ESCUELA SUPERIOR POLITECNICA DEL LITORAL

PROYECTO DE GRADUACION

Previo a la obtención del Título de

INGENIERO EN COMPUTACION

T E M A:

Sistema de Compartición de Documentos

S I C O D

REALIZADO POR:

Tomás Hernández

Ricardo Miño

Tulio Peralta

Fernando Villamar

DIRECTOR DEL TOPICO:

Ing. Guido Caicedo

Guayaquil - Ecuador

-:- 1 9 9 5 -:-

ÍNDICE GENERAL

A TODAS AQUELLAS PERSONAS QUE NOS BRINDARON SU AYUDA DESINTERESADA

ESPECIFICACIONES DEL PROYECTO

DESCRIPCIÓN GENERAL DEL PROYECTO

TÍTULO DEL PROYECTO:

"COMPARTICIÓN DE DOCUMENTOS EN INTERNET BAJO EL MODELO CLIENTE/SERVIDOR"

DEFINICIÓN.-

El proyecto de graduación presentado en este documento, proporcionará al usuario común de la internet, un sistema por medio del cual podrá tener acceso a documentos de otros usuarios y dar acceso de documentos de su propiedad a otros usuarios , es decir, que este usuario podrá ***compartir*** con otros usuarios desde memos y mensajes hasta documentos texto y gráficos completos, como por ejemplo la compartición de documentos de word, hojas de excel , objetos de windows, etc.. La lista de documentos a los que un usuario tiene acceso se mantendrán en lo que llamaremos un buzón, que el usuario podrá ver en todo momento.

En cierto modo lo que se ofrece con este sistema es un ***servicio de compartición de documentos*** entre usuarios o entre un usuario y un grupo de usuarios, donde se entiende como grupo de usuarios a una agrupación de usuarios que tienen acceso al servicio. Los usuarios de la internet que deseen tener acceso a este servicio deberán comunicarse con el administrador de su dominio para suscribirse al mismo.

Este sistema, además, le proporcionará al usuario ciertas facilidades que le permitirán obtener información sobre otros usuarios que tienen acceso al servicio, sobre grupos de usuarios existentes (y que usuarios pertenecen a el), y los usuarios que tienen permisos de acceso a un documento de su propiedad, que además le informará si los usuarios han revisado el documento o no, y si lo han hecho, la fecha en que lo hicieron .

OBJETIVOS.-

Interesar al usuario de internet en nuevas aplicaciones del paradigma cliente/servidor usando TCP/IP.

Proveer una aplicación que permita la comunicación de usuarios remotamente a través de los documentos.

Proveer el nuevo servicio de internet de compartir documentos o cualquier tipo de archivo binario entre usuarios internet.

Proveer al usuario internet de una interface amigable de alto nivel .

REQUERIMIENTOS FUNCIONALES

Crear una aplicación Cliente-Servidor que brinde el servicio de poder compartir documentos entre usuarios de la Internet, el mismo que debe cumplir con los siguientes requisitos del usuario final.

- La aplicación Cliente interactuará con un tipo de usuario: el usuario normal. El usuario normal accesará al servicio común y corriente, es decir, podrá únicamente ver los documentos que le hayan dado permisos, crear y actualizar sus propios documentos, dar y borrar permisos a sus documentos, borrar un documento propio, cambiar su clave, etc.

- La aplicación Cliente deberá ser llamada desde un icono de programa en Windows y además debe ser diseñada casi en su totalidad bajo el GUI, y contendrá entre los recursos básicos la barra de título, la barra de menús, la barra de bitmaps, la barra de estados, cajas de dialogo, etc.

- Luego de ejecutar la aplicación Cliente, el programa presentará la pantalla de presentación respectiva con información de la aplicación, luego debe mostrar una caja de dialogo en la cual el usuario debe llenar dos campos vacíos con el nombre o dirección del host al cual desea establecer la comunicación y también el nombre (login) y la clave (password) del usuario en los campos respectivos del dialogo mostrado. Después de ingresar los datos y presionar el botón Aceptar (Ok), a continuación se presentará la correspondiente bienvenida al servicio de parte de la aplicación y se mostrará y llenará la pantalla con los documentos disponibles.

- Por el contrario si el sistema le dice al usuario que no tiene acceso al servicio, se le brindará a este una segunda oportunidad de ingresar al servicio para lo cual deberá volver a introducir sus datos. Si por el contrario el usuario presiona el botón Cancelar (Cancel) de la caja de dialogo el programa cerrará el dialogo y permanecerá en estado de espera, pero sólo con dos comandos activos: salir y servicio.

- La aplicación Cliente dividirá su pantalla en tres ventanas o divisiones principales: la ventana de documentos, la de usuarios y la lista de grupos.

- Cada una de las ventanas usarán el scrolling o desplazamiento entre sus miembros y deben contener o alojar en su interior las listas de sólo lectura de los documentos disponibles, los usuarios que pertenecen al servicio y la lista de los grupos de usuarios existentes.

- En estas porciones de pantalla se podrá hacer uso del dispositivo apuntador que se este usando ya sea Mouse o trackball, para poder desplazar las listas. Haciendo doble clip con el dispositivo Mouse sobre un documento por ejemplo , el usuario recibirá una copia del archivo correspondiente en un

subdirectorio temporal en donde resida el programa Cliente; luego de efectuado esto el usuario podrá llamar a la aplicación correspondiente y editarlo o verlo según sea necesario.

- Si el documento en cambio es solamente seleccionado con un clip del Mouse, el usuario podrá borrarlo, verlo o modificar sus permisos o cualquier otro comando que este disponible o su ejecución se realice sobre el documento.

- En las listas de usuarios y grupos el usuario sólo podrá ver o chequear una fila seleccionada de la lista.

- Para poder realizar la modificación o actualización de los datos asociados con los documentos, usuarios y grupos, el programa debe mostrar las correspondientes cajas de dialogo con los respectivos campos de datos para ser modificados o llenados.

- Los documentos leídos tendrán una marca especial, que los identificaran de los que no han sido leidos en su correspondiente caja de lista.
- Cada documento leído tendrá un timer o lapso de tiempo de 5 días de permanencia en el "Buzón", transcurrido ó expirado el mismo, este documento debe ser borrado de la lista de documentos disponibles, esto se realiza siempre y cuando el documento referido haya sido leído por el usuario respectivo.

- El usuario que desea compartir un determinado documento con los usuarios del servicio, debe primero crear un archivo de documento dentro de su correspondiente aplicación Windows, grabarlo en el subdirectorio temporal del programa Cliente y luego ejecutar el comando correspondiente para ejecutar esta acción y llenar los datos relacionados con el documento.

- Todas las cajas de dialogo deben tener tres botones standard en cuanto sea posible. Un botón Aceptar, para ejecutar el comando o la acción respectiva. Un botón Cancelar, para omitir la realización del comando y cerrar al mismo tiempo el dialogo. Un botón Ayuda, para ofrecer al usuario una pequeña descripción de la función que trata de realizar el comando.

- En lo posible se requiere que la aplicación ofrezca ayuda en todo instante o estado del sistema, activando esto con la tecla de función F1. Además la ayuda debe ser implementada para todo tipo de comando, menú, acción o tarea que este a punto de ser ejecutada o llamada por iteración del usuario.

- Finalmente, para salir del servicio el usuario puede hacer dos cosas: cerrar la aplicación ó cerrar el servicio.
- Si el usuario elige el comando Salir, el programa primero terminará el servicio y luego cerrará la aplicación, si por el contrario el usuario llama al comando para cerrar o suspender el servicio,

entonces el programa simplemente terminará la conexión del servicio y la aplicación permanecerá en un estado de espera o inicio con sólo dos comandos activos: el comando Salir y el comando Servicio.

REQUERIMIENTOS DE RENDIMIENTO Y CONFIABILIDAD

El sistema a desarrollarse tiene que cumplir con ciertas características básicas y técnicas que brinden al usuario final confiabilidad de alto rendimiento, por lo tanto, debe por lo menos utilizar a nivel de transporte el protocolo TCP ya que éste ofrece transporte de paquetes y entrega confiable. .

Ya que TCP es un protocolo de entrega confiable el grupo de desarrollo se preocupará por el rendimiento del sistema a nivel de la comunicación de las aplicaciones.

Para lograr de esta manera rendimiento y confianza al usuario, la aplicación debe cumplir varias características.

- La comunicación, el transporte de paquetes y los mensajes entre las aplicaciones deben operar de forma transparente al usuario.

- Cada uno de los requerimientos que envía el cliente deben tener una respuesta de parte del programa servidor, es decir, después que el programa cliente envía un mensaje, este tiene que esperar por un mensaje de respuesta del programa servidor.

SOLUCIÓN PROPUESTA.-

Para solucionar y diseñar la aplicación el grupo de desarrollo se ha basado en las siguientes alternativas de solución:

En lo referente a nivel de la capa de transporte se usará el protocolo TCP.

A nivel de las conexiones utilizaremos la conocida abstracción del socket.

Para la implementación de los programas cliente/servidor se utilizará el Microsoft C para Unix y Visual BASIC para Windows; así como también se usará el development para desarrollo en TCP/IP llamado Distinc.

Se diseñará un servidor concurrente orientado a conexión.

La implementación del Cliente será orientado a conexión.

