SOLUCION EXAMEN COMUNICACION DE DATOS
(ref: scdt9812.doc)
23 de DICIEMBRE de 1998.

Problema 1:

El procesador de texto *Palabra*, tiene una opción que guarda sus documentos encriptados con Vigenere ampliado (con letras mayúsculas, minúsculas, espacio y caracteres especiales), con clave de largo máximo de 8 caracteres. Se sabe que este procesador de texto en el archivo en la posición byte 47 en adelante tiene el siguiente texto:

Documento Microsoft Palabra 6.0

Se pide:
a) Explique como reformularía, el Vigenere ampliado.
b) Generar un programa que determine con que clave fue encriptado el documento.

Solución:

a)
Vigenere originalmente es en base a una matriz cuadrada de dimensiones 26 x 26, en el caso más común, donde se consideran 26 caracteres en mayúscula *A..Z*(y se quitan los espacios y caracteres especiales), y para codificar se utiliza el siguiente método, para el caso de codificar el carácter *char*, con el carácter correspondiente *char_clave* de clave:

char_codif=Vigenere(char,char_clave)

siendo Vigenere la matriz cuadrada mencionada anteriormente.

La matriz tiene el siguiente formato:

```
                  A B C D E F G H I J K L M N O P Q R S T U V W X Y Z

1a. fila :    A   B C D E F G H I J K L M N O P Q R S T U V W X Y Z A
2a. fila :    B   C D E F G H I J K L M N O P Q R S T U V W X Y Z A B
3a. fila :    C   D E F G H I J K L M N O P Q R S T U V W X Y Z A B C
4a. fila :    D   E F G H I J K L M N O P Q R S T U V W X Y Z A B C D
5a. fila :    E   F G H I J K L M N O P Q R S T U V W X Y Z A B C D E
                        ..................
                        ..................
26a.fila :    Z   A B C D E F G H I J K L M N O P Q R S T U V W X Y Z
```

Se supone que la matriz tiene las permutaciones de los caracteres a codificar, en cada columna.

En general para codificar un texto, se le quitan los espacios, puntos y caracteres especiales, y todas las letras se pasan a mayúsculas. En el caso a resolver esto no es posible, ya que una vez desencriptado debo obtener el texto tal cual fue escrito, con todo, inclusive espacios, puntos y caracteres especiales, así como las letras en mayúscula, y en minúscula en su forma original. Para resolver esto extiendo la matriz cuadrada, a la dimensión correspondiente a considerar, las mayúsculas, minúsculas y caracteres especiales, y el carácter codificado (*char_codfi*) se encontrará en el mismo rango.

type v_char: [A..Za..z!”·$%&/()=?¿Ç..;:_-]

Por lo tanto, el Vigener ampliado se codifica con las siguiente matriz cuadrada:
char_codif=Vigenere_Amp(v_char,v_char)
donde *v_char* es el tipo definido anteriormente.

b)
Para resolver el siguiente problema, debo deshacer la codificación de Vigenere realizada, conociendo que en cierta parte del texto se encuentra un mensaje conocido. Para esto realizo el siguiente método.

Con la siguiente función, dado el carácter codificado y el que corresponde a la posición, determino la clave con que fue encriptado por Vigenere.

```
Function  Busco_clave(decod_char:char,encod_char: char):char
// esta funcion devuelve para el carácter codificado,
// con que clave fue encriptado.
begin
        carácter=A;
        encontre=false;
        // Busco para este caracter, que clave codifica al
        // carácter codificado.
        while carácter <> ultimo_carácter  and not encontredo
                if encod_char==Vigenere_Amp(decod_char,carácter)
                        encontre=true;
        Busco_clave=carácter
end.
```

Para resolver busco la clave por medio de la función mostrada anteriormente, a partir del texto que se supone que aparece y el texto enciptado. Si esta se repite en una secuencia de aparición, entonces encontré la clave.

Documento Microsoft Palabra 6.0

Considero estas 31 letras conocidas a partir de la posición 47, para determinar cual es la clave. Al ser esta de largo 8 debe repetirse en las 31 posiciones.

