Concórdia Serviço Local de Troca de Mensagens

Carlota Gutiérrez Carabias	e11137
Hugo Revuelta Aja	e11134
Alvaro González Nava	e11136
Dario Prieto Soria	e11130

Carlota Gutiérrez Carabias
Alvaro González Nava
Hugo Revuelta Aja
Dario Prieto Soria

INTRODUCCIÓN

El trabajo desarrollado por nuestro grupo consiste en una aplicación de intercambio de mensajes de texto de forma asíncrona entre usuarios linux. La aplicación trata de ser lo más rigurosa y eficiente posible para garantizar la comunicación y la protección de los mensajes intercambiados entre los usuarios. Se ha tratado de buscar la mejor estructura posible tonto para el almacenamiento de los datos como para todo aquello que precise el código y lo haga más eficiente y legible.

Las diferentes estructuras, explicadas en los siguientes puntos del informe, están pensadas de una manera lo más meticulosa posible para tener en cuenta todos los detalles de la organización que han sido sugeridos. Cada mensaje intercambiado entre los usuarios guarda un formato en el que se pueda recoger la información relevante acerca del mismo, como a la hora que fue enviado, el remitente ...

Una característica muy importante de este proyecto es la capacidad de gestionar grupos dentro del programa. Esta función permite a los usuarios crear y administra grupos a los que les llegan mensajes comunes que todos pueden visualizar. De esta manera se mejora la comunicación entre todos para evitar repetir mensajes.

Durante el siguiente documento, se detallará las diversas fases y decisiones estratégicas tomadas por nuestro grupo a lo largo del desarrollo del proyecto. Se analizarán las decisiones sobre las estructuras utilizadas, decisiones técnicas y soluciones a problemas que se han ido generando. Todo esto busca el propósito de contextualizar de la mejor manera posible el resultado obtenido por nuestro equipo y comprender mejor la aplicación presentada.

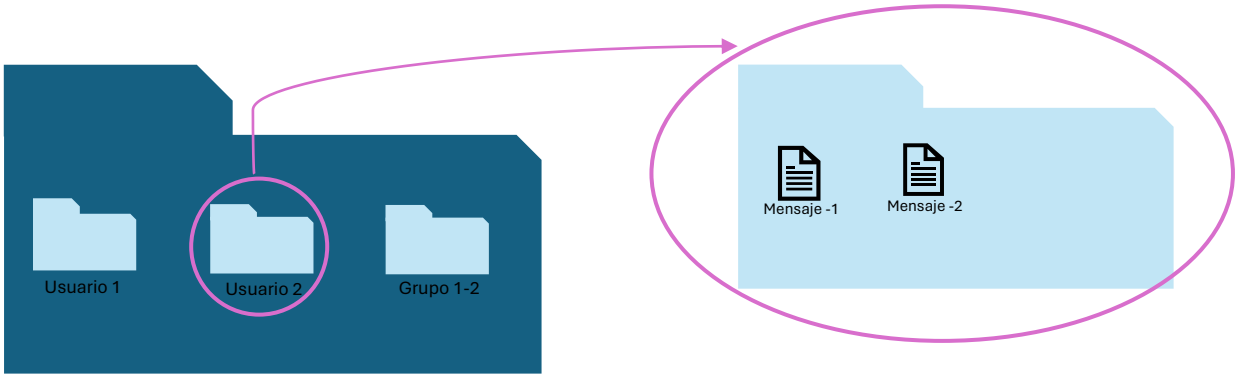
Carlota Gutiérrez Carabias
Alvaro González Nava
Hugo Revuelta Aja
Dario Prieto Soria

ARQUITECTURA DEL SISTEMA

Una de las primeras decisiones cruciales en el desarrollo del proyecto fue establecer la estructura para el almacenamiento de información de los mensajes y los propios mensajes. Tras considerar diversas opciones, la arquitectura que se seleccionó y sobre la cual se ha trabajado es la siguiente:

Cada usuario tiene su propia carpeta donde de irán guardando los distintos mensajes que le son enviados tanto a los usuarios en particular (“mensaje_x.txt”) como los mensajes que receban por pertenecer a un grupo en concreto (“mensaje_grupo_x.txt”). Estas carpetas privadas de usuario deberán crearse en el momento que se activa la aplicación.

Por otro lado, se encuentran las carpetas de grupos, en estas se guarda un documento con la información relevante acerca del grupo, quién lo creo, los miembros del mismo...