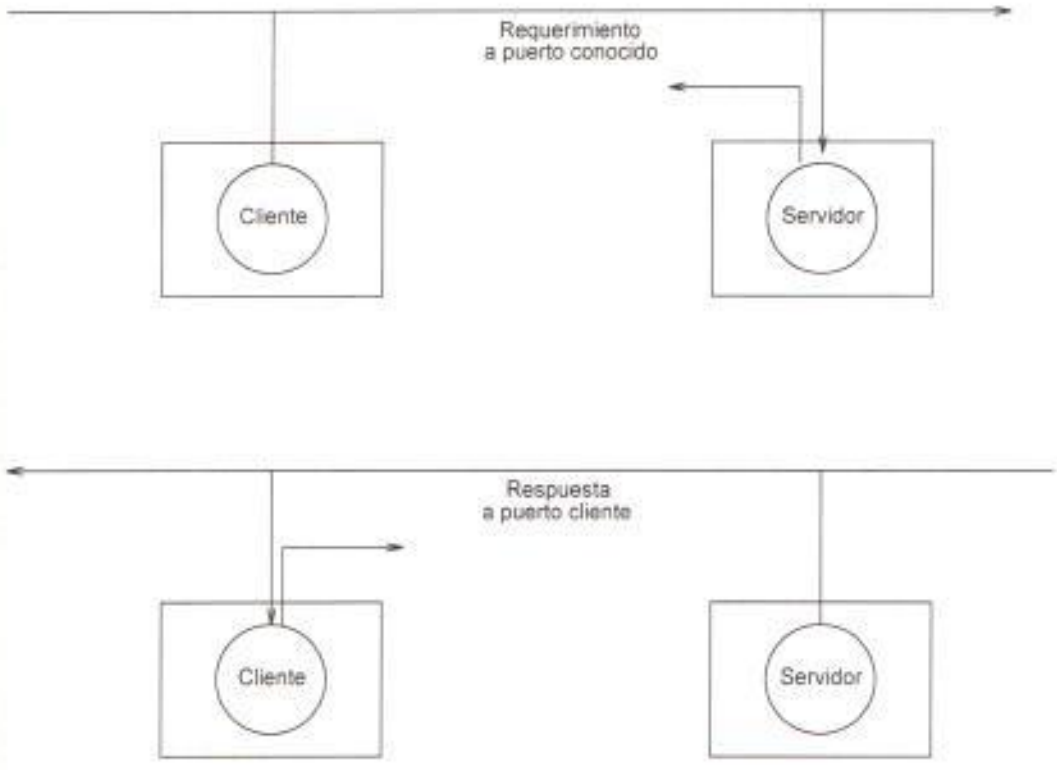
DISEÑO DEL PROTOCOLO DE APLICACIÓN

ARQUITECTURA CLIENTE/SERVIDOR

Para poder implementar el servicio hemos diseñado un protocolo de aplicación, el cual utiliza streams con campos de datos que serán manejadas junto con la abstracción de los sockets.

El esquema de presentación del protocolo se hará desde el punto de vista del programa cliente, es decir, se detallará en primer lugar el requerimiento del Cliente y luego la correspondiente respuesta del servidor.

El protocolo asume que la conexión entre Cliente/Servidor ya está establecida antes de autenticar el servicio y comenzar con el manejo de los requerimientos.



MÁQUINA DE ESTADOS DEL CLIENTE

ESTADOS DEL CLIENTE.

Servicio inactivo: En este estado la aplicación se encuentra en espera de una acción del usuario para generar un requerimiento al servidor.

Servicio Activo: En este estado la aplicación cliente procesa la información que va a enviar al servidor o que ha recibido del mismo, mostrando en todo momento el estado de la aplicación. al usuario.

Enviar requerimiento: Es aquí donde la aplicación cliente formatea los datos para ser empaquetados y transmitidos al servidor .

Recibir respuesta: En este estado la aplicación cliente acepta los datos que le han sido enviados desde el servidor. La aplicación se encarga en este estado de tomar los datos que le han llegado desde el servidor para convertirlos al formato de la aplicación y luego enviarlos al modulo de **Servicio activo.**

Timer: Es un simple repetitivo de espera. En este estado la aplicación esta en un ciclo de espera por algún evento(requerimiento o respuesta).

fin: este estado se concluye con la aplicación..

MAQUINA DE ESTADOS DEL SERVIDOR

ESTADOS DEL SERVIDOR

Socket: En este estado la aplicación servidora crea un socket para desde el cual leerá los datos que le llegan desde el servidor.

Bind: En este estado la aplicación servidor a enlaza el socket a un puerto bien conocido desde el cual se tomarán los datos.

Listen: En este estado el socket es colocado en modo pasivo, es decir quedara escuchando para ver si recibe requerimientos desde la aplicación cliente.

Acept: Aquí la aplicación servidora se mantiene en un lazo esperando recibir un requerimiento de conexión desde algún cliente.

Fork: En este estado la aplicación servidora crea un proceso hijo paralelo al proceso padre, para manejar una conexión, esto es atender todos los requerimientos provenientes desde una aplicación cliente.

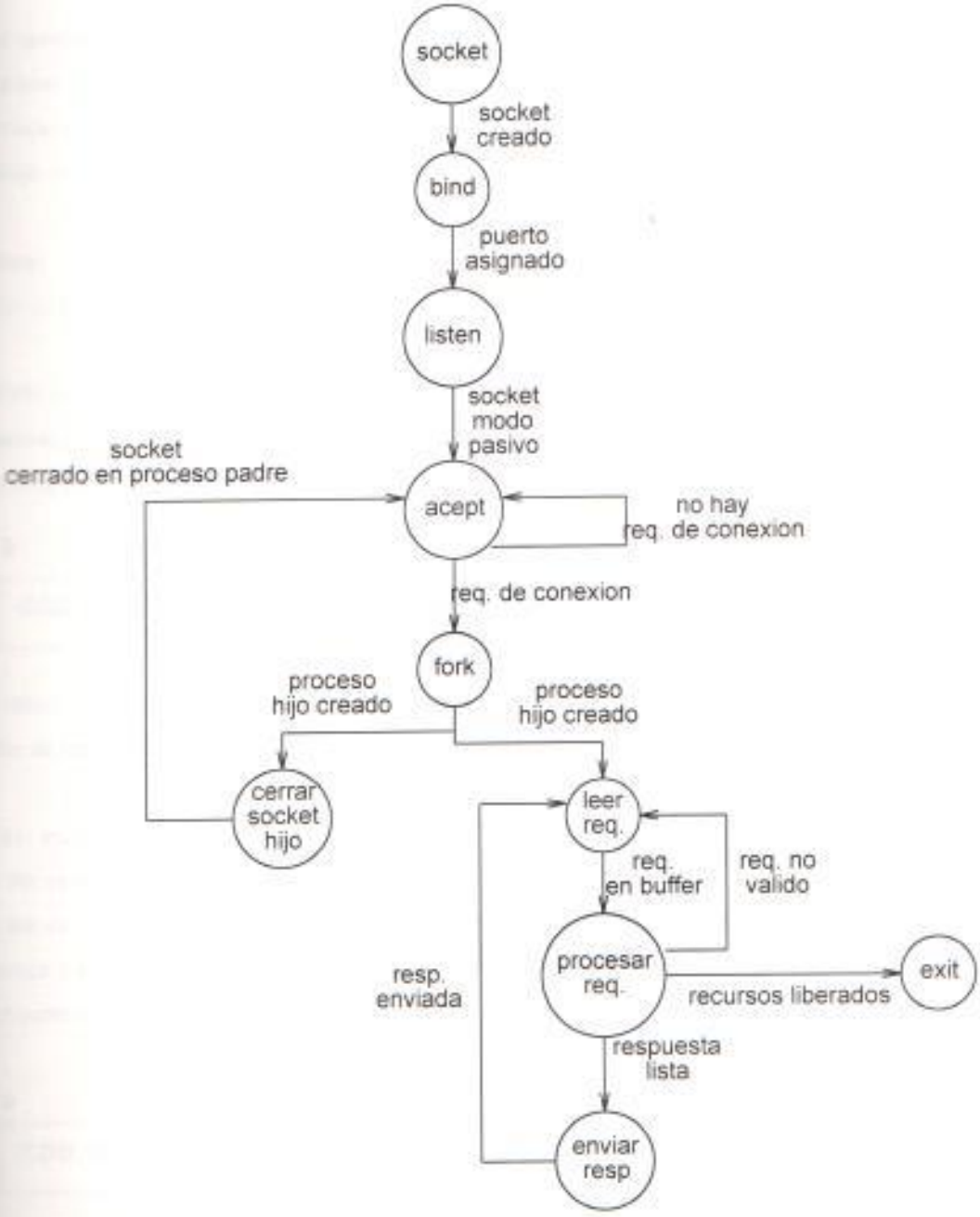
Leer requerimientos: En este estado la aplicación servidora lee desde el socket los requerimientos provenientes desde la aplicación cliente y los coloca en un formato apropiado para su procesamiento.

Procesar Requerimiento: Estado de procesamiento de requerimientos y elaboración de respuestas.

Envío de respuestas: estado de empaquetamiento y envío de respuestas, entendiéndose por empaquetamiento a la colocación de los datos en el formato apropiado para la transmisión através de la red.

Cerrar socket: Estado de transición en que la aplicación servidora cierra el nuevo socket creado en su proceso padre y vuelve al estado **acept** para seguir aceptando nuevos requerimientos de conexión.

Exit: En este estado la aplicación servidora en su proceso hijo que esta manejando una conexión cierra el socket y termina .



SINTAXIS Y SEMÁNTICA DEL PROTOCOLO.

REQUERIMIENTO : Acceso al servicio .

Descripción: Para poder iniciar la sesión con la aplicación servidora el cliente enviará un requerimiento de acceso al servicio. En éste requerimiento el cliente incluirá un login y una clave. El servidor responderá afirmativamente al requerimiento si el login y la clave son válidos de otro modo enviará un mensaje de error indicando cual fue el error.

Cliente:

buffer de requerimiento enviado desde la aplicación cliente hacia el servidor.

cod 01: se refiere al código que identifica el requerimiento para la aplicación servidora.

nombre: se refiere al login del cliente .

clave: se refiere al password del cliente.

