 Proyecto Cupi2	ISIS-1204 Algorítmica y Programación Descripción
Ejercicio:	n12_cupiEmail
Autor:	Equipo Cupi2
Semestre:	2016-1

Enunciado

El objetivo de este ejercicio es modelar CupiEmail, un sistema de correo electrónico donde los usuarios pueden enviar mensajes utilizando sus cuentas de correo registradas en CupiEmail. Los usuarios de CupiEmail únicamente pueden recibir y enviar correos a otros usuario de CupiEmail.

El ejercicio está compuesto de dos aplicaciones que se comunican entre sí usando “sockets” y la persistencia realiza sobre una base de datos relacional. Estas dos aplicaciones son:

1. El servidor CupiEmail, encargado de manejar la información de los usuarios y sus correos en la base de datos.
2. El cliente, que le permite a los usuarios leer y enviar correos.

El servidor CupiEmail es una aplicación que se encarga de: (1) consultar todos los usuarios registrados en el sistema, (2) consultar los usuarios conectados, (3) mostrar la información detallada de un usuario incluyendo número total de mensajes y número total de mensajes de sin leer, y (4) atender las solicitudes de los clientes.

El cliente de cupi-Email puede: (1) crear una cuenta, (2) iniciar sesión, (3) enviar un correo electrónico, y (4) consultar los mensajes que le han enviado.

Persistencia

Para el desarrollo de la aplicación CupiEmail, se utilizará como motor de base de datos el sistema (open source) llamado Derby, desarrollado en el proyecto Apache Derby.

Las ventajas de utilizar un motor de base de datos Derby son:

- Está desarrollado completamente en Java y tiene un sistema de persistencia basado en archivos.
- Permite crear un servidor Derby independiente de la aplicación y establecer la comunicación con esta a través de sockets y desde diferentes máquinas.
- Es una base de datos embebida, es decir, que se puede utilizar como si fuera parte de la aplicación.

La base de datos de la aplicación tiene dos tablas:

- La primera tabla es utilizada para almacenar los datos de los usuarios del sistema. La tabla incluye los siguientes campos: login del usuario, nombre del usuario, apellidos del usuario, contraseña del usuario, total de correos del usuario, total de correos sin leer del usuario, estado de conexión del usuario. La llave primaria de esta tabla es el login de usuario. Esto quiere decir que no pueden existir dos usuarios con el mismo login.
- La segunda tabla es utilizada para almacenar la información de los correos del sistema. La tabla



incluye los siguientes campos: login del usuario remitente, login del usuario destinatario, fecha de envío del correo, asunto del correo, mensaje (contenido del correo), estado de lectura del correo. La llave primaria de esta tabla es una llave compuesta por el login del destinatario, la fecha de envío y el asunto del correo.

La Tabla 1 muestra el diseño de las dos tablas de la base de datos con sus respectivos campos y llaves primarias.

Tabla	Campos	Tipo	Llave primaria
Usuarios	login	varchar(32)	login
	nombres	varchar(50)	
	apellidos	varchar(50)	
	contrasena	varchar(32)	
	conectado	varchar(1)	
Correos	login_remitente	varchar(32)	login_destinatario, fecha_envio, asunto
	login_destinatario	varchar(32)	
	fecha_envio	varchar(20)	
	Asunto	varchar(140)	
	Mensaje	varchar(512)	
	Leído	varchar(1)	

Tabla 1. Diseño de las tablas para la base de datos.

Protocolo de comunicación

A continuación se presenta el protocolo de comunicación que establece cuáles mensajes (y en qué orden) se deberán enviar para realizar cada una de las tareas del sistema. Es responsabilidad tanto del servidor como de los clientes ser capaz de interpretar este protocolo y realizar las tareas necesarias.

En términos generales, los mensajes que se envían tienen el siguiente formato:

<COMANDO>;;<PARÁMETRO1>;;<PARÁMETRO2>; ... <PARÁMETROn>

En dónde:

- <COMANDO>: Indica el tipo de solicitud del usuario al servidor o la respuesta del servidor al usuario.
- <PARAMETRO>: Indica la información necesaria para resolver la petición.
- Para separar el comando de los parámetros, se utiliza el separador “;;”.
- Para separar los parámetros, se utiliza el separador “;”.

Protocolo para iniciar sesión:

Caso 1: El usuario inicia sesión correctamente en el servidor:



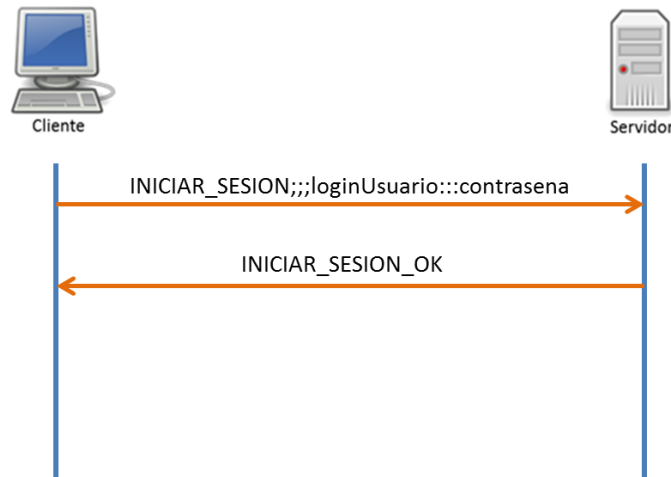


Ilustración 1. Protocolo de inicio de sesión.

En la Ilustración 1 se aprecia el protocolo de inicio de sesión. Este consiste en que el cliente envía un mensaje al servidor con el comando INICIAR_SESION, seguido del login y la contraseña del usuario que desea iniciar sesión. Si el servidor encuentra un usuario con el login y la contraseña correspondiente en la base de datos, entonces envía un mensaje de respuesta con el comando INICIAR_SESION_OK. El servidor guarda una referencia del usuario que inició sesión.

Las Tablas 2 y 3 explican la sintaxis de cada mensaje:

Comando Cliente	Parámetros	Descripción
INICIAR_SESION	loginUsuario	Representa el login del usuario que quiere iniciar sesión.
	contrasena	Representa la contraseña del usuario que quiere iniciar.

Tabla 2. Sintaxis del mensaje del cliente para el protocolo de inicio de sesión.

Comando Servidor	Parámetros	Descripción
INICIAR_SESION_OK	Ninguno	Indica que inicia sesión sin problemas.

Tabla 3. Sintaxis del mensaje del servidor para el protocolo de inicio de sesión.

A continuación, un ejemplo concreto:

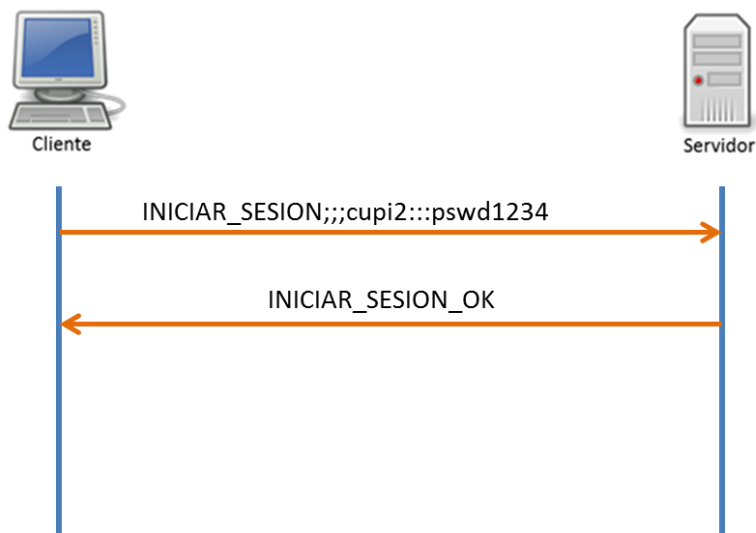


Ilustración 2. Ejemplo del protocolo de inicio de sesión.

En la Ilustración 2 se muestra el escenario en el que el usuario cupi2 desea iniciar sesión. El cliente envía una petición de inicio de sesión con el comando INICIAR_SESION, con login cupi2 y contraseña 1234. El servidor busca en su base de datos un usuario con este login. Al encontrar que el usuario y la contraseña corresponden, le responde al usuario enviando el mensaje con el comando INICIAR_SESION_OK.