Esta nos pareció la forma más ordenada para desarrollar la estructura de almacenamiento de los mensajes. Esto se debe a que cada mensaje es guardado en un fichero único cuyo nombre sirve como identificador (siendo el 1 el primer mensaje enviado e incrementándose conforme se envían más), esto facilita el listado y la lectura de los mensajes. Además, la inclusión del nombre de usuario en cada carpeta simplifica la gestión de permisos, donde el usuario asociado a la carpeta es el responsable de los permisos de acceso y lectura del contenido de dentro de las carpetas.

Una de las estructuras más relevantes en el proyecto es la de los mensajes. Estos son la base de toda la aplicación, sin embargo, la estructura no es demasiado compleja. Los mensajes se almacenan en ficheros en los cuales podemos distinguir dos partes.

La primera parte corresponde a donde se encuentra toda la información que únicamente es relevante para la aplicación. En este apartado se almacena:

Carlota Gutiérrez Carabias Hugo Revuelta Aja
Alvaro González Nava Dario Prieto Soria

- Información sobre si el mensaje fue leído (si/no).
- Fecha y hora a la que se envió el mensaje.
- Emisor del mensaje.

La segunda parte corresponde a la parte que verán los usuarios cuando deseen leer los mensajes. Es decir, el propio contenido del mensaje.

Adentrándonos un poco más en las estructuras presentadas en el código una de las más relevantes es la de la gestión de grupos. Para ello se empleó “struct” en la cual se registra su creador, el nombre del grupo, número de usuarios que componen dicho grupo y los nombres de los usuarios. Esta información es volcada en los archivos de Información dentro de la carpeta del grupo, además se actualizará ante cualquier cambio que realice el creador.

Para que sea más fácil la comprensión del código y posibles actualizaciones en el futuro el código fue dividido en diversos ficheros agrupando las funciones comines entresi. Para ello se han creado 5 ficheros distintos:

En primer lugar, encontramos el documento principal, main.c, en el cual se encuentra básicamente el main de la aplicación. Este se encarga de recoger el comando que desea ejecutar el usuario y redirigir a la función deseada. Además, en este fichero se inicializan aquellas variables que van a ser usadas por todas las funciones.

En segundo lugar, encontramos un fichero denominado datos.h, en el se almacenan tanto las librerías que van a ser usadas todo aquello a lo que deben tener acceso el resto de programas “.c”. Esto último nombrado se refiere a la estructura de grupos y a todas las variables externas a las que se debe poder acceder desde cualquier sitio del programa.

En tercer lugar, el fichero “prototipos.c” es el lugar en el que se encuentran declaradas todas las funciones que se emplean a lo largo del programa. Este fichero actúa como puente entre el main y los archivos donde se almacenan dichas funciones. Además cada función cuenta con un comentario en el que se explica tanto lo que hace la función como para que sirven sus parámetros y en el caso de retornar algo que es y para que sirve, generalmente control de errores.

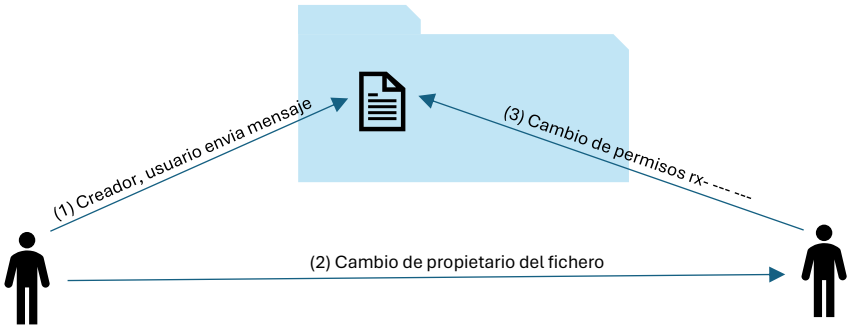
Por último, encontramos dos archivos “.c”, estos son los encargados de almacenar las funciones en sí. Por un lado, encontramos “grupos.c” en la cual se encuentran todas las funciones de gestión de grupos, como crear uno, añadir o eliminar miembros... Por otro lado, encontramos “mensajes.c” en este fichero se encuentra todo lo referido a la gestión de mensajes y a la activación y desactivación del programa.