La posición de comienzo de la clave, después de 47, la puedo determinar, ya que para una clave de largo *l*, esta debe comenzar en :

inic_clve= *47 + l - (47 mod l);*

```
Procedure  desencripto_Vigenere( in texto: sting, out clave: string);
// este procedimiento busca la clave según una matriz
// del sistema llamada Vigenere_Amp
// En conocido(i) esta el texto: “Documento Mic... 6.0”
var
      clve_x: array(1..28) of char;
      aux1,aux2:char;
begin
      // desencripto los 31 caracteres que conozco
      // y los guardo en clve_x
      for i =47 to 78 do
            clve_x(i-46):=Busco_clave(conocido(i-46),texto(i));
      // Busco la secuencia que se repite
      encontre:=false;
      clve_larg:=1;
      while ( clve_larg <= 8 and not encontre )
      begin
            encotre:=true;
            posicOK:=true;
            // Calculo donde comienza la clave para el rango utilizado
            inic_clve= 47 + clve_larg - (47 mod clve_larg);
            // Calculo donde termina la clave para el rango  utilizado
            fin_clave:=78 - (78 mod clve_larg) -1;
            j:=0;
            while  ( j < clve_larg and encontre )
            begin
                  sec:=1;
                  // comparo con todas la misma posición de clave.
                  while ( (inic_clve + sec *clve_larg +  j) < fin_clave and
                           posicOK)
                        begin
                              aux1:=clve_x(inic_clve+j);
                              aux2:=clve_x(inic_clve + j+ sec*clve_larg);
                              if (aux1 <> aux2) then
                                    posicOK=false;
                              end
                              sec:=sec+1;
                        end;
                  if  ((inic_clve + sec *clve_larg) < fin_clave ) then
                        // existe un carácter no coincidente
                        encontre:= false;
                  end;
                  j:= j+1;
            end;
            clve_larg:= clve_larg +1;
      end;
      // escribo el resultado
      for j=1 to clve_larg - 1 do
            clave(j):= clve_x(inic_clve + j - 1);
end.
```

Problema 2:

Se desea implementar un protocolo para la transmisión de voz, donde se deben tener las siguientes consideraciones:

- Es más importante para el receptor contar lo antes posible con los paquetes mas actuales disponibles, que el recibir en orden todos los paquetes emitidos.
- No se considera de interés el reenvío de paquetes perdidos (no reintentar el envío).

Se cuenta con un procedimiento, *paq_voz_enviar(out: paquete_voz)*, que devuelve el siguiente paquete de voz a enviar (se envía con el mismo formato).

Se pide:

a) Justifique que problemas ocasiona utilizar un protocolo tipo TCP orientado a conexión.
b) Implementar el protocolo de envío y recepción, sobre UDP (datagrama de TCP), suponiendo se tienen primitivas udp_send, y udp_receive.

Solución:

a)

Un protocolo del tipo TCP/IP, en su implementación, incluye la retrasmisión de paquetes que se perdieren, además de que entrega al destinatario en orden de envío, y en caso de perderse algún paquete, retrasara la entrega de los siguientes, hasta recuperar el perdido.

Esto no es compatible con los requerimeintos de:

- Es mas importante para el receptor contar lo antes posible con los paquetes mas actuales disponibles, que el recibir en orden todos los paquetes emitidos.
- No se considera de interés el reenvío de paquetes perdidos (no reintentar el envío).

b)

Para la resolución del problema planteado, el UDP, no garantiza que los paquetes recibidos sean en el orden de emitidos, por lo que debo considerar algún método que permita determinar si el paquete recibido es posterior o anterior al anterior.

Para resolver esto existen dos métodos, el primero es simplemente numerar los paquetes en origen, y así podré determinar con que orden fueron emitidos. El segundo método es similar y consiste en colocarle una estampa de tiempo a cada datagrama.

