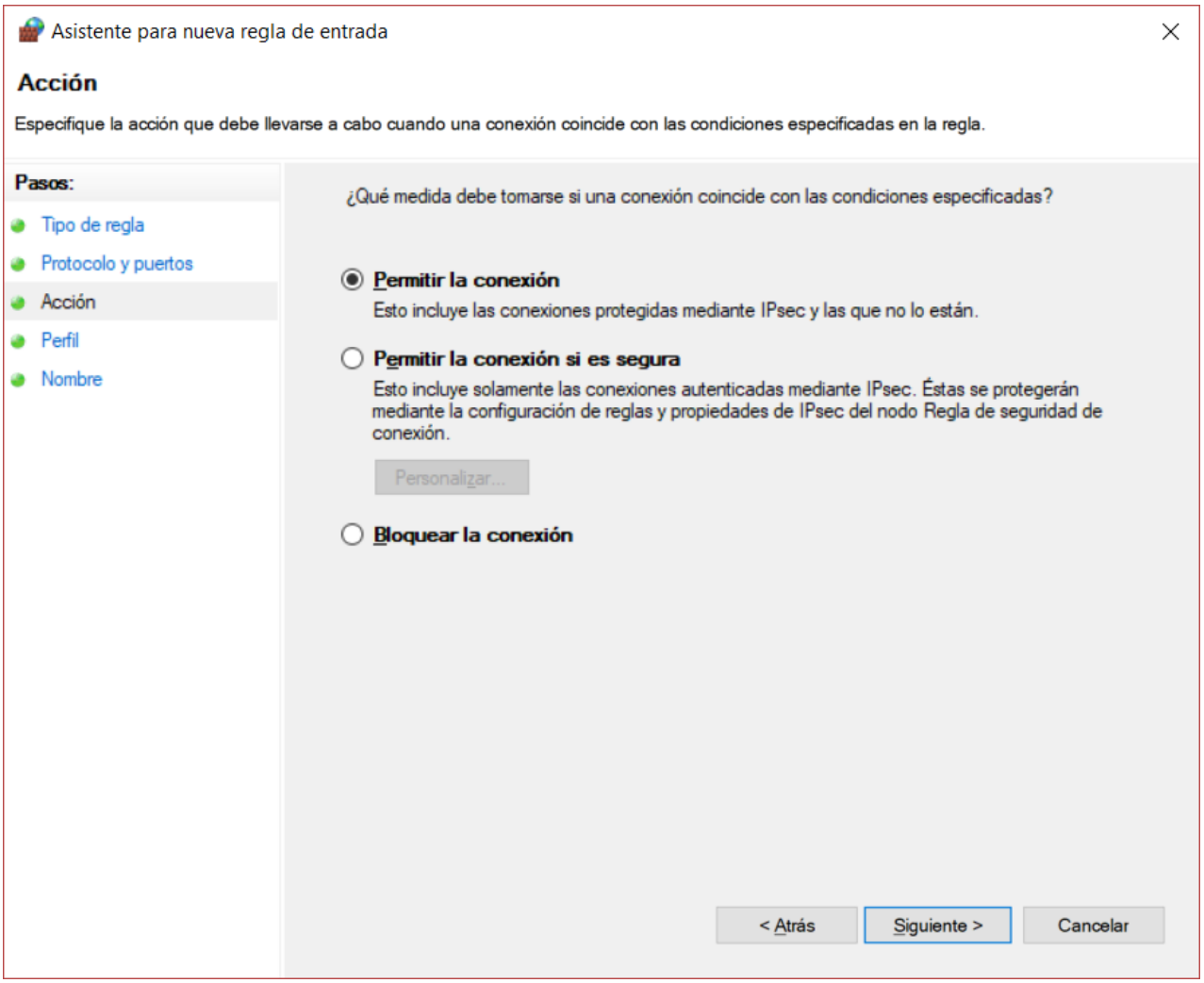
Para crear la regla de entrada deberemos dar click en “Reglas de entrada” y a continuación nos mostrara la pantalla donde comenzaremos a crear la regla.