Esta estructura modular proporciona una organización clara y facilita el mantenimiento y la escalabilidad del sistema a medida que se añaden nuevas funcionalidades.

Carlota Gutiérrez Carabias
Alvaro González Nava
Hugo Revuelta Aja
Dario Prieto Soria

SEGURIDAD

La parte de seguridad sin duda ha sido la más costosa de desarrollar y de lograr llegar a una conclusión favorable. La idea final a la que se ha llegado es otorgar los permisos necesarios a los ficheros de los mensajes. Esto se realiza al enviar el mensaje, cuando se ejecuta la función el usuario que la ejecuta crea un archivo en el que se guarda la información relevante. Una vez ha sido almacenada dicha información se ejecuta un cambio de propietario siendo ahora el usuario remitente el dueño del fichero. Por ultimo se le quitan todos los permisos al fichero a todos los usuarios menos al dueño del mismo.



Conforme íbamos barajando otras opciones tuvimos bastantes conflictos con a quien darles los permisos. Por ejemplo, queríamos que los permisos se gestionaran a partir de las carpetas de los usuarios, queríamos otorgar el permiso de lectura al dueño y el de escritura a los demás. Tras esto nos dimos cuenta que existía un conflicto con lo ideado para los permisos y para el envío de mensajes de grupo, por lo que se deseó cambiar a permiso de lectura y escritura para el propietario y de escritura para el resto. Sin embargo, existían varios conflictos a la hora de crear los ficheros desde los distintos usuarios. Por ese motivo se decidió únicamente modificar los permisos de los ficheros que actúan de almacenamiento de mensajes.

Aun así, se ha mantenido el cambio de dueño a la hora de crear los ficheros de los usuarios. Para de esta manera garantizar que los usuarios puedan entrar en sus propios directorios. Además, si finalmente retomábamos la idea inicial esto facilitaría el proceso de pruebas.

Carlota Gutiérrez Carabias
Alvaro González Nava
Hugo Revuelta Aja
Dario Prieto Soria

PROBLEMAS DE IMPLEMENTACION

Apesar de las ideas planteadas en la etapa del planificacion del proyecto muchas de ellas no se han podido llevar acabo por problemas transcurridos durante la implementación.

Por ejmplo ya hemos mencionado anteiormente las dificultades encontradas a la hora de implementar la mejor seguridad viable. La idea principal de que los ficheros fueran los que tuvieran la barrera de escritura o lectura no fue posible. Y no solo eso, en la ultima versión de nuestro programa el unico usuario capaz de enviar mensajes es el root, puesto que el resto no tiene el permiso necesario para crear archivos dentro de las distintas carpetas. Este erro ha sido uno de los más trabajados puesto a que la decisión final fue que todos lo usuarios tuvieran todos los permisos sobre las los diferentes directorios y aun así el error de permiso sigue existiendo en la version entregada.

Al solo poder el root enviar mensajes el comando “concordia-responder” no fue implementado y por tanto todo lo que conlleva. Esto quiere decir que en el mensaje no se guarda ninguna informacion acerca de si se trata de una respuesta a otro o no. La idea principal era que el formato del mensaje cambiara si esto llegara a funcionar. El nuevo formamo tendria que recoger:

- Información sobre si el mensaje fue leído (si/no).
- Fecha y hora a la que se envió el mensaje.
- Emisor del mensaje.
- Mensaje al que responde
- Contenido del mensaje

De esta manera se podría guardar un registro de los mensajes que han sido respondidos. En nuestra idea los mensajes de grupo se responderían al usuario que mando el primer mensaje para garantizar la privacidad del contenido del mismo, ya que si quisiera enviar un mensaje a todo el grupo ya vendría dado por el comando “concordia-enviar”.

Una vez implementada la parte de gestión de grupos nos dimos cuenta de un problema bastante importante. Los usuarios a la hora de arrancar cada uno su programa este se realiza de manera que cada uno tiene su propio proceso con su propia información sobre las variables. Esto impedía que todos los usuarios fueran conocedores de los grupos creados por otros usuarios. Por lo que solo podían mandar mensajes a los grupos creados por ellos mismo. Sin embargo, nosotros queríamos que cualquiera pudiera enviar un mensaje a cualquier grupo.

