U N I V E R S I D A D N A C I O N A L

A U T O N O M A D E M E X I C O

COLEGIO DE CIENCIAS Y HUMANIDADES
INSTITUTO DE INVESTIGACIONES EN MATEMATICAS APLICADAS
Y EN SISTEMAS IIMAS-UNAM

==========

EL SISTEMA DE TRANSPORTE PROCESO-A-PROCESO PARA LAS PDP-11 DE LA RED DE COMPUTADORAS DEL IIMAS

Por: José Guillermo Villagómez Cardona

T E S I S

QUE PRESENTA COMO PARTE DE LOS REQUISITOS
PARA OBTENER EL GRADO DE:

MAESTRO EN CIENCIAS DE LA COMPUTACION

México, D.F. Septiembre de 1982

UNAM – Dirección General de Bibliotecas Tesis Digitales Restricciones de uso

Indice

LA RED LOCAL DEL IIMAS

LAS CAPAS DE TRANSMISION E INTER-RED

LA ESTACION DE TRANSPORTE EN EL SISTEMA RSX-11M

LA CAPA DE TRANSPORTE:
IMPLEMENTACION DE HERMES EN EL SISTEMA RSX-11M

Lista de Figuras

PROLOGO

El exitoso establecimiento de la primera red de computadoras ARPANET en 1971, así como la rapidez con que declinaron los costos del equipo de computación, aunados con la atractiva idea que significaba el gran potencial de compartir recursos informáticos geográficamente dispersos, ha impulsado el desarrollo casi explosivo del campo de redes. Muchas redes de computadoras están siendo desarrolladas o planeadas en todo el mundo,y sus aplicaciones incluyen las educacionales, las comerciales y militares.

Esta corriente también hizo eco en un grupo de investigadores del Instituto de Investigaciones en Matemáticas Aplicadas y en Sistemas (IIMAS) de la Universidad Nacional Autónoma de México, quienes se propusieron implantar una red local de computadoras para interconectar los equipos actuales (y futuros) de la institución en un sistema que permita hacer uso de ellos en forma racional y eficiente. No se descartó la posibilidad futura de interconectar esta red a redes locales de otras instituciones, o a redes públicas de mayor magnitud.

He participado desde hace mas de un año en el desarrollo del software de la red y el trabajo que ahora presento como tesis es la implantación de los diseños y las valiosas ideas propuestas por el Dr. Gursharan S. Sidhu, coordinador del proyecto "REDLAC". Estas propuestas dieron como resultado un software de elevado

nivel de sofisticación y eficiencia.

El contenido de este trabajo se puede resumir en los siguientes puntos:

1. Una estación de transporte en lenguaje ensamblador y sus manejadores de líneas para el sistema operativo RT-11 (para uso en la PDP-11/10);
2. Una estación de transporte tipo ACP para el sistema operativo RSX-11M, junto con los manejadores de línea corespondientes (para uso en las PDP-11/34 y 11/40 de la red);
3. Implementación en ambos sistemas de las rutinas de interface que permitan a los usuarios el acceso a la red, mediante el uso de primitivas tales como: activar un puerto, liberar un puerto, enviar una carta, recibir una carta, terminar una sesión, etc.
4. Documentación e instructivo para el uso del servicio que los sistemas anteriores ofrecen.

El capítulo 1 de este documento contiene una introducción al concepto de redes de computadoras, destacando algunos aspectos de la tecnología de redes. El capítulo 2 explora la tendencia actual en la organización de redes, como una serie de capas o niveles, para lo cual se describe la arquitectura de referencia propuesta (con fines de estandarización) por la International Standard Organization (ISO); también se establece su relación con la arquitectura de redes como ARPANET, DECNET y XEROX. El capítulo 3 es una advertencia a los problemas que surgen al contemplar la interconección de redes (incompatibles).

El capítulo 4 contiene una sinopsis de las características de la red local del 11MAS y una discusión global de su arquitectura.

En los capítulos 5,6 y 7 se detallan las capas más inferiores de este diseño, correspondientes a la capa de transmisión, la capa inter-red y la capa de transporte; junto con una discusión de los protocolos en cada nivel y los detalles de su implementación en el sistema RSX-11M. En el capítulo 8 se detalla la implementación de esta arquitectura en el sistema RT-11.

Por último, el capítulo 9 es un resúmen de los logros alcanzados, incluyendo aquí algunas sugerencias sobre las aplicaciones futuras, que pueden desarrollarse fundamentadas en este trabajo. Los apéndices I y II son los manuales para el usuario de los servicios de transporte en los sistemas RSX-11M y RT-11.

Expreso mi gratitud al Dr. Sidhu, director de tesis, por su incansable y desinteresado apoyo, y dedico con mucho cariño este trabajo, a mi esposa Susana, mi hijo Santiago y a mis padres.

IIMAS-UNAM
Mexico,D.F. Septiembre de 1982

En los capítulos 5,6 y 7 se detallan las capas más inferiores de este diseño, correspondientes a la capa de transmisión, la capa inter-red y la capa de transporte; junto con una discusión de los protocolos en cada nivel y los detalles de su implementación en el sistema RSX-11M. En el capítulo 8 se detalla la implementación de esta arquitectura en el sistema RT-11.

Por último, el capítulo 9 es un resúmen de los logros alcanzados, incluyendo aquí algunas sugerencias sobre las aplicaciones futuras, que pueden desarrollarse fundamentadas en este trabajo. Los apéndices I y II son los manuales para el usuario de los servicios de transporte en los sistemas RSX-11M y RT-11.

Expreso mi gratitud al Dr. Sidhu, director de tesis, por su incansable y desinteresado apoyo, y dedico con mucho cariño este trabajo, a mi esposa Susana, mi hijo Santiago y a mis padres.

IIMAS-UNAM
Mexico,D.F. Septiembre de 1982

1.

REDES DE COMPUTADORAS

1.1 Introducción

El desarrollo tecnológico en este último tiempo en los campos de la computación y de las telecomunicaciones, ha cambiado substancialmente el modelo de organización de un CENTRO DE CÓMPUTO, y el sistema de una sola computadora grande atendiendo el trabajo de muchos usuarios, está reemplazándose por el de un gran número de computadoras autónomas pero interconectadas que efectúan el trabajo. A estos sistemas se les denomina REDES DE COMPUTADORAS.

En esta definición, se entiende que dos computadoras están interconectadas, si son capaces de intercambiar información, y el requisito de que sean autónomas excluye de nuestra definición aquellas en que existe una relación maestro-esclavo, como el caso de una computadora que puede arrancar, detener o controlar a otra, ni tampoco el caso de una gran computadora con lectoras de tarjetas o terminales remotas.

Para dar una percepción mas profunda, sobre el alcance de la definición, es útil la analogía con lo que ocurre dentro de una sola computadora. Ahí se encuentran un conjunto de actividades, a las cuales se llaman procesos, cooperando entre si para cumplir las tareas que los usuarios les encomiendan. Esta cooperación entre los diversos procesos requiere de algún sistema de

intercomunicación entre ellos, y este último lo provee el SISTEMA OPERATIVO. Pues bien una red, extiende esta posibilidad de cooperación a todos los procesos ubicados en todas las computadoras conectadas a la red, estableciendo un sistema de comunicación proceso a proceso, el que a su vez emplea algún sistema de comunicación de datos.

Entre los hechos que han influido en el desarrollo de este campo pueden mencionarse:

- La necesidad de ciertas organizaciones de poder obtener y correlacionar la información procesada aisladamente en diferentes lugares, y el poder compartir los datos, programas y recursos.
- El deseo de alta confiabilidad, al tener recursos alternativos, en el caso de falla de un integrante de la red.
- La relación costo benificio de una computadora pequeña en relación a las grandes, ha planteado el diseño de sistemas en base a varios procesadores.
- El deseo de explotar las ventajas de la computación distribuída, pues al dedicar cada máquina a una función diferente, la implementación es mas simple o mas eficiente.

Según la distancia entre procesadores y de acuerdo a su grado de acoplamiento, estos sistemas reciben diferentes nombres:

- Máquinas de PROCESAMIENTO EN PARALELO, si la distancia entre pocesadores es del órden de 0.1 metro, pudiendo trabajar varios procesadores simultáneamente en el mismo programa.
- MULTIPROCESADORES con distancias de hasta 1 metro, y que se comunican compartiendo memoria principal.
- Las REDES propiamente dichas, donde las computadoras se

comunican intercambiando mensajes, llamadas REDES LOCALES si la distancia entre procesadores está entre 10 metros hasta el órden de 1 kilómetro, y REDES LARGAS si la distancia es mayor de 10 kilómetros.

1.2 Estructura de una red

Uno de los trabajos pioneros en este campo fue el de ARPANET, que sin duda ha aportado la mayor cantidad de información en este campo y estableció la pauta de la terminología básica que se aplicará también en este documento. En cualquier red existe una colección de máquinas para el servicio de usuarios denominadas ANFITRIONES, y constituyen lo que se llama la RED EXTERNA. Estos anfitriones están interconectados por la RED INTERNA que se encarga de acarrear los mensajes entre ellos. (Ver la Figura 1-1).

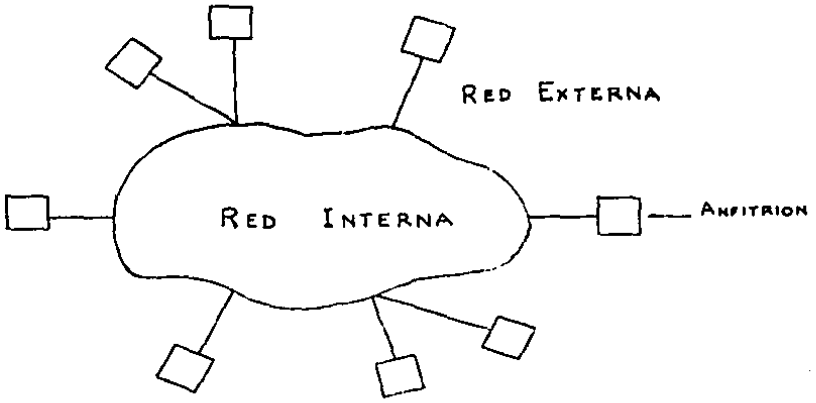


Figura 1-1: Estructura de una red

La red interna consiste de todos los equipos, medios de

comunicación y procedimientos necesarios para lograr la intercomunicación de los anfitriones. Aunque existen muchas variantes, en casi la mayoría de las redes, la red interna consiste de dos componentes básicos, los nodos de conmutación y las líneas de transmisión. Los nodos se conocen en Arpanet como IMP (Interfase Message Processors) y son computadoras especializadas, y las líneas de comunicación se conocen con el nombre de circuitos o canales, y de acuerdo a la topología de interconección las redes pueden clasificarse en:

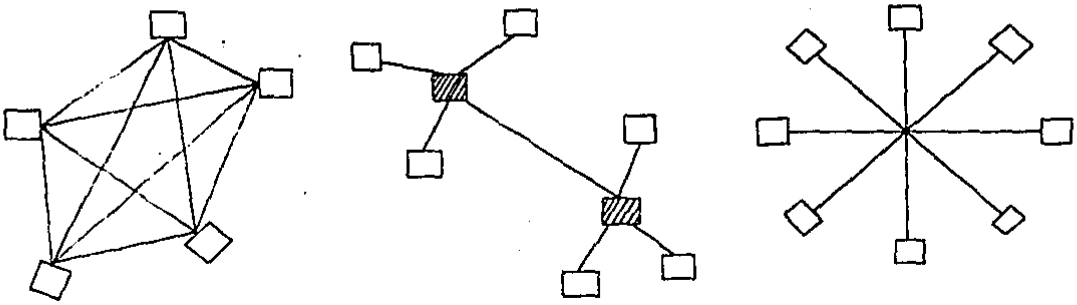
1.2.1 Redes clásicas

Estas redes se caracterizan porque las líneas de comunicación de la red interna son cables o líneas telefónicas, cada uno conectado a un par de nodos, pudiendo tratarse de redes totalmente conectadas, parcialmente conectadas, en estrella, etc. como se ilustra en la figura 1-2, y cuyos criterios de diseño persiguen el lograr confiabilidad, utilizando suficiente redundancia, y eficiencia disminuyendo los retrazos.

De acuerdo a la técnica básica de transmisión utilizada, este tipo de redes pueden ser:

- Circuitos conmutados

  En esta técnica, es indispensable el establecer de antemano un ruta física definida y dedicada entre los anfitriones de orígen y destino, igual a lo que ocurre en el sistema telefónico. Los retrazos originados por el establecimiento de esta conección previa, resultan indeseables para muchas aplicaciones de computación, peró tiene la ventaja de que una vez establecida, los retrazos son mínimos, y no existe el peligro de la



Totalmente conectada Parcialmente conectada Red en estrella

Figura 1-2: Algunas topologías de redes

congestión.

- Paquetes conmutados

  En este caso, no existe una trayectoria física establecida de antemano, entre el enviador y el receptor. Cuando el anfitrión orígen envía un bloque de datos, este es almacenado, en el primer nodo intermedio, y dependiendo del tráfico actual en las líneas ahí conectadas, selecciona una línea disponible por la cual el nodo vuelve a transmitir el paquete. Si el paquete fué recibido sin error entonces el nodo receptor devuelve al emisor un mensaje de reconocimiento (ACK). Además si después de un cierto tiempo, el nodo no ha recibido el ACK, se retransmite el paquete, y de esta manera se eliminan errores por medio de una estrategia de retransmisión. Este proceso es repetido de nodo a nodo, hasta que el paquete arriba a su destino. Una red que utiliza este principio es conocida con el nombre de GUARDA REEXPIDE.

La diferencia clave entre "circuitos conmutados" y "paquetes conmutados", está en que el primero, reserva de antemano el ancho de banda requerido, mientras que el segundo lo toma y libera de acuerdo a sus necesidades. Con circuitos conmutados, cualquier ancho de banda no usado se desperdicia, mientras que con paquetes

conmutados, este puede ser utilizado por paquetes que van a otros destinos ya que los circuitos no están nunca dedicados, sin embargo un tráfico repentino puede colmar la capacidad de almacenamiento de un nodo, provocando pérdida de paquetes.

1.2.2 Redes no clásicas

La idea básica en estas redes es el usar un medio compartido, utilizando una técnica de difusión (broadcast). En este diseño hay un simple canal de comunicación compartido por todos los nodos. El mensaje enviado por uno es recibido por todos y si se especifica en el mensaje una dirección de destino entonces los nodos a los cuales no corresponde esta dirección ignoran el mensaje.

Existen aquí también algunas alternativas:

- El canal de comunicación tiene la forma de un BUS, pudiendo ser este un cable coaxial o un par trenzado, con velocidades de transmisión del órden de 1 a 10 megabits/seg. Existen varios mecanismos para controlar el acceso al bus y resolver conflictos de colisiones cuando dos o mas nodos quieren transmitir simultáneamente. Un ejemplo de este tipo de redes es ETHERNET.

- El canal de comunicación es un SATELITE o un SISTEMA DE RADIO. Cada nodo tiene una antena a través de la cual puede enviar y recibir mensajes. Todos los nodos pueden escuchar los mensajes emitidos por el satélite y a veces también escuchar las transmisiones de sus nodos vecinos.

- REDES EN ANILLO que también utilizan un cable que conecta punto a punto las estaciones consecutivas, y en donde cada bit recorre todo el anillo y los mensajes no son retransmitidos por el siguiente nodo. Al igual que los otros sistemas de comunicación existen también reglas para regular el acceso al anillo.

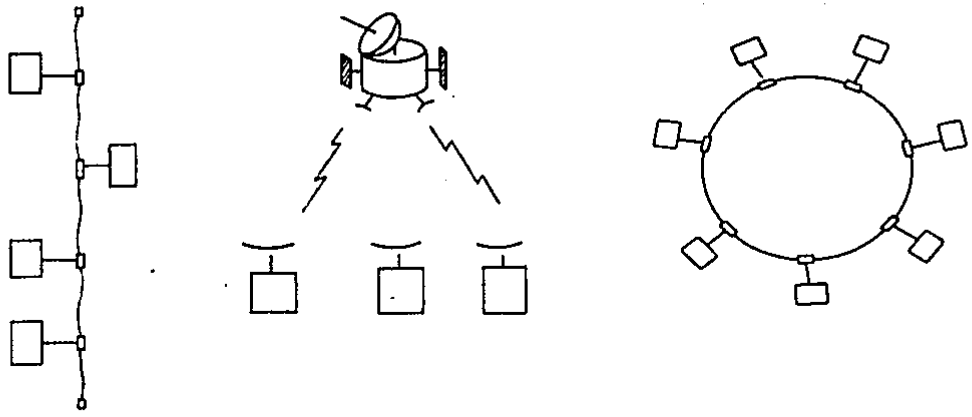


Figura 1-3: Redes de difusión (Broadcast)

En este tipo de redes la forma de asignación del canal puede ser estática o dinámica. Una asignación estática consiste en dividir el tiempo en intervalos discretos y mediante un sistema de "round-robin" permitir que cada nodo transmita cuando le toca su turno. Sin embargo este método tiene el problema de desperdicio del canal, si el nodo al que le toca su turno no tiene nada que transmitir. Los métodos de asignación dinámica de un canal pueden ser centralizados o descentralizados. En los primeros existe una unidad que arbitra quien va a continuación, y en los descentralizados cada nodo debe decidir por si mismo si quiere transmitir o no.

1.3 Redes locales

Puesto que el presente trabajo corresponde al desarrollo del software para la red local de computadoras del IIMAS, haremos énfasis en las características de una red local, cuyos distintivos sobresalientes son:

- Un diámetro del órden de un kilómetro.
- Velocidades de transmisión sobre 1 megabit/seg
- Pertenecea una sola organización

La mayoría de las redes locales utilizan la técnica de difusión en su red interna de comunicación, y aquí estriba la diferencia clave con sus parientes cercanas las redes largas, ya que los diseñadores de redes largas, están forzados por razones económicas o legales a usar redes telefónicas públicas, sin importar si son o no técnicamente apropiadas. En contraste nada impide a los diseñadores de redes locales, el utilizar sus propios cables con un elevado ancho de banda, con la ventaja de que los bajos retrazos y tasas de error hacen posible que los sistemas de procesadores distribuidos, compartan información que antes solo era accesible a sistemas de multiprocesadores que compartían la memoria principal.

El alto ancho de banda puede ser explotado para simplificar la estructura de control de los protocolos de comunicación, los encabezados de los paquetes pueden arreglarse para simplificar el procesamiento involucrado en crear o interpretar estos, y el

sistema de DATAGRAMAS resulta más adecuado y económico que el de CIRCUITOS VIRTUALES. Su mayor atractivo es la habilidad de cada estación (terminal o computadora) para verificar el canal de comunicación, antes de intentar usarlo.

El deseo de lograr las ventajas antes mencionadas motivó a los diseñadores de la red local del IIMAS, a selecionar una red de alta velocidad (10 megabits/seg) con la técnica de difusión de tipo Ethernet, utilizando un cable coaxial. Sin embargo como el desarrollo del hardware adecuado toma su tiempo, se pensó en desarrollar paralelamente una red punto a punto de tipo guarda reexpide, utilizando líneas de comunicación de 9600 bits/seg y cuya implementación ha sido concluida.

2.

ARQUITECTURA DE LOS SERVICIOS EN REDES

Una técnica empleada en el diseño de la mayoría de las redes, consiste en organizar a éstas como una serie de capas o niveles, cada una construída sobre su predecesora. El número, el nombre y las funciones de cada capa varían de red en red. Sin embargo en todas las redes, el propósito de cada capa es ofrecer ciertos servicios a las capas superiores, liberando a éstas de los detalles de como los servicios ofrecidos están implementados. Esta técnica si bien no reduce la complejidad del diseño, ayuda a manejarlo, transformando este, en varios problemas de diseño mas pequeños y manejables, es decir el diseño de cada capa.

Las reglas, convenciones y formatos de mensajes usados en la "conversación" entre entidades del mismo nivel de jerarquía en dos diferentes máquinas, se conocen como PROTOCOLOS, como se ilustra en la Fig 2-1. En realidad, ningún dato es directamente transferido entre una entidad de la capa N en una máquina, y otra entidad de la misma capa en otra (excepto en el caso de la capa más inferior). En lugar de ello, cada capa pasa datos e información de control a la capa inferior inmediata, hasta alcanzar la última capa donde existe una comunicación física con la otra máquina. El conjunto de capas y protocolos constituyen la ARQUITECTURA DE UNA RED.