Servidor:

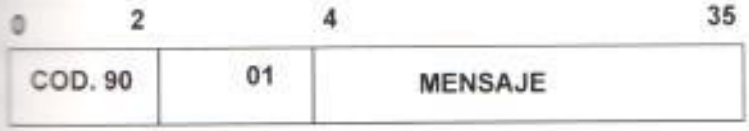
buffer de respuesta enviado desde la aplicación servidora a la aplicación cliente.

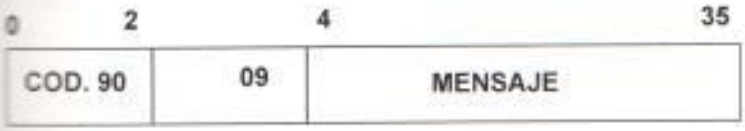
cod31: indica al cliente que la operación fue satisfactoria .

cod 90: es un código que indica al cliente que un error ha ocurrido.

01y 09: es un código de error que le indica al cliente el tipo de error que ha ocurrido.

mensaje : es un mensaje de error adicional por si hay fallas de sistema sobre las que la aplicación no tiene control.





REQUERIMIENTO: Lista de documentos.

Descripción: Por medio de este requerimiento el cliente obtendrá una lista de todos los documentos de su propiedad, es decir los documentos que el ha colocado en su buzón y todos los documentos que no son de su propiedad, a los cuales tiene acceso, indicando la fecha en que el documento fue colocado en el buzón del cliente y si ha sido revisado o no por él.

Cliente:
buffer de requerimiento enviado desde la aplicación cliente hacia el servidor

cod02: este es el código que le indica al servidor de que requerimiento se trata.

0	2
COD. 02	

Servidor:
buffer de respuesta enviado desde la aplicación servidora a la aplicación cliente.

COD32: Este código le indicará a la aplicación cliente que se le esta enviando el nombre de un documento al que el tiene acceso con n toda la información pertinente .
COD33: Este código le indica a la aplicación cliente que todos los documentos a los que el tiene acceso le han sido enviados.
COD34: este código le indica a la aplicación cliente que se le están enviando el nombre de un documento de su propiedad con toda la información pertinente.
COD 90: es un código que indica al cliente que un error ha ocurrido.
#E: es un código de error que le indica al cliente el tipo de error que ha ocurrido.
Nombre: Este campo del buffer de envío indica el login del cliente propietario de un documento.
Documento: Este campo del buffer indica el nombre del documento.
Leído: Este campo es una bandera que le indica a la aplicación cliente si el documento ha sido leido por él o no.
F.publ. : Este campo le informa a la aplicación cliente de la fecha en que el documento fue colocado en su buzón.
F.lectura: Este campo le informa al cliente la fecha en que el usuario reviso un documento, si es que lo hizo.
mensaje : es un mensaje de error adicional por si hay fallas de sistema sobre las que la aplicación no tiene control.

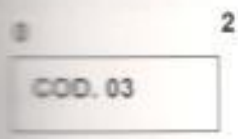
REQUERIMIENTO: obtener lista de usuarios.

Descripción: Por medio de este requerimiento la aplicación cliente informa al servidor que necesita obtener una lista de todos los usuarios que tiene acceso al servicio.

Cliente:

buffer de requerimiento enviado desde la aplicación cliente a la servidora.

CODIGO : este es el código que le indica al servidor de que requerimiento se trata.



Servidor:

buffer de respuesta enviado desde la aplicación servidora a la aplicación cliente.

COD35: Este código le indicará a la aplicación cliente que se le esta enviando el nombre de un usuario.

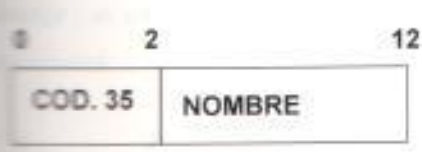
COD33: Este código le indica a la aplicación cliente que todos los nombres de usuarios han sido enviados.

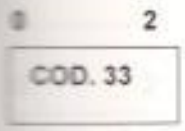
COD 90: es un código que indica al cliente que un error ha ocurrido.

01: es un código de error que le indica al cliente el tipo de error que ha ocurrido.

Nombre: Este campo del buffer de envío indica el login de un cliente.

mensaje : es un mensaje de error adicional por si hay fallas de sistema sobre las que la aplicación no tiene control.





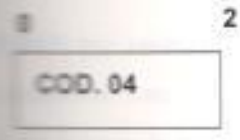
REQUERIMIENTO: Obtener lista de grupos

Descripción: Por medio de este requerimiento la aplicación cliente informa al servidor que requiere obtener una lista de todos los grupos de usuarios que existen para este servicio.

Cliente:

buffer de requerimiento enviado desde la aplicación cliente a la servidora.

COD04 : este es el código que le indica al servidor de que requerimiento se trata.



Servidor:

buffer de respuesta enviado desde la aplicación servidora a la aplicación cliente.

COD36: Este código le indicará a la aplicación cliente que se le esta enviando el nombre de un grupo.

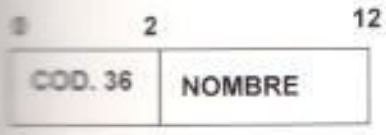
COD33: Este código le indica a la aplicación cliente que todos los nombres de grupos han sido enviados.

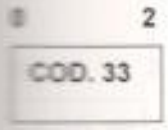
COD 90: es un código que indica al cliente que un error ha ocurrido.

01: es un código de error que le indica al cliente el tipo de error que ha ocurrido.

Nombre: Este campo del buffer de envío indica el login de un cliente.

mensaje : es un mensaje de error adicional por si hay fallas de sistema sobre las que la aplicación no tiene control.





REQUERIMIENTO :Obtener documento.

Descripción: Por medio de este requerimiento la aplicación cliente le indica al servidor que desea que le envíe un documento (archivo texto ó gráfico). La aplicación cliente deberá especificar el nombre del documento que desea se le envíe.

Cliente:

buffer de requerimiento enviado desde la aplicación cliente a la servidora.

COD05: Por medio de este código la aplicación servidora determina de que requerimiento se trata.

Documento: Este campo del buffer le indica al servidor el nombre del documento que la aplicación cliente desea se le envíe.

Buffer de respuesta enviado desde la aplicación servidora a la aplicación cliente.

: Este código le indicará a la aplicación cliente que se le esta enviando la ruta donde se encuentra localizado el documento deseado.

: es un código que indica al cliente que un error ha ocurrido.

: es un código de error que le indica al cliente el tipo de error que ha ocurrido.

: Este campo del buffer de envío indica el path completo de la localización del documento. .

: es un mensaje de error adicional por si hay fallas de sistema sobre las que la aplicación no tiene control.

2	83
COD. 37	RUTA

0	2	4	34
COD. 90	01|02|03|12|13	MENSAJE	

REQUERIMIENTO: Archivar documento.

Descripción: Por medio de este requerimiento la aplicación cliente le indica al servidor que desea que le envíe un documento (archivo texto ó gráfico). La aplicación cliente deberá especificar el nombre del documento que desea se le envíe.

Buffer de requerimiento enviado desde la aplicación cliente a la servidora.

: Por medio de este código la aplicación servidora determina de que requerimiento se trata.

Documento: Este campo del buffer le indica al servidor el nombre del documento que la aplicación cliente desea se le envíe.

0	2	14
COD. 06	DOCUMENTO	

Servidor:

buffer de respuesta enviado desde la aplicación servidora a la aplicación cliente.

CODIGO: Este código le indicará a la aplicación cliente que se le esta enviando la ruta en la que el documento debe ser grabado.

COD 90: es un código que indica al cliente que un error ha ocurrido.

01: es un código de error que le indica al cliente el tipo de error que ha ocurrido.

Ruta: Este campo del buffer de envío indica el path completo del directorio hogar asignado al usuario.

mensaje : es un mensaje de error adicional por si hay fallas de sistema sobre las que la aplicación no tiene control.

REQUERIMIENTO: Añadir permisos a documento

Descripción: Por medio de este requerimiento la aplicación cliente le informa al servidor que desea dar permisos a usuario s o grupos del servicio de un documento de su propiedad que el especifica.

Cliente:

buffer de requerimiento enviado desde la aplicación cliente a la servidora.

CODIGO: Por medio de este código la aplicación servidora determina de que requerimiento se trata.

Documento: Este campo del buffer que la aplicación cliente envía al servidor, indica el nombre del documento del que desea dar permisos

Nombre: este campo del buffer que se envía indica el nombre del usuario ó grupo al cual se le esta dando permiso de acceso para el documento.

Servidor:

buffer de respuesta enviado desde la aplicación servidora a la aplicación cliente.

CODIGO: Este código le indicará a la aplicación cliente que el procesamiento del requerimiento fue satisfactorio.

CODIGO: es un código que indica al cliente que un error ha ocurrido.

MENSAJE: es un código de error que le indica al cliente el tipo de error que ha ocurrido.

mensaje : es un mensaje de error adicional por si hay fallas de sistema sobre las que la aplicación no tiene control.

0	2
COD. 30	

0	2	4	34
COD. 90	01|05|06| 07	MENSAJE	

REQUERIMIENTO: Eliminar permisos de documento

Descripción: Por medio de este requerimiento la aplicación cliente le informa al servidor que desea quitar permisos a usuario s o grupos del servicio de un documento de su propiedad que el especifica.

Cliente:

buffer de requerimiento enviado desde la aplicación cliente a la servidora.

CODIGO: Por medio de este código la aplicación servidora determina de que requerimiento se trata.