Caso 2: Un usuario no puede iniciar sesión en el servidor:

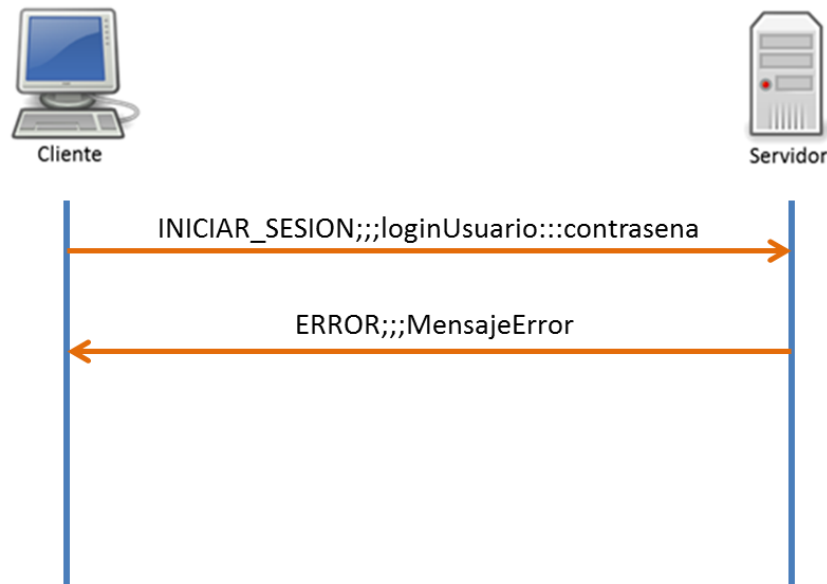


Ilustración 3. Protocolo de inicio de sesión cuando ocurre un error.

La Ilustración 3 muestra el protocolo de inicio de sesión en presencia de un error. Esto ocurre cuando el cliente envía un mensaje al servidor con el comando INICIAR_SESION junto con login y contraseña pero el servidor no encuentra en su base de datos un usuario con el login dado, o la contraseña no corresponde a la del usuario. Por lo tanto, le envía al programa cliente un mensaje con el comando ERROR y el mensaje de error. Las Tablas 4 y 5 explican la sintaxis de cada mensaje:

Comando Cliente	Parámetros	Descripción
INICIAR_SESION	loginUsuario	Representa el login del usuario que desea iniciar sesión.
	contrasena	Representa la contraseña del usuario que desea iniciar sesión.

Tabla 2. Sintaxis del mensaje del cliente para el protocolo de inicio de sesión.

Comando Servidor	Parámetros	Descripción
ERROR	MensajeError	Representa el mensaje de error que se quiere informar al usuario.

Tabla 3. Sintaxis del mensaje del servidor para el protocolo de inicio de sesión (caso de error).

A continuación, un ejemplo concreto:



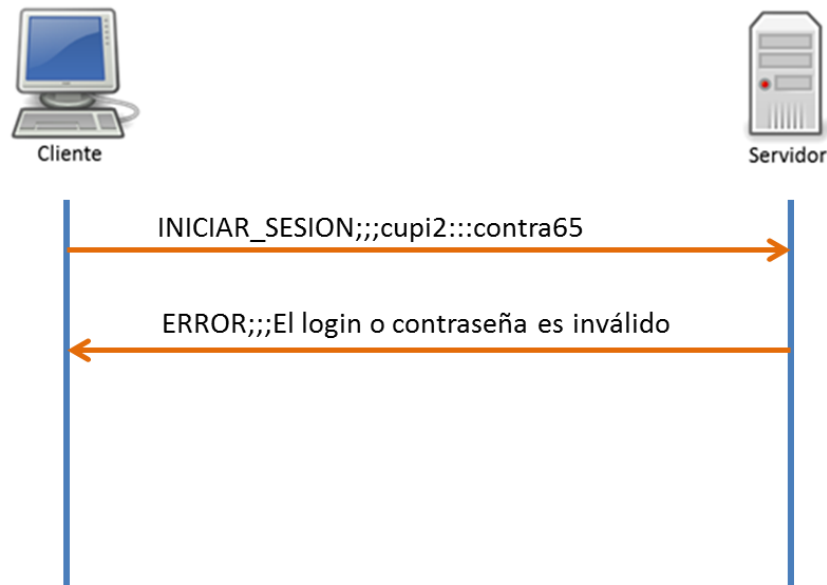


Ilustración 4. Ejemplo del protocolo de inicio de sesión cuando ocurre un error

La Ilustración 4 muestra el siguiente ejemplo: el cliente envía una petición de inicio de sesión con el comando INICIAR_SESION, con login cupi2 y contraseña contra65. El servidor no encuentra un usuario con este login y contraseña, y por lo tanto envía el mensaje de error con comando: ERROR y el mensaje “El login o contraseña es inválido”.

Protocolo para crear la cuenta de un usuario:

Caso 1: El usuario crea la cuenta exitosamente.

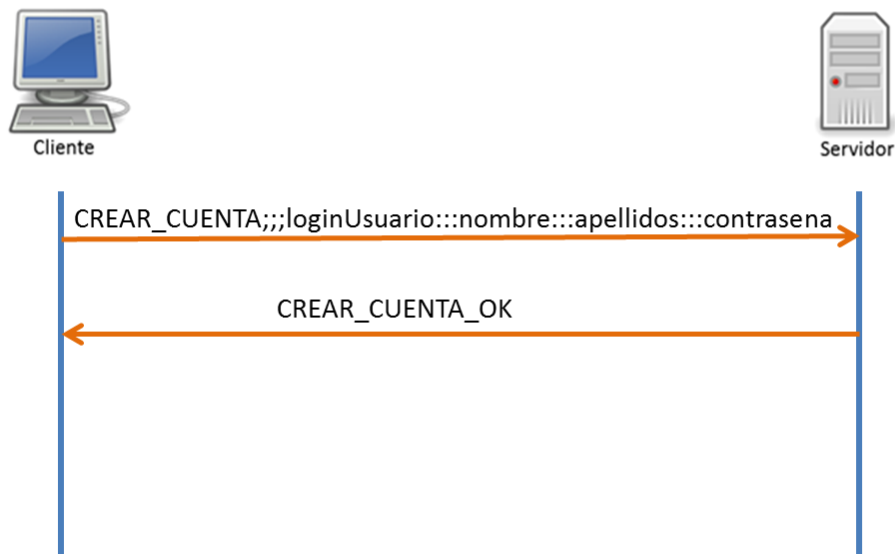


Ilustración 5. Protocolo para crear una cuenta de usuario.

En la Ilustración 5 se aprecia el protocolo de crear cuenta que consiste en que el programa cliente envía un mensaje al servidor con el comando CREAR_CUENTA, seguido del login, el nombre, los apellidos y la contraseña del usuario que desea crear la cuenta. Si el servidor encuentra que no existe ningún usuario con ese login en la base de datos, entonces agrega el nuevo usuario a la base de dato y envía un mensaje de respuesta con el comando CREAR_CUENTA_OK. El servidor guarda una referencia del usuario que creó la



cuenta.

Las Tablas 6 y 7 explican la sintaxis de cada mensaje:

Comando Cliente	Parámetros	Descripción
CREAR_CUENTA	loginUsuario	Representa el login del usuario que quiere crear la cuenta.
	nombre	Representa el nombre del usuario que quiere crear la cuenta.
	apellidos	Representa los apellidos del usuario que quiere crear la cuenta.
	contrasena	Representa la contraseña del usuario que quiere crear la cuenta.

Tabla 6. Sintaxis del mensaje del cliente para el protocolo de crear cuenta.

Comando Servidor	Parámetros	Descripción
CREAR_CUENTA_OK	Ninguno	Indica que se creó la cuenta sin problemas.

Tabla 7. Sintaxis del mensaje del servidor para el protocolo de crear cuenta.