Entre cada par de capas adyacentes hay una INTERFACE, la cual

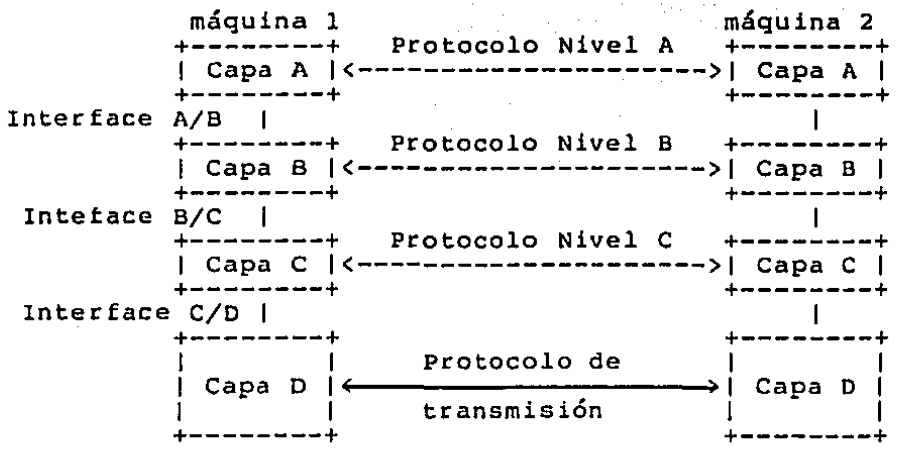


Figura 2-1: Capas,protocolos e interfaces

define las operaciones y servicios que la capa inferior ofrece a la superior, y así uno puede cambiar cualquier capa sin efecto ninguno sobre las demás, siempre y cuando mantenga invariante la interface de la capa reemplazada con sus vecinos inmediatos.

Entre los problemas claves que influyen en el diseño de los protocolos pueden mencionarse aquí:

- El direccionamiento; puesto que una red tiene muchos computadores y dentro de ellos múltiples procesos, se necesita alguna forma de direccionamiento para que un proceso en una máquina pueda especificar con quien desea dialogar.
- El modo de transferencia de datos; si los datos viajan en ambas direcciones pero no simultáneamente (half-duplex), o si viajan en ambas direcciones simultáneamente (full-duplex).
- El control de errores en las líneas, usando códigos para detectar y aún corregir errores.

- El control de flujo, para preservar el órden de los mensajes y para sincronizar el diálogo entre anfitriones de diferente velocidad.
- Resolver las diferencias entre las longitudes de mensajes aceptados por las diferentes máquinas, rompiendo los mensajes en pedazos y volviéndolos a reemsamblar en el destino.
- Las decisiones de ruteo en el caso de trayectorias alternas.
- El multiplexaje y demultiplexaje de varias conversaciones no relacionadas a través de un solo canal lógico.

2.1 El modelo propuesto por ISO

La International Standards Organization (ISO) ha propuesto un modelo de red estructurado en siete capas denominado Modelo de Referencia para la Interconección de Sitemas Abiertos (OSI), como se ilustra en la Fig. 2-2, y en las secciones 2.1.1 a 2.1.7 discutiremos brevemente cada una de ellas.

2.1.1 La capa física (The Physical Layer)

En el diseño de esta capa, los problemas son de tipo mecánico eléctrico, para asegurar la transmisión correcta de bits sobre un canal de comunicación. Los aspectos que interesan aquí tienen que ver con el medio de transmisión elegido, ya sean líneas telefónicas, cables coaxiales, microondas, satélites, etc; si la red usa la técnica de circuitos conmutados o paquetes conmutados. Su función es controlar el ancho de banda, multiplexar el canal de comunicación, modular las señales y detectar los errores en las líneas de transmisión, y aún corregirlos utilizando códigos para corrección de errores, aunque en la práctica es mas común

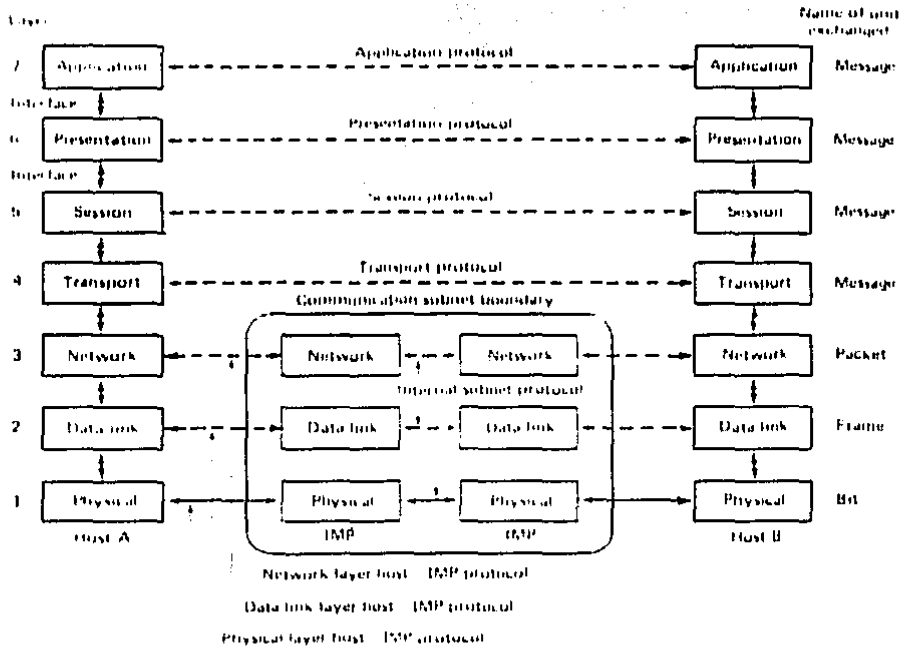


Figura 2-2: Modelo de referencia ISO

simplemente el detectar los errores y retransmitir los bloques erróneos.

2.1.2 La capa de control de datos (The Data Link Layer)

Su función es tomar el servicio de la capa física, y transformarlo en una línea que aparece libre de errores de transmisión. Esta tarea la realiza rompiendo los datos de entrada en marcos de datos (frames), transmitiéndolos secuencialmente y procesando los marcos de reconocimiento (ACKs) enviados por el receptor. Reconoce los límites de un marco, agregándoles cadenas especiales de bits al inicio y final de un marco.

Cuando el ruido en las líneas destruye un marco, esta capa se encarga de retransmitirlo. Sin embargo múltiples retransmisiones del mismo marco pueden producir duplicados, y es responsabilidad de esta capa resolver los problemas causados por daños, pérdida y duplicación de marcos, de modo que la capa superior pueda asumir que está trabajando con una línea virtual libre de errores. Se encarga además de sincronizar la comunicación entre un enviador rápido y un receptor lento, a base de mecanismos para conocer cuanto espacio de buffers tiene el receptor en ese momento.

2.1.3 La capa de la red (The Network Layer)

Esta capa controla la operación de la red interna. Entre otras cosas ésta determina las características principales de la interface nodo-anfitrión , acepta los mensajes desde el anfitrión orígen, los transforma en paquetes y los rutea en la red interna. Este ruteo puede hacerlo en base a tablas estáticas alambradas previamente en la red, o puede ser dinámico dependiendo de la carga en las líneas.

Controla la congestión en la red interna, cuando hay demasiados paquetes en las líneas de comunicación, e incluye también algunas funciones de contabilidad para el cobro de servicios.

2.1.4 La capa de transporte (The Transport Layer)

En contraste con las tres capas más inferiores, cuyas funciones son el ruteo de paquetes, la formación de marcos y su transmisión entre "máquinas adyacentes", la capa de transporte tiene la tarea

de proporcionar un servicio de transporte fin-fin confiable y eficiente entre los "procesos", mas bien que entre máquinas. Las diferencias entre las tres capas inferiores, y las capas 4 a 7 las cuales son fin-fin están ilustradas en la Figura 2-2; y los programas que comprenden la capa de transporte corren únicamente en los anfitriones, mientras que las tres capas inferiores, están presentes tanto en los anfitriones como en los nodos (IMPs).

Los problemas centrales de esta capa, son el asignar nombres y direcciones a los procesos, establecer y terminar conecciones, el control del flujo, multiplexaje, recuperación de errores y sincronización.

La capa de transporte determina también el tipo de servicio ofrecido a la capa de sesión, en suma a los usuarios de la red, pudiendo ser de CIRCUITOS VIRTUALES si entrega los mensajes en el órden en que fueron enviados; de DATAGRAMAS si se trata de mensajes aislados sin garantía en el órden de entrega o de eliminación de duplicados; o de DIFUSION si los mensajes se difunden a múltiples destinos.

La capa de transporte es a menudo implementada como parte del sistema operativo y se denomina ESTACION DE TRANSPORTE, en contraste la capa de la red es implementada en el anfitrión a manera de un manejador de entrada y salida, y las dos capas inferiores son normalmente implementadas en hardware.

2.1.5 Capa de sesión (The Session Layer)

La capa de sesión es la interfase entre el usuario y la red. Con esta capa el usuario negocia el establecimiento de una conección con un proceso de otra máquina. Una conección entre usuarios es llamada usualmente una sesión. Una sesión puede ser usada para permitir a un usuario registrarse (log-on) en un sistema remoto, o transferir un archivo entre dos máquinas.

Otras funciones de la capa de sesión corresponden al control y recuperación de fallas en las conecciones de transporte. Por ejemplo en sistemas de bases de datos distribuídas, es necesario que una transacción complicada, no falle en la mitad, pues dejaría la base de datos inconsistente. La capa de sesión puede asegurar que un grupo de mensajes no sean entregados al usuario, hasta que todos hayan arribado. Este mecanismo asegura que una falla en el software o hardware de la red interna, no deje a medias una transacción crítica.

En algunas redes, las capas de transporte y de sesión están entremezcladas, o esta última no existe en lo absoluto.

2.1.6 La capa de presentación (The Presentation Layer)

La capa de presentación proporciona una solución general para cierto tipo de funciones que son requeridas muy a menudo, en lugar de dejar que cada usuario de una solución aislada a sus problemas.

Entre los servicios proporcionados por esta capa, cabe

mencionarse la compactación de texto, el envío de mensajes en clave por motivos de seguridad, la conversión entre códigos de caracteres, el resolver las diferencias de formato entre los archivos de diferentes computadores, y la incompatibilidad entre diferentes tipos de terminales, etc. En definitiva la capa de presentación trata de aliviar estos problemas.

2.1.7 La capa de aplicación (The Application Layer)

El límite entre la capa de presentación y la de aplicaciones, marca la separación entre el dominio de los diseñadores de redes, y el dominio de los usuarios de redes. El contenido de la capa de aplicación, depende exclusivamente del usuario, pues él determina que programas desea correr y que protocolos usar. Además no se ha estandarizado ningún tipo de protocolos para dicha capa.

Sin embargo algunos problemas son comunes a muchas aplicaciones. Por ejemplo el lograr la transparencia de la red, escondiendo al usuario la distribución física de los recursos. El dividir un problema entre varias máquinas para obtener el mayor beneficio de la red. Las bases de datos distribuidas generan también problemas interesantes a la capa de aplicación. Cabe mencionar que la atención a esta capa es reciente, y los resultados son pocos.

2.2 Arquitectura de algunas redes y su interrelación con ISO

En esta sección haremos una comparación entre el modelo propuesto por ISO y la arquitectura de algunas redes conocidas, la red ARPANET,la red DECNET, y la red de XEROX, las cuales si bien utilizan el concepto de capas, ninguna de ellas se ajusta al modelo de referencia. En la Figura 2-3 y en las seciones 2.2.1 a 2.2.3 explicaremos la relación aproximada.

<table>
<tr><th></th><th>ISO</th><th>ARPANET</th><th>DECNET</th><th>XEROX</th></tr>
<tr><td>7</td><td>Application</td><td>User</td><td rowspan="2">Application</td><td>Filing, Printing Clearinghouse</td></tr>
<tr><td>6</td><td>Presentation</td><td>Telnet, Ftp</td><td>Courier</td></tr>
<tr><td>5</td><td>Session</td><td colspan="3"></td></tr>
<tr><td>4</td><td>Transport</td><td>Host - Host
Source To Destinat IMP</td><td>Network Services</td><td>Transport Interprocess</td></tr>
<tr><td>3</td><td>Network</td><td rowspan="2">IMP-IMP</td><td>Transport</td><td>Internetwork</td></tr>
<tr><td>2</td><td>Data Link</td><td>Data Link Control</td><td rowspan="2">Transmission</td></tr>
<tr><td>1</td><td>Physical</td><td>Physical</td><td>Physical</td></tr>
</table>

Figura 2-3: Relación de arquitecturas en varias redes

2.2.1 La red ARPANET

Esta red es un creación de ARPA (Advanced Research Projects Agency), y constituye el trabajo pionero en el campo, cubriendo ahora casi la mitad del mundo con mas de 100 computadoras

interconectadas.

El protocolo IMP-IMP corresponde a una mezcla de los protocolos de las capas 2 y 3 de ISO. La tercera capa contiene un elaborado mecanismo de ruteo, e incluye además un mecanismo que explícitamente verifica la correcta recepción en el nodo (IMP) destino de cada uno de los paquetes enviados por el nodo (IMP) orígen; estrictamente hablando este mecanismo constituye otra capa de protocolo y se lo denomina el protocolo IMP orígen a IMP destino, y aunque éste no existe en el modelo ISO, por sus características se acerca mas a la capa 3.

ARPANET tiene además dos protocolos de transporte, el protocolo original NCP diseñado teniendo en cuenta una red interna perfecta, este protocolo funciona bien dentro de ARPANET mismo, no así cuando se conecta a redes de datagramas no confiables, por lo cual se desarrolló el nuevo protocolo TCP. No existen las capas de sesión y presentación, aunque algunas de las funciones de la capa de presentación, están incluídas en ciertos servicios de alto nivel como Telnet para el manejo de sesiones remotas, y en el FTP para transferencia de archivos.

2.2.2 La red DECNET

Esta red fué desarrolada por Digital Equipment Corporation, y está orientada a establecer redes privadas de equipos Digital. Su arquitectura se conoce con el nombre de DNA (Digital Network Architecture).

En estas redes no existe distinción entre anfitriones y nodos (IMPs). DECNET es justamente una colección de máquinas (nodos), algunas de las cuales pueden correr programas de usuarios, y otras hacer conmutación de paquetes, o ambas cosas a la vez, pudiendo las funciones de cada máquina llegar a cambiar con el tiempo.

La arquitectura DNA tiene cinco capas, y la capa física, la capa de control de datos, la capa de transporte y la capa de servicios de la red, corresponden casi exactamente a las cuatro capas inferiores de ISO, excepto por el intercambio de nombres de las capas 3 y 4. No existe una capa de sesión, y la última capa de aplicación es una mezcla de las capas ISO de presentación y de aplicaciones.

2.2.3 La red de XEROX

Esta red es propiedad de XEROX Corporation. La arquitectura de sus protocolos puede subdividirse en cinco capas, y las funciones de cada una de ellas y su interrelación con ISO es la siguiente:

La capa de transmisión corresponde aproximadamente a las capas 1 y 2 y parte de 3 del modelo ISO, y comprende varios protocolos dependiendo de las disciplinas de transmisión, pudiendo ser Ethernet, líneas de 9600 bps usando protocolos HDLC, o X.25. La capa inter-red se aproxima mas a la capa 3 de ISO y su función es asignar direcciones de orígen y destino, el ruteo de los paquetes, y de escoger el medio de transmisión; sin embargo no

existe en esta capa ninguna garantía de secuencia de mensajes, ni supresión de duplicados, funciones encomendadas a la capa de transporte para comunicación entre procesos, al igual que la capa 4 de ISO. Esta nueva capa toma a su cargo las retransmisiones, el control de secuencia y duplicados, y el control de flujo, existiendo en esta capa algunas alternativas de protocolos, según el tipo de servicio requerido.

No existen protocolos correspondientes a la capa de sesión, existen sin embargo la capa de presentación o de enlace, para la estructuración de datos y su interacción entre procesos, y la capa de aplicación con diversos protocolos de alto nivel para menejo de archivos, funciones de impresión, distribución de mensajes, etc.

3.

INTERCONECCION DE REDES

3.1 Introducción

La proliferación de un gran número de redes de computadoras ha traído como consecuencia su interconección, y en particular con el surgimiento de redes locales, la interconección de éstas es algo imprescindible. La interconección puede ser motivada ya sea por economía o simplemente para atender las necesidades de los usuarios de una red local. Por ejemplo una red local, puede proporcionar un medio económico de conectar un cierto número de anfitriones dentro de una area pequeña a una o mas redes largas. Los ahorros así obtenidos resultan obvios comparados con la situación en que cada computador vaya a ser conectado a mas de una red larga. En lugar de ello, cada anfitrión está conectado solo a una red local y un anfitrión (u otro equipo especial) llamado "Compuerta" (Gateway) conecta la red local con cada una de redes largas, como se ilustra en la Figura 3-1

Los ahorros en costos son significativos, aún en situaciones en las cuales los anfitriones locales están conectados a una sola red larga, por dos razones: primero por que el hardware de la interfase para redes locales es menos caro que para redes largas y segundo porque solo se requiere un puerto a la red larga, en lugar de un puerto por cada computador local.

Entre las motivaciones para el uso de interconección de redes

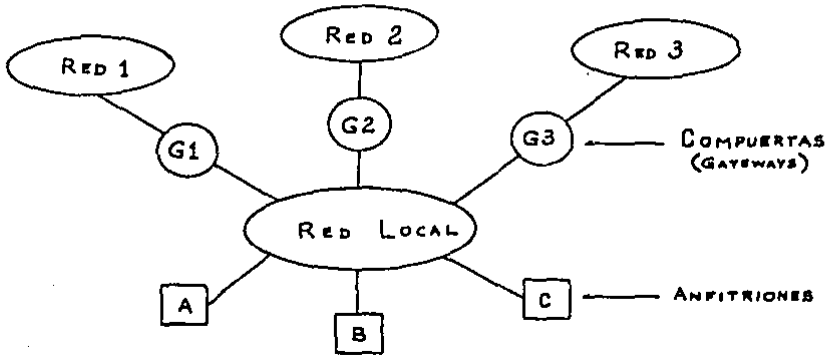


Figura 3-1: Interconeción de redes

pueden mencionarse:

- Establecer un sistema de correo, con el cual los mensajes entre los usuarios de redes locales, pueden intercambiarse a través de una red larga.
- El acceso a recursos especializados de cómputo, ocasionalmente requeridos por los usuarios de una red local, pero demasiado caros para mantenerlos localmente, por lo cual se necesita la interconección de la red local a la grande.
- La necesidad de comunicación entre las redes locales mantenidas por una empresa en sus centros principales.

La interconección de redes locales a redes largas, presenta sus problemas como también sus beneficios. En algún punto los protocolos usados dentro de la red local, deben ser compatibles con el de las redes largas. La compatibilidad puede ser alcanzada ya sea adoptando en la red local los protocolos de la red larga, o llevando a cabo la transformación de protocolos en los mensajes que pasan a través de la compuerta. Para enviar un mensaje a otra red, el anfitrión orígen direcciona el mensaje a

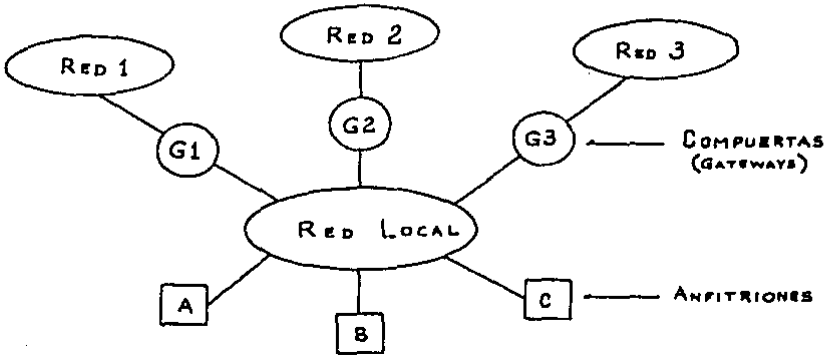


Figura 3-1: Interconeción de redes

pueden mencionarse:

- Establecer un sistema de correo, con el cual los mensajes entre los usuarios de redes locales, pueden intercambiarse a través de una red larga.
- El acceso a recursos especializados de cómputo, ocasionalmente requeridos por los usuarios de una red local, pero demasiado caros para mantenerlos localmente, por lo cual se necesita la interconección de la red local a la grande.
- La necesidad de comunicación entre las redes locales mantenidas por una empresa en sus centros principales.

La interconección de redes locales a redes largas, presenta sus problemas como también sus beneficios. En algún punto los protocolos usados dentro de la red local, deben ser compatibles con el de las redes largas. La compatibilidad puede ser alcanzada ya sea adoptando en la red local los protocolos de la red larga, o llevando a cabo la transformación de protocolos en los mensajes que pasan a través de la compuerta. Para enviar un mensaje a otra red, el anfitrión orígen direcciona el mensaje a

la compuerta, de la misma manera en que lo hace a otro anfitrión usando el protocolo estandard de transporte de la red. La compuerta convierte entonces el mensaje al formato de la segunda red, y direcciona éste a la próxima compuerta usando el protocolo de transporte de la segunda red.

3.2 Algunos problemas Inter-Red

3.2.1 Direccionamiento y ruteo

Uno de los principales problemas, es el asignar direcciones a las entidades de la inter-red, existiendo dos estrategias de direcionamiento: las direcciones "jerárquicas", y las direcciones "planas".