Documento: Este campo del buffer que la aplicación cliente envía al servidor indica el nombre del documento del que desea quitar permisos

Nombre: este campo del buffer que se envía indica el nombre del usuario ó grupo al cual se le esta quitando permiso de acceso para el documento.

0	2	14
COD. 08	DOCUMENTO	NOMBRE

Servidor:

Buffer de respuesta enviado desde la aplicación servidora a la aplicación cliente.

CODIGO: Este código le indicará a la aplicación cliente que el procesamiento del requerimiento fue satisfactorio.

CODIGO: es un código que indica al cliente que un error ha ocurrido.

01,06,08,10,13: es un código de error que le indica al cliente el tipo de error que ha ocurrido.

MENSAJE : es un mensaje de error adicional por si hay fallas de sistema sobre las que la aplicación no tiene control.

0	2
COD. 30	

0	2	4	34
COD. 90	01|06|08| 10|13	MENSAJE	

REQUERIMIENTO: Obtener permisos de documento

Descripción: Por medio de este requerimiento la aplicación cliente le informa al servidor que desea obtener una lista de todos los usuarios y grupos de usuarios que tengan permiso de acceso al documento que se especifica.

Cliente:

Buffer de requerimiento enviado desde la aplicación cliente a la servidora.

CODIGO: Por medio de este código la aplicación servidora determina de que requerimiento se trata.

DOCUMENTO: Este campo del buffer que la aplicación cliente envia al servidor indica el nombre del documento del cual desea obtener los permisos.

0	2	14
COD. 09	DOCUMENTO	

Servidor:

buffer de respuesta enviado desde la aplicación servidora a la aplicación cliente.

COD35: Este código le indicará a la aplicación cliente que se esta enviando los nombres de usuarios.

COD36: Este código le indicará a la aplicación cliente que se están enviando nombres de grupos.

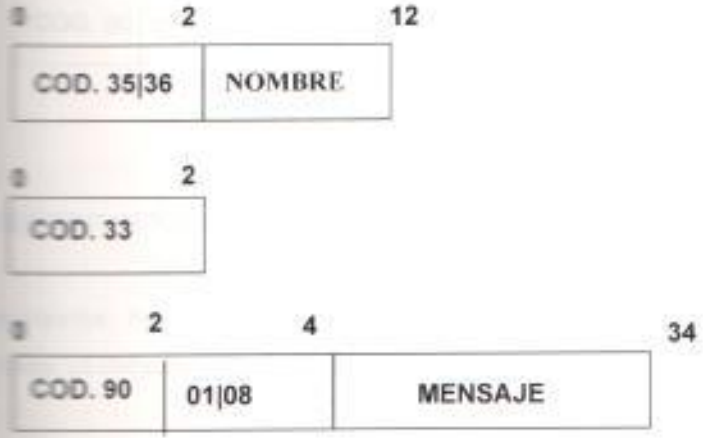
COD33: Este código le indicará a la aplicación cliente que se enviaron todos los nombres de usuarios independientes y grupos.

Nombre: Este campo del buffer de envío del servidor al cliente, le indica a la aplicación cliente los nombres de usuarios y de grupos.

COD 90: es un código que indica al cliente que un error ha ocurrido.

ERR: es un código de error que le indica al cliente el tipo de error que ha ocurrido.

mensaje : es un mensaje de error adicional por si hay fallas de sistema sobre las que la aplicación no tiene control.



REQUERIMIENTO: Actualiza clave del usuario

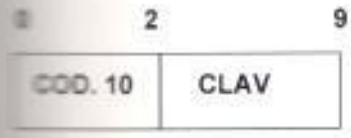
Descripción: Por medio de este requerimiento la aplicación cliente informa a la servidora que desea hacer una modificación de la clave de acceso.

Cliente:

buffer de requerimiento enviado desde la aplicación cliente a la servidora.

COD10: Por medio de este código la aplicación servidora determina de que requerimiento se trata.

Clave: Este campo del bufer le indica a la aplicación servidora la nueva clave.



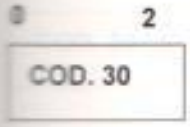
Servidor:

Buffer de respuesta enviado desde la aplicación servidora a la aplicación cliente.

COD30: Este código indica al aplicación cliente que hubo éxito .

COD 90: es un código que indica al cliente que un error ha ocurrido.

09: es un código de error que le indica al cliente el tipo de error que ha ocurrido.



REQUERIMIENTO: Eliminar-documento

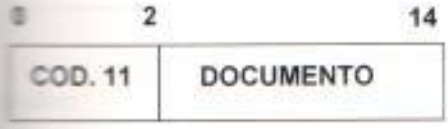
Descripción: Por medio de este requerimiento la aplicación cliente le indica al servidor que desea eliminar un documento determinado.

Cliente:

Buffer de requerimiento enviado desde la aplicación cliente a la servidora.

COD11: Este código le indica al servidor de que requerimiento se trata.

Documento:este campo del buffer indica el nombre del documento que se desea eliminar.



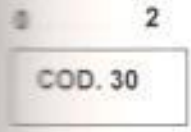
Servidor:

Buffer de respuesta enviado desde la aplicación servidora a la aplicación cliente.

COD.30: Este código le indicará a la aplicación cliente que hubo éxito.

COD.90: es un código que indica al cliente que un error ha ocurrido.

MSJ2: es un código de error que le indica al cliente el tipo de error que ha ocurrido.



REQUERIMIENTO: Quitar-documento

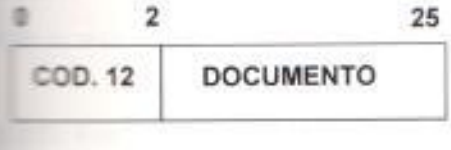
Descripción: Por medio de este requerimiento la aplicación cliente le indica al servidor que desea borrar del buzón un documento al cual el tiene acceso.

Cliente:

Buffer de requerimiento enviado desde la aplicación cliente a la servidora.

COD12: Este código le indica al servidor de que requerimiento se trata.

Documento:este campo del buffer indica el nombre del documento que se desea eliminar del buzón.



Servidor:

Buffer de respuesta enviado desde la aplicación servidora a la aplicación cliente.

COD.30: Este código le indicará a la aplicación cliente que hubo éxito.

COD.90: es un código que indica al cliente que un error ha ocurrido.

MSJ.03: es un código de error que le indica al cliente el tipo de error que ha ocurrido.

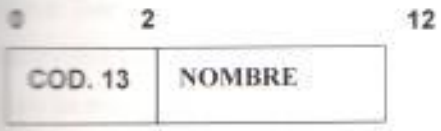
Servidor: responde

REQUERIMIENTO: Obtener lista de usuarios por grupos

Descripción: Por medio de este requerimiento la aplicación cliente le indica al servidor que le envíe una lista de los usuarios que pertenecen a un grupo determinado.

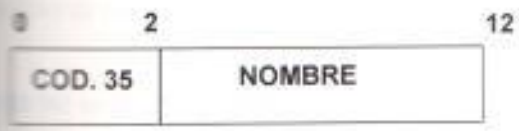
Cliente:
buffer de requerimiento enviado desde la aplicación cliente a la servidora.

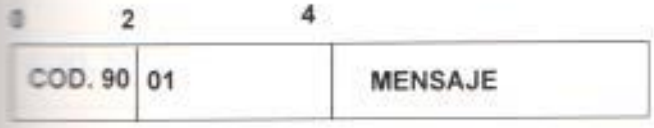
COD13: Este código le indica al servidor de que requerimiento se trata.
Nombre:este campo del buffer indica el nombre del grupo del cual se desea obtener la lista de usuarios.



Servidor:
buffer de respuesta enviado desde la aplicación servidora a la aplicación cliente.

COD35: Este código le indicará a la aplicación cliente que se esta enviando los nombres de usuarios.
COD33: Este código le indicará a la aplicación cliente que se enviaron todos los nombres de usuarios independientes y grupos.
Nombre: Este campo del buffer de envío del servidor al cliente, le indica a la aplicación cliente los nombres de usuarios y de grupos.
COD 90: es un código que indica al cliente que un error ha ocurrido.
IE: es un código de error que le indica al cliente el tipo de error que ha ocurrido.
mensaje : es un mensaje de error adicional por si hay fallas de sistema sobre las que la aplicación no tiene control.





REQUERIMIENTO: Obtener lista de grupos por usuarios

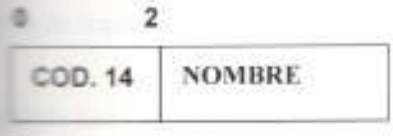
Descripción: Por medio de este requerimiento la aplicación cliente le indica al servidor que le envíe una lista de los grupos a los que un usuario ha sido asignado.

Cliente:

buffer de requerimiento enviado desde la aplicación cliente a la servidora.

COD14: Este código le indica al servidor de que requerimiento se trata.

Nombre: este campo del buffer indica el nombre del usuario del cual se desea obtener la lista de grupos a los que ha sido asignado..



Servidor:

buffer de respuesta enviado desde la aplicación servidora a la aplicación cliente.

COD36: Este código le indicará a la aplicación cliente que se esta enviando los nombres de grupos de usuarios.

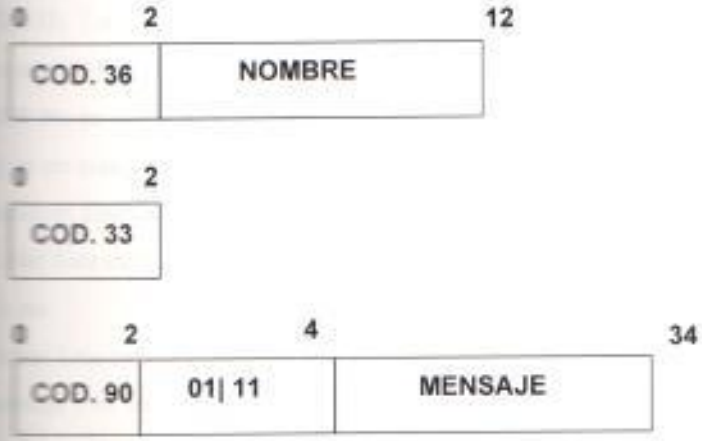
COD33: Este código le indicará a la aplicación cliente que se enviaron todos los nombres de usuarios independientes y grupos.