A continuación, un ejemplo concreto:

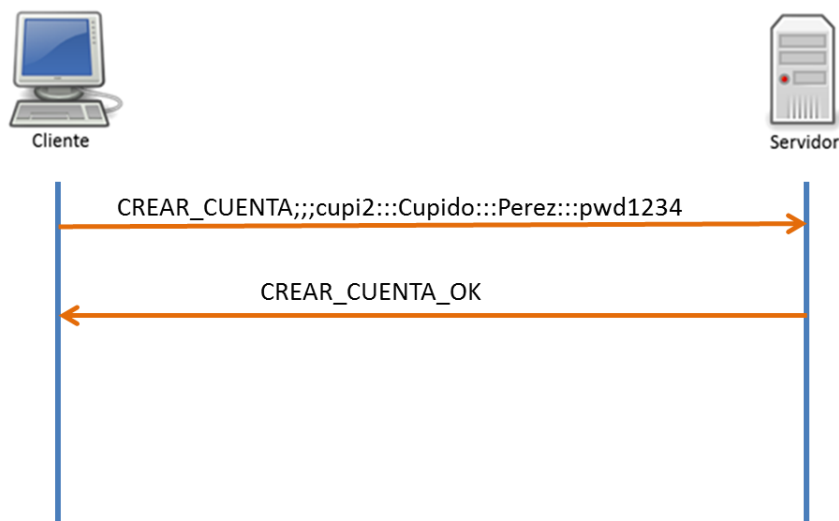


Ilustración 6. Ejemplo de protocolo para crear una cuenta de usuario.

La Ilustración 6 muestra el siguiente ejemplo: el usuario envía una petición de crear cuenta con el comando **CREAR_CUENTA**, con login **cupi2**, nombre **Cupido**, apellidos **Perez** y contraseña **pwd1234**. El servidor no encuentra un usuario con este login, por lo tanto envía el comando **CREAR_CUENTA_OK**.

Caso 2: Ya existe un usuario con el login que se desea crear y no se puede crear la cuenta.





Ilustración 7. Protocolo para crear una cuenta de usuario cuando ocurre un error.

En la Ilustración 7 se aprecia el protocolo de crear cuenta cuando ocurre un error que consiste en que el programa cliente envía un mensaje al servidor con el comando **CREAR_CUENTA** seguido del login, nombre, apellidos y contraseña del usuario que desea crear la cuenta. El servidor encuentra que ya existe un usuario con ese login en la base de datos, entonces envía un mensaje de respuesta con el comando **ERROR**, seguido de un mensaje de error.

Las Tablas 8 y 9 explican la sintaxis de cada mensaje:

Comando Cliente	Parámetros	Descripción
CREAR_CUENTA	loginUsuario	Representa el login del usuario que quiere crear la cuenta.
	nombre	Representa el nombre del usuario que quiere crear la cuenta.
	apellidos	Representa los apellidos del usuario que quiere crear la cuenta.
	contrasena	Representa la contraseña del usuario que quiere crear la cuenta.

Tabla 8. Sintaxis del mensaje del cliente para el protocolo de crear cuenta.

Comando Servidor	Parámetros	Descripción
ERROR	MensajeError	Mensaje que indica el error al crear la cuenta

Tabla 9. Sintaxis del mensaje del servidor para el protocolo de crear cuenta cuando ocurre un error.

A continuación, un ejemplo concreto:

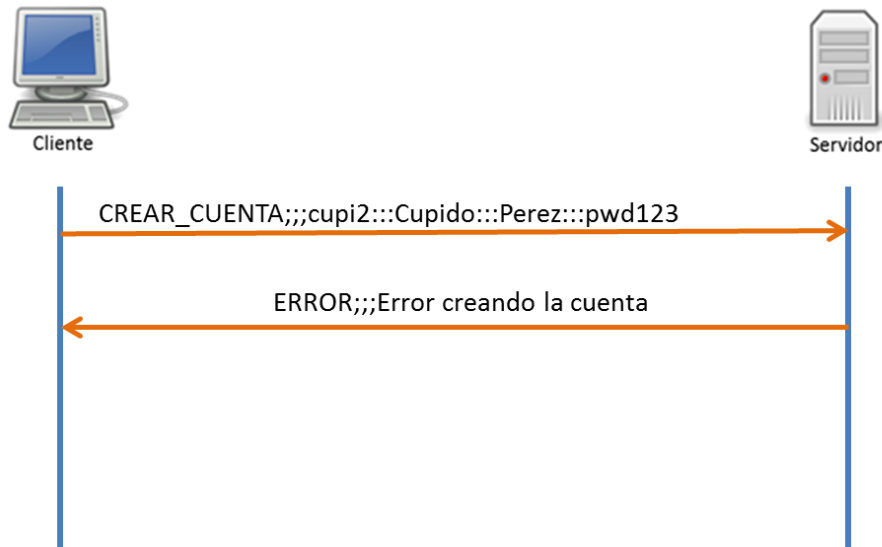


Ilustración 8. Ejemplo de protocolo para crear una cuenta de usuario cuando ocurre un error.

La Ilustración 8 muestra el siguiente ejemplo: el usuario envía una petición de crear cuenta con el comando CREAM_CUENTA, con login cupi2, nombre Cupido, apellidos Perez y contraseña pwd123. El servidor encuentra un usuario con este login, y por lo tanto envía el comando ERROR con el mensaje: Error creando la cuenta.

Protocolo para cerrar sesión:

Caso 1: El usuario puede cerrar sesión sin problemas.

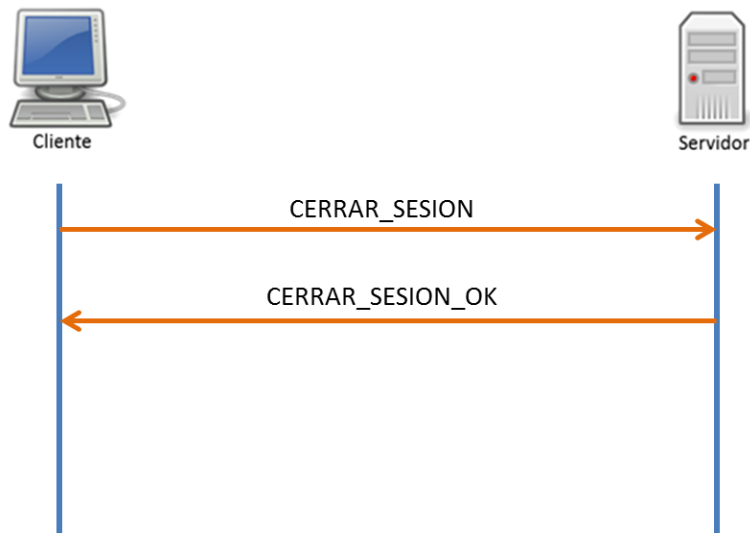


Ilustración 9. Protocolo para cerrar sesión de usuario.

En la Ilustración 9 se aprecia el protocolo de cerrar sesión que consiste en que el programa cliente envía un mensaje al servidor con el comando CERRAR_SESION. El servidor logra cerrar la sesión exitosamente y envía un mensaje de respuesta con el comando CERRAR_SESION_OK.

Las Tablas 10 y 11 explican la sintaxis de cada mensaje:



Comando Cliente	Parámetros	Descripción
CERRAR_SESION	Ninguno	Indica que se quiere cerrar la sesión.

Tabla 10. Sintaxis del mensaje del cliente para el protocolo de cerrar sesión.

Comando Servidor	Parámetros	Descripción
CREAR_CUENTA_OK	Ninguno	Indica que se cerró sesión sin problemas.

Tabla 11. Sintaxis del mensaje del servidor para el protocolo de cerrar sesión.

A continuación, un ejemplo concreto:

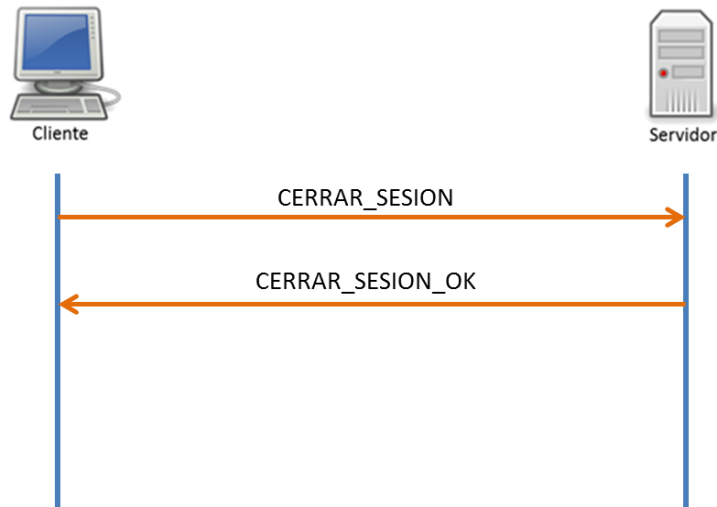


Ilustración 10. Ejemplo de protocolo para cerrar sesión de usuario.