Las direcciones jerárquicas consisten en una secuencia de campos que pueden desglosarse, y guardan relación con la localización geográfica de la entidad, tal es el caso de la numeración telefónica que contiene el código del país, el código del area, y el número del abonado:

<dirección> = <pais><area><abonado>

En cambio las direcciones planas, no contienen ninguna relación de geografía, y pueden asignarse en base a un contador que es incrementado cada vez que se requiere una nueva dirección.

CCITT en su recomendación X.121 propone un sistema de direcionamiento jerárquico para redes públicas, consistente en un campo de 14 dígitos decimales como se indica en la figura:

la compuerta, de la misma manera en que lo hace a otro anfitrión usando el protocolo estandard de transporte de la red. La compuerta convierte entonces el mensaje al formato de la segunda red, y direcciona éste a la próxima compuerta usando el protocolo de transporte de la segunda red.

3.2 Algunos problemas Inter-Red

3.2.1 Direccionamiento y ruteo

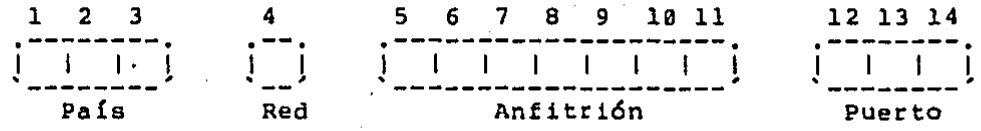
Uno de los principales problemas, es el asignar direcciones a las entidades de la inter-red, existiendo dos estrategias de direcionamiento: las direcciones "jerárquicas", y las direcciones "planas".

Las direcciones jerárquicas consisten en una secuencia de campos que pueden desglosarse, y guardan relación con la localización geográfica de la entidad, tal es el caso de la numeración telefónica que contiene el código del país, el código del area, y el número del abonado:

<dirección> = <pais><area><abonado>

En cambio las direcciones planas, no contienen ninguna relación de geografía, y pueden asignarse en base a un contador que es incrémentado cada vez que se requiere una nueva dirección.

CCITT en su recomendación X.121 propone un sistema de direcionamiento jerárquico para redes públicas, consistente en un campo de 14 dígitos decimales como se indica en la figura:



Para países que esperan tener muchas redes públicas, involucradas en tráfico internacional, se les han asignado múltiples códigos, y así por ejemplo USA tiene los códigos 310 al 329, permitiéndole manejar hasta 200 redes de tipo internacional.

Las direcciones jerárquicas hacen el ruteo mas fácil, puesto que para manejar un paquete en tránsito, basta identificar el primer campo de la dirección para decidir si el destino está en otro país, y rutear el paquete a la compuerta adecuada, sin la necesidad de enterarse de los detalles del direccionamiento interno dentro de cada país. En el país de destino, se hará uso del segundo campo para rutear el paquete dentro de las redes del propio país, y finalmente dentro de la red en cuestión se hará uso de los últimos campos para rutear el paquete al anfitrión de destino. Los algoritmos de ruteo pueden ser de tipo estático, en base a tablas que contienen las rutas para los posibles destinos, o pueden ser dinámicos, basando sus decisiones en estimaciones y medidas del tráfico actual y de la topología, pudiendo estos últimos tener un control centralizado, o estar distribuido en cada uno de los anfitriones que conforman la red.

Otra ventaja del direccionamiento jerárquico, es que cada país es dueño de decidir como asignar los anfitriones a sus redes, y

seleccionar para cada uno los puertos que convengan, sin necesidad de coordinar su asignación con otros anfitriones o alguna autoridad central. Una desventaja de este modo de direcionamiento, radica en que si un proceso o entidad emigra a otra máquina, no puede llevar consigo su dirección original.

Las direcciones planas, menos comunes tienen propiedades inversas, el ruteo se vuelve mas difícil, pero las entidades móviles conservan sus direcciones. La asignación de este tipo de direciones es mas complicada, para asegurarse de que no ocurran duplicados.

3.2.2 Fragmentación de paquetes

Cada red impone algún tamaño máximo en sus paquetes. Estos límites tienen varias causas, entre ellas:

- Hardware
- Sistema operativo (todos los buffers son de 512 bytes)
- Protocolos (número de bits del campo de longitud del paquete)
- Estándares (inter)nacionales
- Reducir probabilidad de errores que inducen retransmisiones
- Evitar el monopolio de un canal por paquetes muy largos

El problema se presenta cuando un paquete grande debe atravesar una red cuyo tamaño máximo de paquete es demasiado pequeño. Básicamente la única solución al problema consiste en que las compuertas rompan estos paquetes grandes en fragmentos, enviando cada fragmento como un paquete separado, y reensamblándolos en el destino.

Existen dos estrategias para reconstruir el paquete original.

La primera consiste en hacer que la fragmentación provocada por una red que soporta paquetes pequeños sea transparente a las redes subsecuentes por las cuales el paquete atraviesa. Esta opción se ilustra en la figura 3-2. Cuando un paquete sobredimensionado llega a una compuerta, ésta lo rompe en fragmentos. Cada fragmento es direcionado a la misma compuerta de salida, donde las piezas son reensambladas. De esta manera el paso a través de una red de paquetes pequeños es totalmente transparente, y las redes subsecuentes no se enteran de la fragmentación ocurrida.

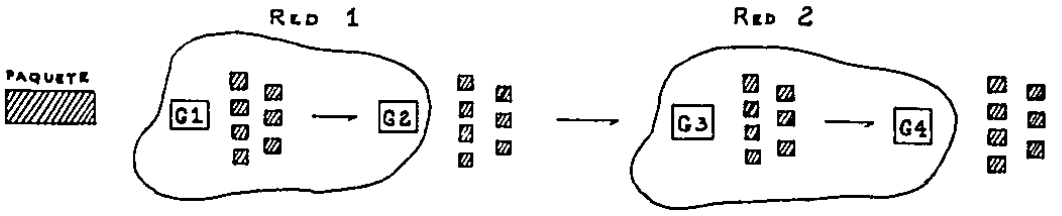
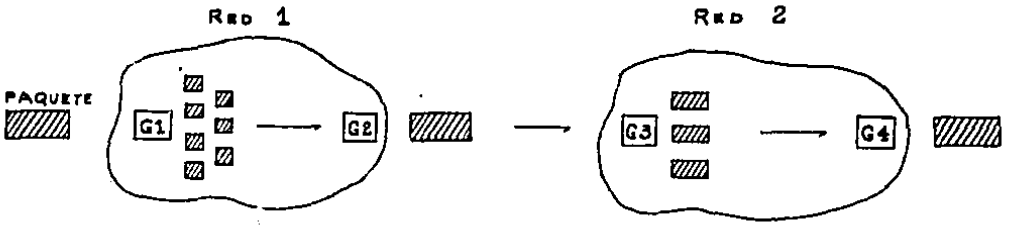


Figura 3-2: Fragmentación inter-red transparente

La fragmentación transparente es simple pero tiene algunos problemas. Por un lado la compuerta de salida debe conocer cuando ha recibido todas las piezas del paquete, así que un campo de cuenta o una marca de fin de paquete debe ser añadido a cada fragmento. Por otro lado, todos los paquetes deben salir por la misma compuerta, y un tercer problema es el overhead de repetidas

fragmentaciones y reensamblajes de un paquete grande que atraviesa varias redes de paquetes pequeños.

La otra estrategia de fragmentación consiste en no reensamblar los fragmentos en las compuertas intermedias. Una vez que un paquete ha sido fragmentado, cada pieza es tratada como si fuera un paquete original. El reensamblaje ocurrirá solo en el anfitrión destino. Ver la figura 3-3.



G1 fragmenta un paquete grande | G2 reensambla los fragmentos | G3 vuelve a fragmentar | G4 reensambla nuevamente

Figura 3-3: Fragmentación inter-red no-transparente

Este sistema tiene también algunos problemas. Por ejemplo requiere que cada anfitrión tenga la capacidad de reensamblar los paquetes. Otra dificultad es el "overhead" que la fragmentación de un paquete grande introduce, por cuanto cada fragmento debe llevar un encabezado, y este overhead permanece durante todo el tiempo que dura la transmisión. Una ventaja de este método es que pueden utilizarse multiples compuertas de salida.

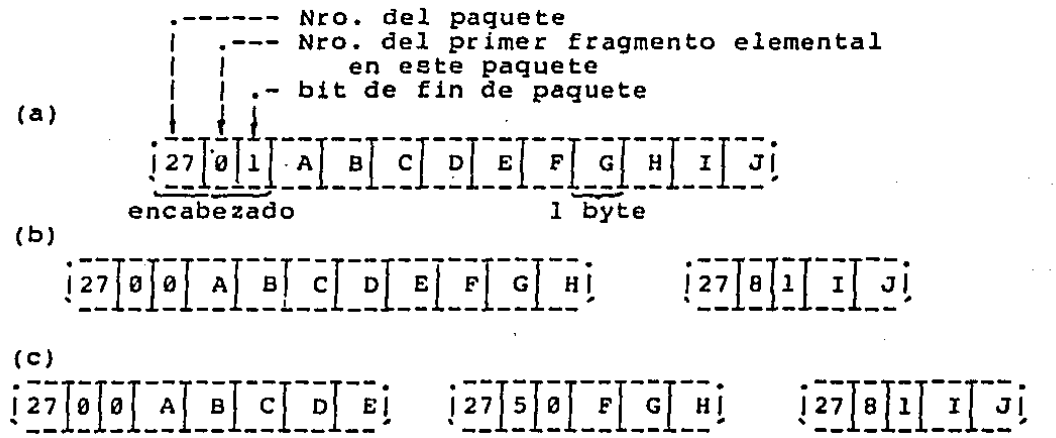
Cuando un paquete es fragmentado, los pedazos deben ser

numerados de tal forma que la cadena original pueda ser reconstruida. Una alternativa es una numeración en forma de árbol. Si el paquete 0 debe ser fragmentado, las piezas se numeran como 0.0, 0.1, 0.2,...etc. Si estos fragmentos a su vez se subdividen en la próxima compuerta, las piezas se numeran como 0.0.0, 0.0.1, 0.0.2...etc.

Sin embargo esta forma de numeración puede causar problemas en el caso de retransmisiones, pues si un paquete que originalmente es fragmentado en 4 pedazos 0.0, 0.1, 0.2, 0.3, y el fragmento 0.1 se extravía, este fragmento deberá ser retransmitido. Si embargo si la ruta elegida ahora por el enviador atraviesa una red que obliga a dividir el paquete original en solo dos pedazos, el paquete retransmitido 0.1 corresponderá a los fragmentos 0.2 y 0.3 enviados por la ruta anterior, provocando una reconstrucción errada del paquete.

Un sistema de numeración mejor consiste en definir un tamaño elemental de fragmento suficientemente pequeño que lo tolere cualquier red. Cuando el paquete es fragmentado todas las piezas son de tamaño elemental, excepto la última que puede ser más corta. Un paquete inter-red puede contener varios fragmentos por razones de eficiencia, y el encabezado inter-red debe proporcionar el número original del paquete, y el número del primer fragmento elemental contenido en el paquete, y como es usual debe existir un bit indicando que el último fragmento elemental contenido dentro del paquete inter-red es el último del

paquete original. En la figura 3-4 se ilustra el caso en que el tamaño elemental del fragmento es 1 byte.



Fragmentacion cuando el tamano elemental de datos es 1 byte
(a) Paquete original conteniendo 10 bytes de datos
(b) Fragmentos al pasar una compuerta con un tamano de 8 bytes
(c) Fragmentos al pasar una compuerta con un tamano de 5 bytes

Figura 3-4: Fragmentación en paquetes de tamaño elemental Estrategia de numeración

3.2.3 Problemas políticos

La interconección de redes, nos está conduciendo a estar más cerca del establecimiento de la teleinformática a nivel internacional. Sin embargo esto ha traído consigo varios problemas, la mayoría de corte político y económico. Por ejemplo la interconección de redes norteamericanas a redes europeas ha creado una situación no siempre sana para las redes europeas, por cuanto las tarifas de las redes norteamericanas son muy atractivas al usuario europeo, y para proteger sus intereses los

europeos han elevado tanto el nivel de las tarifas trasatlánticas, que parece ser un ejemplo de proteccionismo a través de tarifas.

4.

LA RED LOCAL DEL IIMAS

4.1 Descripción

En el Instituto de Investigaciones en Matemáticas Aplicadas y en Sistemas (IIMAS) se formó hace algún tiempo un grupo de investigadores, con el propósito de diseñar una red local para enlace de las computadoras de la institución. La figura 4-1 muestra la topología de interconección propuesta, y en las referencias [11] y [14] pueden consultarse las consideraciones que determinaron el diseño.

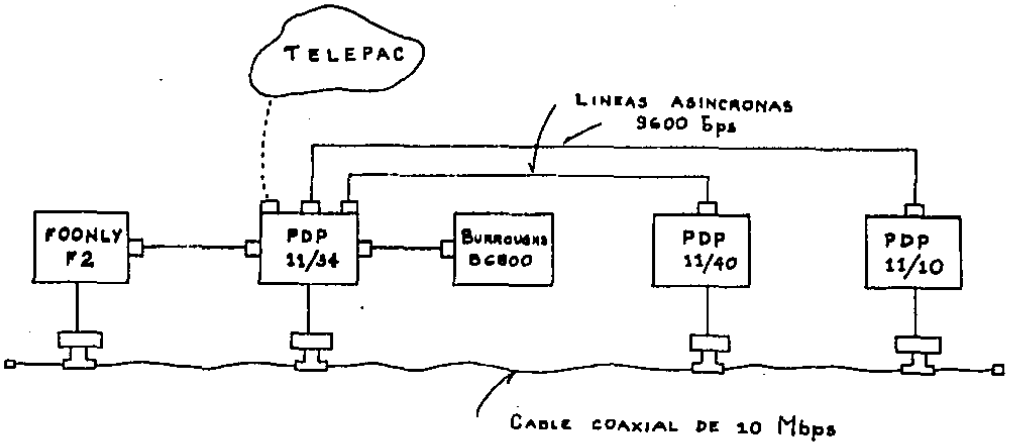


Figura 4-1: Topología de la red IIMAS

Los eqipos PDP-11 están conectados por líneas full-duplex punto a punto de 9600 bps usando interfases asíncronas tipo DL-11, y con la técnica guarda - reexpide. En breve se conectarán también

por medio de un cable coaxial que alcanza la velocidad de 10 millones de bits por segundo, con una técnica de difusión de paquetes en un medio compartido parecido a Ethernet, para lo cual se están construyendo equipos especiales consistentes en unos dispositivos denominados Controladores de Transmisión de Datos (CTD) y Transmisores - Receptores (TR).

Entre la PDP 11/34 y la Burroughs B6800 existe una conección temporal mediante líneas de 9600 bps half-duplex que permiten la transferencia de archivos. Igualmente la conección entre la FOONLY F2 y la PDP 11/34 es una línea full-duplex de 9600 bps que permite enviar listados desde la F2 a la impresora Printronix de la PDP 11/34. Estas conecciones son provisionales hasta que se logre el desarrollo de las Estaciones de Transporte para incorporar estas dos máquinas en la arquitectura descrita en la próxima sección.

Se tiene en miras la conección a la red pública TELEPAC, y por intermedio de esta a las redes locales de otras instituciones, como también a redes largas de otros países.

4.2 Arquitectura de la red

En [13] se establecieron ya los lineamientos de la arquitectura propuesta, y en esta sección nos limitaremos a una breve sinopsis. Este diseño atiende a los objetivos propuestos por ISO, aunque la arquitectura de referencia a sido simplificada a cuatro capas principales, correspondientes a la Capa de

Transmisión en el nivel inferior, la Capa Inter-red, la Capa de Transporte, y la Capa de Aplicaciones, como se muestra en la figura 4-2.

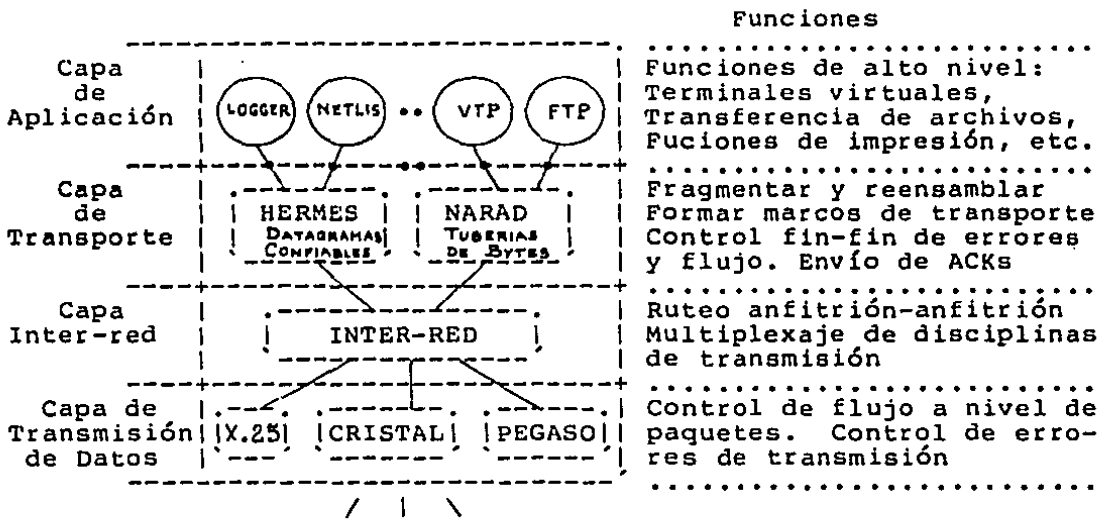


Figura 4-2: Arquitectura de la Red IIMAS

4.2.1 Capa de Transmisión

Constituye la más inferior en jerarquía de la red y en este nivel se encuentran los manejadores de las líneas, pudiendo existir varios según la disciplina de transmisión elegida. Por el momento solo existe CRISTAL, que es un protocolo para la operación bidireccional (full-duplex) de las líneas de baja velocidad, y permite transferir bloques de información de tamaño variable en forma transparente, o sea independiente de su contenido. PEGASO es el nombre asignado al manejador de acceso

al sistema de transmisión a alta velocidad en etapa de implementación, y en este nivel se incluirá también el protocolo X.25 para conecciones a redes públicas.

4.2.2 Capa Inter-red

Esta capa se encarga de escoger el medio de transmisión adecuado con el propósito de rutear el paquete a su destino. El protocolo de esta capa es de anfitrión a anfitrión.

4.2.3 Capa de Transporte

Proporciona a los procesos de aplicación la posibilidad de enviar o recibir cartas a procesos ubicados en cualquier procesador de la red. Se la conoce con el nombre de estación de transporte y su responsabilidad principal es el de asegurar la entrega de las cartas encomendadas a ella, siendo este tipo de consideraciones los que determinan el diseño de los protocolos para la interacción entre las estaciones de transporte. Los protocolos en esta capa son de tipo fin-fin y son de dos clases:

- El protocolo HERMES inplementado en esta capa ofrece el servicio de Datagramas Confiables, que acepta mensajes ya divididos en paquetes, y transporta (en la red) cada paquete independientemente de los otros. Cada paquete es retransmitido, mientras no se reciba el reconocimiento de su entrega, haciendo caso omiso de paquetes que llegen duplicados. No trata de ensamblarlos en mensajes, ni pone atención a su posición secuencial dentro del mensaje.

- El protocolo NARAD, actualmente en desarrollo y discutido en [7], ofrecerá el servicio de Tuberías de Bytes, cuyo propósito fundamental es el de liberar a los usuarios de los problemas de paquetes duplicados o en desórden, proporcionándole una conección puerto a puerto, a manera de una tubería bidireccional por la

cual se pueden intercambiar cadenas de bytes.

Un proceso en la capa de aplicaciones que desea enviar un mensaje a otro proceso, lo pasa a la capa de transporte (o sea, a la estación de transporte ubicada en su procesador). Dicha estación toma el mensaje, de tamaño limitado si el servicio solicitado es el de Datagramas, o de tamaño variable, si el servicio requerido es el de Tuberías de Bytes. En este último caso la estación rompe el mensaje en fragmentos y agrega a cada uno de ellos la información necesaria para entregarlos al proceso destino y reensamblarlos si es necesario. Cabe mencionar que las partes del mensaje junto con esta información de control se denominan Marcos de Transporte, y constituyen las unidades de transferencia entre la capa de transporte y la de transmisión. Además puesto que debe atender simultáneamente las necesidades de varios procesos en la capa de aplicaciones, la estación de transporte deberá realizar un multiplexeo (y a su vez demultiplexeo) de todos pedidos de la capa de aplicaciones. Por último cabe destacar que la capa de transporte, libera a los procesos de aplicaciones de la necesidad de tomar en cuenta cualquier aspecto particular de la capa de transmisión.