Nombre: Este campo del buffer de envío del servidor al cliente, le indica a la aplicación cliente los nombres de usuarios y de grupos.

COD 90: es un código que indica al cliente que un error ha ocurrido.

0,11: es un código de error que le indica al cliente el tipo de error que ha ocurrido.

mensaje : es un mensaje de error adicional por si hay fallas de sistema sobre las que la aplicación no tiene control.



REQUERIMIENTO: Obtener lista de usuarios que han leído un documento

Descripción: Este requerimiento le indica a la aplicación servidora que se desea obtener una lista de los usuarios que han leído el documento especificado.

Cliente:

buffer de requerimiento enviado desde la aplicación cliente a la servidora.

COD15: Este código le indica al servidor de que requerimiento se trata.

Documento:este campo del buffer indica el nombre del documento del cual se desea obtener la lista de usuarios que lo han leído.

[illegible]:

[illegible] de respuesta enviado desde la aplicación servidora a la aplicación cliente.

[illegible]: Este código le indicará a la aplicación cliente que se le esta enviando el nombre de un usuario que tiene permiso de acceso al documento especificado.

[illegible]: Este código le indica a la aplicación cliente que todos los usuarios que tienen acceso al documento han sido enviados.

[illegible]: es un código que indica al cliente que un error ha ocurrido.

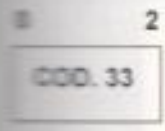
[illegible]: es un código de error que le indica al cliente el tipo de error que ha ocurrido.

[illegible]: Este campo del buffer de envio indica el login del cliente propietario de un documento.

[illegible]: Este campo es una bandera que le indica a la aplicación cliente si el documento ha sido leido por [illegible].

[illegible]: Este campo le informa al cliente la fecha en que el usuario reviso un documento, si es que lo hizo.

[illegible] : es un mensaje de error adicional por si hay fallas de sistema sobre las que la aplicación no tiene control.



[illegible]**REQUERIMIENTO:** Cerrar conexión

[illegible]:

[illegible] de requerimiento enviado desde la aplicación cliente a la servidora.

Descripción: Prepara el fin de la conexión, es decir, avisa al servidor el cierre de la conexión.

COD99: Este código le indica a la aplicación servidora que debe cerrar la conexión.

Mensaje : es un mensaje que le indicara al usuario si la terminación de su sesión fue normal o no.

0	2	32
COD. 99	MENSAJE	

Servidor:

Buffer de respuesta enviado desde la aplicación servidora a la aplicación cliente.

COD30: Este código le indica a la aplicación cliente que la operación cerrar conexión fue exitosa.

0	2
COD. 30	

CÓDIGOS DE ERROR .

En este apéndice se detallan los códigos de error que maneja la aplicación.

CÓDIGO	DESCRIPCIÓN
01	Error de lectura en el servidor
02	Documento no existe
03	No hay acceso a un documento
04	Requerimiento no válido
05	Grupo ya tiene permiso
06	Grupo no existe
07	Usuario ya tiene permiso
08	No existe permiso para un documento
09	Usuario no existe.
10	Grupo no tiene permiso
11	Usuario no asignado a grupos
12	Error no se puede abrir archivo
13	Usuario no tiene permiso

JUSTIFICACIÓN DEL DISEÑO

Esta sección trata de explicar y justificar el diseño elaborado para la comunicación entre los programa cliente y servidor.

Para que el sistema ofrezca el servicio mencionado las explicaciones deben seguir un protocolo determinado establecer la comunicación al nivel más elevado, es por esto que el protocolo pretende eliminar las redundancias operativas para desarrollar un control mas significativo y eficaz.

El formato de las cadenas están compuestos de campos de datos de longitud fija y presididos siempre por un campo que identifica el código de la transacción. Los campos de longitud fija se seleccionaron para facilitar la captura y procesamiento de los datos.

Cada una de las cadenas posee un número variable de campos. Esto se lo diseñó de ésta manera el flujo de campos vacios o innecesarios, y así mejorar la transmisión de aquellos.

En general, el diseño del protocolo y de sus formatos de cadenas pretende evitar la transmisión de cadenas largas e innecesarias para de esta forma mejorar la eficiencia de la comunicación.

DISEÑO DEL SERVIDOR.

Funcionalidad del Servidor

El servidor será capaz de:

1.- Permitir a varios clientes accesar al servicio de comparición de documentos simultáneamente.
2.- Manejar los permisos pertinentes para acceso a documentos.
3.- Manejar grupos y usuarios.
4.- Almacenar los documentos en modo binario.
5.- Trabajar en plataforma UNIX-BSD.
6.- Usar SOCKETS para establecer el enlace de comunicación con los clientes.
7.- Transferir datos por paquetes (TCP)

Tipo de Servidor y su justificación

Se ha tomado la decisión de implementar un servidor concurrente orientado a conexión basados en el tipo de servicio que se pretende dar, ya que los requerimientos de compartición de documentos prevé una

carga más o menos moderada. Porque de usar un modelo iterativo, solo un cliente podría ser atendido a la vez, y si bien existen requerimientos que podrán ser atendidos en corto tiempo, otros como por ejemplo la transferencia de archivos, podrían tomar demasiado tiempo lo que implicaría la espera excesiva de otros clientes que estén en cola esperando ser atendidos.

El servidor concurrente, por el contrario, podrá solventar una acogida favorable a este servicio, dando atención simultánea a los clientes, y proporcionando tiempos de respuesta relativamente cortos, haciéndolo mas eficiente.

El programa servidor será orientado a conexión, ya que el protocolo TCP provee de todos los servicios de entrega confiable y además desde el punto de vista del programador resulta un poco menos complicada su implementación.

El servidor además será Stateless pues los requerimientos son independientes, y por lo tanto no se requiere mantener almacenada información de estatus de requerimientos.

DISEÑO DE LOS DATOS MANEJADOS EN EL SERVIDOR

Estructuras.

Las estructuras de datos manejadas en el servidor, las mismas que representan tablas de información son las que se muestran a continuación:

1.-USUARIOS
- PASSWORD
- LOGIN

2.-GRUPOS
- NOMBRE_GRUPO
- LISTA_DE_USUARIOS

3.-DOCUMENTOS
- NOMBRE_DOCUMENTO
- LEÍDO
- FECHA PUBLICACIÓN
- FECHA ACCESO

4.-LISTA_COMPLETA_USUARIOS
- DOCUMENTO
- LISTA_DE_USUARIOS

5.-LISTA_COMPLETA_GRUPOS
- DOCUMENTO
- LISTA_DE_GRUPOS

Donde:

PASSWORD.-

Password del cliente que lo identifica como un cliente valido del servicio.

Valor almacenado: valor del campo CLAVE de la estructura interna manejada por el servidor.

LOGIN.-

nivel: nombre que se asigna al cliente del servicio.

Valor almacenado: valor del campo NOMBRE de la estructura interna manejada por el servidor.

NOMBRE_GRUPO.-

Nombre de grupo creado por el usuario(administrador).

Valor almacenado: valor del campo NOMBRE de la estructura interna manejada por el servidor.

LISTA_DE_USUARIOS.-

Lista de usuarios que pertenecen a un grupo(separada por comas).

Valor almacenado: tomado de los campos NOMBRE ó PERMISOS (dependiendo del requerimiento), de la estructura interna manejada por el servidor.

NOMBRE_DOCUMENTO.-

Nombre de los documentos almacenados por el cliente.

Valor almacenado: tomado del campo NOMB_DOCUMENTO de la estructura interna manejada por el servidor.

LEÍDO.-

Indica si un documento ha sido leído o modificado por el cliente

Valor almacenado: es el valor del campo LEÍDO tomado de la estructura interna manejada por el servidor

FECHA PUBLICACIÓN.-

Fecha de publicación del documento.

FECHA ACCESO.-

Fecha en la cual el cliente leyó un documento.

LISTA_DE_USUARIOS.-

Lista de los clientes que han leído un documento especificado.

Valor almacenado: es tomado del campo login de la tabla LISTA_COMPLETA_USUARIOS.

LISTA_DE_DOCUMENTOS.-

Lista de documentos que un cliente ó un grupo de clientes puede accesar (documentos que no son propiedad de un cliente ó clientes de un grupo especifico).

Valor almacenado: tomado del campo NOMB_DOCUMENTO de la estructura interna manejada por el servidor

COMPROMISOS.

Para la implementación del servidor se ha sacrificado recursos para ganar en eficiencia, el servidor concurrente orientado a conexión utiliza mas recursos en cuanto a memoria, ancho de banda, etc. Gana en eficiencia, pues desde el punto de vista del cliente, el no tiene que esperar demasiado tiempo para la atención a determinados requerimientos (en contraparte al servidor iterativo el cual mantendría en espera a los clientes que estén accesando al servicio mientras atiende un requerimiento de un cliente particular), y es relativamente mas rápido a mayor carga.

El servidor se compromete a:

- Dar servicio a aquellos clientes que consten en su tabla de usuarios.
- Permitir acceso a documentos a los que los clientes tengan permisos.
- Brindar al cliente información completa acerca de documentos de su propiedad y los que no son de su propiedad, pero a los que ellos tienen permisos.
- Brindar información completa a un determinado cliente acerca de todos los clientes que tienen acceso a este servicio.
- Brindar información al cliente acerca de todos los grupos que existan para este servicio.
- Proporcionar información acerca de los usuarios que pueden accesar a un documento especifico(propiedad del cliente que hace el requerimiento).
- Borrar documentos de su lista de documentos a los que tiene acceso.