Resolución por el segundo método planteado:

El formato del datagrama entonces será el siguiente:

```
type datagrama: record of
                    tiempo: timestamp;
                    datos: string;
                end.
```

Los procesos que se requieren para la resolución son los siguientes:

```
Procedure  envio ( );
var
paq:strig;
data: datagrama;

begin
        while (true)
        begin
                paq_voz_enviar(paq);
                data.datos:=paq;
                data.tiempo:=NOW();
                udp_send(data)
        end;
end.

Procedure  recibo ( );
var
ultimo:timestamp;
paq:strig;
data: datagrama;

begin
        ultimo=NOW();
        while (true)
        begin
                udp_receive(data);
                if  ( data.tiempo > ultimo ) then
                begin
                        entrego(data.datos);
                        ultimo:=data.tiempo;
                end;
        end
end.
```

Problema 3:

Se cuenta con un servidor, que brinda servicios a través de la red, en los que se requiere estar ”autenticado”. Este servidor, no permite a los usuarios una interfase de comandos para establecer o cambiar el password de acceso al mismo. Además existe un usuario, *administrador*, que puede modificar el password de cualquier usuario, por medio de la siguiente primitiva del sistema operativo:

cambio_passwd user passwd_anterior passwd_nuevo

que devuelve OK, si cambio el password o ERROR en caso de problemas.

Se pide:

a) Especificar en un lenguaje de alto nivel, un servicio sobre una punto conocido de conexión de capa 4, donde dicho servicio se ejecuta con permisos de *administrador*, y permita modificar el password de un usuario del servidor, y el cliente para su utilización.

b) Que problemas de seguridad a nivel de escuchas en la red puede tener este protocolo, y que posibles soluciones existen.

Solución:

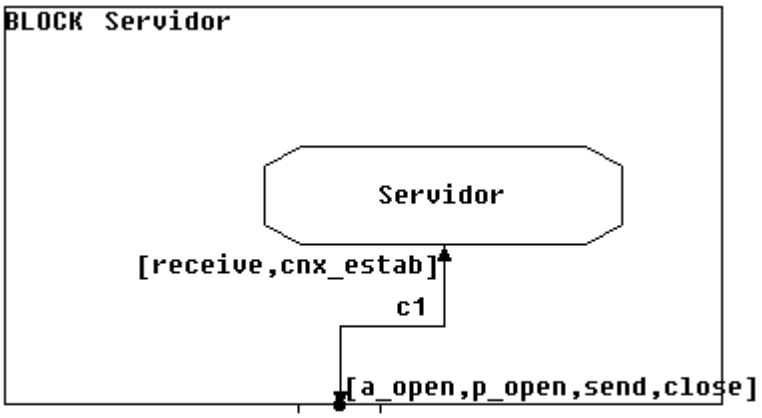
El protocolo propuesto funciona de la siguiente manera (suponiendo que S es el servidor y C el cliente que desea cambiar el passwd):

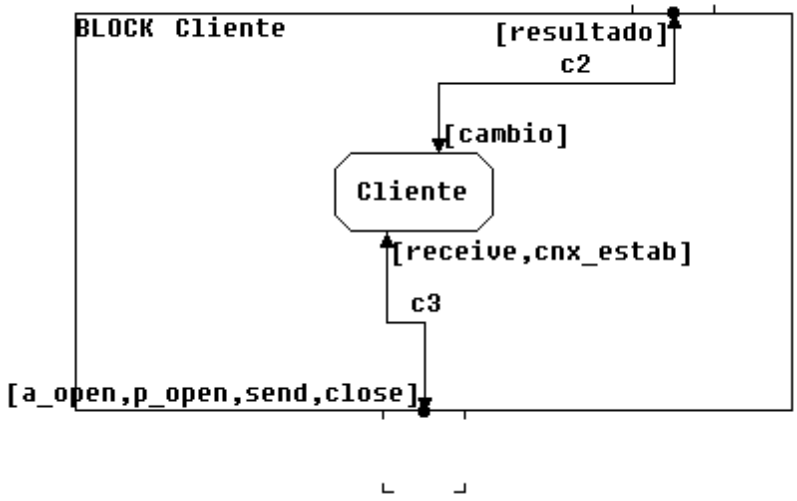
S: hola protocolo cambio de password\n
C: user *nombre_usuario*\n
S: por favor envie su password\n
C: pass *password_actual*\n
S: por favor envie su nuevo password\n
C: newpass *password_nuevo*\n
S: *resultado_cambio_passwd*\n
C: quit\n
S: <closes connection>
C: <closes connection>

Los términos que aparecen en itálica, corresponden a las variables del cambio de password. Utilizaré un protocolo de capa 4 con primitivas tipo TCP/IP, a_open, p_open, receive, send, y close.