La Ilustración 10 muestra el siguiente ejemplo: el usuario envía una petición de cerrar sesión con el comando CERRAR_SESION. El servidor no tiene problemas al cerrar sesión, y por lo tanto envía el comando CERRAR_SESION_OK.

Caso 2: Se presenta un error al intentar cerrar la sesión.



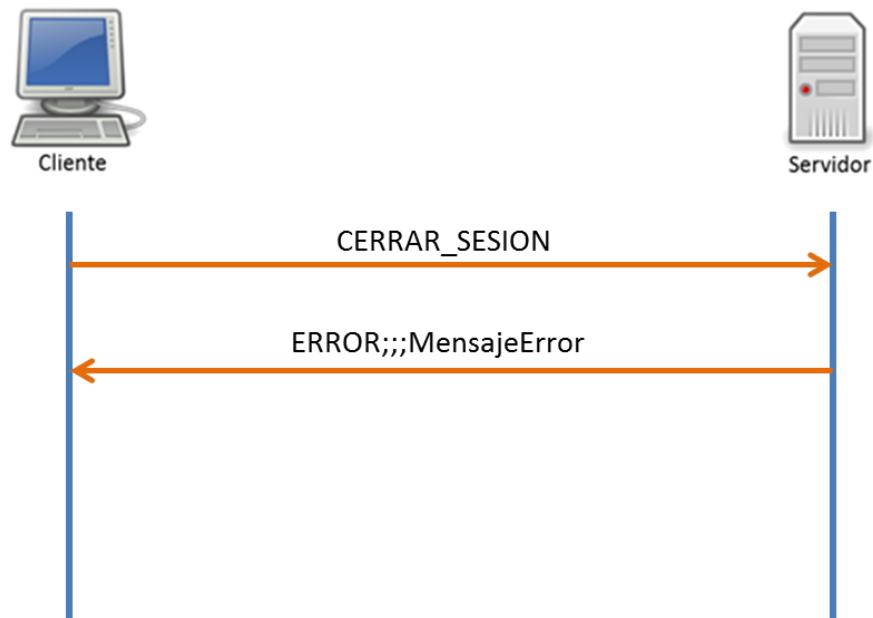


Ilustración 11. Protocolo para cerrar sesión de usuario cuando ocurre un error.

En la Ilustración 11 se aprecia el protocolo de cerrar sesión cuando ocurre un error que consiste en que el programa cliente envía un mensaje al servidor con el comando CERRAR_SESION. El servidor encuentra algún problema para cambiar el estado del usuario con ese login en la base de datos, entonces envía un mensaje de respuesta con el comando ERROR, seguido de un mensaje de error.

Las Tablas 12 y 13 explican la sintaxis de cada mensaje:

Comando Cliente	Parámetros	Descripción
CERRAR_SESION	Ninguno	Indica que el usuario desea cerrar sesión.

Tabla 12. Sintaxis del mensaje del cliente para el protocolo de cerrar sesión.

Comando Servidor	Parámetros	Descripción
ERROR	MensajeError	Mensaje que indica el error al cerrar sesión.

Tabla 13. Sintaxis del mensaje del servidor para el protocolo de cerrar sesión cuando ocurre un error.

A continuación, un ejemplo concreto:



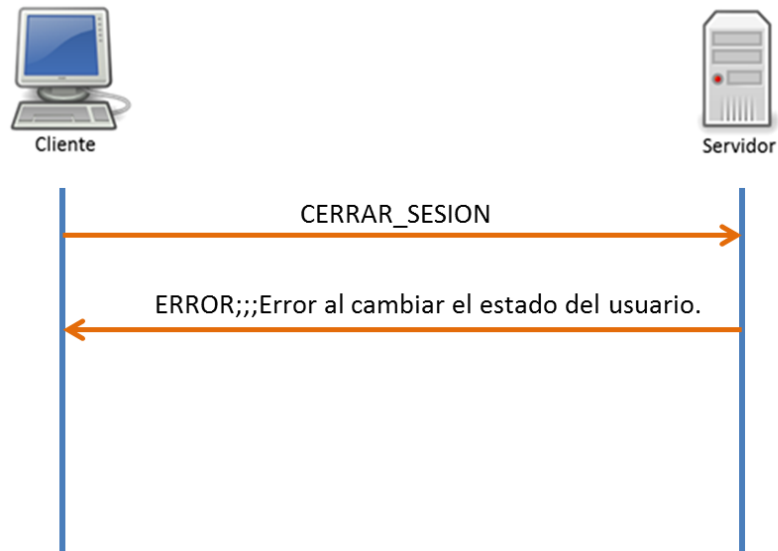


Ilustración 12. Ejemplo de protocolo para cerrar sesión de usuario cuando ocurre un error.

La Ilustración 12 muestra el siguiente ejemplo: el usuario envía una petición de cerrar sesión con el comando CERRAR_SESION. El servidor encuentra problemas al cerrar la sesión del usuario, y por lo tanto envía el comando ERROR con el mensaje: Error al cambiar el estado del usuario.

Protocolo para marcar un correo como leído:

Caso 1: El correo se puede marcar como leído.

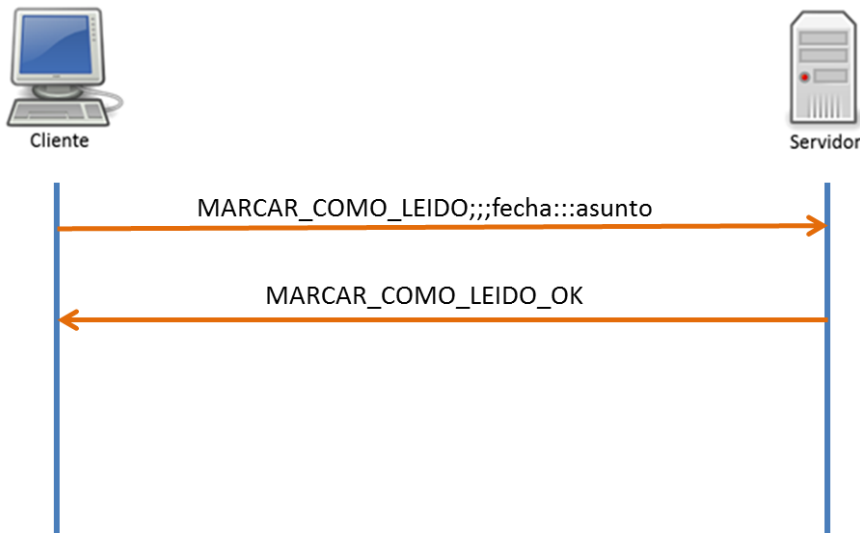


Ilustración 13. Protocolo para marcar un correo como leído

En la Ilustración 13 se aprecia el protocolo de marcar un correo como leído, que consiste en que el programa cliente envía un mensaje al servidor con el comando MARCAR_COMO_LEIDO seguido de la fecha del correo y el asunto del correo. Si el servidor encuentra que existe algún correo con esa fecha y login en la base de datos, entonces envía un mensaje de respuesta con el comando MARCAR_COMO_LEIDO_OK. El servidor guarda cambia el estado del correo a leído.

Las Tablas 14 y 15 explican la sintaxis de cada mensaje:

Comando Cliente	Parámetros	Descripción
MARCAR_COMO_LEIDO	fecha	Representa la fecha de envío del correo.
	asunto	Representa el asunto del correo.

Tabla 14. Sintaxis del mensaje del cliente para el protocolo de marcar un correo como leído.

Comando Servidor	Parámetros	Descripción
MARCAR_COMO_LEIDO_OK	Ninguno	Indica que el correo se marcó como leído sin problemas.

Tabla 15. Sintaxis del mensaje del servidor para el protocolo de marcar un correo como leído.