Diseño de la aplicación servidora.

SISTEMA DE COMPARTICION DE DOCUMENTOS
Diagrama Esquematico de la aplicacion

ESTABLECER CONEXIÓN.

-Este modulo se encarga de inicializar las variables del sistema, y establecer la conexion entre la aplicación cliente y la servidora.

-Declaración e inicialización de variables.

-Localizar un socket.

-Enlazar el socket a puerto bien conocido.

-Determinar el numero máximo de clientes en la cola .

-Dejar el socket en modo pasivo.

-Aceptar y establecer conexión.

MANEJADOR DE REQUERIMIENTOS.

Este modulo es quien procesa los requerimientos de la aplicación cliente una vez establecida la conexión.

INICIALIZACIÓN DE ESTRUCTURA.

-Inicializa las estructuras con valores defaults.

LLENAR ESTRUCTURA.

Este modulo se encarga de obtener los datos que le son enviados desde el cliente y llenar las debidas estructuras con las que el trabaja.

-leer buffer desde el socket

-obtener los datos desde el buffer

-colocar los datos obtenidos en el campo correspondiente d e la estructura.

PROCESAMIENTO DE REQUERIMIENTOS.

ACCESO AL SERVICIO.

Obtiene del socket los campos de identificación del usuario y los valida.

-leer los campos NOMBRE Y CLAVE de la estructura

-comparar los valores leídos con valores de la tabla USUARIOS

-si el cliente existe

-crear nuevo socket para atender los requerimientos del cliente

-sino

-enviar mensaje de error al cliente

-terminar conexión.

PEDIR DOCUMENTO

Se encarga de buscar en su repositorio un documento especificado por el cliente para su envio.

-leer campo NOMB_DOCUMENTO de la estructura

-buscar NOMB_DOCUMENTO en la tabla DOCUMENTOS

-si existe

-obtener datos de tabla

-colocar datos en los campos apropiados de la estructura

-sino

-enviar mensaje de error al cliente.

GRABAR DOCUMENTO

Toma el documento que el cliente le envía y lo graba en su repositorio.

-leer campo NOMB_DOCUMENTO

-crear archivo con NOMB_DOCUMENTO

-mientras no sea fin de archivo

-leer campo DATOS de la estructura

-escribir el dato leído en archivo creado

OBTENER LISTA DE USUARIOS

Obtiene la lista de usuarios que tienen acceso al servicio y se la envía al cliente.

-obtener datos de la tabla USUARIOS

-colocar datos de clientes en la estructura uno a la vez

OBTENER LISTA DE GRUPOS.

Obtiene la lista de grupos existentes en el servicio y se la envía al cliente.

-obtener datos de la tabla de GRUPOS

-colocar los datos de grupos en la estructura uno a la vez

SETEAR PERMISOS DE DOCUMENTO.

Asigna permisos de acceso a un documento que el cliente propietario especifique.

-obtener datos de campos DOCUMENTO y PERMISOS de la estructura

-asignar permisos a un documento actualizando tabla LISTA_COMPLETA_USUARIOS.

PEDIR PERMISOS DE UN DOCUMENTO

Se encarga de obtener los permisos que un documento especificado tiene y se los envía al cliente.

-leer campo NOMB_DOCUMENTO de la estructura

-si documento existe

-obtener permisos de tablas LISTA_COMPLETA_USUARIOS y LISTA_COMPLETA_GRUPOS

-colocar permisos en el campo PERMISOS de estructura

-sino

-envía mensaje de error al cliente

MODIFICAR CLAVE DE ACCESO DEL CLIENTE

Da la opción de cambiar la clave de acceso de un usuario.

-leer campo CLAVE de estructura

-actualizar tablas de USUARIOS y GRUPOS

ELIMINAR DOCUMENTO.

Borra físicamente un documento del repositorio y actualiza las tablas.

-leer campo NOMB_DOCUMENTO de la estructura

-si el nombre de documento existe en tabla DOCUMENTOS

-actualizar tabla DOCUMENTOS y todas las otras que resulten afectadas por la operación

-sino

-envía mensaje de error al cliente

MANEJADOR DE RESULTADOS.

Formatea los resultados obtenidos del procesamiento de los requerimientos y los prepara para ser manejados por la aplicación servidora.

LLENAR BUFFER

Coloca los datos procesados en la estructura de envío (Buffer).

-llenar los campos de la estructura con los datos que sean necesarios

-si es transferencia de documento hacia el cliente.

-leer archivo y llenar campo DATOS de la estructura

-sino

-llenar buffer con los datos actuales de la estructura.

-volver al paso 2 de ser necesario.

ENVÍO DE DATOS.

Transmite los datos hacia la aplicación cliente a través de la red.

-colocar buffer en el socket.

CERRAR CONEXIÓN.

Libera los recursos que esta utilizando la aplicación servidora y termina.

-enviar todos los datos que tenga para ese cliente

-cerrar todos los descriptores asociados al proceso actual

-salir al sistema.

ADMINISTRACIÓN DEL SERVIDOR.

La administración del servidor se llevara a cabo mediante el programa ASCDOC, el cual es una herramienta que usa una interface a base de menúes, para permitirle al usuario autorizado, realizar alteraciones a los archivos que de otra manera solamente podrían ser accesados en tiempo de ejecución desde la aplicación cliente.

Esta aplicación, permitirá al usuario administrador ingresar, modificar, y eliminar datos de usuarios, grupos y documentos. Además le provee un listado de los usuarios que tienen acceso al sistema, los grupos autorizados para usar este servicio y los documentos compartidos por los usuarios del servicio. Se pide confirmación de cada operación que se lleve a cabo.

El administrador usa una interface a base de menúes, donde el usuario deberá escoger de entre las opciones que se le dan, la que específicamente necesite.

```
SISTEMA DE COMPARICIÓN DE ARCHIVOS
==================================

UTILITARIO DE ADMINISTRACIÓN
============================

1. Usuarios
2. Grupos
3. Salir
Escoja una opción (1-3), o q para salir:_
```

Menú principal

Para mas detalles, revisar: Manual del Administrador de la aplicación servidora

DISEÑO DEL CLIENTE

FUNCIONALIDAD DEL CLIENTE

Esta sección contiene aquellos compromisos que la aplicación cliente pretende llevar a cabo a la realidad para ofrecer funcionalidad total o parcial a aquellos requerimientos de usuarios un tanto exigentes.

La aplicación cliente ofrece total funcionalidad al usuario, cuando en el menú principal se presenten todos los comandos que el operador del programa puede ejecutar; además de contener comandos deshabilitados en ciertos estados del programa, el menú contiene comandos dinámicos que se visualizan cuando la operación de aquellos es válida.

La aplicación cliente deberá mostrar al usuario en todo momento el contenido de su buzón de documento s incluyendo documentos que son de su propiedad como aquellos a los que tiene permiso de acceso por parte de otras aplicaciones, ademas le ofrecerá facilidades para obtener una amplia variedad de información acerca de los documentos, así como de los clientes que tengan acceso al servicio, como son: proveer listas de usuarios del servicio, grupos definidos para este servicio, proveer listas de documentos, listas de usuarios que tienen acceso a un documento, listas de clientes que pertenencen a un grupo dado , listas de grupos a los que un usuario ha sido asignado, obtener documentos desde el buzón , colocar documentos en el buzón y borrar documentos de su propioedad del buzón.

Además, el programa cliente ofrecerá manipulación y control directo de los errores del sistema incluidos aquellos relacionados a la comunicación, para que el usuario pueda fácilmente depurar y entender errores y procedimientos.

DATOS MANEJADOS POR EL CLIENTE

La presente sección pretende informar a los desarrolladores futuros la agrupación y relación de los datos que el cliente lee y transmite al servidor. Todos los datos se listan a continuación:

Tipo	Nombre	Longitud	Descripción
char	código	2	código de transacción
char	nombre	10	nombre de usuario o grupo
char	clave	6	clave del usuario
char	documento	12	nombre del documento con extensión
char	fecha	8	fecha de creación del documento
char	tipo	1	s = leido, n= no leido
char	modo	1	r = lectura , w = escritura
char	id	4	identificador de usuario o grupo
char	permisos	80	lista de usuarios y grupos con permisos
char	grupo	20	nombres de grupo
char	datos	2048	bloques archivo texto o binario
char	mensaje	30	mensajes de error .

DISEÑO DEL CLIENTE

SELECCIÓN DEL MODELO

Esta sección describe entre otras cosas el porque se ha seleccionado el modelo Cliente-Servidor entre otros existentes. El paradigma Cliente-Servidor es el patrón primario de la iteración entre las aplicaciones cooperativas. Este paradigma forma las bases de la mayoría de las comunicaciones de redes. Este nos ayuda a entender el fundamento sobre el cual los algoritmos distribuidos son creados e implementados.

Básicamente el modelo Cliente-Servidor ha sido seleccionado porque la construcción de los programas de interacción resulta sencilla y fácil.

La aplicación Cliente que se ha diseñado, acepta un comando del usuario, lo transforma en un requerimiento que es enviado al programa servidor y espera la respuesta de parte del programa servidor; el programa Cliente termina cuando el usuario cierra la aplicación ó una vez que este halla usado el servidor varias veces y no quiera continuar, es decir, en otros términos se deslogonee sin cerrar la aplicación.

En conclusión, los procesos que usan la comunicación de red caen dentro del patrón de uso llamado modelo Cliente-Servidor.