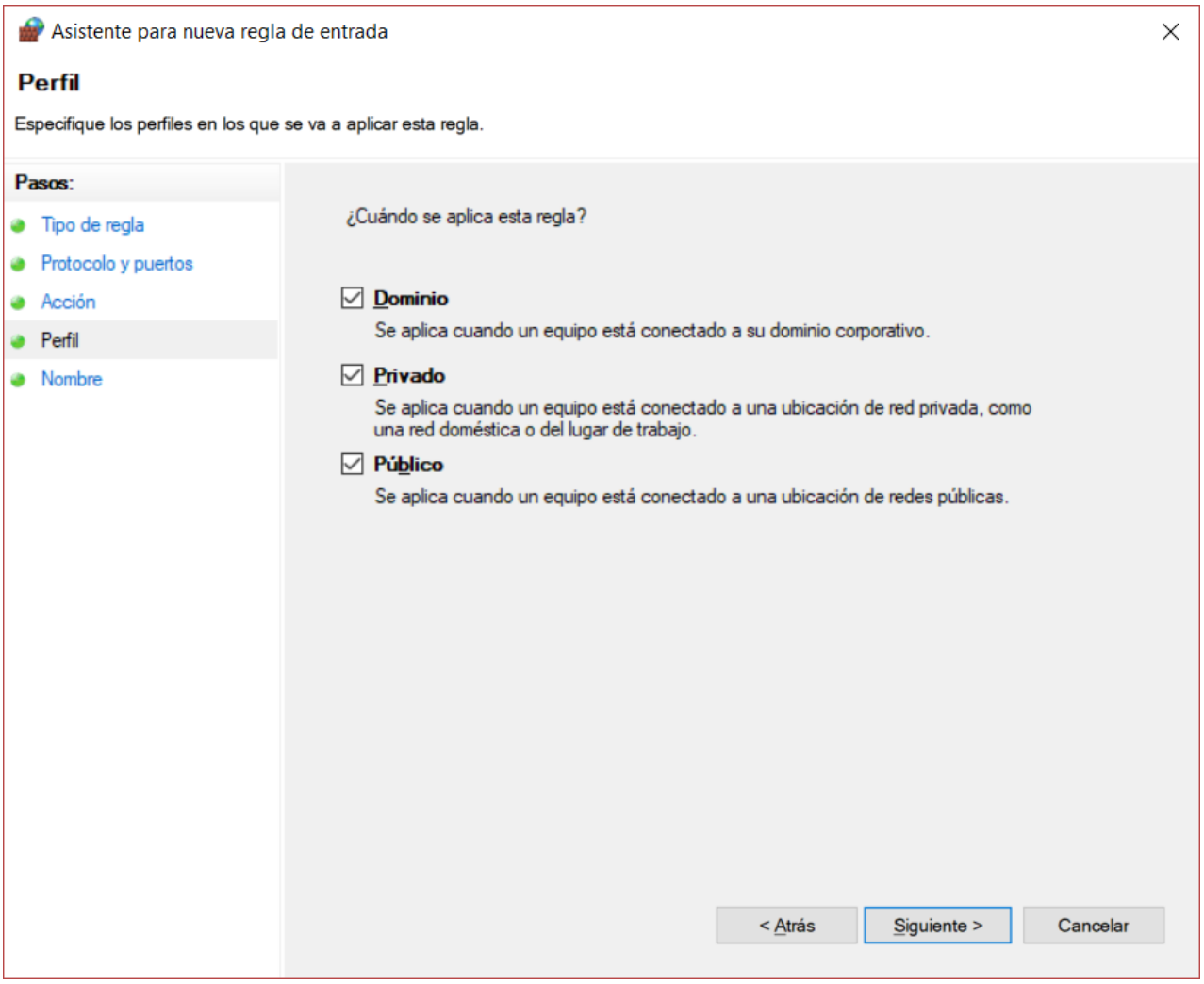
Una vez mostrada la pantalla de creación de regla marcaremos la opción de “Puerto” y daremos en el botón “Siguiente” mostrando una ventana para ingresar información del puerto, en este caso, ingresaremos el puerto 8015 con TCP debido a que con esta tecnología funciona nuestra aplicación como se muestra a continuación.