A continuación, un ejemplo concreto:

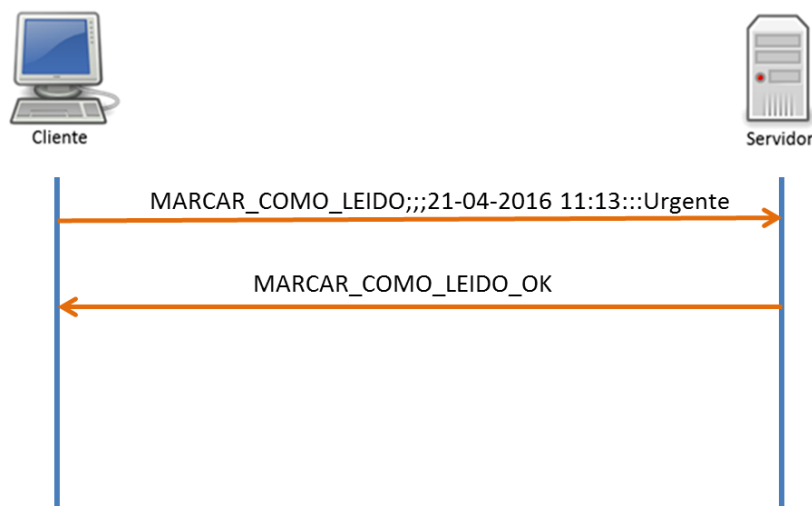


Ilustración 14. Ejemplo de protocolo para marcar un correo como leído.

La Ilustración 14 muestra el siguiente ejemplo: el usuario envía una petición de crear cuenta con el comando `MARCAR_COMO_LEIDO`, con fecha 21-04-2016 11:13 y asunto “Urgente”. El servidor encuentra el correo con esa fecha y asunto por lo tanto envía el comando `MARCAR_COMO_LEIDO_OK`.

Caso 2: El correo no se puede marcar como leído.



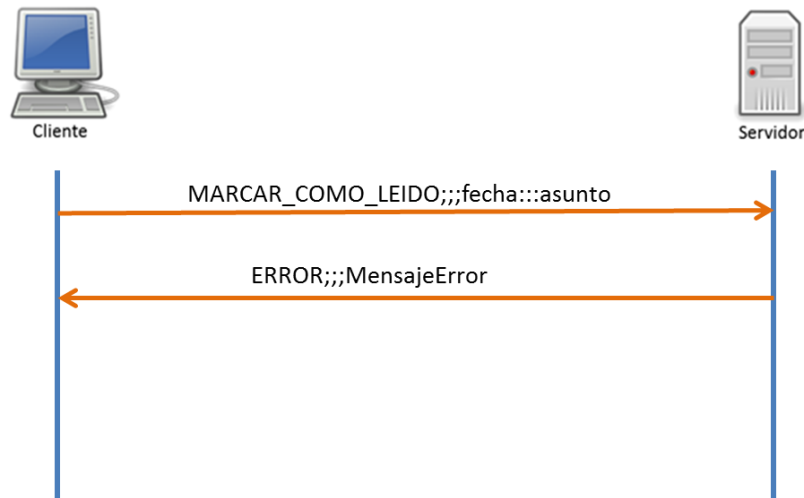


Ilustración 15. Protocolo para marcar un correo como leído cuando ocurre un error.

En la Ilustración 15 se aprecia el protocolo de marcar un correo como leído cuando ocurre un error que consiste en que el programa cliente envía un mensaje al servidor con el comando `MARCAR_COMO_LEIDO`, seguido de la fecha y el asunto del correo. El servidor encuentra que no existe correo con ese asunto y fecha de envío en la base de datos, entonces envía un mensaje de respuesta con el comando `ERROR`, seguido de un mensaje de error.

Las Tablas 16 y 17 explican la sintaxis de cada mensaje:

Comando Cliente	Parámetros	Descripción
MARCAR_COMO_LEIDO	fecha	Representa la fecha de envío del correo.
	asunto	Representa el asunto del correo.

Tabla 16. Sintaxis del mensaje del cliente para el protocolo de marcar un correo como leído.

Comando Servidor	Parámetros	Descripción
ERROR	MensajeError	Mensaje que indica el error al marcar como leído el correo.

Tabla 17. Sintaxis del mensaje del servidor para el protocolo de marcar un correo como leído cuando ocurre un error.

A continuación, un ejemplo concreto:

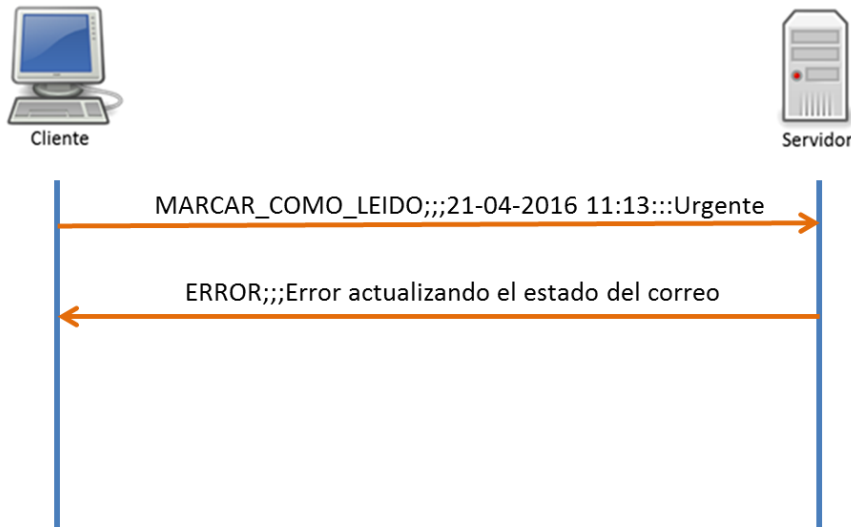


Ilustración 16. Ejemplo de protocolo para marcar un correo como leído cuando ocurre un error.

La Ilustración 16 muestra el siguiente ejemplo: el usuario envía una petición de marcar un correo como leído con el comando `MARCAR_COMO_LEIDO`, con fecha 21-04-2016 11:13 y asunto Urgente. El servidor encuentra un usuario con este login, y por lo tanto envía el comando `ERROR` con el mensaje: Error actualizando el estado del correo.

Protocolo para consultar los correos:

Caso 1: No hay problemas al consultar los correos del usuario.

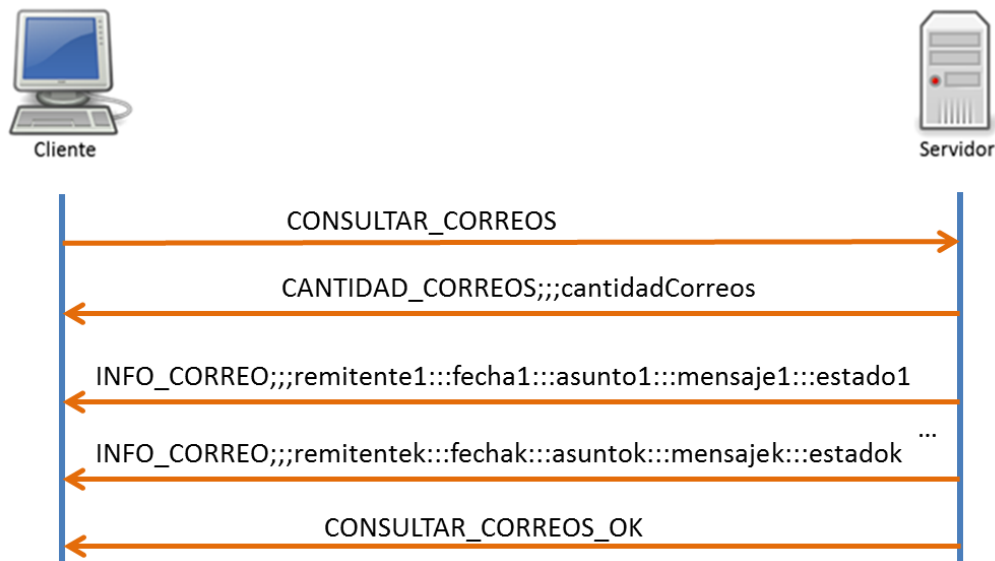


Ilustración 17. Protocolo para consultar los correos del usuario.

En la Ilustración 17 se aprecia el protocolo de consultar correos que consiste en que el programa cliente envía un mensaje al servidor con el comando `CONSULTAR_CORREOS`. Si el servidor no tiene problemas al consultar los correos del usuario en la base de datos, entonces envía un mensaje de respuesta con el comando `CANTIDAD_CORREOS`, seguido de la cantidad de correos del usuario. Por cada correo, el servidor envía un mensaje con el comando `INFO_CORREO`, seguido del remitente, fecha de envío, asunto del correo, mensaje



del correo y estado del correo (leído o no).