4.2.4 Capa de Aplicaciones

La constituyen todos los procesos de usuario que hacen uso de la estación de transporte. Los protocolos en este nivel tienen que ver ahora con el contenido de los datos y el control o manipulación de recursos. Tales protocolos se denominan

normalmente Protocolos de Control. En esta capa están los protocolos para el manejo de terminales virtuales (VTP), transferencia de archivos (FTP), funciones de impresión (NETLIS), etc.

4.3 Comentarios

La red IIMAS es una colección de máquinas que actúan a la vez como nodos y anfitriones. Las ventajas de su arquitectura jerárquica con capas independientes, resultan de su modularidad, ya que los cambios en una capa solo requieren cambios menores en las vecinas y facilitan el desarrollo y mantenimiento del sistema. Claro es además que cambios internos a cualquier capa son irrelevantes para las demás, siempre que estos no afecten las características de las interfaces entre las capas.

En los capítulos siguientes se discutirán las especificaciones de los protocolos en las tres capas mas inferiores, y los detalles de la implementación de éstos bajo los sistemas operativos RSX-11M y RT-11 de las máquinas PDP-11. Además durante la fase de prueba de las estaciones se desarrollaron algunos programas a nivel de aplicación, y se mencionan aquí dos de ellos que permanecen ahora como utilerías: el LOGGER para activar procesos remotos, y NETLIS que permite listar los archivos de otras máquinas en la impresora Printronix de la máquina PDP 11/34.

5.

LAS CAPAS DE TRANSMISION E INTER-RED

5.1 Estructura de la interface entre la red y el procesador

Todo el control de transmisión de datos en las líneas de comunicación entre las computadoras PDP-11 que integran la Red Local del IIMAS de baja velocidad, lo efectúan las interfaces DL11. Estas interfaces manejan una comunicación full-duplex entre el computador y un canal serial asíncrono, como lo son en este caso las líneas de comunicación. La naturaleza full-duplex de la interface hace factible el iniciar una operación de transmisión, mientras se mantiene pendiente una operación de recepción.

El DL-11 lleva a cabo operaciones de conversión de una cadena de bits representando caracteres, de serie a paralelo cuando estos se reciben por la línea, y de paralelo a serie cuando estos se transmiten por la línea, todo por medio de un circuito denominado UART (Universal Asynchronous Receiver-Transmitter). El sistema es muy flexible y ha sido programado para recibir y transmitir caracteres de 8 bits a una velocidad de 9600 bits por segundo. Adicionalmente se habilitó el chequeo de la paridad, para verificar la paridad par de cada caracter recibido.

La interface entre el DL11 y su programa manejador que corre en el computador PDP-11, es a través de un conjunto de cuatro registros direccionables. Ellos son: 1) Registro para el estado

de recepción (RCSR); 2) Registro para almacenar el caracter recibido (RBUF); 3) Registro para el estado de transmisión (XCSR); y 4) Registro para almacenar el caracter a transmitirse (XBUF).

Cuando una cadena de bits llega a través de la línea, el UART los acumula hasta formar un caracter completo, y entonces lo transfiere al registro de recepción RBUF almacenando el caracter en el byte inferior, y grabando en ciertos bits del byte superior información sobre si el caracter recibido es o no correcto. Al mismo tiempo en el registro RCSR se informa que un caracter está listo y se despierta un mecanismo de iterrupciones con el fin de que el manejador del dispositivo transfiera el caracter recibido hacia la memoria del computador.

La operación inversa de transmisión lleva a cabo una conversión de paralelo a serie del caracter transferido desde el UNIBUS. El registro XBUF almacena el caracter a transmitirse antes de ser serializado por el UART, y el registro XCSR se encarga de mantener la sincronía con el manejador informándole del momento en que puede transmitir otro caracter.

Los detalles del funcionamiento de esta interface y sus registros puede consultarse en el manual [6].

Cabe mencionar aquí que para reducir la tasa de los errores de transmisión en las líneas de conección de la red de baja velocidad, se construyeron sistemas especiales de hardware

de recepción (RCSR); 2) Registro para almacenar el caracter recibido (RBUF); 3) Registro para el estado de transmisión (XCSR); y 4) Registro para almacenar el caracter a transmitirse (XBUF).

Cuando una cadena de bits llega a través de la línea, el UART los acumula hasta formar un caracter completo, y entonces lo transfiere al registro de recepción RBUF almacenando el caracter en el byte inferior, y grabando en ciertos bits del byte superior información sobre si el caracter recibido es o no correcto. Al mismo tiempo en el registro RCSR se informa que un caracter está listo y se despierta un mecanismo de iterrupciones con el fin de que el manejador del dispositivo transfiera el caracter recibido hacia la memoria del computador.

La operación inversa de transmisión lleva a cabo una conversión de paralelo a serie del caracter transferido desde el UNIBUS. El registro XBUF almacena el caracter a transmitirse antes de ser serializado por el UART, y el registro XCSR se encarga de mantener la sincronía con el manejador informándole del momento en que puede transmitir otro caracter.

Los detalles del funcionamiento de esta interface y sus registros puede consultarse en el manual [6].

Cabe mencionar aquí que para reducir la tasa de los errores de transmisión en las líneas de conección de la red de baja velocidad, se construyeron sistemas especiales de hardware

(Line-Drivers) que convierten las señales RS-232 emitidas por las interfases DL-11, en una transmisión diferencial más inmune a ruidos o interferencias en las líneas.

5.2 Manejadores de líneas y su protocolo CRISTAL

Para el control de las líneas de comunicación se desarrollaron dos manejadores: RX para recepción y TX para transmisión. Su función es aceptar o transmitir bloques de bytes sin considerar el contenido, aunque debe reconocer el inicio y final de un bloque, además de detectar los errores causados por distorsiones en las líneas. El protocolo empleado a este nivel se conoce como PROTOCOLO DE CONTROL DE LINEA, y consiste en añadir al inicio y fin de un paquete de datos una secuencia especial de bytes como se indica a continuación:

<SYN><SYN><SYN><SYN><SYN><SOH>.....datos...<ETX><BCC>

Los caracteres <SYN> sirven como caracteres de sincronía, y el receptor debe recibir por lo menos dos de ellos (contiguos) antes de esperar un <SOH>. El caracter <SOH> identifica el inicio de los datos de un bloque y <ETX> su final. Además se incluye un byte de chequeo <BCC> calculado como el OR exclusivo de la cadena comprendida entre los caracteres <SOH> y <ETX> incluyendo estos. Si dentro de la cadena de datos ocurren caracteres de control (como son <SOH>,<ETX>,<SYN> y <DLE>) el manejador transmisor los distingue anteponiendo a cada uno de ellos el caracter <DLE>.

El manejador de transmisión TX se encarga de enviar los caracteres a la línea (o sea al DL-11), y de preparar los caracteres de control incluído el cálculo del BCC. En cambio el manejador de recepción RX debe detectar el arribo de una cadena de datos, eliminar los caracteres de control, verificar si hubo algún error registrado por el hardware a causa de ruido, distorsiones en la línea, o superposición de caracteres, debe iniciar el cálculo del BCC de este paquete, filtrar los <DLE> que enmascaran caracteres de control dentro del texto, y finalmente comparar el BCC calculado con el que arriba por la línea, e informar del éxito o fracaso de la operación. Cualquier error provoca que el paquete se rechaze sin que el usuario se percate de ello.

5.3 La Capa Inter-red

Esta capa la constituye el software que se encarga del ruteo de los "Marcos de transporte" y de seleccionar la disciplina de transmisión, que por el momento corresponde únicamente a la de líneas de baja velocidad, pero que en el futuro deberá integrar además el de la línea de difusión de alta velocidad (PEGASO), y la de X.25 para una conección eventual a la red pública TELEPAC.

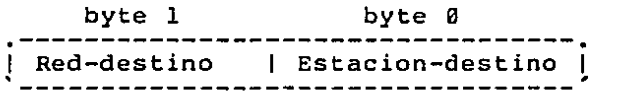


Figura 5-1: Encabezado Inter-red

Esta capa añade al marco de transporte un ENCABEZADO INTER-RED conformado por dos bytes con el formato indicado en la figura 5-1, y su propósito es facilitar el ruteo de los marcos, por ahora únicamente entre las estaciones de la red local. El ruteo de los marcos es muy simple y está implementado en cada máquina en base a una tabla estática que contiene el identificador de la línea de transmisión de acuerdo a la estación de destino. Sin embargo esta capa deberá recibir especial atención en el futuro cuando se incremente el número de componentes de la red, y haya que integrar algoritmos de ruteo más complejos.

6.

LA ESTACION DE TRANSPORTE EN EL SISTEMA RSX-11M

6.1 Antecedentes

El desarrollo del software para las estaciones de transporte en el sistema RSX-11M, para el caso de las PDP 11/34 y 11/40 ha ido evolucionando como resultado de la experiencia adquirida en este campo. Originalmente este software fue concebido como una tarea normal desarrollada en PASCAL y mas tarde traducida al lenguaje "C". Sin embargo, de las pruebas a las cuales hemos sometido este sistema hemos detectado varios inconvenientes acarreados por diversos factores, principalmente porque siendo un programa normal no se tiene acceso directo a las facilidades del sistema operativo. Se ha detectado por ejemplo un alto "overhead" en la comunicación de un proceso de aplicación con la estación de transporte, por el intercambio de mensajes tanto para despertar la atención del otro, como para solicitar o entregar servicios. De hecho la interface entre ellos hace uso de una area común fija, en donde se encuentran los buffers para recepción por las líneas y un número limitado de buffers de envío, lo cual en determinado momento puede restringir el uso eficiente de la red y limitar el número de usuarios. Además es necesario ligar con cada tarea de la capa de aplicaciones una cantidad considerable de código para manejar el acceso a dicha area común, y para los mecanismos de enviar solicitudes y recibir mensajes desde la estación de transporte.

6.2 Evolución del software

El factor determinante en la evolución de este software, fue la idea de establecer una similitud entre las funciones de una estación de transporte en la red y las que lleva a cabo el sistema de acceso a los archivos de una máquina. Es claro que una tarea que requiere abrir un archivo o cerrarlo, o leer o escribir datos en los mismos, solicita dicha operación al Sistema Operativo a través de una directiva QIO dirigida al manejador de la unidad de disco. Sin embargo no es posible ni conveniente implementar manejadores de complejidad tal que puedan atender a pedidos de esta naturaleza. Por este motivo el Sistema Operativo dispone de un mecanismo para interceptar dichos pedidos y pasarlos a un proceso especial (Ancillary Control Process) conocido con el nombre de F11ACP, el cual los traduce en uno o mas pedidos de menor complejidad, para a su vez regresar éstos al manejador.

De la misma manera la solicitud de envío de un mensaje por la red es una operación complicada, involucrando el uso de varios macanismos como son la formación de los "Marcos de Transporte", su envío por la ruta adecuada, la retransmisión de estos al no recibir un ACK en un tiempo preestablecido, etc. Además resultará muy conveniente el poder solicitar las operaciones de transporte con un simple QIO, el cual en forma transparente al proceso solicitante se dirija a un proceso especial el ACP de la red, que se encarge de realizar todos los aspectos del protocolo

correspondiente.

Por estos razonamientos se decidió construir para las dos PDP-11 que operan bajo el sistema operativo RSX-11M, una estación de transporte basada en un ACP y que no difiere de la anterior a nivel de arquitectura ni de protocolos, sinó en la forma de atención a sus funciones. Esta estación es ahora una tarea privilegiada escrita en ensamblador, intimamente ligada al sistema operativo y que se encarga de interceptar todo el proceso de entrada y salida a través de las líneas de conexión entre las computadoras a fin de realizar todo el trabajo adicional necesario, como el manejo de protocolos, ruteo de paquetes, comunicación con el usuario, control de errores fin-fin, manejo de timeouts, etc.

Las ventajas logradas con este tipo de estación, resaltan en la disminución drástica del overhead de comunicación entre tareas, la eliminación del area común de la interface con la capa de aplicación permitiendo ahora que los mismos buffers del usuario se utilicen en el envío de cartas, una simplificación notable de la interface con el usuario, y un mayor control y recuperación de situaciones anormales provocadas por los programas de aplicación.

El inconveniente principal para el desarrollo de este tipo de software, radica en que Digital Equipment Corporation (DEC) no incluye en ninguna parte de su documentación como se puede implementar un ACP. De hecho ellos no consideran dentro del

ámbito de los usuarios de sus sistemas, ni necesidad ni conveniencia de hacer ACPs por los mismos. Esta información tuvo que ser extraída de los listados del Sistema Operativo específicamente de los módulos $DRQIO, $IOSUB y $REQSB.

6.3 Detalles de la implementación como ACP

La implementación del ACP de la red está ligado intimamente al desarrollo previo del manejador de un pseudo dispositivo "NX", al cual el usuario dirijirá las solicitudes, antes de que sean interceptadas por el ACP. Este manejador tiene en sus estructuras de datos algunas características especiales a mencionarse:

- Los códigos de las funciones permisibles se declaran en el DCB (Device Control Block) del manejador, marcándolas como funciones ACP y de control, y su valor será mayor a siete (7).
- Al asignar el UCB (Unit Control Block) debe declararse que se trata de un dispositivo montable como un canal de comunicación por medio de los bits DV.MNT y DV.COM, y se marcará el bit US.MNT para indicar que la unidad está inicialmente desmontada.
- Adicionalmente el UCB incluirá dos palabras adicionales para almacenar en ellos un apuntador al TCB (Task Control Block) del ACP de la red (NETACP), y un apuntador al VCB (Volume Control Block). Estas dos palabras son inicializadas con cero, tomando su valor correcto el momento de montar el ACP a través del comando MOU del MCR.

6.4 Flujo general de operación

Con el propósito de ilustar la interacción entre el usuario y la estación de transporte, en el gráfico 6-1 se muestra el flujo de las solicitudes y mensajes intercambiados entre ellos. Los

procesos involucrados coresponden al programa de aplicación, la estación propiamente dicha (NETACP), el manejador de un pseudo dispositivo de comunicación entre el usuario y el ACP, y los manejadores de las líneas de transmisión. Se supone además que estos procesos están inmersos en el sistema operativo, que en definitiva es quien controla las actividades y rutea los pedidos entre uno y otro.

En esta figura las solicitudes del usuario dirigidas al ACP a través del manejador NX, y las emitidas por el mismo ACP a las líneas de transmisión se muestran como líneas continuas de un solo trazo. Las líneas de trazo doble que parten desde el proceso de usuario, atraviezan el ACP (líneas entrecortadas), y finalmente llegan a los manejadores de las líneas de transmisión, representan los mensajes que el usuario envía a las líneas y que son tomados directamente desde sus buffers sin almacenamientos intermedios. Las líneas de trazo doble que van desde los manejadores de las líneas de recepción al ACP, representan los mensajes que arriban por las líneas y son recibidos en buffers del propio ACP. Las líneas dobles que parten del ACP al usuario, representan la entrega de dos tipos de mensajes, los que arribaron por las líneas y que son copiados a los buffers del usuario desde los buffers del ACP, y los devueltos por el ACP como respuesta a las solicitudes emitidas por el usuario.

Para detallar el flujo de la operación hemos elegido como ejemplo la actividad de envío de una carta:

IMPLEMENTACION DE LA ESTACION DE TRANSPORTE EN P D P - 11 BAJO EL SISTEMA OPERATIVO R S X - 11 M

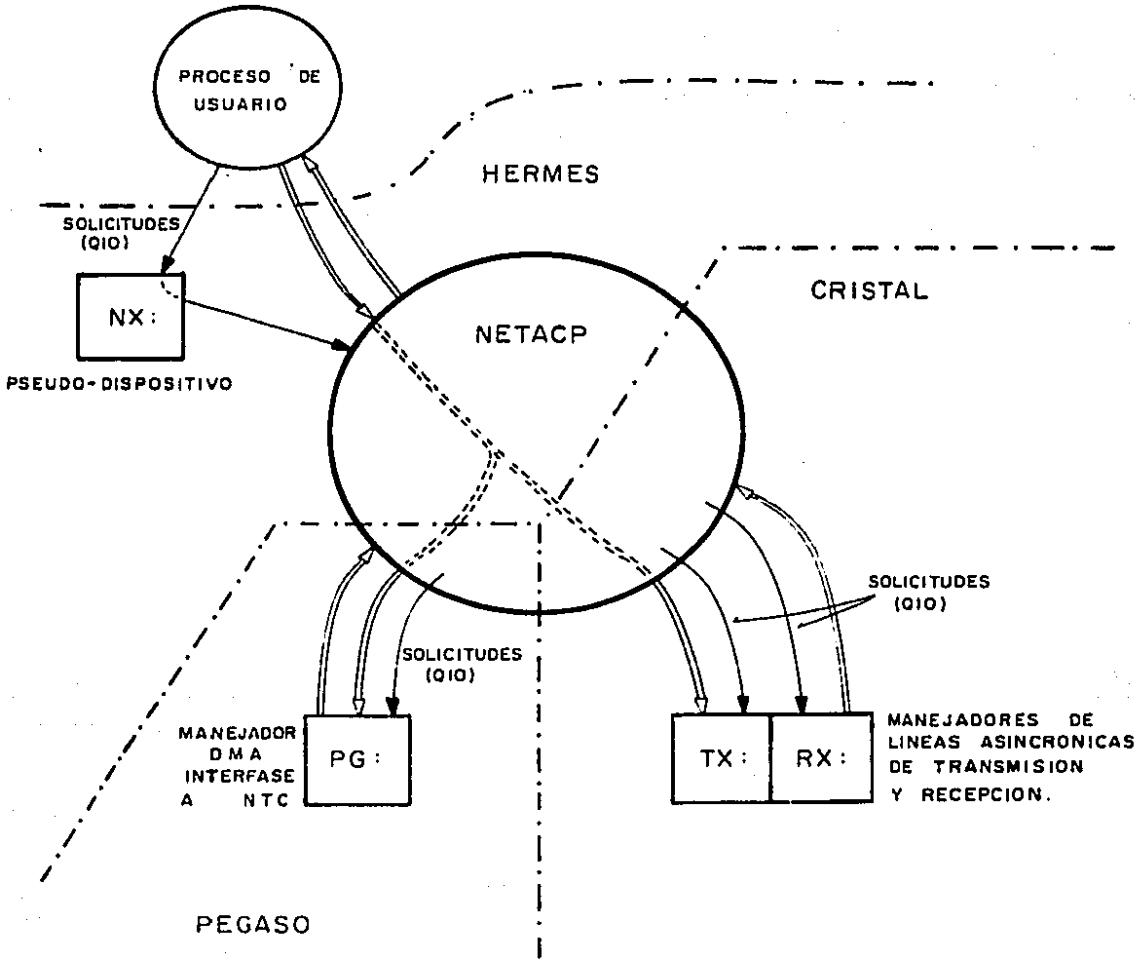


Figura 6-1: Diagrama de Operación

1. El proceso del usuario (1) emite una directiva QIO cuyos parámetros especiales corresponden a la dirección del buffer con la carta, el tamaño del texto, y las direcciones de orígen y destino.

2. El Ejecutivo toma ahora el control de ejecución y el despachador de directivas $EMTRP transfiere ésta al módulo $DRQIO.

3. $DRQIO verifica que la función es Legal y de Control y construye un paquete de I/O, insertando textualmente en él los parámetros especiales de la directiva, y finalmente rutea el paquete y lo pasa al manejador del pseudo dispositivo NX (2).

4. El manejador NX toma el control del paquete de I/O inmediatamente, (es decir antes de que este haya sido puesto en su cola de pedidos) con el fin de verificar que la función solicitada corresponde al envío de un mensaje, y pueda chequear y relocalizar la dirección del buffer con la carta, (calcular su dirección física) en el contexto de la tarea del usuario y finalmente insertar el paquete en su propia cola de pedidos, para que el mismo manejador NX lo recupere ahora en forma normal a través de la rutina del sistema $GTPKT.

5. La rutina $GTPKT examina el paquete de I/O y al verificar que la función solicitada es de tipo ACP, inserta el paquete en la cola de recepción de NETACP (3). Cabe destacar que todas las funciones legales soportadas por el manejador del pseudo dispositivo NX son de tipo ACP, y por consiguiente al llamar a la rutina $GTPKT, todos los paquetes de I/O para este dispositivo pasarán a la cola de recepción de NETACP.

6. El Ejecutivo activa ahora el ACP de la Red (NETACP) apagando su bit de stop.

7. NETACP es una tarea privilegiada que extrae directamente las solicitudes de su cola de recepción. Si la función solicitada corresponde en este caso al envío de una carta, procede a formar el marco de transporte, inicializar un contador de retransmisiones, especificar un tiempo en espera (time-out) para el reconocimiento de su entrega, rutear el marco y solicitar su envío al manejador de la línea de transmisión correspondiente (4).

8. NETACP informa al usuario del resultado del envío así como de cualquier otra solicitud recibida, a través de

la rutina del sistema $IOFIN, que se encarga de escribir el resultado en una area especificada por el usuario, activar una rutina asíncrona o prender una bandera para despertar su atención, según la preferencia del usuario.