Asistente para nueva regla de entrada
Protocolo y puertos
Especifique los puertos y protocolos a los que se aplica esta regla.
Pasos:
Tipo de regla
Protocolo y puertos
Acción
Perfil
Nombre
¿Se aplica esta regla a TCP o UDP?
TCP
UDP
¿Se aplica esta regla a todos los puertos locales o a unos puertos locales específicos?
Todos los puertos locales
Puertos locales específicos:
8015
Ejemplo: 80, 443, 5000-5010
< Atrás
Siguiente >
Cancelar

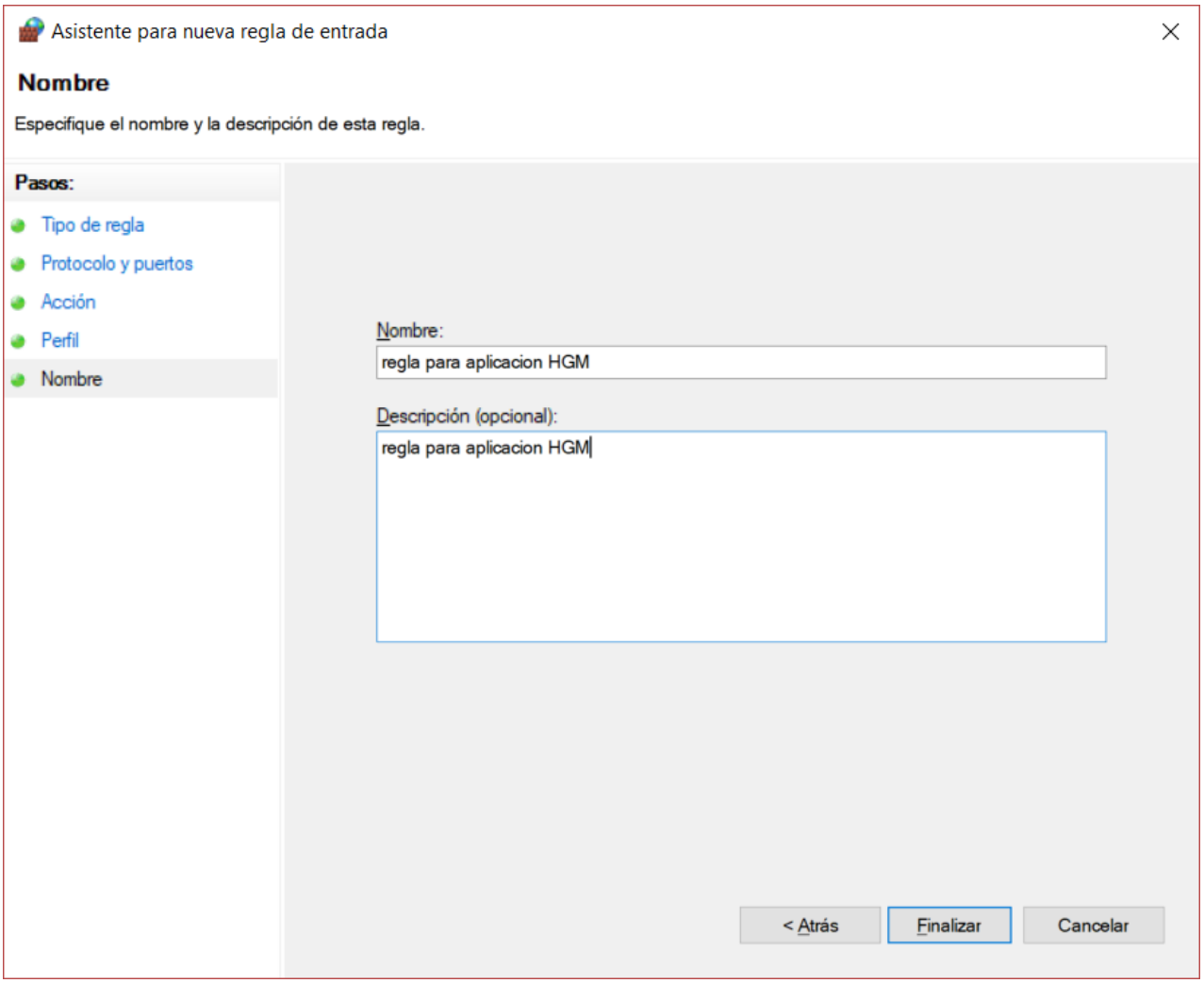
En la siguiente ventana le daremos en “permitir la conexión” para poder crear el enlace entre la aplicación cliente y aplicación servidor.