Las Tablas 18 y 19 explican la sintaxis de cada mensaje:

Comando Cliente	Parámetros	Descripción
CONSULTAR_CORREOS	Ninguno	Indica que el usuario quiere consultar sus correos.

Tabla 18. Sintaxis del mensaje del cliente para el protocolo de consultar los correos.

Comando Servidor	Parámetros	Descripción
CANTIDAD_CORREOS	Cantidad	Cantidad de correos del usuario
INFO_CORREO	Remitente	Remitente del correo.
	Fecha	Fecha de envío del correo
	Asunto	Asunto del correo.
	Mensaje	Mensaje (contenido) del correo.
	Estado	Estado del correo (leído o no).
CONSULTAR_CORREOS_OK	Ninguno	Indica que se consultaron los correos sin problema.

Tabla 19. Sintaxis del mensaje del servidor para el protocolo de consultar los correos.

A continuación, un ejemplo concreto:

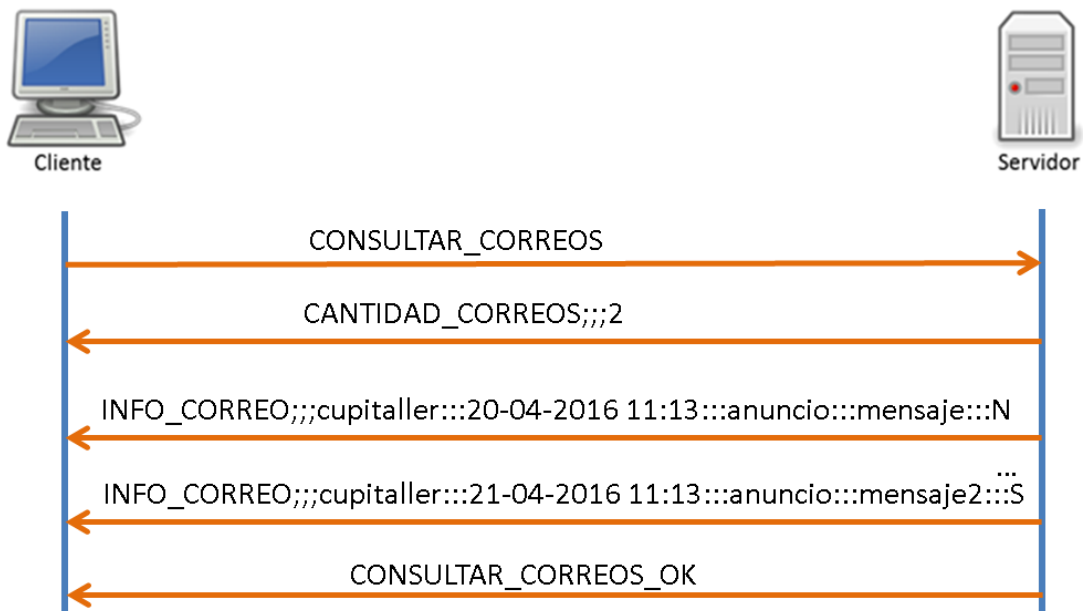


Ilustración 18. Ejemplo de protocolo para consultar los correos del usuario.

La Ilustración 18 muestra el siguiente ejemplo: el usuario envía una petición de consultar los correos con el comando CONSULTAR_CORREOS. El servidor responde con el comando CANTIDAD_CORREOS, seguido de la cantidad 2. Después envía 2 mensajes con el comando INFO_CORREO, uno por cada correo: 1) remitente cupitaller, fecha 20-04-2016 11:13, asunto “anuncio”, mensaje “mensaje”, estado no leído “N”, 2) remitente cupitaller, fecha 21-04-2016 11:13, asunto “anuncio”, mensaje “mensaje2”, estado leído “S”. Por último, el servidor envía el mensaje de CONSULTAR_CORREOS_OK para indicar que la consulta fue exitosa.



Caso 2: Hay problemas al consultar los correos del usuario.

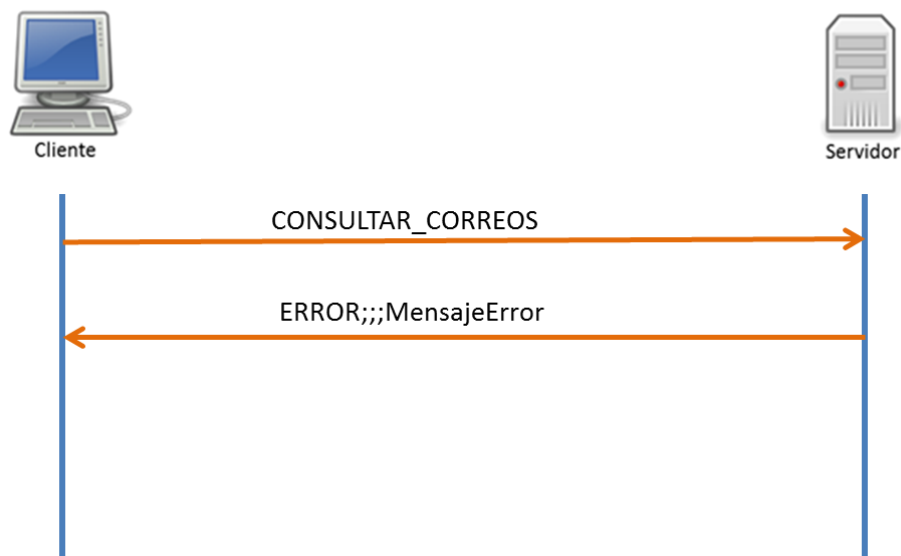


Ilustración 19. Protocolo para crear una cuenta de usuario cuando ocurre un error.

En la Ilustración 19 se aprecia el protocolo de consultar correos cuando ocurre un error. Consiste en que el programa cliente envía un mensaje al servidor con el comando CONSULTAR_CORREOS. El servidor tiene problemas en consultar los correos del usuario en la base de datos, entonces envía un mensaje de respuesta con el comando ERROR, seguido de un mensaje de error.

Las Tablas 20 y 21 explican la sintaxis de cada mensaje:

Comando Cliente	Parámetros	Descripción
CONSULTAR_CORREOS	Ninguno	Indica que el usuario quiere consultar sus correos.

Tabla 20. Sintaxis del mensaje del cliente para el protocolo de consultar correos.

Comando Servidor	Parámetros	Descripción
ERROR	MensajeError	Mensaje que indica el error al consultar los correos.

Tabla 21. Sintaxis del mensaje del servidor para el protocolo de consultar correos cuando ocurre un error.

A continuación, un ejemplo concreto:



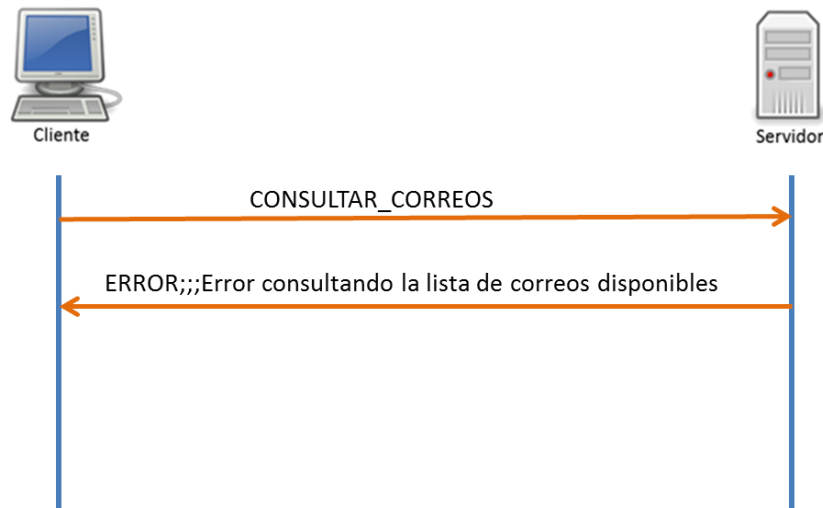


Ilustración 20. Ejemplo de protocolo para consultar correos del usuario cuando ocurre un error.

La Ilustración 20 muestra el siguiente ejemplo: el usuario envía una petición de consultar correos con el comando CONSULTAR_CORREOS. El servidor encuentra un problema al consultar los correos del usuario, y por lo tanto envía el comando ERROR con el mensaje: Error consultando la lista de correos disponibles.