Una vez identificado el problema se barajaron distintas soluciones como el de incluirlo a la hora de activar y que este fuera un paso necesario para todos los usuarios. Pero

existía el problema de que las carpetas de usuario solo se necesitan crear una vez. Por ello la última solución y la implementada en la versión enviada fue la de crear una función “actualizar()” en ella se comprueban una a una las diferentes carpetas de los grupos identificadas como “MG_G + nombre_grupo”. Entra en el fichero de información para comprobar si hay grupos que no son conocidos por el usuario que ejecuta el programa o si estos han sido modificados. Por eso la función se rebe realizar después de invocar cualquier comando.

Carlota Gutiérrez Carabias
Alvaro González Nava
Hugo Revuelta Aja
Dario Prieto Soria

MANUAL DE USO

Todo el programa se basa en comandos ejecutados en el terminar. Por este motivo cada usuario podrá ejecutar el programa desde su propia terminal. Al arrancarlo aparecerá el terminal de la aplicación el cual consiste en la palabra identificadora “concorida-” y a partir de ahí se escribe el comando deseado.

Lo primero que se debe hacer es ejecutar el comando “ativar”, cuya funcionalidad consiste en generar la estructura necesaria durante la ejecución del programa. Esta estructura consiste en los directorios de usuarios y el almacenamiento de los datos que no será visto por el usuario.

Por otro lado tenemos la función “desativar” la cual en resumen sería lo contrario a la primera mencionada. Se encarga de eliminar toda la estructura y los ficheros que se han ido generando durante la ejecución del programa.

Otro comando bastante relevante que aporta nuestro programa es el de “help”. Este ha sido creado con la finalidad de que la experiencia sea más fácil para el usuario. Si se ejecuta en la pantalla se mostrará el nombre de todos los comandos y una breve explicación de lo que hacen.

Tras esto todos los demás comandos podríamos dividirlos en dos grupos fijándonos en su funcionalidad.

Por un lado, encontramos los comandos destinados a la gestión de mensajes:

- concordia-enviar: tras ejecutarlo solicitará el usuario o grupo al que se quiera enviar el mensaje y el contenido del mismo.
- concordia-listar: este comando sirve para listar todos los mensajes que no han sido leidos por el usuario que lo ejecuta. Si al final del comando se le añade “-a” se mostraran todos los mensajes, leidos y no leidos.
- concordia-ler: si un usuario desea leer un mensaje en concreto debera ejecutar dicho comando. Este le solicitara el nombre completo del mensaje que desea leer.
- concordia-remover: la función del comando es la de eliminar un mensaje en concreto, para ello la aplicación solicitará el nombre completo del mensaj que desea eliminar.

Por otro lado, encontramos los comandos de gestión de grupos:

- concordia-grupo-criar: para crear un grupo, el usuario que ejecuta la instrucción será el creador del grupo. En esta no se especifican los miembros solo el nombre del nuevo grupo.
- concordia-grupo-remover: sirve para eliminar un grupo en concreto por lo que se le solicita el nombre. Sin embargo solo el creador tiene el poder de eliminar un grupo.
- concordia-grupo-listar: el comando se encarga de listar los miembros de un grupo en especifico.
- concordia-grupo-destinario-adicionar: sirve para añadir un nuevo miembro a tu grupo. Esta función solo la puede hacer el creador.
- concordia-grupo-destinario-remover: elimina un miembro en especifico del grupo. Solo el creador puede hacerlo.

Carlota Gutiérrez Carabias
Alvaro González Nava
Hugo Revuelta Aja
Dario Prieto Soria

CONCLUSION

Lo que podemos sacar en conclusión tras todo lo comentado y nuestra experiencia personal a la hora de ejecutar el proyecto al completo es que el tema de seguridad es la parte más incompleta del proyecto. A pesar de nuestros esfuerzos por construir un software seguro aún existen aspectos a mejorar con el tema de los permisos.

Sin embargo, podemos garantizar que el proyecto esta lo más completo posible abarcando casi todos los aspectos que se necesitaban. Ya que se han implementado prácticamente todos los comandos solicitados y se han añadido nuevos para la mejor experiencia del usuario.

Además, esta modulado y estructurado de una manera que sea fácilmente comprensible y legible para alguien externo. Esto es un gran punto a favor no solo para la revisión de alguien externo si no para la organización y la actualización de las diferentes versiones que se han ido dando durante su elaboración.

En resumen, se ha intentado realizar lo mejor posible y se ha conseguido la gran parte de los objetivos básicos que se plantearon en el inicio del proyecto.