En la siguiente ventana dejamos por default las marcas en dominio, privado y público.

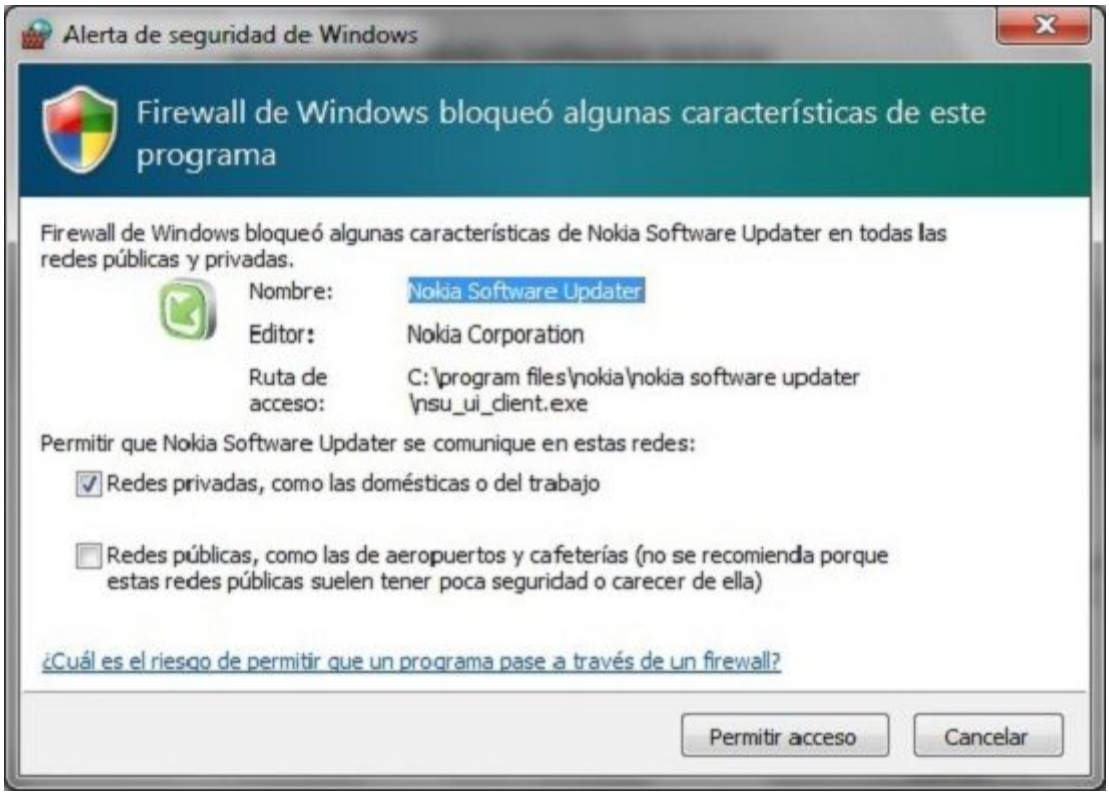


Para finalizar podemos ponerle nombre a la regla para ubicarla dentro de todas las reglas que vienen por defecto en el firewall de Windows.