Protocolo para enviar un correo:

Caso 1: El correo se puede enviar sin problemas.

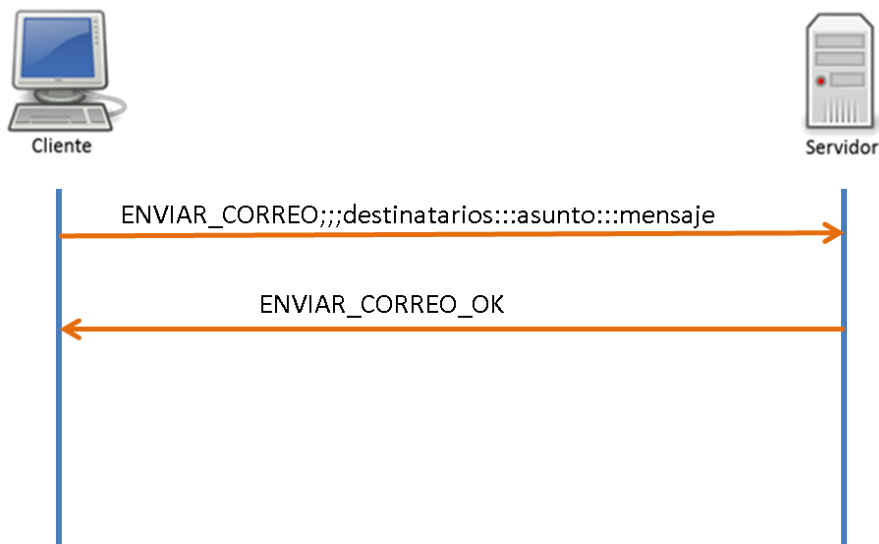


Ilustración 21. Protocolo para enviar un correo.

En la Ilustración 21 se aprecia el protocolo de enviar un correo, que consiste en que el programa cliente envía un mensaje al servidor con el comando ENVIAR_CORREO, seguido del destinatario, asunto y mensaje. Si el servidor no encuentra problemas al registrar el nuevo correo en la base de datos, entonces envía un mensaje de respuesta con el comando ENVIAR_CORREO_OK.

Las Tablas 22 y 23 explican la sintaxis de cada mensaje:



Comando Cliente	Parámetros	Descripción
ENVIAR_CORREO	Destinatarios	Lista de logins de los destinatarios, los logins se separan con coma (,).
	Asunto	Asunto del correo
	Mensaje	Mensaje del correo

Tabla 22. Sintaxis del mensaje del cliente para el protocolo de enviar un correo.

Comando Servidor	Parámetros	Descripción
ENVIAR_CORREO_OK	Ninguno	Indica que se envía el correo sin problemas.

Tabla 23. Sintaxis del mensaje del servidor para el protocolo de enviar un correo.

A continuación, un ejemplo concreto:



Ilustración 22. Ejemplo de protocolo para enviar un correo.

La Ilustración 22 muestra el siguiente ejemplo: el usuario envía una petición de enviar correo con el comando ENVIAR_CORREO seguido de destinatarios “cupitaller”, asunto “importante” y mensaje “mensaje”. El servidor no tiene problemas al registrar el correo, y por lo tanto envía el comando ENVIAR_CORREO_OK.

Caso 2: Hay un error al momento de enviar el correo.



Ilustración 23. Protocolo para enviar un correo cuando ocurre un error.

En la Ilustración 23 se aprecia el protocolo de enviar correo cuando ocurre un error. El cliente envía un mensaje con el comando ENVIAR_CORREO, seguido de los destinatarios, asunto y mensaje del correo. El servidor tiene problemas al registrar el correo en la base de datos y envía al cliente el comando ERROR con su mensaje de error.

Las Tablas 24 y 25 explican la sintaxis de cada mensaje:

Comando Cliente	Parámetros	Descripción
ESCRIBIR_CORREO	destinatarios	Lista de logins de los destinatarios, los logins se separan con coma (,).
	asunto	Asunto del correo
	mensaje	Mensaje del correo

Tabla 24. Sintaxis del mensaje del cliente para el protocolo de enviar correo.

Comando Servidor	Parámetros	Descripción
ERROR	MensajeError	Mensaje que indica el error al escribir el correo.

Tabla 25. Sintaxis del mensaje del servidor para el protocolo de enviar correo cuando ocurre un error.

A continuación, un ejemplo concreto:

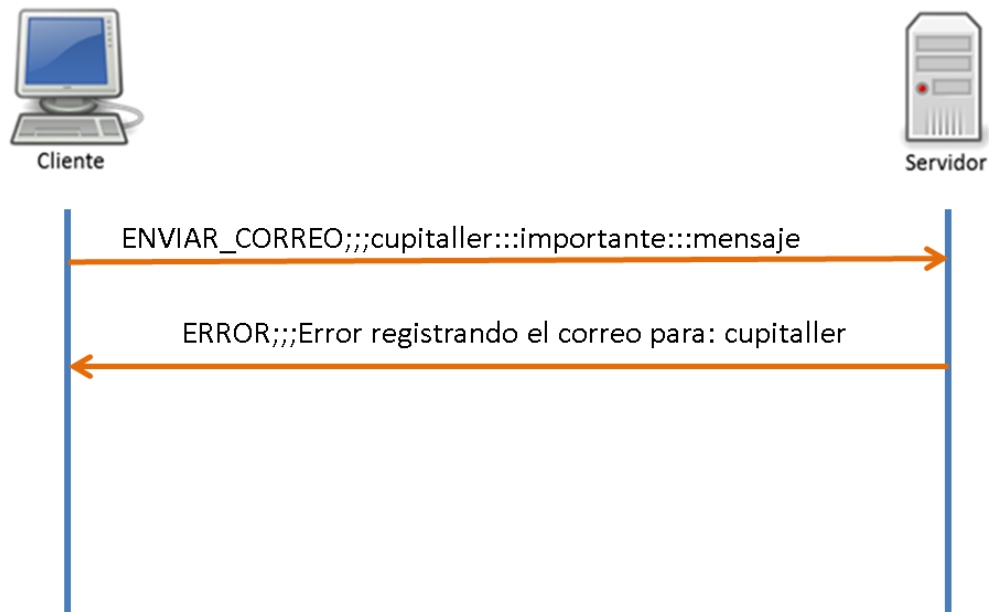
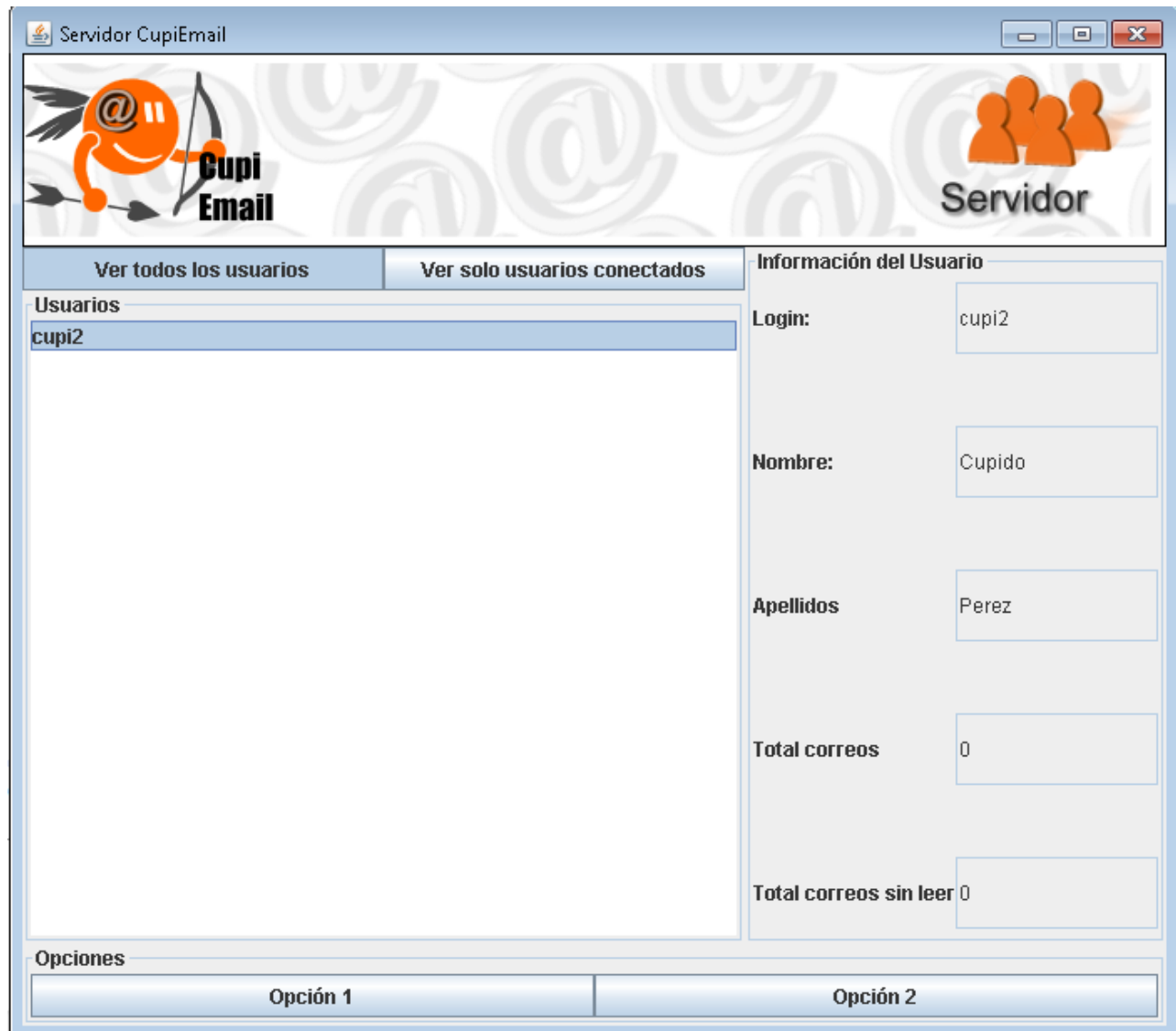