DISEÑO DEL ALGORITMO CLIENTE

El diseño del programa Cliente es usualmente más fácil que la creación del programa servidor, en cuanto se refiere al algoritmo que debe controlar los procesos de comunicación y conexión a la red y al servidor.

Para construir el Cliente de acuerdo al diseño del servidor se utilizará el protocolo de transporte TCP, haciendo de ésta manera la tarea de programación en red más fácil, ya que como es sabido por todos, TCP manipula toda la confiabilidad y los problemas de control de flujo que surgen cuando se establece una comunicación en Internet.

Nuestro Cliente TCP sigue a más de los pasos del algoritmo básico del Cliente mostrado más adelante, otros pasos que ayudan a resolver o manipular y procesar los requerimientos de conexión y del propio usuario del servicio.

Las próximas secciones van desarrollando y extendiendo este algoritmo paso a paso, hasta llegar a obtener un procedimiento más o menos detallado al máximo que resuelva o más bien que brinde sin problemas el servicio deseado por el usuario de un manera totalmente transparente para este.

ARQUITECTURA DEL CLIENTE.

ALGORITMO BÁSICO

1. Encontrar dirección IP y puerto del servidor.
2. Obtener un socket.
3. Seleccionar un puerto de protocolo.
4. Conectar el socket.
5. Establecer la comunicación.
6. Cerrar la conexión.

ALGORITMO CLIENTE: NIVEL 1

1. Inicializar variables y estructuras.
2. Encontrar la dirección IP y el número de puerto del protocolo.
3. Obtener un socket.
4. Dejar a TCP seleccionar un puerto de protocolo no usado.
5. Conectar el socket al servidor.
6. Enviar los requerimientos y esperar las respuestas.
7. Cerrar la conexión.

ALGORITMO CLIENTE: NIVEL 2

1. Inicializar.
   1.1 Definir constantes, variables y estructuras.
   1.2 Iniciar variables y estructuras.
   1.3 Obtener y manipular los datos iniciales o argumentos.

2. Dirección IP y número de puerto.
   2.1 Buscar IP
   2.2 Buscar puerto

3. Obtener un socket.

4. Inicializar las estructuras del socket.

5. Conectar el socket usando la estructura del socket.

6. Requerimientos y respuestas.
   6.1 Enviar requerimiento de servicio.
   6.2 Si no hay servicio ir al paso 7.
   6.3 Capturar eventos y obtener requerimiento n
   6.4 Llamar procedimiento n
   6.5 Si fin de servicio ir al paso 7 sino ir al paso 6.3

7. Cerrar el servicio
   7.1 Enviar mensaje de fin de conexión
   7.2 Cerrar la conexión del socket.

Como se nota claramente, sólo existen dos niveles de depuración del algoritmo básico, ya que hemos creído que el desarrollo a un nivel más nos llevaría a introducirnos en los detalles de programación, los cuales serán explicados en las próximas secciones con lujo de detalles.

Tal como se visualiza que cada uno de los niveles, en especial el último nivel de desarrollo del algoritmo describe paso a paso en forma clara y precisa cada uno de los procedimientos que controlan principalmente las tareas de comunicación a nivel por supuesto de las aplicaciones Cliente-Servidor.

FLUJO DEL SOFTWARE CLIENTE

Para muchos lectores y Analistas-Programadores todavía se les hace más fácil el entendimiento de un programa mediante los diagramas de flujo; es por esto que nos permitimos mostrarles además el flujo del programa principal como una extensión de los algoritmos desarrollados en la sección previa. Así podemos entender de manera gráfica la secuencia principal del algoritmo que el grupo de desarrollo diseño para ofrecer el servicio deseado a los usuarios de Internet.

INTERFACES

INTERFACES CLIENTE - TCP/IP

Debido a que generalmente el software del protocolo TCP/IP reside dentro del Sistema Operativo de la máquina, la interface exacta entre un programa de aplicación Cliente y los protocolos TCP/IP depende en los detalles del Sistema Operativo que se esté usando o bajo el cual se va a implementar el software Cliente, por lo que esto no se encuentra especificado en el protocolo standard de TCP/IP.

Basados en la información investigada para la implementación del programa Cliente,hemos examinado la interface del socket usada en BSD-Unix y además hemos adoptado para la implementación del programa Cliente el viejo paradigma utilizado por Unix: open-read-write-close.

La aplicación Cliente usará TCP, ya que ésta creará un socket, enlazará las direcciones end-point local y remota al socket, abrirá la conexión con la dirección remota (dirección IP y número de puerto del servidor), luego se comunicará usando las primitivas o librerías write (para enviar una pregunta) y read (para leer una respuesta), y finalmente cuando el usuario termine o desee salir de la aplicación Cliente, el programa procederá entonces a cerrar el socket.

Además de todo esto, el programa Cliente usará rutinas de librería para crear y manipular direcciones IP, para convertir datos cuando sea necesario entre el formato de la máquina local y el byte de orden standard de red; y también usará el FTP cuando sea requerido su servicio, es decir, el intercambio de archivos entre los programas en este caso el software Cliente y Servidor.

Esta interface ha sido seleccionada para la implementación del software Cliente, ya que, la interface socket se ha convertido en la más popular y es además soportada ampliamente por muchos vendedores de software de desarrollo de TCP/IP. Por otro lado los vendedores que no ofrecen facilidades de sockets en sus sistemas operativos, a menudo proveen las librerías de socket que permiten a la aplicación usar llamadas al socket aun cuando el Sistema Operativo básico tenga disponible para el programador un conjunto diferente de llamadas al sistema.

Este justamente es el caso nuestro, ya que como nuestro sistema operativo no las incluye, hemos adquirido las librerías de socket por un tercer fabricante para que la aplicación Cliente haga las llamadas respectivas, tal como lo explicaremos con más detalle en las próximas secciones incluidas en este documento.

INTERFACE CLIENTE - USUARIO

La aplicación Cliente usará como interface para los usuarios del programa los recursos ofrecidos por Windows y conocidos con el nombre de GUI's, en otras palabras, los usuarios tendrán acceso al servidor mediante una interface gráfica y amigable cuya manipulación y acceso se hará en su mayor parte utilizando el dispositivo apuntador o Mouse que el sistema de cómputo tenga instalado para el uso.

La interface del programa Cliente contiene barras de títulos, barras de menús, barras de estado, barras de desplazamiento, barras de herramientas, ventanas, listas combinadas, cajas de diálogo standard de Windows y diseñadas también por el propio equipo de desarrollo.

Para obtener este tipo de aplicaciones que usen y corran los recursos de Windows se usará como antes se había mencionado un Development Kit de desarrollo para TCP/IP, llamado Distinct, el cual contiene las rutinas de librerías para los sockets tanto para Visual BASIC así como para Microsoft C.

Por lo tanto, para desarrollar la inteface gráfica hemos usado el programa de desarrollo de aplicaciones Visual BASIC junto con las librerías de socket añadidas a los recursos de programación o caja de herramientas de programación. Las próximas secciones describirán con lujo de detalles la interface gráfica para el usuario que será usada para aceptar los requerimientos de los usuarios del cliente.

DESCOMPOSICIÓN MODULAR

Esta sección describe de manera detallada todos y cada uno de los módulos principales en que se ha dividido el programa Cliente. En primer lugar se muestra una lista completa en orden alfabético de los módulos del programa. Por último se realiza la descripción de cada uno de estos módulos en donde se especifica entre lo más sobresaliente la función del módulo, los parámetros de entrada del módulo y además los valores que retorna dicho módulo si es que alguno de ellos lo hace.

Lista Alfabética de los Módulos del Sistema:

Abrir_Servicio
Ayuda_BoxDlg
Borrar_Doc
Borrar_Perm
Cerrar_Servicio
Enviar_Doc
Get_Lista_Doc
Get_Lista_Grupo
Get_Lista_User
Grupos_User
Ingresar_Host
Iniciar_Var
Mensaje_BoxDlg
Mostrar_Inicio
Mostrar_Formato
Recibir_Doc
Set_Clave
Usuarios_Grupo

A continuación se procederá a efectuar la descripción de cada uno de los módulos mencionados anteriormente.

Abrir_Servicio

Función: este módulo esconde los procedimientos para accesar al servicio, es decir, que obtiene los datos de un usuario y con éstos envía un requerimiento al servidor para pedir acceso al servicio que se especifica.

Parámetros: este procedimiento maneja los datos del usuario (el nombre y su clave).

Retorno: en este módulo se retorna un valor entero que especifica un índice de texto para ser visualizado después de ejecutado el módulo.

Ayuda_BoxDlg

Función: aquí se esconden los procedimientos que destinados a mostrar al usuario una pantalla de ayuda según el contexto o el estado del programa.

Parámetros: este procedimiento recibe como parámetro un valor entero que especifica el índice del texto de ayuda que se pretende mostrar en el dialogo.

Retorno: no retorna valor alguno.

Borrar_Doc

Función: el módulo mencionado esconde todos los procedimientos destinados a eliminar un documento determinado. Primero capta los eventos de la interface y luego envía un requerimiento para que el servidor borre el documento especificado.

Parámetros: los parámetros que recibe este procedimiento es el nombre del documento que se pretende borrar.

Retorno: este procedimiento retorna un mensaje satisfactorio de la operación o uno de error indicando que no ha sido posible efectuar la operación deseada.

Borrar_Perm

Función: este módulo esconde también los detalles de programación o implementación para eliminar un permiso de un documento. Esto lo realiza una vez que el procedimiento halla sido activado por el comando de menú en la interface del usuario.

Parámetros: los valores que recibe son el nombre del usuario y el documento que se pretende eliminar del buzón.

Retorno: esta función retorna un valor que es un índice al mensaje que se muestra después de ejecutado el módulo.