Y con eso queda creada la regla de entrada, para la regla de salida se repite exactamente el mismo proceso, mostrando las mismas ventanas y especificando el mismo puerto (8015).

Para el caso de Windows 7 es mas sencillo ya que al ejecutar cualquier aplicación, ya sea la del cliente o la del servidor nos mostrará la siguiente pantalla donde marcaremos las dos casillas, la de redes privadas y la de redes públicas, después daremos click en “permitir acceso”.



Con esto, queda solucionado el problema de bloqueo de redes en Windows 7 y en Windows 10.

E. Manual técnico para el usuario.

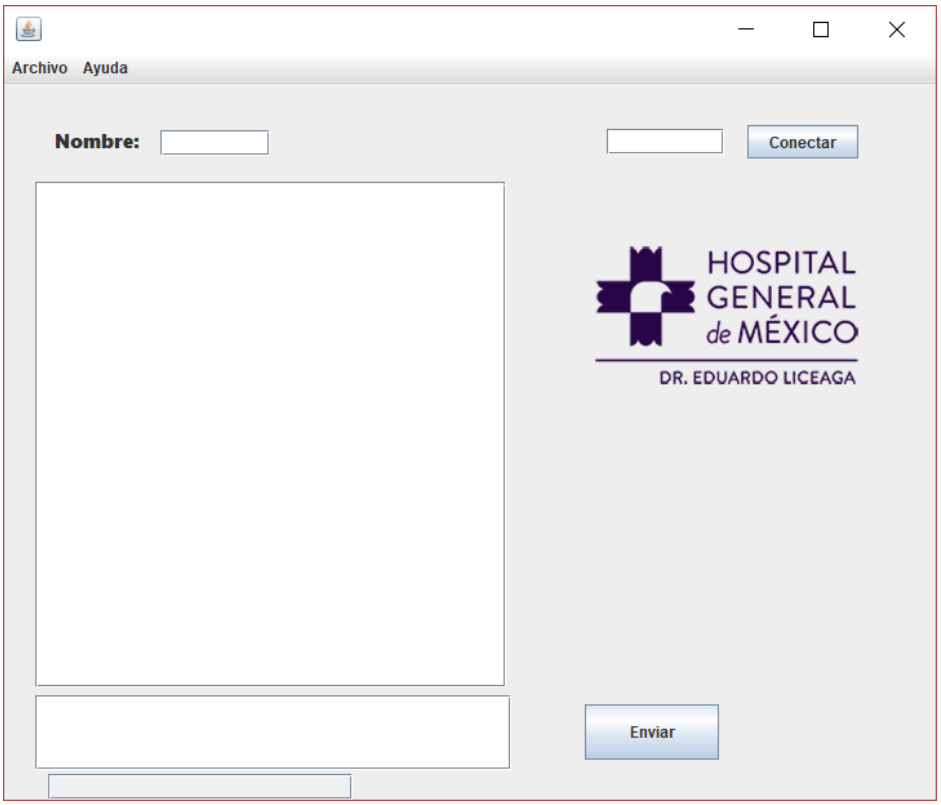
El personal de sistemas le instalará en su escritorio el acceso al sistema SCI (sistema de comunicación interna) y dejándolo correctamente configurado para su uso.