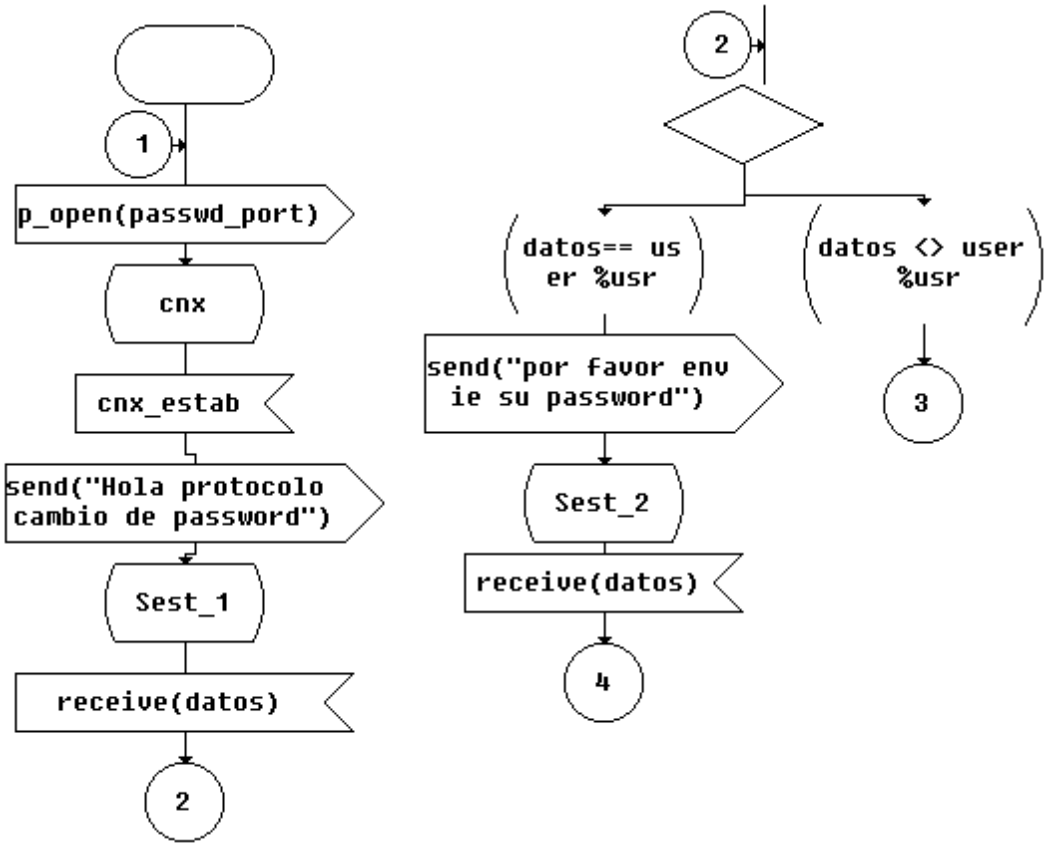
La interface manejada con el cliente son las siguientes señales:

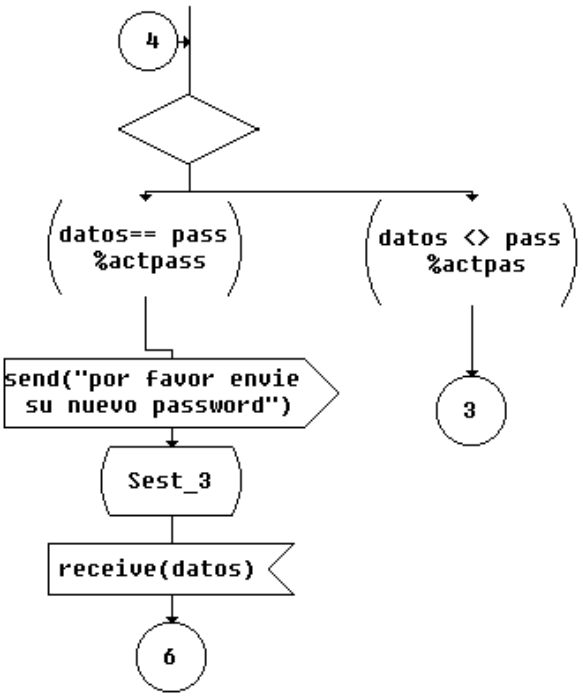
cambio(user,passwd,new_passwd)
resultado(texto)