9. Cuando el ACP a terminado su trabajo y no hay mas solicitudes en su cola de recepción puede detener su ejecución, prendiendo su bit de stop a través de la llamada a la rutina del sistema $STPCT.

6.5 Características especiales

Una de las características notables de NETACP es su facultad para accesar directamente los buffers de los usuarios, a través del uso directo del mecanismo para el manejo de memoria de los sistemas PDP-11. En las siguientes líneas describiremos someramente este mecanismo, aunque para mayores detalles, el lector puede referirse a [9].

Los sistemas PDP-11 tienen una longitud de palabra de 16 bits, sin embargo el UNIBUS y el CPU pueden manejar direcciones de hasta 18 bits de longitud, y es así que mientras una palabra puede contener referencias a direcciones en el rango de 32K palabras (64K bytes), el CPU y el UNIBUS pueden referirse a direcciones de hasta 128K palabras (256K bytes). Esta característica ha sido aprovechada para extender la memoria física de la máquina hasta 128K palabras de 16 bits y permitir que varios programas de usuario con un tamaño máximo de 32K palabras, residan simultáneamente en memoria. El manejador de memoria interpreta las direcciones de 16 bits como direcciones virtuales, y combinando estas con la información almacenada en

ciertos registros de hardware llamados REGISTROS DE PAGINAS ACTIVAS (APRs), calcula la dirección física de 18 bits en forma automática y transparente al usuario.

Cada registro APR asigna o mapea las direcciones virtuales a direcciones físicas de una página de 4K palabras en cualquier parte de la memoria física. La memoria de la máquina se considera dividida lógicamente en bloques de 32 palabras y los registros de mapeo almacenan la dirección de inicio de una página, expresada como el número del bloque físico en los primeros 12 bits.

La información básica necesaria para la construcción de la dirección física se extrae de la dirección virtual, la cual se interpreta de la manera siguiente:

- Los bits 13-15 determinan el número del registro de mapeo APR0-APR7.
- Los bits 0-12 contienen la dirección relativa al inicio de una página (2^{13} = 8K bytes), donde los bits 6-12 contienen el número del bloque dentro de la página (2^{7} = 128 bloques), y los bits 0-5 contienen el desplazamiento dentro de ese bloque (2^{6} = 64 bytes).

La construción de la dirección física es ilustrada en la figura 6-2.

Ahora bien para que NETACP pueda accesar los buffers del usuario es indispensable que conozca su dirección física, y de esto se encarga el manejador NX el cual antes de pasar las solicitudes al ACP, transforma las direcciones virtuales de los

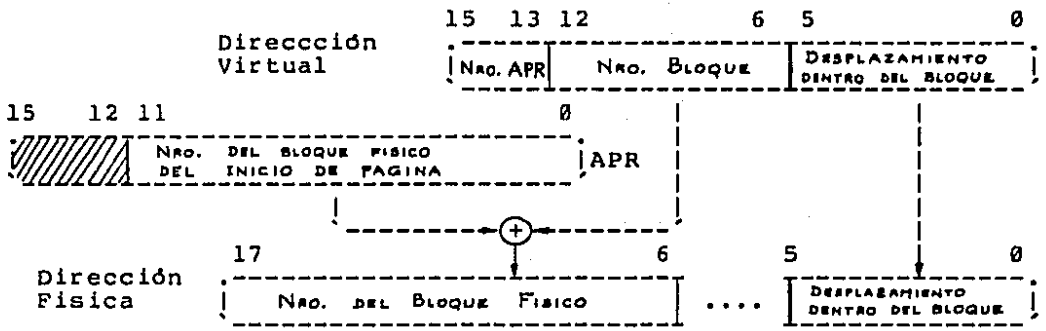


Figura 6-2: Construcción de una dirección física

buffers en direcciones físicas en el contexto de la tarea del usuario, almacenando el resultado en dos palabras, donde la primera contiene el número del bloque físico, y la segunda el valor del desplazamiento dentro del bloque (DIB). Además se preden los bits 14 y 15 de esta segunda palabra con el propósito de que este desplazamiento, al ser tratado por el ACP como una dirección virtual, le corrresponda el registro de mapeo APR6.

El procedimiento de acceso empleado por NETACP consiste en salvar el contenido de su registro actual de mapeo APR6 en el stack y cargarlo luego con la nueva constante de relocalización, correspondiente al contenido de la primera palabra de la dirección física de un buffer. La segunda palabra se trata normalmente como si fuese la dirección virtual del buffer del usuario, y el manejador de memoria se encarga de transformar ésta en la dirección física correcta sirviéndose del nuevo contenido de APR6.

Sin embargo el código del ACP que maneja este tipo de operaciones no deberá estar mapeado con APR6, ya que no es posible direccionar el código del ACP y el buffer del usuario simultáneamente, pues aunque les corresponde el mismo APR las constantes de relocalización son diferentes para cada caso. Además si se toma en cuenta que el programa NETACP es una tarea privilegiada con acceso a las rutinas del ejecutivo, su APR inicial de mapeo será el APR4 o el APR5 según el tamaño con el que se generó el sistema operativo, el cual consumirá los primeros registros de mapeo. Por lo tanto el código de las rutinas que accesan los buffers de usuarios deberán localizarse con cuidado al inicio del programa.

Cabe mencionar también que el código de NETACP no podrá sobrepasar el tamaño de 12K u 8K palabras, según que su registro inicial de mapeo sea el APR4 ó APR5 respectivamente. Ninguna parte de su código deberá estar mapeado con el APR7, por cuanto esta tarea accesa la "Página de I/O" y por tanto el APR7 queda reservado para dicha función.

7.

LA CAPA DE TRANSPORTE: IMPLEMENTACION DE HERMES EN EL SISTEMA RSX-11M

7.1 Descripción y especificación del servicio HERMES

Aunque HERMES puede multiplexar los pedidos de varios usuarios a la vez y las actividades solicitadas repercuten en otras que se ejecutan asincrónicamente. Con el propósito de su documentación describiremos las actividades principales junto con las estructuras de datos que manejan, como si se tratasen de actividades completamente independientes y de ejecución secuencial. Estas actividades se reducen a:

1. Inicializar la estación de transporte
2. Activar y liberar puertos públicos o privados
3. Envío de datagramas
4. Recepción de mensajes por las líneas
5. Envío de ACKs
6. Manejo de time-outs
7. Revisión de las tareas que usan la estación
8. Finalizar la actividad de la estación de transporte

Para hacer mas objetiva la descripción que sigue, se utilizarán en ella los nombres simbólicos de las constantes empleadas en NETACP (v.g. NBFREC = número de buffers del ACP para la recepción de mensajes que llegan por las líneas).

7.1.1 Inicializar la estación de transporte

La operación de montar el ACP de la red a través del comando MOU, ocasiona que el MCR genere un paquete especial de I/O, donde inserta como código de función el valor 5 y como parámetros especiales las direcciones del VCB (Volumen Control Block) y el de un bloque de comunicación (Communication Parameter Block).

Este paquete lo pasa directamente el MCR a la cola de recepción del ACP, explicándose de esta manera como pudo filtrarse una función que no es de tipo ACP, y hemos aprovechado este pedido para inicializar la estación de transporte con las siguientes acciones:

- Devolver al ejecutivo el area correspondiente al "Communication Parameter Block", ya que no es utilizado en ningún momento por el ACP.
- Asignar números lógicos a las líneas de comunicación de datos. Las líneas de recepción corresponden a las unidades del dispositivo RX y las de transmisión al dispositivo TX. Estos números se asignan secuencialmente a partir del valor inicial LUNINI, comenzando por las líneas de recepción y luego las de transmisión. En la figura 7-1 se muestra esta asignación para el caso de la Estación ET-1 (PDP 11/34) de la cual parten dos líneas de recepción y dos de transmisión a las estaciones ET-2 y ET-3.

  La asignación de estos números lógicos, determina la tabla de ruteo para cada estación de transporte.
- Estas líneas de comunicación son ligadas al ACP a fin de que los usuarios no tengan acceso directo a ellas, si no es a través de la estación de transporte.
- El ACP dispone de NBFREC buffers para la recepción de los mensajes que arriban por las líneas, y se sirve de ellos para activar y mantener pendientes NLEC lecturas por cada línea de recepción, a fin de captar los mensajes que llegan en cualquier momento.

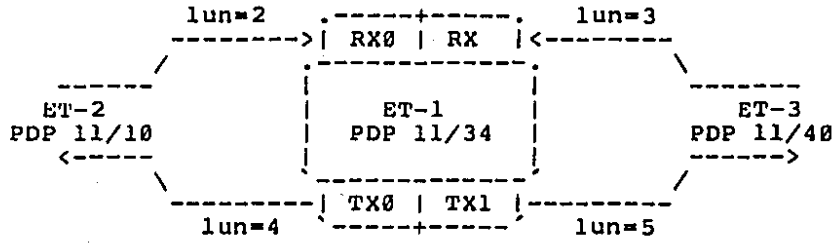


Figura 7-1: Identificadores (LUNs)

- Se activa el LOGGER que es una tarea complementaria del ACP.
- Finalmente se especifica un time-out de limpieza, para activar el "Revisador de tareas", que se encarga de examinar de cuando en cuando la lista de tareas activas y eliminar aquellas que terminaron sin liberar sus puertos en la estación de transporte.

El nombre de las rutinas involucradas en estas actividades corresponde a: MOUNT, INITPT, REZAGO, LNREAD y BUSBUF.

7.1.2 Activar y liberar puertos

La estación de transporte concede a los procesos que utilizan sus servicios un identificador único denominado puerto. Internamente HERMES define estos puertos por medio de registros como se ilustra en la figura 7-2, guardando en la primera palabra el estado del puerto, en la segunda un apuntador al TCB (Task Control Block) de la tarea propietaria del puerto, las dos palabras siguientes son utilizadas como ligas a las solicitudes

para recepción y envío de datagramas a través de este puerto, y las dos palabras restantes son empledas por NARAD [7] para el servicio de tuberias de bytes (Byte-pipes).

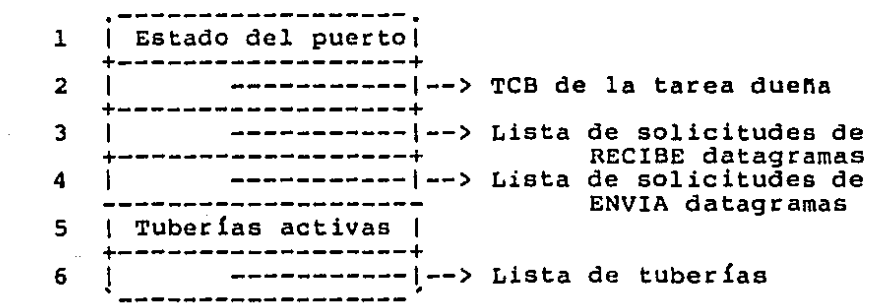


Figura 7-2: Estructura de un puerto

Los registros descriptores de los puertos son extraídos del area de memoria dinámica del ejecutivo con cada solicitud de ACTIVAR PUERTO, y se ligan a una lista de puertos activos interviniendo para el efecto las rutinas ACTPUB, ACTPRV e CRPTDB.

La diferencia entre los puertos públicos y privados radica unicamente en el procedimiento de su asignación. Los primeros se asignan a cualquier tarea y su identificador puede variar dependiendo de la disponibilidad de los puertos en el momento de la asignación; en cambio los puertos fijos corresponden exclusivamente a una tarea particular y su idenficador es siempre el mismo.

Es imprescindible el que una tarea antes de concluir su

operación libere los puertos que activó, proceso que no solo representa el hecho de eliminar el registro de la lista de puertos activos y devolverlo al ejecutivo, sinó que adicionalmente se cancelan las solicitudes de envío o recepción de cartas que pudiesen estar pendientes en este puerto. Intervienen en esta actividad las rutinas LIBPTO y KILLPT.

7.1.3 Envío de Datagramas

En esta actividad la respuesta de HERMES al envío de un datagrama puede no ser inmediata, ya que la estación espera la llegada del acuse de recibo para garantizar al usuario que el mensaje llegó efectivamente a su destino. Sin embargo durante este intervalo HERMES puede admitir otras solicitudes o atender a otras que quedaron pendientes. El flujo general de operación es el siguiente:

- Hemos mencionado ya que toda solicitud llega a la estación a través de un paquete de I/O, y para el caso presente este lleva entre sus parámetros, la dirección del buffer portador del texto, su tamaño en bytes y las direcciones de orígen y destino. La rutina SENDGM verifica estos datos, y cualquier error provoca la cancelación inmediata del pedido, informando la causa en el bloque de I/O estatus especificado por el usuario.

- Si la verificación es satisfactoria, la estación busca un identificador para la carta (rutina DAIDCA) y procede a construir el Marco de Transporte, preparando el encabezado de la carta dentro del espacio reservado en el mismo buffer del usuario con el formato que se indica en la figura 7-3.

- El paquete de I/O con la solicitud original de envío es conservado por HERMES, ligándolo al puerto de orígen en espera del arribo del reconocimiento de su entrega, y con el propósito de manejar el protocolo de

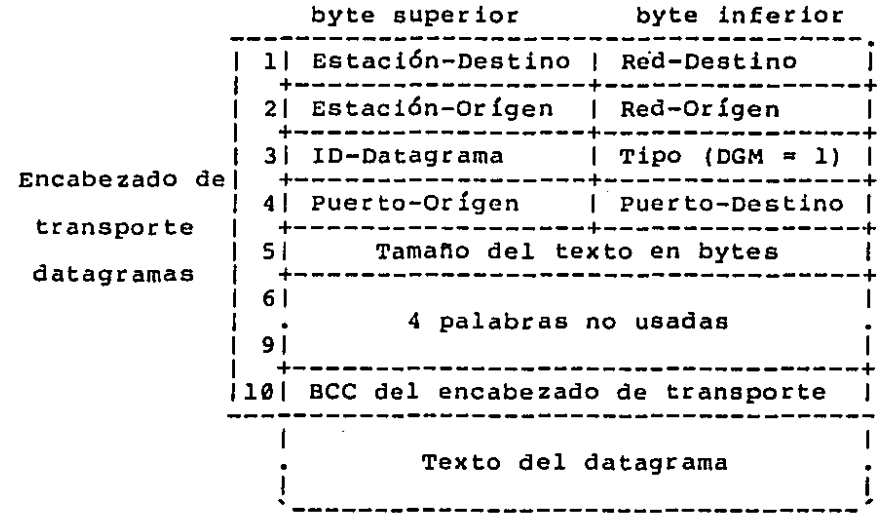
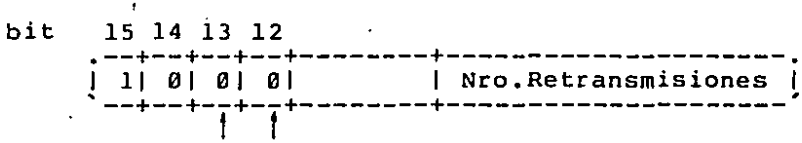


Figura 7-3: Marco de transporte de HERMES

retransmisiones y time-outs almacena adicionalmente ciertos datos de contexto, como son el identificador de la carta, junto con el de su puerto de orígen, un contador de retransmisiones y el estado de la carta. Las dos últimas palabras en el paquete de solicitud son utilizados como un bloque de I/O estatus para la directiva QIO que lanza el paquete a la línea. En la figura 7-4 se muestra la disposición de estos datos:

- En el campo de ESTADO de la carta, los bits 14 y 15 son cero y uno respectivamente e identifican que el buffer con la carta es del usuario. El bit 13 es cero o uno según que la carta esté con transporte o transmisión. El bit 12 igualmente es cero o uno según que la carta recibió o no el acuse de recibo.



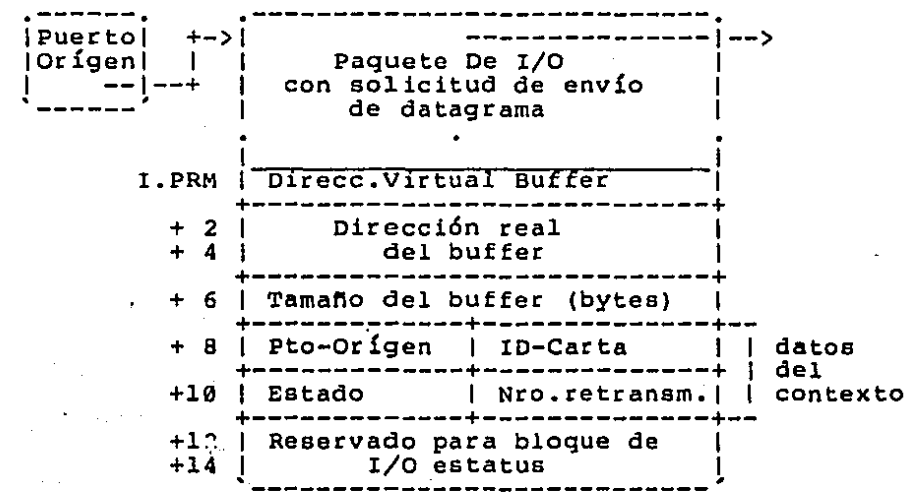


Figura 7-4: Solicitud de envío de datagrama en espera de ACK

```
|  |_ 1 si recibió ACK
|____ 1 si el buffer está con transmisión
```

- La estación solicita un time-out de tres segundos (rutina TMOPED), tiempo en el cual si no se recibe el ACK, el datagrama volverá a ser retransmitido.

- El datagrama se somete ahora a transmisión, proceso del que se encarga la rutina RUTEA y dependiendo de la dirección de destino se toman las acciones siguientes:

  * Si el destinatario está en la misma estación, la rutina LLGFRM verifica si éste tiene un buffer para recibir el mensaje, y en caso afirmativo copia ahí el texto (rutina COPYBF) y da por terminado el envío, cancela el time-out, e informa al usuario del resultado del envío. En caso negativo se devuelve el buffer con el datagrama a la capa de transporte (ESTADO bit 13 = cero), en espera del vencimiento del time-out para nuevamente intentar su entrega.

* Si el destinatario está en otra estación, se elige la ruta adecuada y se lanza el datagrama por la línea a través de un QIO, especificando como bloque de I/O estatus las dos últimas palabras del paquete de solicitud. Al término de la operación se devuelve el buffer a la capa de transporte en espera del ACK, o del vencimiento del time-out para su retransmisión.

7.1.4 Recepción de mensajes por las líneas

La estación de transporte mantiene durante su operación lecturas activas por cada línea de recepción (NLEC). Los mensajes que arriban en cualquier momento son recibidos en los buffers del propio ACP, cuya estructura se ilustra en la figura 7-5.

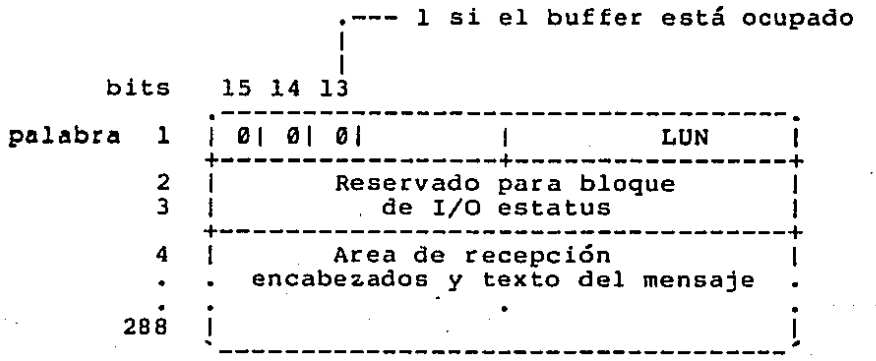


Figura 7-5: Buffer del ACP para recepción

En el byte inferior de la primera palabra, se almacena el número lógico (LUN) de la línea de recepción a la cual ha sido asignado, útil al final de un envío para actualizar el contador de lecturas activas por dicha línea. En el byte superior, los

bits 14 y 15 son igual a cero y constituyen el identificador para los buffers de recepción del ACP, mientras que el bit 13 contiene un cero o uno según que el buffer esté libre o ocupado respectivamente.

El bloque de las dos palabras siguientes está reservado para recibir el I/O estatus de la operación de lectura en la cual el buffer ha sido involucrado. El texto del mensaje se deposita a partir de la cuarta palabra, pudiendo ser un datagrama o un acuse de recibo.

El mecanismo de recepción emplea una rutina asíncrona (RCVFIN) al término de la llegada de un mensaje, la cual recibe la dirección del bloque del I/O estatus a fin de verificar el resultado de la operación, y al mismo tiempo localizar el texto recibido. Es también su responsabilidad el volver a activar una nueva lectura para esta línea sirviéndose de otro buffer libre del ACP.