Una vez instalado, se mostrará en su escritorio de la siguiente manera:

En esta aplicación darle doble click para ejecutarlo mostrando la siguiente ventana

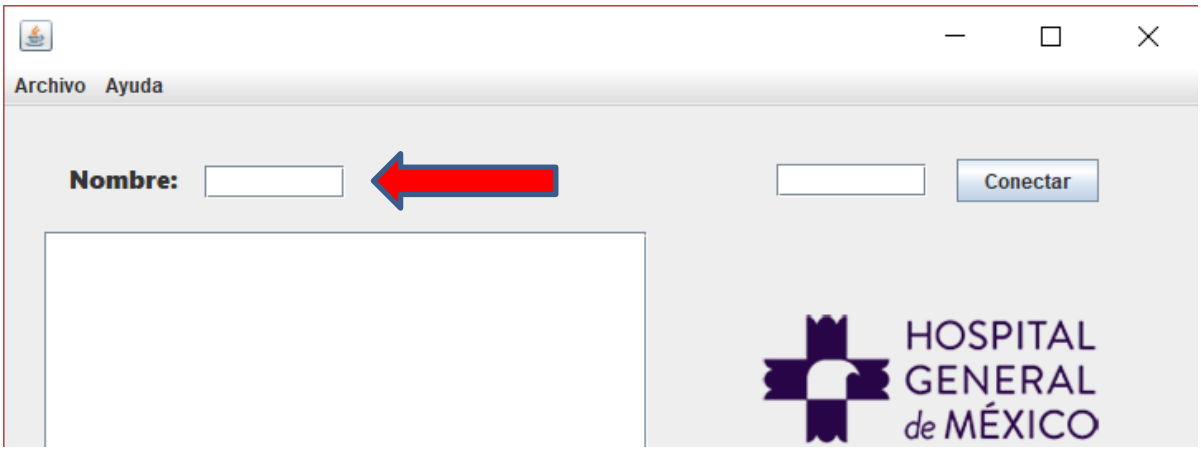
(Pantalla de carga de la aplicación cliente).

La siguiente imagen es la ventana principal de la aplicación del cliente

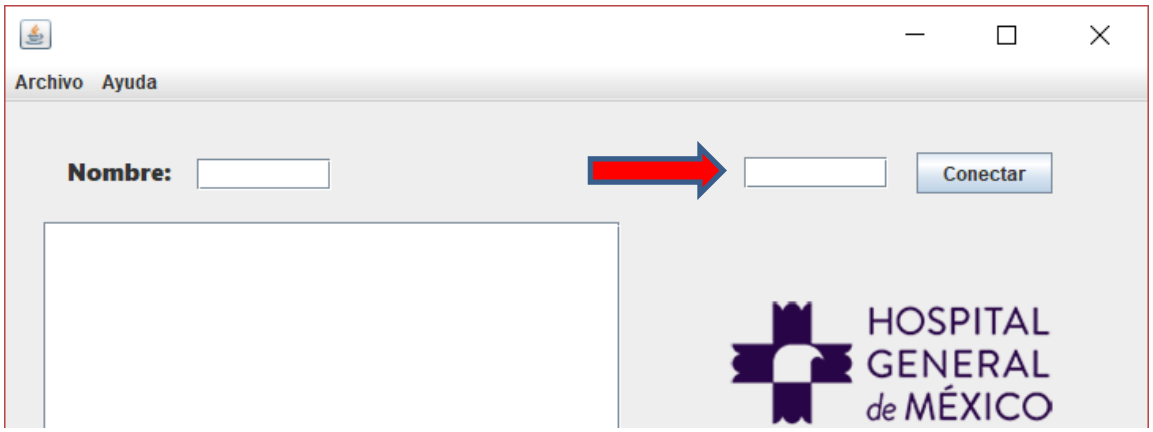


La aplicación cliente cuenta con campos que se mencionan y describen a continuación:

- Campo donde el usuario solicitante o cliente ingresa su nombre.