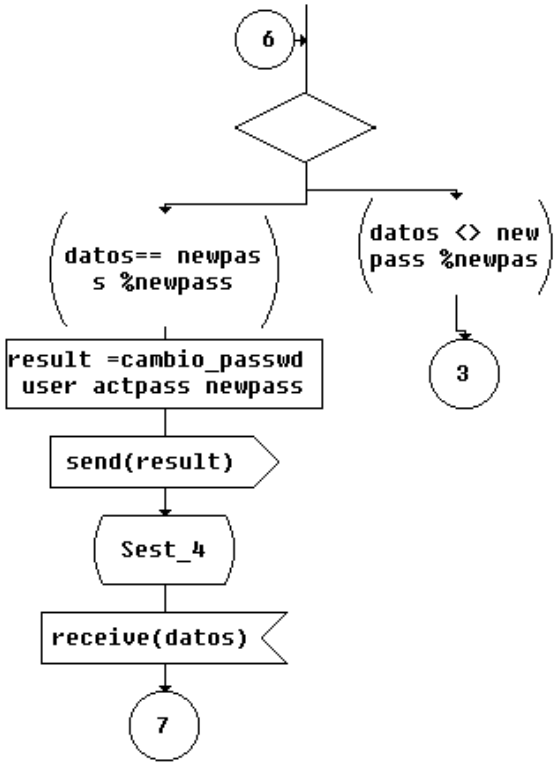

BLOCK Servidor
Servidor
[receive,cnx_estab]
c1
[a_open,p_open,send,close]

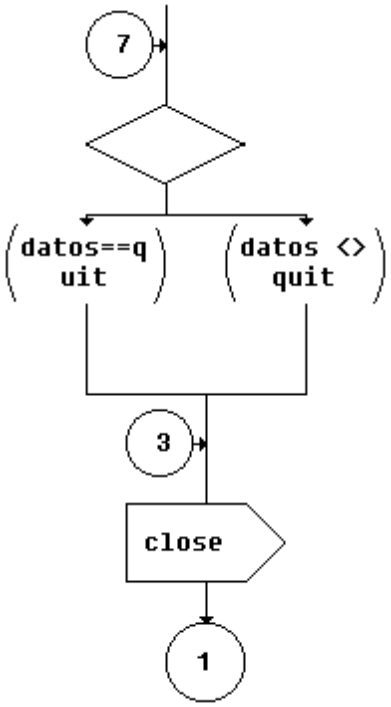

BLOCK Cliente
[resultado]
c2
[cambio]
Cliente
[receive,cnx_estab]
c3
[a_open,p_open,send,close]