Ilustración 24. Ejemplo de protocolo para enviar un correo cuando ocurre un error.

La Ilustración 24 muestra el siguiente ejemplo: el usuario envía una petición de enviar correo con el comando ENVIAR_CORREO, con destinatarios cupitaller, asunto “importante” y mensaje “mensaje”. El servidor tiene problemas al registrar el correo para el destinatario cupitaller, y por lo tanto envía el comando ERROR con el mensaje: “Error registrando el correo para: cupitaller”.

Interfaz

Interfaz del Servidor



The screenshot shows the 'Servidor CupiEmail' window. It features a header with the CupiEmail logo (an orange '@' character with wings and a bow) and the word 'Servidor' next to three orange person icons. Below the header, there are two tabs: 'Ver todos los usuarios' and 'Ver solo usuarios conectados'. The 'Ver todos los usuarios' tab is active, showing a list of users with 'cupi2' selected. To the right, the 'Información del Usuario' section displays details for the selected user: Login: cupi2, Nombre: Cupido, Apellidos: Perez, Total correos: 0, and Total correos sin leer: 0. At the bottom, there is an 'Opciones' section with two buttons: 'Opción 1' and 'Opción 2'.

Ver todos los usuarios		Ver solo usuarios conectados	
Usuarios		Información del Usuario	
cupi2		Login:	cupi2
		Nombre:	Cupido
		Apellidos	Perez
		Total correos	0
		Total correos sin leer	0
Opciones			
Opción 1		Opción 2	

Interfaz del Cliente

Ventana inicial del cliente que permite crear una cuenta o iniciar sesión.

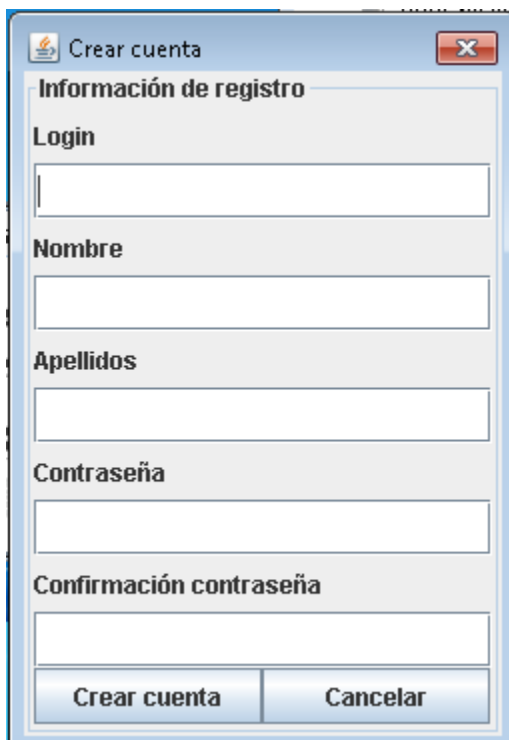


The screenshot shows the 'CupiEmail' client window. It features the CupiEmail logo (an orange '@' character with wings and a bow) and the text 'Bienvenido a CupiEm@ail'. To the right, there are three buttons: 'Crear cuenta', 'Iniciar sesión', and 'Salir'.

 Bienvenido a CupiEm@ail	Crear cuenta
	Iniciar sesión
	Salir



Ventana para crear una cuenta de usuario



Crear cuenta

Información de registro

Login

Nombre

Apellidos

Contraseña

Confirmación contraseña

Crear cuenta **Cancelar**

Ventana para iniciar sesión de usuario.



Iniciar sesión

Inicio de Sesión

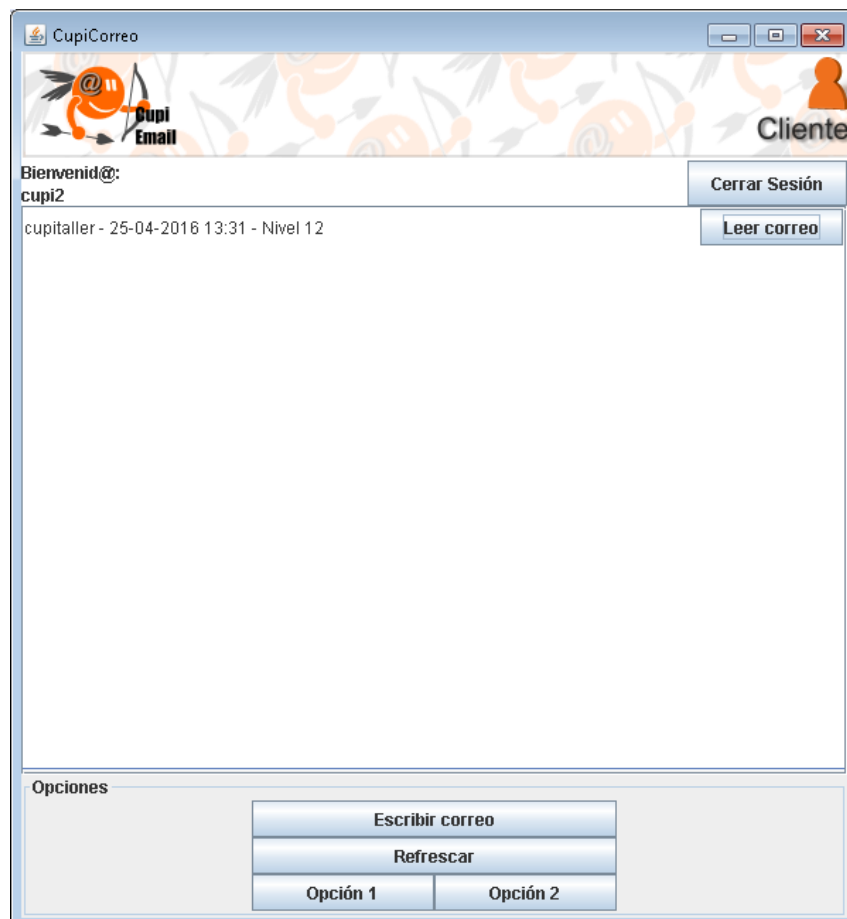
Usuario

Contraseña

Iniciar sesión **Cancelar**

Ventana de cliente con correos





Ventana para enviar un correo.

The screenshot shows a "Nuevo correo" (New email) window. It has a title bar with the text "Nuevo correo" and a close button. The form contains two input fields: "Para:" with the value "cupi2" and "Asunto:" with the value "Nivel 12". Below these fields is a large text area containing the text "Publicar ejercicio de Nivel 12 pronto." At the bottom of the window are two buttons: "Enviar correo" and "Cancelar".