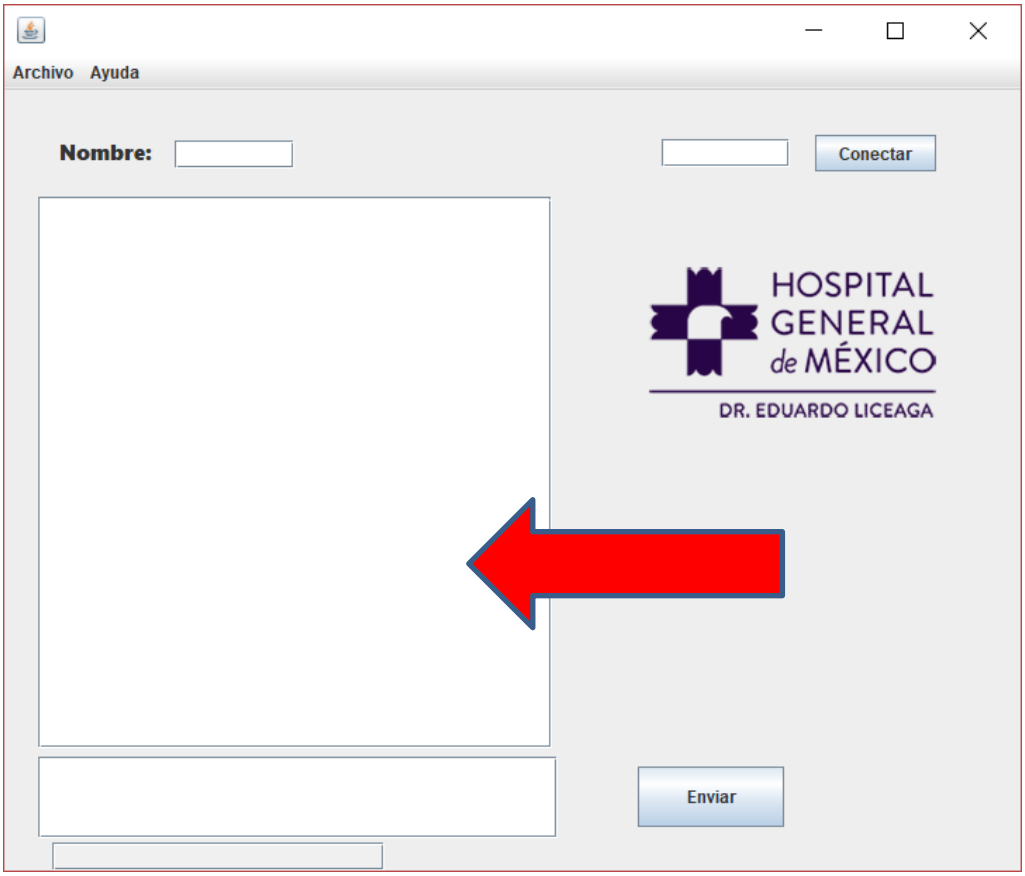
- Campo donde se ingresa la dirección IP del servidor a donde quiere conectarse o con que personal del área de sistemas quiere contactar y su respectivo botón para conectar.