Process Server

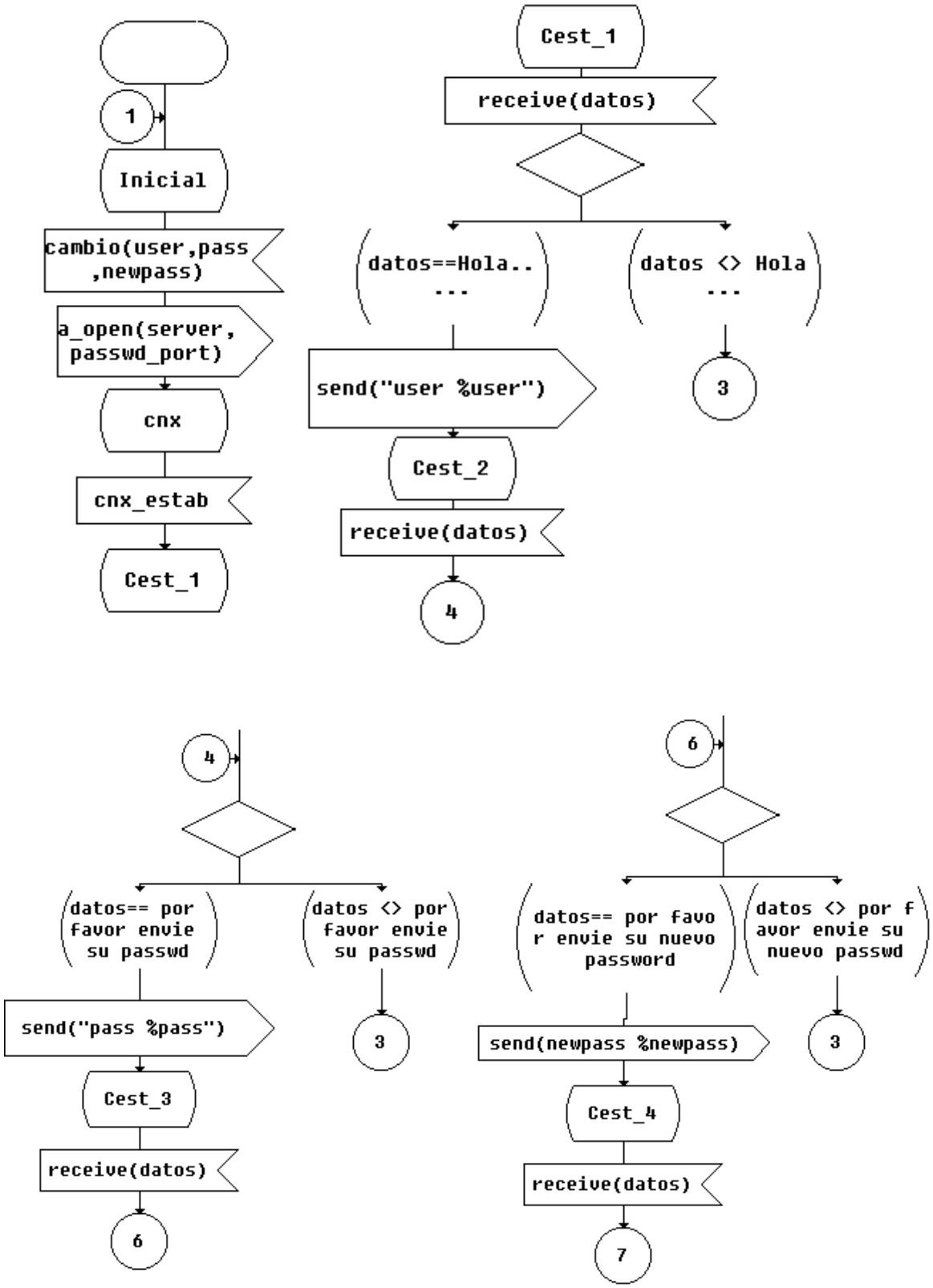


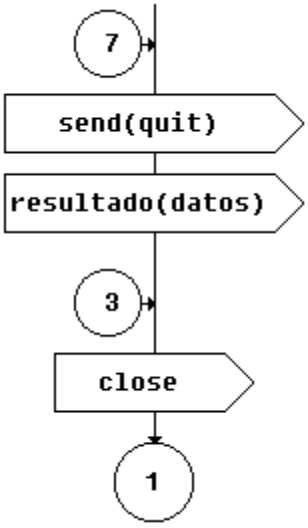

4
datos== pass %actpass
datos <> pass %actpas
send("por favor envie su nuevo password")
3
Sest_3
receive(datos)
6


6
datos== newpas s %newpass
datos <> new pass %newpas
3
result =cambio_passwd user actpass newpass
send(result)
Sest_4
receive(datos)
7


7
datos==q uit
datos <> quit
3
close
1

Process Cliente




7
send(quit)
resultado(datos)
3
close
1