El mensaje recibido es examinado por la rutina RUTEA, y si la dirección de destino corresponde a otra estación, se trata de un mensaje en tránsito, en cuyo caso se elige la ruta adecuada y se vuelve a lanzar el mensaje a la línea, al término de la cual se procede a liberar el buffer. En cambio si el mensaje fué dirigido a esta estación, las acciones que se toman dependen de:

- Si el mensaje es un datagrama se copia éste a un buffer del usuario de destino, y se envía el reconocimiento de su entrega. Pero si no hay quien lo reciba se ignora

bits 14 y 15 son igual a cero y constituyen el identificador para los buffers de recepción del ACP, mientras que el bit 13 contiene un cero o uno según que el buffer esté libre o ocupado respectivamente.

El bloque de las dos palabras siguientes está reservado para recibir el I/O estatus de la operación de lectura en la cual el buffer ha sido involucrado. El texto del mensaje se deposita a partir de la cuarta palabra, pudiendo ser un datagrama o un acuse de recibo.

El mecanismo de recepción emplea una rutina asíncrona (RCVFIN) al término de la llegada de un mensaje, la cual recibe la dirección del bloque del I/O estatus a fin de verificar el resultado de la operación, y al mismo tiempo localizar el texto recibido. Es también su responsabilidad el volver a activar una nueva lectura para esta línea sirviéndose de otro buffer libre del ACP.

El mensaje recibido es examinado por la rutina RUTEA, y si la dirección de destino corresponde a otra estación, se trata de un mensaje en tránsito, en cuyo caso se elige la ruta adecuada y se vuelve a lanzar el mensaje a la línea, al término de la cual se procede a liberar el buffer. En cambio si el mensaje fué dirigido a esta estación, las acciones que se toman dependen de:

- Si el mensaje es un datagrama se copia éste a un buffer del usuario de destino, y se envía el reconocimiento de su entrega. Pero si no hay quien lo reciba se ignora

éste, y en cualquier caso finalmente se libera el buffer.

- En cambio si el mensaje es un ACK, se extrae el identificador del datagrama al que corresponde, y se informa al usuario que su envío fué exitoso, liberando a continuación el buffer de recepción.

7.1.5 Envío de ACKs

Esta actividad es complementaria a la de recepción por las líneas, cuando el datagrama dirijido a esta estación pudo ser entregado a su destinatario, entonces la rutina ENVACK prepara un mensaje de acuse de recibo dirijido al remitente, y con este propósito el ACP dispone de NBFACK buffers de 15 palabras, cuya estructura se indica en la figura 7-6.

En la primera palabra de cada buffer, los bits 14 y 15 son uno y cero respectivamente e identifican los buffers para envío de ACKs, mientras que el bit 13 registra el estado del buffer, un "0" o "1" según que el buffer esté libre u ocupado. Las dos palabras siguientes son utilizadas como un bloque de I/O estatus en la directiva QIO que lanza a la línea el mensaje de ACK y cuyo texto es grabado en el buffer a partir de la quinta palabra. El contenido de este mensaje corresponde al encabezado de la carta recibida (11 palabras), invirtiendo las direcciones de orígen y destino, su código es ahora 2 (=ACK), y el tamaño del texto es cero.

La búsqueda de un buffer libre lo efectúa la rutina BUSBUF, y del envío se encarga RUTEA que elige la ruta adecuada y lanza el

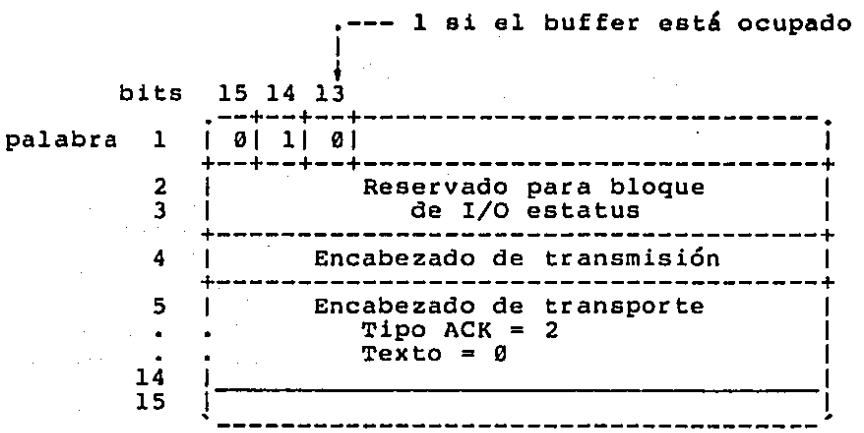


Figura 7-6: Buffer para envío de ACKs

mensaje a través de un QIO. La rutina asíncrona ENVFIN al término de la operación, libera el buffer para que pueda ser reutilizado.

7.1.6 Manejo de time-outs

Por cuanto en el sistema RSX-11M V3.1 no existe una forma selectiva de cancelar los time-outs emitidos a través de la directiva MRKT, se desarrolló en la estación de transporte un procedimiento ad-hoc para el manejo de éstos, consistente en guardar cada solicitud de time-out en un registro, en el que se incluye: la magnitud del tiempo en ticks, un identificador, y el tipo de time-out (rutina TMOPED). Estos registros se toman de una lista de registros libres y se ligan en órden creciente de la magnitud de tiempo en una lista de time-outs activos como se

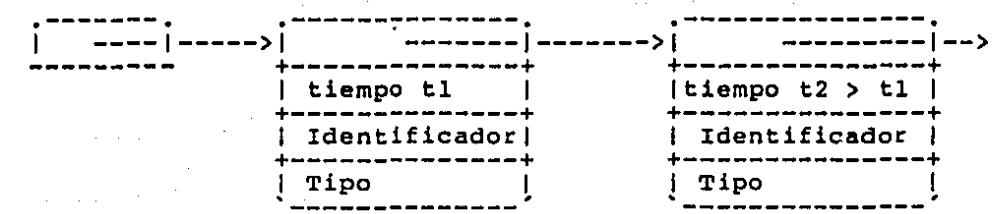


Figura 7-7: Lista de time-outs activos

indica en la figura 7-7.

Los tipos de time-outs en la Estación de Transporte corresponden a:

- TIPO 1 : Lo solicita la rutina SENDGM al enviar un datagrama, y corresponde al intervalo de espera al término del cual sinó arribó el reconocimiento de su entrega, volverá a ser retransmitido. Su magnitud es de TRES SEGUNDOS y el identificador del time-out lo constituyen el identificador del datagrama en el byte inferior, y el puerto de envío en el byte superior.

- TIPO 2: Lo solicita la rutina LIBPTO cuando el pedido de liberar un puerto no se satisface, ya que en ese instante la solicitud de envío de una carta pendiente de ese puerto no pudo cancelarse, por estar la carta en transmisión. Este time-out es de UN SEGUNDO y su identificador es la dirección del paquete de I/O portador de la solicitud de liberar puerto. Al vencer éste, se intenta de nuevo liberar el puerto.

- TIPO 3: Lo solicita la rutina DMOUNT cuando al tratar de desmontar el ACP, no se pudo liberar un puerto por la causa mencionada en el párrafo anterior. El time-out es de UN SEGUNDO, antes de volver a intentar la acción y su identificador es la dirección del paquete de I/O con la solicitud de desmontar el ACP.

- TIPO 4: Lo solicita la rutina LIMPIA y su magnitud es de UN MINUTO, intervalo con el cual se revisan las tareas, para eliminar aquellas que al concluir no

liberaron sus puertos en la estación de transporte. El identificador del time-out es en este caso igual a cero.

El estado de los time-outs es examinado con intervalos regulares de diez ticks por una rutina asíncrona TMORLJ, la cual incrementa un contador de ticks. Un time-out se da por terminado cuando su magnitud es menor o igual que este contador, en cuyo caso se lo elimina de la lista y la rutina TMOFIN selecciona la acción correspondiente. Además con esta organización es posible cancelar los time-outs en forma selectiva, operación indispensable en el protocolo de envío de datagramas al recibir el reconocimiento de su entrega. Este proceso es responsabilidad de la rutina TMOCAN, la cual localiza el time-out por su identificador y devuelve su registro a la lista de libres.

7.1.7 Revisión de las tareas que usan la Estación

Para evitar el que una tarea mantenga amarrados indefinidamente los puertos que activó en la estación de transporte, ya sea porque olvidó liberarlos o porque terminó su ejecución anormalmente, se aprovechó el comportamiento del Ejecutivo, el cual no retira una tarea de su lista de procesos activos, mientras estén pendientes solicitudes de I/O registradas por un contador (T.IOC) en el TCB (Task Control Block) de la tarea en cuestión.

Con este propósito se implementó en HERMES un mecanismo, consistente en incrementar o decrementar artificialmente este

contador de solicitudes directamente en el TCB de la tarea, cada vez que ésta activa o libera un puerto, consiguiendo de esta forma que si este contador es diferente de cero al momento de concluir su ejecución, el Ejecutivo la retendrá en memoria, marcando esta tarea para aborto en espera de que se completen o cancelen sus solicitudes pendientes.

La rutina LIMPIA del ACP periódicamente examina la lista de tareas activas del sistema y libera los puertos de aquellas que están marcadas para aborto, permitiendo en esta forma que el Ejecutivo complete el proceso y elimine de memoria la tarea. De ahí que el abortar una tarea que utiliza los servicios de la estación de transporte puede llevar considerable tiempo, específicamente hasta un minuto, que es el intervalo con el cual se efectúa la limpieza.

Durante la operación de revisión es indispensable el impedir que se modifiquen las listas del sistema, para lo cual la rutina LOCKL del ACP deshabilita el "checkpointing" y el "shuffling" de tareas. Al término de la revisión, la rutina UNLKL restaura las condiciones iniciales.

7.1.8 Finalizar la actividad de la estación de transporte

Esta actividad final es provocada por la operación de desmontar el ACP a través del comando DMO del MCR, el cual genera un paquete especial de I/O cuyo código de función es 6, y lo inserta directamente en la cola de recepción de NETACP. La rutina DMOUNT

toma a cargo esta solicitud y efectúa los siguientes pasos:

- Notifica al LOGGER que su operación debe concluir.
- Libera los puertos que aún permanecen activos.
- Cancela los time-outs que aún están pendientes.
- Cancela todas las lecturas y escrituras emitidas por el ACP a las líneas.
- Elimina cualquier paquete de I/O que llegó posteriormente a la cola de recepción del ACP.
- Devuelve al Ejecutivo el area correspondiente al VCB (Volumen Control Block).
- Finalmente pasa el control a la rutina $DREXT del Ejecutivo, y ésta retira el ACP de la memoria.

8.

LA CAPA DE TRANSPORTE: IMPLEMENTACION DE HERMES EN EL SISTEMA RT-11

8.1 Descripción general

Entre las computadoras que integran la red local del IIMAS está una PDP 11/10, con una capacidad de 32K palabras de 16 bits de memoria principal. Su sistema operativo RT-11 (V02B) permite el uso del sistema por un solo usuario a la vez, aun cuando existe la posibilidad de que dos programas residan simultáneamente en memoria, si el sistema es controlado por el monitor de "Foreground/Background". El programa de foreground recibe la máxima prioridad, mientras que el de background únicamente recibe el control si el otro se detiene en espera de algún evento externo, y es interrumpido en cualquier instante en que la tarea de foreground requiere del procesador.

Esta configuración del sistema determinó el diseño de la estación de transporte HERMES como un programa que corre en foreground, por cuanto cierto tipo de eventos externos tales como el arribo de mensajes por las líneas, requieren de atención inmediata, mientras que el programa de aplicación menos crítico es ejecutado en background.

La versión de HERMES en este sistema fué también escrita en lenguaje ensamblador MACRO-11, aunque es mas simple que su homólogo en el sistema RSX-11M pues se trata de un programa

normal de usuario. Su arquitectura es modular y compatible con el resto de estaciones y el servicio que presta es el de datagramas confiables, a través de la disciplina de transmisión CRISTAL que maneja la red de baja velocidad.

Para el proceso en la capa de aplicación, la interface aplicación/transporte es semejante a una colección de puertos, que al ser asociados por el proceso de usuario se activan para permitir el flujo bidireccional de datos, pudiendo el usuario transportar marcos (o cartas) a cualquier otro puerto de la red.

El proceso enviador proporciona a HERMES el texto de la carta y la especificación del destino, y éste se preocupa de la entrega del mensaje al puerto indicado, correspondiendo su actividad a:

- Construir el marco de la carta, añadiendo al texto de la misma el encabezado de transporte cuyo formato puede consultarse en la fig 7-3, antes de pasarlo a transmisión.
- Si dentro de un pèríodo especificado (3 segs) no se recibe el reconocimiento de su entrega (ACK), el marco es retransmitido.
- Si se recibe el ACK, el proceso enviador es informado de la entrega exitosa del marco.
- Si después de un número máximo de retransmisiones no se recibe el ACK, el marco es rechazado (abortado) y el proceso enviador es informado adecuadamente.

En cambio si arriba un mensaje a esta estación y:

- El puerto destino está inactivo, el marco recibido es ignorado y descartado.
- Si el puerto destino está activo, el proceso asociado,

es informado de su arribo y se devuelve un ACK al enviador, garantizando así la entrega efectiva del mensaje.

8.2 Algunas características de HERMES en RT-11

Aunque hemos dicho que la arquitectura de todas las estaciones de la red es invariante, cada una tendrá sus propios detalles locales de implementación dependiendo de la máquina y su sistema operativo, y destacamos aquí algunos aspectos en este sistema particular:

- No existe la necesidad de multiplexaje entre varios usuarios, pues el sistema soporta un solo programa de aplicación, lo cual simplifica el dieño de las estructuras de datos.

- Todas las direcciones que se manejan son reales, y no existe aquí la complicación del mecanismo para el manejo de memoria de los sistemas PDP 11 mayores.

- No existe limitación en cuanto al acceso de los programas a toda la memoria de la máquina, y aunque el mal uso puede provocar la falla del sistema, se aprovechó esta facilidad para que tanto HERMES como el programa del usuario, compartan una area común de buffers para el envío o recepción de mensajes por las líneas, localizada en foreground.

- La cooperación entre HERMES y el programa de aplicación, es posible a través del intercambio de mensajes en bloques de tamaño variable, por medio de las directivas del sistema SEND/RECEIVE-DATA. Este mecanismo es utilizado en las solicitudes dirijidas a la estación y en la entrega de servicios al usuario.

- Una facilidad muy importante en este sistema, es el que proporciona la directiva MARK-TIME para el manejo de time-outs, que a diferencia del sistema RSX-11M permite identificar cada time-out como también cancelar selectivamente cualquiera de ellos, sin necesidad de elaborar rutinas especiales o estructuras de datos adicionales.

- La intérface entre HERMES y el programa de aplicación,

fué implementada como un juego de rutinas en lenguaje MACRO-11, que el usuario deberá ligar con su programa, y su manejo es talvés mas complicado que el de las directivas QIO de la interface en el sistema RSX-11M.

- La interface entre la computadora y las líneas de comunicación es un DL-11 que opera en full-duplex a una velocidad de 9600 bits/seg, y al igual que en el sistema RSX-11M es controlada por dos manejadores (handlers) de línea RX para recepción y TX para transmisión, cuyos protocolos de control de línea, ya fueron discutidos en el Capítulo 5.

8.3 Algoritmos

Con el propósito de documentar el programa, se describirán aquí las actividades principales de HERMES, junto con sus estructuras de datos principales:

8.3.1 Inicialización

El arranque inicial de la estación de transporte implica las acciones siguientes:

- La rutina INICOM inicializa los apuntadores y estado de los buffers para envío y recepción de datagramas que se encuentran en el area común compartida con el usuario. Inicializa los registros de contexto y los liga a los buffers de envío y prepara también los buffers para envío de ACKs, como se muestra en la figuras 8-1 y 8-2.
- La rutina RCVD abre un canal de comunicación con background para recibir las solicitudes del usuario.
- La rutina INICNL abre un canal de escritura a la línea, e inicializa una cola de espera, donde se ligarán los buffers que deben transmitirse por la línea, sean estos de datagramas o ACKs.
- Se abren varios canales de lectura a las líneas, y la rutina READ activa en cada uno de ellos una lectura utilizando los buffers de recepción.

En este punto se suspende la ejecución del programa principal,

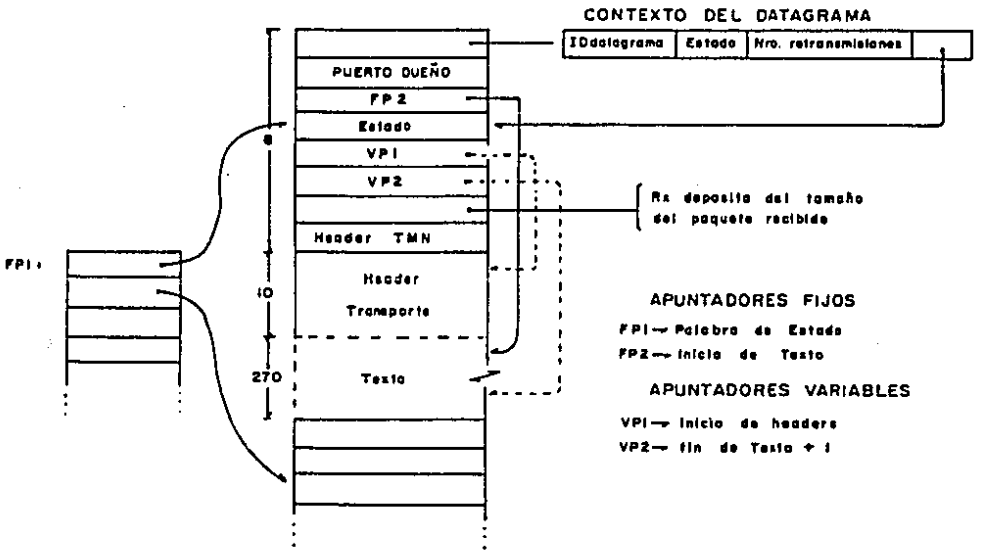


Figura 8-1: Buffers de Envío/Recepción de Datagramas

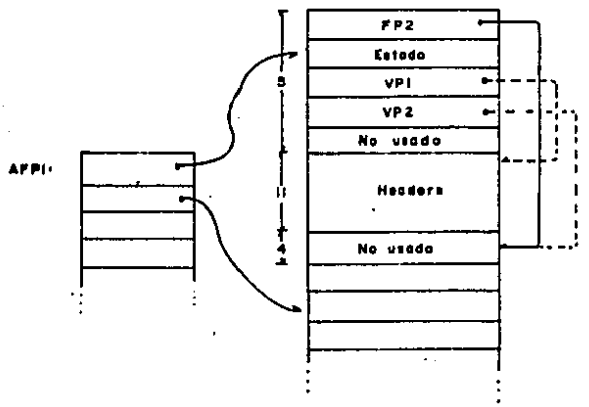


Figura 8-2: Buffers para envío de ACKs

actuando ahora solo las rutinas asíncronas que se activan al término de ciertos eventos que correponden a:

8.3.2 Llegada de una solicitud de usuario

Las solicitudes son interceptadas por la rutina RCVMSG y las acciones que toma dependen del tipo de solicitud pudiendo ser:

- Solicitud de sincronía

  La rutina SNCRON devuelve al programa de aplicación, la dirección de inicio del area común de buffers para envío de datagramas.

- Activar un puerto

  La rutina ACTV verifica que el identificador del puerto solicitado es válido y el puerto está libre, en cuyo caso lo activa y devuelve al usuario el resultado de la operación.

- Liberar un puerto

  La rutina DEACTV verifica si el puerto objeto de la solicitud es válido y activo en cuyo caso lo libera. En cualquier caso devuelve al usuario el resultado de la acción.

- Enviar un datagrama

  La rutina ENVCAR toma a cargo la solicitud y:

  * Verifica los parámetros de envío correspondientes a las direcciones de orígen y destino, y si hubo error cancela la operación.

  * Asigna un identificador al datagrama y prepara el correspondiente registro de contexto, guardando el identificador, el estado del buffer con la carta, e inicializando el contador de retransmisiones.

  * Prepara el encabezado de transporte y pasa la carta a transmisión.

  * Emite un time-out en espera del ACK para el datagrama enviado.

* Devuelve al usuario el resultado de la operación.

- Liberar un buffer y el canal que lo recibió

  Esta solicitud la origina el usuario para avisar a la estación que puede liberar el buffer con el datagrama recibido, pues ya lo copió en un buffer de su propiedad. La rutina LIBCNL libera el canal de recepción y el buffer correspondiente y revisa todos los canales para activar lecturas a la línea de aquellos que estén desocupados.

8.3.3 Recepción de un mensaje por la línea

Al término de una lectura por la línea la rutina asícrona RECLIN se encarga de verificar si la recepción fué correcta, y ajustar los apuntadores VP1 al inicio del encabezado de transmisión y VP2 al fin del texto. Revisa también el encabezado de transmisión, y si hay error libera el canal de recepción, o en caso contrario pasa el buffer con el mensaje a transporte.