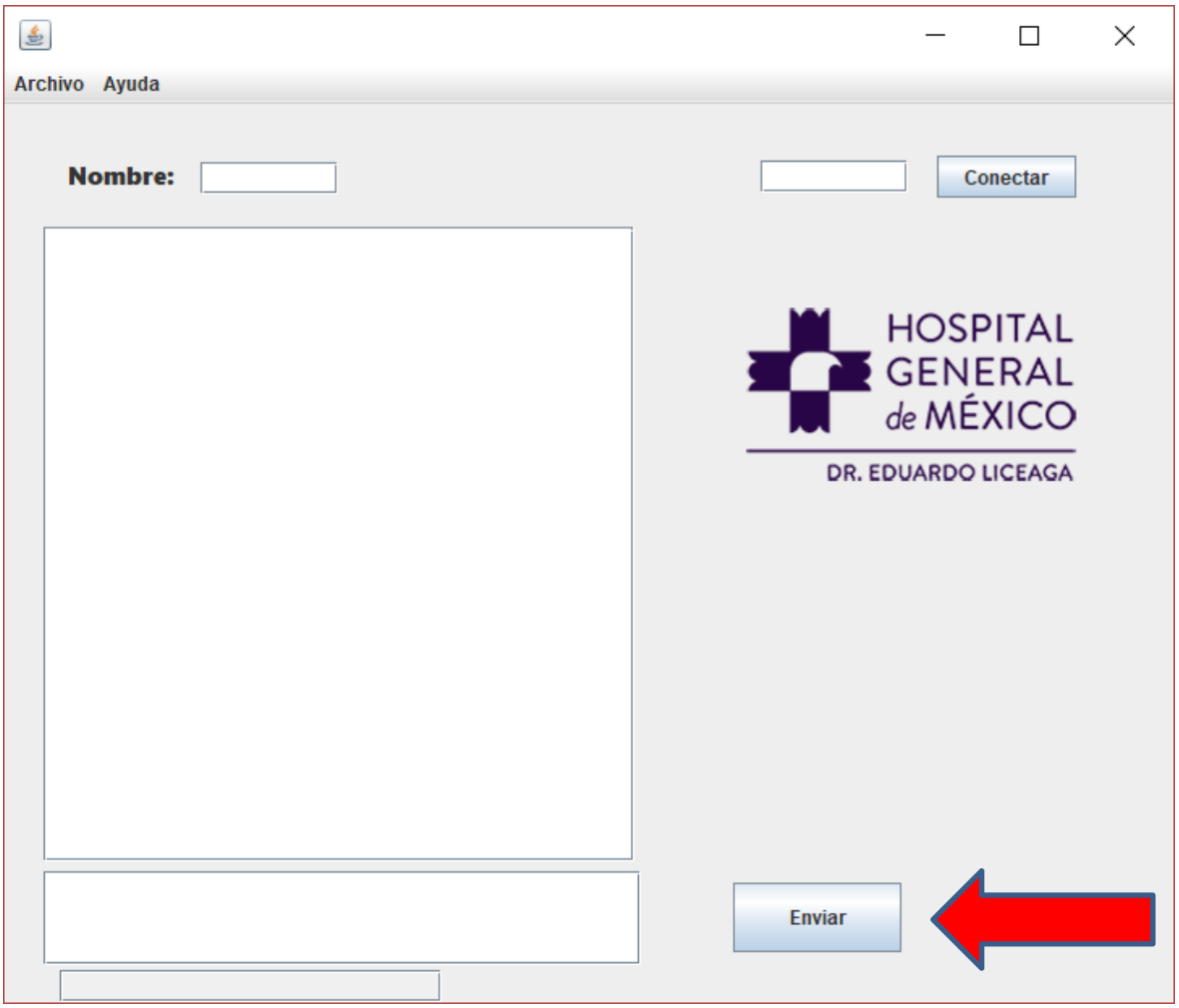
El siguiente listado muestra las direcciones del personal que usted desea contactar:

Nombre del personal	IP de contacto
Sinue González Caballero	192.168.9.240
Gerardo Romero Sánchez	192.168.9.241
Pedro Anzures Niño	192.168.9.242
Guillermo Vázquez Alcaraz	192.168.9.243
Gonzalo Paniagua Ochoa	192.168.9.244
Alejandro Ortega Velázquez	192.168.9.245
José Luis Alejo Vargas	192.168.9.246
Israel Blancas Rodríguez	192.168.9.247
Horacio Valencia Miranda	192.168.9.248

- Campo de texto del chat.



- Campo de texto donde el usuario escribe con su respectivo botón para enviar el mensaje.



Cualquier duda o aclaración sigue existiendo la extensión telefónica 1141 para reportar cualquier incidente relacionado con bienes informáticos.