Cerrar_Servicio

Función: este procedimiento esconde los detalles de los procedimientos destinados para enviar un mensaje al servidor de cierre de servicio y conexión. Una vez desde luego que el usuario ha activado dicho módulo.

Parámetros: este procedimiento no recibe ningún tipo de valor para poder efectuar la operación.

Retorno: el mensaje que se retorna es uno de error o de ejecución efectiva del comando.

Enviar_Doc

Función: el módulo esconde todos los detalles de implementación de los procedimientos destinados a transportar los datos de un documento y enviar un requerimiento de creación de un documento para el servicio.

Parámetros: los parámetros son el nombre del documento y el nombre del usuario.

Retorno: del mismo modo se retorna un mensaje final al usuario para indicarle la ejecución satisfactoria del módulo o el tipo de error si es que ha ocurrido alguno.

Get_Lista_Doc

Función: este módulo esconde los detalles implementados en los procedimientos para lograr enviar un requerimiento para obtener la lista de documentos disponibles para un usuario específico; para luego ser visualizados en la porción de la pantalla destinada para el efecto.

Parámetros: aquí se envía únicamente el nombre del usuario que está en el servicio.

Retorno: se retorna cada uno de los datos relacionados con los documentos como son el nombre del documento, el modo, el tipo y el usuario que lo creó; ó en su defecto un mensaje de error.

Get_Lista_Grupo

Función: de igual modo aquí se esconde los detalles de los procedimientos que envían un requerimiento para obtener la lista de los grupos existentes en el servicio; para luego ser visualizados en pantalla.

Parámetros: este procedimiento no necesita algún tipo de parámetro para ser ejecutado.

Retorno: retorna los datos de cada grupo que pertenece al servicio como son el nombre del grupo y su identificador.

Get_Lista_User

Función: este módulo de igual modo esconde los detalles de los procedimientos que envían un requerimiento para obtener la lista de los usuarios que pertenecen al servicio; para luego ser visualizados en pantalla.

Parámetros: este procedimiento no necesita algún tipo de parámetro para ser ejecutado.

Retorno: retorna los datos de cada usuario que pertenece al servicio como son el nombre del usuario, su identificador y el grupo al que pertenece.

Grupos_User

Función: este módulo esconde los detalles de programación destinados a mostrar una caja de dialogo con campos que muestran los nombres de grupos de un usuario determinado.

Parámetros: este procedimiento requiere el nombre del usuario como parámetro.

Retorno: aquí se retorna la lista de nombres de grupos de un usuario.

Ingresar_Host

Función: este módulo esconde los detalles de programación necesarios para visualizar una caja de dialogo que tenga campos para llenar el nombre del host destino, el nombre del usuario y su clave de acceso.

Parámetros: este procedimiento no requiere ningún valor como parámetro.

Retorno:aquí se retorna el nombre del host destino, el nombre del usuario y la clave.

Iniciar_Var

Función: este módulo esconde los detalles de programación necesarios para implementar e inicializar las variables y constantes.

Parámetros: este procedimiento no necesita que se le ingrese un determinado valor.

Retorno: este procedimiento no retorna ningún tipo de dato.

Mensaje_BoxDlg

Función: este procedimiento esconde los detalles de programación necesarios para implementar un caja de dialogo de mensajes que el sistema pretende mostrar cuando sea necesario.

Parámetros: recibe como parámetro la cadena de caracteres que se desea mostrar.

Retorno: este procedimiento no retorna ningún tipo de valor o dato.

Mostrar_Inicio

Función: este procedimiento esconde todos los detalles de programación utilizados para implementar y mostrar una pantalla de presentación al inicio de la ejecución de la aplicación.

Parámetros: este procedimiento no necesita ningún tipo de parámetro para su ejecución.

Retorno: no existe valor de retorno después de su ejecución.

Mostrar_Formato

Función: este módulo esconde los detalles de los procedimientos necesarios para la implementación visualización de la pantalla de trabajo que el usuario necesita manipular.

Parámetros: requiere como parámetros las listas de los documentos, la lista de los usuarios y la lista de los grupos que se van a mostrar.

Retorno: este procedimiento no requiere retornar algún valor después de su ejecución.

Recibir_Doc

Función: el módulo esconde todos los detalles de implementación de los procedimientos destinados a transportar los datos de un documento desde el servidor hasta el cliente.

Parámetros: los parámetros son el nombre del documento y el nombre del usuario.

Retorno: del mismo modo se retorna un mensaje final al usuario para indicarle la ejecución satisfactoria del módulo o el tipo de error si es que ha ocurrido alguno.

Set_Clave

Función: este módulo esconde los detalles de programación de los procedimientos destinados para enviar un requerimiento de actualización de la clave de un usuario.

Parámetros: los valores que se ingresan al procedimiento son la clave del usuario.

Retorno: este procedimiento retorna un mensaje al usuario indicándole la ejecución efectiva del comando o algún error detectado en el sistema.

Usuarios_Grupo

Función: este módulo esconde todos los detalles de implementación de los procedimientos necesarios para el envío de un requerimiento para obtener una lista de usuarios de un grupo.

Parámetros: los valores pasados al procedimiento son el nombre del grupo.

Retorno: así mismo se retorna un mensaje indicándole al usuario la ejecución efectiva del comando o el tipo de error que se ha detectado.

COMPROMISOS DE DISEÑO

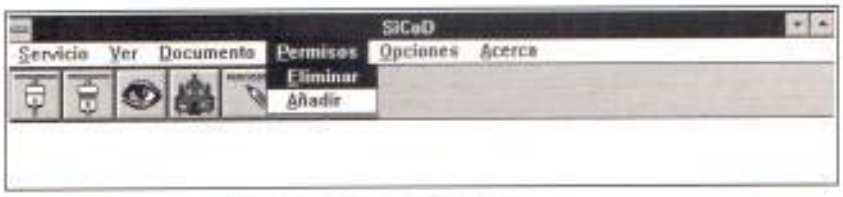
Esta sección menciona aquella especificaciones que se diseñaran e implementaran de forma básica, es decir, todos aquellos compromisos que el grupo de desarrollo implementará en el programa cliente.

El grupo de desarrollo siguiendo los standard de los sistemas abiertos se compromete a revelar en este documentos todas las especificaciones de diseño del programa, tales como el algoritmo, los formatos de los datos y sobre todo el código de implementación.

Referente a la implementación, se anexara en el documento final el código de todas las subrutinas implementadas en Visual BASIC para su mayor comprensión y mantenimiento. El diseño del programa cliente será en su mayoría orientado a objeto, aunque muchos programadores digan que Visual Basic esta orientado a eventos, el grupo de desarrollo implementará todo cuanto sea posible la programación de objetos, ya que Visual al menos lo soporta.

La aplicación cliente se diseñara totalmente con una interface gráfica para el usuario final (GUI), es decir utilizando e implementando la mayoría de los recursos ofrecidos para Windows 3.1 o superior, como son las barras de menús, de herramientas, listas combinadas y objetos OLE si es posible.

Entre los compromisos mas importantes tenemos la manipulación y transferencia de los errores del sistema y su comunicación hacia el usuario; implementación de los procedimientos de arrastrar y soltar y sobre todo la implementación de varias secciones con un solo programa cliente mediante la utilización, implementación y codificación de una interface de múltiplos documentos (MDI). De este modo se podrán mantener varias secciones de diferentes usuarios con el programa servidor .



ALCANCES Y RESTRICCIONES

- El Sistema prácticamente no tiene restricciones en cuanto al manejo de la cantidad de grupos y usuarios, ya que, el sistema en realidad puede aceptar cualquier cantidad de grupos. Pero hay que dejar en claro que en cuanto al número de usuarios que se puede manejar para cada grupo existe un número máximo de 50 usuarios por grupo. Esta decisión resultó de la limitación de recursos de almacenamiento que ofrece el sistema de computación tanto de memoria virtual como de la dinámica.

- En lo que respecta a los permisos dados para un documento determinado, tiene una restricción de 50 permisos ó nombres de grupos y/o usuarios del servicio por cada documento puesto a disposición del sistema. Por otro lado hay que anotar que la cantidad de documentos que pueden ser almacenados en el sistema servidor depende sólo y exclusivamente de la cantidad de espacio disponible para el programa servidor.

- El sistema desarrollado puede transportar cualquier tipo de archivo desde el cliente al servidor o viceversa, ya que el sistema utiliza el servicio bien conocido como el FTP . Por este motivo si se quiere lograr mayor eficiencia de transmisión de los documentos se debe en lo posible enviar archivos de tamaño más o menos pequeño para no sobrecargar el tráfico de la red.

- Existen restricciones además en lo referente a la longitud máxima de los campos utilizados por el servidor y el cliente, como: el nombre de los grupos deben ser de hasta 10 caracteres, la clave del usuario hasta 6 caracteres y la restricción más importante para el funcionamiento efectivo del cliente es el nombre del documento de hasta 12 caracteres divididos en los primeros 8 caracteres de un nombre seguido por un punto y finalmente una extensión necesaria de 3 caracteres, los cuales deben ser txt, doc, xls y BMP para que el Cliente pueda verlos sin ningún problema. Tenemos que recalcar que la parte del nombre del documento no debe contener el carácter ascci del guión (-) , ya que esto haría caer la transferencia del archivo, por lo tanto se requiere que se utilicen solamente caracteres comunes y corrientes y dejar a un lado los caracteres especiales.

- Una restricción a nivel del programa Cliente es que el usuario del sistema debe seleccionar siempre un ítem de las listas que aparecen en la interface para poder ejecutar algún comando del menú de opciones principal. Otra restricción es el acceso de solamente un usuario por cada ejecución del programa Cliente en la máquina local.