En la capa de transporte la rutina RECTPT verifica ahora la validez de los parámetros del encabezado de transporte. Igualmente si hay error se libera el canal, o de lo contrario analiza el tipo de mensaje recibido pudiendo ser:

- Datagrama

  Al recibir un datagrama desde la línea, la rutina RCVCAR comunica a su vez al usuario su llegada. Si el usuario no responde, se procede a liberar el canal de recepcción, de lo contrario llama a ENVACK para preparar el envío del reconocimiento de la entrega.

- ACK

  Este tipo de mensaje lo procesa la rutina RCVACK, y de acuerdo a su identificador busca el datagrama en los registros de contexto. Si no lo encuentra ignora el

ACK liberando el canal de recepción. En caso contrario procede a cancelar el time-out para este datagrama y liberar su buffer, y por último informa al usuario de la llegada del acuse de recibo y libera el canal de recepción.

8.3.4 Fin de la transmisión de un mensaje por la línea

Al término de una escritura por la línea, la rutina asíncrona WRTRUT libera el canal de escritura y devuelve el buffer a transporte, y si hay otro mensaje listo para envío en la cola de espera, lo lanza por la línea marcando nuevamente el canal como ocupado.

8.3.5 Finaliza un time-out para envío de datagrama

La rutina CRONOS busca en los registros de contexto el identificador del time-out vencido, y si el número de retransmisiones del datagrama correspondiente no sobrepasa un cierto límite, lo vuelve a retransmitir, activando un nuevo time-out. En cambio si se agotaron las retransmisiones, avisa al usuario que su envío se abortó, y libera el buffer con el datagrama y su registro de contexto.

9.

CONCLUSIONES

Es indudable que un conocimiento sólido en el tema de redes de computadoras se consigue implementándolas, y este es precisamente el objetivo que hemos logrado, pues el tiempo empleado en este trabajo nos ha retribuido con un un modesto pero valioso bagaje de experiencia en los problemas y soluciones de este campo. El sistema implementado no solo es el resultado de seleccionar buenas herramientas, sinó el justificar su elección con criterios adecuados, y en los párrafos siguientes resumiremos los puntos claves de este diseño, como también las ventajas obtenidas y los problemas que aún subsisten:

9.1 Diseño de la arquitectura de la red

La idea de una arquitectura estructurada en capas no es nueva, pues ya ha venido aplicándose en otras redes y aún hay propuestas para su estandarización. El problema radica aquí en definir perfectamente los límites de las capas y los servicios que ofrecen a las capas vecinas, y aunque toda jerarquía es en cierto modo una burocracia, ésta puede ser eficiente, si se evitan duplicar las funciones en los diferentes niveles.

Una arquitectura de este tipo, facilita el manejo de la complejidad del diseño, y además en una red con computadores heterogéneos, el diseño e implementación de cualquier capa en las diferentes máquinas está restringido a comportarse externamente

de acuerdo a los protocolos del sistema, pero los aspectos técnicos y funcionales de su diseño pueden elegirse libremente en cada máquina, y no serán visibles a otras entidades excepto quizás por variaciones de eficiencia. Los únicos puntos en los cuales la organización física de cada capa es visible, son las interfaces, y si éstas se mantienen constantes, puede cambiarse cualquier capa de una red sin efecto ninguno sobre las demás.

9.2 Diseño de los protocolos

El diseño de los protocolos debe garantizar la comunicacíon entre las capas del mismo nivel, y sus funciones corresponden a la asignación de direcciones, la eliminación de errores, la recuperación de situaciones de error, el control del flujo y la sincronización. Tomando en cuenta las características actuales de la red del IIMAS, se ha dado prioridad a los protocolos de la capa de transmisión y la capa de transporte, pero en el futuro la interconección de esta red a otras, obligará incluir también un protocolo inter-red para el ruteo de los marcos entre las redes interconectadas.

Protocolos de Transmisión

En este nivel existen dos tipos de protocolos: el "Protocolo de Control de Linea", implementado en los manejadores de las líneas, y cuya función es identificar los límites de un marco de transporte, rodeando a éste de una secuencia especial de caracteres. Los errores en este nivel se detectan a través de un

BCC (Block Check Character), y las pruebas realizadas demuestran que este protocolo es confiable, y su simplicidad redunda en un código reducido para los manejadores, condición indispensable de los manejadores eficientes. Sin embargo subsiste la probabilidad de errores, pues pueden perderse caracteres y no ser detectados por el BCC, pero teniendo en cuenta que la red va a evolucionar a una de alta velocidad de tipo Ethernet, se mejorará la confiabilidad de la transmisión a través de un CRC (Cyclic Redundancy Check), calculado velozmente por el hardware.

El segundo tipo de protocolo en este nivel es el "Protocolo Inter-red", previsto para el ruteo de los marcos entre las redes interconectadas, aunque por ahora limitado a los anfitriones de la red local. El ruteo está implementado en base a una tabla estática que contiene los identificadores de las líneas de acuerdo a la estación de destino. Si se conserva esta red de tipo guarda-reexpide, y se incrementa su número de nodos, existiendo la posibilidad de rutas alternas, será indispensable mejorar los algoritmos de ruteo, sin embargo con la red de difusión se elimina el ruteo, pues los marcos se difunden por igual a todas las estaciones, captándolo únicamente la estación a la cual va dirigido.

Protocolo de Transporte

El "Protocolo de Transporte HERMES" está orientado al servicio de Datagramas Confiables, donde los paquetes se transmiten

aisladamente sin garantía de secuencia o eliminación de duplicados, relegando esta responsabilidad a la capa de aplicaciones. Sin embargo aunque este protocolo es adecuado para una gran cantidad de aplicaciones, pueden existir otras, donde resulta ineficiente o demasiado caro. Este problema se detectó con el desarrollo de otra tesis cuyo tema era el manejo de terminales virtuales, y donde el transporte de paquetes muy pequeños puede provocar un "overhead" de envío de datagramas y ACKs, incrementando además la probabilidad de paquetes duplicados o fuera de orden.

Estas consideraciones impulsaron la necesidad de implementar el nuevo protocolo de transporte NARAD. Esta nueva ampliación que no es parte de este trabajo está ya especificado y actualmente es implementado por el Dr. G. Sidhu, y permitirá ofrecer a los usuarios de la estación el servicio de Tuberías de Bytes, en base a un protocolo parecido al TCP (Transmission Control Protocol) de ARPANET, aunque simplificado teniendo en cuenta las características de nuestra red. Los detalles y especificaciones de este nuevo protocolo pueden consultarse en [7].

9.3 Diseño del software

El software fué diseñado en forma modular, reflejando la idea de una arquitectura dividida en capas, lo cual facilita la depuración y el mantenimiento del sistema. El avance más significativo, se logró en las máquinas PDP-11 con sistema operativo RSX-11M, en donde las Estaciones de Transporte gozan de

de privilegios para efectuar un multiplexaje efectivo de los usuarios que utilizan sus servicios, y monitorear a cada uno de ellos. En realidad el software de la red, puede considerarse como una extensión del ejecutivo, pues se trata de un proceso privilegiado de control ACP (Ancillary Control Process), desde el cual se puede accesar las funciones del ejecutivo, compartir el código de sus rutinas, su espacio de memoria dinámica, las estructuras de datos y hacer uso directo del mecanismo de manejo de memoria para accesar los buffers de los usuarios.

Una ventaja indiscutible de la asociación del software de transporte con el ejecutivo fué la simplificación de la interface con el usuario, pues las solicitudes se canalizan ahora a través de las directivas QIO del sistema, que ofrecen una gama muy amplia de facilidades, en contraste con la interface original que obligaba a ligar con cada programa de usuario una cantidad considerable de código y compartir con la estación de transporte una area común de "buffers" para el envío o recepción de mensajes.

Cabe destacar que el problema más grave en el desarrollo de este software, fué la falta de documentación al respecto, y la única alternativa fué leer los listados fuentes del sistema. Sin embargo esta experiencia destacó la bondad del sistema RSX-11M por la facilidad de incorporar nuevos manejadores y otros procesos especiales sin tener que modificar explícitamente su código, a diferencia de otros sistemas (v.g. TENEX), en los cuales actividades de este tipo obligarían a regenerar el

sistema, después de modificar su código, con el riesgo de corromperlo.

Este nuevo tipo de software demanda usar un lenguaje altamente confiable y flexible para manejar los recursos del sistema, y por tal motivo se eligió el lenguaje ensamblador MACRO-11 auxiliado por macros de alto nivel [5]. Este lenguaje es compatible con el código del sistema operativo, y si bien es cierto que no es transportable, queda justificado plenamente por su confiabilidad en el desempeño de actividades concurrentes, características de este tipo de programación, ya que no necesita de intermediarios para el manejo de las directivas del ejecutivo, y tiene el control de las banderas de eventos, y otros parámetros, que en lenguajes de alto nivel, se esconden al usuario, ya que fueron diseñados con otros propósitos.

Otra ventaja de programar en MACRO, fué la optimización del código, aspecto este muy importante, teniendo en cuenta las limitaciones de espacio, y el hecho de que el programa estará permanentemente en memoria, para atender en forma rápida y eficiente las solicitudes del usuario y el arribo de mensajes. La tarea en cuestión tiene un tamaño de 5K palabras de memoria, en contraste con la Estación original en Pascal que ocupaba 12K palabras. Los manejadores de linea RX y TX y del pseudo dispositivo NX consumen un total de 800 palabras.

9.4 Recomendaciones

Quedan aún pendientes el desarrollo del software, para el manejo de otras disciplinas de transmisión, el de la red de alta velocidad de tipo Ethernet, cuyo hardware está a punto de concluir, y el manejo del protocolo X.25 para una conexión eventual a una red pública, que dada la modularidad del programa, afectará únicamente a la capa de transmisión, y precisará el desarrollo de un manejador adicional.

Se recomienda también el desarrollo de un juego de rutinas que facilite a los usuarios en lenguajes de alto nivel, el acceso a los servicios de la red, liberándolo de los detalles y sutilezas de las directivas QIO, para lo cual será indispensable el conocer las peculiaridades de cada lenguaje como son las banderas que cada uno de ellos emplea internamente, y la asignación de identificadores o números a las unidades lógicas, a fin de no interferir con los que utilizen las rutinas de la interface Usuario/Estación de Transporte.

Por último, cabe destacar que el establecimiento de la red de Transporte estará ligado intimamente al desarrollo posterior de aplicaciones muy útiles y de interés creciente, y de hecho constituye los cimientos sobre los cuales se implementarán trabajos tales como:

- Terminales virtuales: que permita desde una terminal accesar a una variedad de periféricos y computadoras remotas.

- Procesamiento distribuido: donde el usuario somete su trabajo a la red, sin especificar un anfitrión particular donde éste se ejecute.

- Bases de datos distribuidas: para manejar bancos de datos, los cuales aún cuando están dispersos en distintos anfitriones, pueden ser usados como si estuvieran en un solo lugar.

- Correo electrónico y teleconferencia: para intercambiar información en forma económica y eficiente.

APENDICE I

HERMES: SISTEMA DE TRANSPORTE PROCESO-A-PROCESO DE DATAGRAMAS CONFIABLES PARA LA RED LOCAL DEL IIMAS

Manual del usuario en el sistema RSX-11M

RED LOCAL Nro. 1 - ESTACION Nro. 1

Gursharan S. Sidhu
José Villagómez C.

IIMAS, UNAM, México

Junio de 1982

I.1. RESUMEN

El propósito de este manual es el de proporcionar toda la información necesaria sobre la interface a través de la cual los usuarios interesados, pueden accesar los servicios del transporte de datagramas confiables proporcionados por HERMES en las máquinas PDP-11 bajo el sistema operativo RSX-11M, versión 3.1.

I.2. Introducción

La filosofía con la cual se desarrolló HERMES en el sistema RSX-11M, permite visualizar a éste como un dispositivo más de entrada y salida, a través del cual se reciben o envían mensajes, y como tal todas las solicitudes de servicios son dirigidas directamente al manejador de este dispositivo por medio de las directivas QIO del sistema.

Con esta característica, el usuario dispone de una gran flexibilidad para solicitar sus servicios, ya que el sistema dispone de un conjunto de macros que facilitan el uso de las directivas QIO. Cada programa o tarea es capaz de accesar este dispositivo de uso público, y las solicitudes se disponen en cola y se procesan subsecuentemente de acuerdo a la prioridad de la tarea que las emitió.

El uso eficiente de este nuevo conjunto de facilidades, demandará que el usuario tenga algún conocimiento del lenguaje Macro-Ensamblador para PDP-11, experiencia con algunas directivas del sistema y le sean familiares conceptos tales como manejo de eventos, estatus de una directiva, estatus de un pedido de I/O y el uso de rutinas asíncronas.

Para los programadores en lenguajes de alto nivel, se podrán elaborar facilmente un conjunto de rutinas, dentro de la librería de cada lenguaje, de modo que el usuario solo deba conocer el nombre y la función de la rutina en cuestión y el de los

parámetros a insertar. Esta implementación carece de dificultad si se conoce la forma en que cada lenguaje maneja los parámetros de subrutinas, a fin de poder interactuar correctamente con las rutinas escritas en Macro.

Sin embargo en el presente documento nos concretaremos a demostrar las facilidades de transporte de HERMES, a través del uso directo de las directivas del sistema, para lo cual a manera de introducción se revisarán someramente los mecanismos de entrada y salida en el sistema RSX-11M, ya que para éste propósito el usuario deberá referirse a los manuales [2] y [3].

I.3. Entrada y salida en el sistema RSX-11M

Las operaciones de entrada y salida en el sistema RSX-11M son extremadamente flexibles y los programas dirigen sus solicitudes de I/O a unidades lógicas, las cuales han sido asociadas previamente con la unidad de un dispositivo físico particular, siendo cada programa capaz de establecer su propia correspondencia entre las unidades físicas y los números de unidades lógicas LUNs (Para el caso de la Estación de Transporte la tarea del usuario debe asignar un número lógico a la unidad cero del dispositivo denominado NX). Una vez establecido este canal lógico de comunicación con el dispositivo, el usuario está en libertad de solicitar los servicios de este a través de las directivas del sistema QIO o QIOW, según su conveniencia.

Debe notarse que la tarea no accesa directamente el dispositivo

físico, sinó que hace uso de los servicios proporcionados por el Ejecutivo, puesto que este es capaz de hacer un multiplexeo efectivo del uso de un dispositivo por muchos usuarios. El Ejecutivo rutea el pedido al manejador del dispositivo y lo pone en cola de acuerdo a la prioridad de la tarea que lo emitió, y las operaciones de I/O se realizan concurrentemente con otras actividades en el sistema.

Antes de poner en cola una solicitud, el Ejecutivo la somete a una serie de chequeos y si hay errores, el pedido se rechaza y advierte al usuario prendiendo el bit de carry, de modo que resulta una buena práctica de programación verificar si hubo rechazo de la directiva con la instrucción BCS (Branch if Carry Set) en la instrucción siguiente al QIO.

Una vez que el pedido es pasado a la cola del manejador, el sistema no espera a que este termine, sinó que continúa con la ejecución de la tarea del usuario y si, en un punto determinado la tarea del usuario no puede continuar sinó ha concluido la operación del QIO, esta deberá hacer uso de una bandera (event-flag) en la directiva QIO y emitir una directiva WAITFOR, especificando la misma bandera, en el punto donde debe ocurrir la sincronización. Una manera más económica de alcanzar esta sincronización, es por medio de la directiva QIOW, que automáticamente espera a que la directiva QIO concluya su operación, antes de retornar el control a la tarea.

Cada directiva QIO o QIOW especifica una serie de parámetros, que sirven para identificar y poner en cola el pedido, siendo estos: el código de la función a realizar, el número lógico asociado con la unidad física, una bandera para sincronizar el proceso de I/O, la dirección de una area para recibir errores o resultados de la operación, la dirección de una rutina asíncrona que deberá ejecutarse al término de la operación de la directiva, y ciertos parámetros adicionales que dependen del tipo de dispositivo o de la función, como son direcciones y tamaños de buffers.

Existen un conjunto de macros que facilitan el uso de las directivas QIO. Estas macros residen en la librería LB:[1,1]RSXMAC.SML, y basta incluir su nombre en la instrucción .MCALL para que estas puedan ser utilizadas por el programa del usuario. Algunos de los parámetros de las directivas QIO son opcionales, pero su espacio debe reservarse, ya que durante la expansión de una macro QIO, se asigna un valor de cero para todos los parámetros nulos (omitidos).

I.4. Formato de la macro QIO

Los argumentos que deben especificarse en la llamada a esta macro pueden ser diferentes para cada dispositivo accesado y para cada función solicitada, sin embargo el formato de la llamada es común a todos los dispositivos y es el siguiente:

```
QIO$C  fcn,lun,[efn],[pri],[isb],[ast][,<p1,p2,...p6>]
```

donde los paréntesis rectangulares [] especifican un parámetro opcional, o un parámetro dependiente de la función. Si se requieren los parámetros <p1,p2,...p6> dependientes de la función, estos deberán encerrarse con paréntesis angulares. En los párrafos siguientes se discute brevemente el uso de estos parámetros.

- El parámetro "fcn" es un nombre simbólico que representa el identificador de la función solicitada. Este nombre es de la forma IO.xxx donde xxx idenfica la operación particular solicitada. Así por ejemplo la solicitud a HERMES para activar un puerto público se identifica con el código IO.RNA. Cabe mencionar que en el caso de la Estación de Transporte la interpretación de estos códigos es exclusiva del manejador del dispositivo NX, sin ninguna relación con la interpretación convencional descrita en la tabla 4-1 del manual "Guide to Write an I/O Driver" [3].

- El parámetro "lun" representa el número de la unidad lógica, asociado a la unidad del dispositivo físico que va a ser accesado por la solicitud de I/O. Esta asociación es específica de la tarea que hace el pedido y debe ser efectuada previamente, mediante el uso de la directiva ALUN o mediante un comando del MCR antes de la ejecución de la tarea.

- El parámetro "efn" es un número que representa la bandera (event-flag) que se asocia con el pedido y puede incluirse opcionalmente en una solicitud de QIO o QIOW. Esta bandera se apaga cuando la solicitud es puesta en cola y se prende al concluir la operación. Si la tarea emitió una directiva QIOW la ejecución se suspende automáticamente hasta el fin de la ejecución del pedido, en caso contrario la ejecución de la tarea continúa en paralelo junto con la ejecución del pedido de I/O.

- El parámetro opcional "pri" no se emplea, ya que en el sistema RSX-11M el pedido asume automáticamente la prioridad de la tarea que lo emitió, por lo tanto bastará con reservar el espacio de este parámetro.

- El parámetro opcional "isb" identifica la dirección de un bloque de dos palabras en el cual se deposita el

código que representa el estatus final de la operación. Este código de éxito o error se almacena en el byte inferior de la primera palabra del bloque. En el byte superior se regresa cierta información dependiente del dispositivo. En el caso de ciertas funciones como la de lectura desde una terminal, se deposita en la segunda palabra el número de bytes transferidos si la operación es exitosa, o en el caso de HERMES al concluir la operación de activar un puerto, se deposita en la segunda palabra el identificador del puerto activado. Este bloque de estatus puede omitirse, si el usuario no tiene intenciones de revisar el resultado de la operación.

- El parámetro opcional "ast" especifica la dirección de una rutina asíncrona que deberá ejecutarse al término de un pedido, interrumpiendo la secuencia normal de ejecución de la tarea. Si el usuario no desea este procesamineto asíncrono, este parámetro puede omitirse o reemplazarse por un valor de cero.

- Los parámetros adicionales <p1,p2,...p6> dependen de la función solicitada, y el número de estos puede variar entre cero y seis. Típicamente estos parámetros contienen la dirección de un buffer, su longitud en bytes, etc. El significado preciso de estos parámetros será discutido para el caso de HERMES, al detallar cada función en particular.

I.5. Proceso de la directiva QIO

Durante el ensamblado de un programa de usuario, la expansión de la macro que corresponde a una directiva del sistema, genera una area de memoria contigua y de longitud fija (DPB) en donde se almacenan los parámetros especificados en la llamada a la directiva. En el caso de la directiva QIO este bloque tiene una longitud de 12 palabras, almacenando en el byte inferior de la primera palabra, el identificador de la directiva (1 para QIO y 3 para QIOW), y en el segundo byte el tamaño del bloque en palabras.

Al momento de la ejecución, el Ejecutivo usa los argumentos almacenados en cada DPB, y genera para cada pedido un paquete de I/O (I/O packet) en el area de almacenamiento dinámico del sistema. Este paquete es entonces puesto en la cola de la unidad del dispositivo al cual va dirigido el pedido, de acuerdo a la prioridad de la tarea que lo emitió. El manejador del dispositivo examina esta cola y extrae el pedido con mas alta prioridad y lo ejecuta, repitiendo este proceso hasta que no haya mas pedidos en la cola.

Al terminar la ejecución de un pedido de I/O el Ejecutivo declara un evento significativo, y puede prender una bandera, provocar la ejecución de una rutina asíncrona y/o notificar el estatus de la operación, dependiendo de los argumentos especificados en la llamada original de la directiva.

I.6. Formas de la directiva QIO

Existen tres formas distintas de llamada a una directiva del sistema, y resumimos estas aplicándolas a la directiva QIO$; sin embargo las características de cada forma se aplican también a otras directivas del sistema como QIOW$, ALUN$, etc.

1. QIO$ genera al momento de ensamblado el bloque de parámetros (DPB), mas nó las instrucciones necesarias para ejecutar el pedido. Esta forma de pedido se ejecuta usando la macro DIR$. Esto resulta útil si el mismo DPB va a ser usado en diferentes puntos de la tarea, y tiene la ventaja de que puede ser modificado por la tarea al momento de ejecución.

2. QIO$C genera al momento de ensamblado el bloque de parámetros (DPB) y además el código para ejecutar el

Al momento de la ejecución, el Ejecutivo usa los argumentos almacenados en cada DPB, y genera para cada pedido un paquete de I/O (I/O packet) en el area de almacenamiento dinámico del sistema. Este paquete es entonces puesto en la cola de la unidad del dispositivo al cual va dirigido el pedido, de acuerdo a la prioridad de la tarea que lo emitió. El manejador del dispositivo examina esta cola y extrae el pedido con mas alta prioridad y lo ejecuta, repitiendo este proceso hasta que no haya mas pedidos en la cola.

Al terminar la ejecución de un pedido de I/O el Ejecutivo declara un evento significativo, y puede prender una bandera, provocar la ejecución de una rutina asíncrona y/o notificar el estatus de la operación, dependiendo de los argumentos especificados en la llamada original de la directiva.

I.6. Formas de la directiva QIO

Existen tres formas distintas de llamada a una directiva del sistema, y resumimos estas aplicándolas a la directiva QIO$; sin embargo las características de cada forma se aplican también a otras directivas del sistema como QIOW$, ALUN$, etc.

1. QIO$ genera al momento de ensamblado el bloque de parámetros (DPB), mas nó las instrucciones necesarias para ejecutar el pedido. Esta forma de pedido se ejecuta usando la macro DIR$. Esto resulta útil si el mismo DPB va a ser usado en diferentes puntos de la tarea, y tiene la ventaja de que puede ser modificado por la tarea al momento de ejecución.

2. QIO$C genera al momento de ensamblado el bloque de parámetros (DPB) y además el código para ejecutar el

pedido. Este DPB es generado en una sección separada del programa llamada $DPB$$. Esta forma es útil cuando el pedido de I/O se ejecuta en un solo lugar del programa.

3. QIO$S genera el bloque de parámetros de la directiva (DPB) en el stack, y también genera el código para ejecutar el pedido. Esta forma es útil para programas reentrantes, puesto que el bloque de parámetros se genera dinámicamente al momento de ejecución.

Los parámetros para las formas QIO$ y QIO$C deben ser expresiones válidas para las directivas de generación de datos en ensamblador tales como .WORD y .BYTE . En cambio los parámetros para la forma QIO$S deben ser operandos válidos para una instrucción tipo MOV y MOVB. Los siguientes ejemplos hacen referencia a los mismos parámetros en las tres formas diferentes de llamada:

```
QIO$  IO.ACW,4,10.,,IOSB,AST1,<BUFCAR,50.>
QIO$C IO.ACW,4,10.,,IOSB,AST1,<BUFCAR,50.>
QIO$S #IO.ACW,#4,#10.,,#IOSB,#AST1,<#BUFCAR,#50.>
```

I.7. Estatus de la ejecución de un QIO

Existen dos clases de condiciones de estado cuando el usuario lanza un pedido de I/O:

1. Condiciones de estado de la directiva, las cuales identifican la aceptación o el rechazo de la directiva QIO. El código que identifica este tipo de condición se regresa en una localidad de nombre simbólico $DSW (Directive Status Word), y el usuario la puede examinar para verificar el estado de la directiva. Estos códigos pueden indicar cualquiera de las condiciones siguientes:

- Aceptación de la directiva.
- Error en la especificación del buffer.
- Bandera (event-flag) errada.
- Número lógico de la unidad inválido o no asignado.
- El identificador o tamaño del DPB es inválido.
- Isuficiente memoria dinámica para generar el I/O packet.

2. Condiciones del estado de éxito o fracaso de la operación de I/O. Los errores detectados por el manejador, incluyen condiciones tales como:

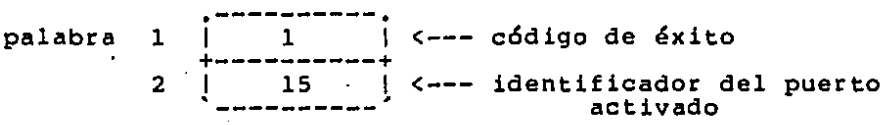
   - Dispositivo fuera de servicio.
   - La tarea no tiene privilegios para ejecutar el pedido.
   - La función solicitada es inválida para este dispositivo.
   - El buffer está fuera del rango de direcciones de la tarea, etc.

Estos códigos de operación pueden ser examinados en el bloque de I/O estatus especificado en la directiva QIO al término de la ejecución del pedido. Este bloque consta de dos palabras con el formato siguiente:

- El byte inferior de la primera palabra recibe el código de estatus, positivo si hubo éxito o negativo si hubo error.
- El byte superior de la primera palabra recibe información dependiente del dispositivo. (Este byte no es utilizado por HERMES, y el código de estatus ocupa toda la primera palabra).
- La segunda palabra contiene información dependiente de

la función, como el número de bytes transferidos o procesados si la operación es de lectura o escritura. Si la solicitud fue dirigida a HERMES, esta palabra contiene información diversa, tal como el identificador del puerto, si la operación fué la de activar un puerto, etc.

La siguiente ilustración muestra un bloque de I/O estatus al término de la ejecución de una solicitud de activar puerto:



I.8. Servicios de la Estación de Transporte HERMES

Como ya hemos señalado anteriormente, la Estación de Transporte HERMES es percibida por el usuario como un dispositivo mas de entrada y salida, denominado NX, y todos los servicios que este presta deben ser solicitados por medio de las directivas QIO del sistema, dirigidas a la unidad cero (Ø) del dispositivo NX.

Los servicios básicos ofrecidos se reducen a seis como se indica en el cuadro 9-1.

Advertimos al usuario que los códigos de cada función, no tienen ninguna relación con el significado descrito en los manuales del sistema, válido unicamente para los dispositivos estandar de Digital.

El usuario está en libertad de hacer uso de cualquiera de las tres formas distintas de llamada a las directivas QIO o QIOW

	Código	Función
1	IO.RNA	Activar un puerto público
2	IO.FNA	Activar un puerto fijo
3	IO.ENA	Liberar un puerto público o fijo
4	IO.ACW	Envio de datagramas
5	IO.ACR	Recibir un datagrama
6	IO.ACE	Averiguar los puertos de otra tarea

Figura 9-1: Códigos de servicios HERMES

según convenga a su programación, sin embargo la descripción individual de los primitivos básicos, la ilustraremos sirviéndonos de la forma QIO$C, de acuerdo a las convenciones estipuladas en la sección 3.

Los códigos del estado de la directiva QIO o QIOW son devueltos en la localidad cuyo nombre es $DSW y pueden consultarse en la Figura 9-2. El usuario los podrá examinar sirviéndose de los nombres simbólicos como se indica en la siguiente instrucción:

```
CMPB  $DSW, #IE.UPN   ;memoria dinámica insuficiente ?
```

Los códigos del resultado de la operación ocupan la primera palabra del bloque de I/O estatus y se detallan para cada operación solicitada, y el usuario puede examinarlo como se indica en la instrucción:

```
CMP  IOSB, #IS.SUC  ;hubo éxito en la operación ?
```

Si el código de la función solicitada es inválido, la primera palabra del bloque de I/O estatus contendrá el valor -2.

Por último, si en la llamada de la directiva, el usuario especifica la dirección de una rutina asíncrona (AST), al pasar el control a la ejecución de ésta, su stack queda en el siguiente estado:

SP+10	Máscara para la bandera de evento.
SP+06	PS (Program Status) de la tarea antes de entrar a la rutina asíncrona.
SP+04	PC (Program Counter) de la tarea antes de entrar a la rutina AST.
SP+02	DSW (Directive Status Word) de la tarea antes de la entrada a AST.
SP+00	Dirección del bloque de I/O estatus, o cero, si no se especificó al lanzar la directiva QIO.

Es responsablidad del usuario el remover del stack la dirección del bloque de I/O estatus antes de la ejecución de la directiva ASTX para salir de la rutina asíncrona.

I.8.1. Activar un puerto público

Todo proceso de aplicación que utilice la red, deberá registrarse en la Estación de Transporte correspondiente, solicitando un puerto, el cual es simplemente un identificador a nivel de transporte, para cada entidad de la red en la capa de aplicaciones.

Las Estaciones de Transporte en el sistema RSX-11M disponen de un conjunto de veinte puertos públicos, cuyo rango de numeración varía de 11 a 30, y cuando una tarea de usuario solicita un

CÓDIGOS DEL ESTADO DE LA DIRECTIVA QIO O QIOW REGRESADOS POR EL EJECUTIVO EN LA LOCALIDAD $DSW

+01. IS.SUC	La directiva fue aceptada
-01. IE.UPN	La memoria dinámica es insuficiente
-05. IE.ULN	El número de la unidad lógica no fue asignado
-06. IE.HWR	El manejador del dispositivo no ha sido cargado.
-96. IE.ILU	El número de la unidad lógica es inválido
-97. IE.IEF	El identificador de la bandera de evento es inválido (>64 o <0).
-98. IE.ADP	Parte del DPB o del bloque de I/O estatus esta fuera del espacio de direcciones de la tarea.
-99. IE.SDP	El identificador de la directiva (DIC) o el tamaño del DPB es inválido.

Figura 9-2:

puerto público, HERMES busca el primer puerto libre y lo asocia con el nombre de la tarea, y devuelve al usuario el identificador asignado. Una tarea de aplicación puede estar asociada a varios puertos, sin embargo cada puerto es propiedad de una sola tarea.

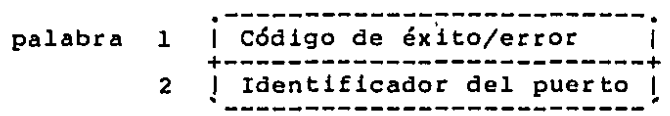
El formato de la solicitud es el siguiente:

```
QIO$C  IO.RNA,lun,[efn],[pri],[isb],[ast]
```

IO.RNA = Código de la función solicitada
lun = Número de la unidad lógica
efn = Número de la bandera de evento
pri = prioridad, aunque ignorada debe estar presente
isb = Dirección del bloque de I/O estatus
ast = Dirección de una rutina asíncrona

El bloque de I/O estatus aunque es opcional, en este caso es indispensable, ya que HERMES se sirve de el, para devolver al

usuario el resultado de la operación con el formato indicado:



Los posibles códigos corresponden a:

1 = Exito de la operación, en cuyo caso la segunda palabra contiene el identificador del puerto asignado.

-6 = Error, los puertos públicos se agotaron.

I.8.2. Activar un puerto fijo

Esta función es similar a la anterior, con la diferencia, de que el proceso de aplicación en este caso, especifica el número del puerto con el que le interesa asociarse. El uso de puertos fijos queda restringido a ciertas tareas que prestan servicios especializados y frecuentes, tales como el LOGGER, el servicio de terminales virtuales NETVTP, el servicio de transferencia de archivos NETFTP, o el listador de archivos NETLIS. Estos puertos operan exactamente igual que los puertos públicos, pero su ventaja radica en que el usuario que desea hacer uso de las tareas especiales, conoce de antemano el identificador del puerto destino al que dirigirá sus mensajes.

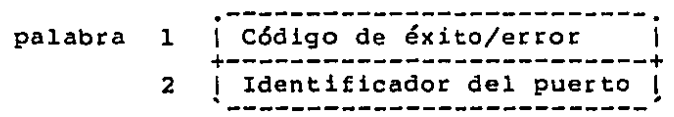
HERMES dispone de un conjunto de 10 puertos fijos, cuyo rango de numeración varía de 1 a 10. Estos puertos no podrán ser activados si la tarea no es instalada con el nombre correcto al cual se ha asociado un determinado puerto fijo, como puede consultarse en la figura 9-3.

El formato de la solicitud es el siguiente:

QIO$C IO.FNA,lun,[efn],[pri],[isb],[ast],<puerto>

IO.FNA = Código de la función solicitada
puerto = Identificador del puerto fijo solicitado
..... Los parámetros restantes son estándares.

El resultado de la operación puede verificarse en el bloque de I/O estatus, cuyo formato es el siguiente:



Los posibles códigos de I/O estatus son los siguientes:

1 = Exito, el puerto se activó.

-6 = Error, el identificador no corresponde al rango de puertos fijos.
-7 = Error, el puerto ya se encontraba activo.

-15 = Error, la tarea solicitante, no tiene derecho a este puerto.

En cualquier caso la segunda palabra contiene el identificador del puerto solicitado.

I.8.3. Liberar un puerto

Cuando una tarea de aplicación, ya no tiene necesidad de los puertos que activó en la Estación de Transporte, ya sea por cambio de actividad, o porque terminó su ejecución, es recomendable el que libere los puertos públicos o fijos que activó, para que estos queden ahora disponibles para otras tareas que los necesiten, y para que la terminación de la tarea en cuestión sea normal.

PROCESOS DE APLICACIÓN ESPECIALES Y PUERTOS FIJOS ASOCIADOS

	Nombre	Puerto fijo	Función
	LOGGER	1	. Activar otras tareas . Informar sobre los puertos de otras tareas
	NETLIS	2	. Listador de archivos
*	NETVTP	3	. Servicio de terminales virtuales
*	NETFTP	4	. Servicio de transferencia de archivos

* Estos programas están todavía en desarrollo.

Figura 9-3:

Conviene destacar el hecho, de que si una tarea concluye sin haber liberado sus puertos, esta permanecerá en la memoria, esperando que la estación de transporte sea quien libere sus puertos y la force a abortarse, proceso que puede tardar hasta un minuto, intervalo con el cual HERMES revisa la lista de tareas y cancela aquellas que terminaron en forma indebida con el mensaje:

```
TASK " name " TERMINATED
TASK EXIT WITH OUTSTANDING IO
```

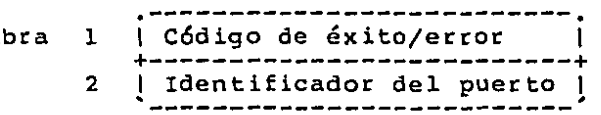
No puede una tarea de usuario liberar los puertos de otra tarea. La liberación de un puerto implica además, que los pedidos pendientes de envío o recepción de mensajes a través de este puerto, sean cancelados.

El formato de la solicitud es el siguiente:

QIO$C IO.ENA,lun,[efn],[pri],[isb],[ast],<puerto>

IO.ENA = Código de la función solicitada
puerto = Identificador del puerto a liberarse
..... Los parámetros no mencionados son estándares

El resultado de la operación puede examinarse en el bloque de I/O estatus, cuyo contenido es el indicado:



Los posibles códigos de I/O estatus son los siguientes:

1 = Exito, el puerto se liberó

-6 = Error, el identificador del puerto es inválido, está fuera de rango.
-7 = Error, el puerto ya ha sido liberado previamente.

-15 = Error, el puerto pertenece a otra tarea.

En cualquier caso, la segunda palabra del bloque contiene el identificador del puerto objeto del pedido.

I.8.4. Envío de datagramas

Esta función permite a los procesos de usuario la posibilidad de enviar datagramas a procesos ubicados en cualquier procesador de la red, y es responsabilidad del usuario el preparar un buffer de su propiedad con el texto del mensaje y cierta información de control como son el tamaño del texto y las direcciones de orígen y destino, pudiéndose consultar estas últimas en la Figura 9-5.

Conviene destacar que HERMES efectúa la transferencia del mensaje, directamente desde el buffer del usuario a las lineas de

transmisión, cuando el destino está en otro procesador, o a otro buffer de usuario, si el destino está en la misma estación, sin almacenamientos intermedios o duplicación innecesaria de la información. Este hecho demanda que el usuario, reserve en el buffer del mensaje, cierto espacio adicional que será utilizado por la Estación para datos de control y que junto con el texto, constituye el "Marco de Transporte". Una vez hecha la solicitud de envío, el usuario pondrá cuidado en no alterar este buffer, mientras no reciba la respuesta del resultado del envío.

El formato del buffer en el cual se almacena el mensaje a enviar, reserva once (11) palabras antes del inicio del texto. El texto del mensaje iniciará a partir del byte inferior de la doceava palabra y su tamaño no sobrepasará los 538 caracteres (bytes). El gráfico 9-4 despejará cualquier duda:

El formato de la solicitud de envío es el siguiente:

```
QIO$C  IO.ACW,lun,[efn],[pri],[isb],[ast],<p1,p2,p3,p4,p5,p6>

    IO.ACW = Código de la función solicitada
        p1 = dirección virtual del inicio del buffer
        p2 = tamaño del buffer en bytes
             (tamaño del texto + 22 bytes reservados)
        p3 = Identificador del puerto de envío
        p4 = Identificador de la red de destino
        p5 = Identificador de la estación destino
        p6 = Identificador del puerto destino
    ......   Los parámetros omitidos son estándares.
```

Ahora bien el protocolo de envío, hace indispensable que el usuario incluya en la solicitud la dirección de un bloque de I/O estatus, a fin de conocer el resultado de la operación, pues

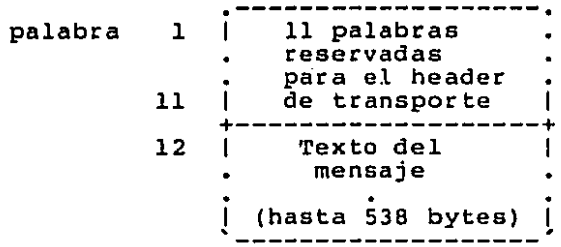
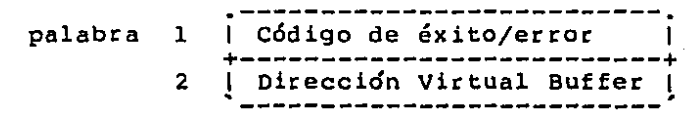


Figura 9-4: Formato del buffer para envío de Datagramas

aparte de si el envío se rechazó por error en los datos de control, una vez que el datagrama es enviado y HERMES recibe un reconocimiento de su entrega (ACK), se deposita en la primera palabra de este bloque, el código de éxito, o si después de un cierto número de retransmisiones no recibe el ACK, se deposita en la primera palabra el código de aborto. En cualquier caso la segunda palabra del bloque de I/O estatus regresa la dirección virtual del buffer enviado.

El resultado del envío en el bloque de I/O estatus tiene el siguiente formato:



Los posibles códigos de I/O estatus son los siguientes:

1 = Exito, el datagrama recibió el reconocimiento de su entrega.
-3 = Error, se abortó el envío, no hay quien reciba el datagrama.
-4 = Error, HERMES no consiguió identificador para el datagrama.
-6 = Error, buffer ilegal,fuera del espacio de direcciones de la tarea.
-10 = Error, identificador de la Red destino es inválido.
-11 = Error, identificador de la Estación destino es inválido.
-14 = Error, el puerto de orígen es inválido.
-15 = Error, el puerto de orígen pertenece a otra tarea.
-16 = Error, el puerto de orígen está inactivo.
-17 = Error, el tamaño del paquete es inválido.

Por último, se recuerda al usuario que en el servicio de DATAGRAMAS existe la posibilidad de paquetes duplicados o fuera de órden, inconveniente que será resuelto por los servicios de NARAD, que garantizan la secuencia y el control de flujo de mensajes.

I.8.5. Recibir un datagrama

Cuando un proceso de aplicación intercambia mensajes a través de la Estación de Transporte, necesita disponer de buffers para recibir en ellos los mensajes dirigidos a él, desde otros procesos, y el informar a HERMES de la existencia de estos buffers, es el objetivo de esta función.

Estos buffers para recepción de datagramas dispondrán de una area suficiente tanto para recibir el texto del mensaje, como

DIRECCIONES DE TRANSPORTE EN LA RED LOCAL NRO. 1 DEL IIMAS

ESTACION DE TRANSPORTE			PUERTOS	
Ubicacion	Sistema Operat	Ident	Fijos	Public
PDP 11/34 Dept. Ciencias de la Computac.	RSX-11M (20k)	1	1 - 10	11-30
PDP 11/10 Maestria Ciencias Computac.	RT-11	2	-	1 -32
PDP 11/40 RESMAC	RSX-11M (16k)	3	1 - 10	11-30

Figura 9-5:

información de control. Cuando HERMES recibe un datagrama e identifica que su destino es un proceso de aplicación en esta misma estación, revisa inmediatamente si éste tiene registrado algún buffer para recepción de datagramas y en caso afirmativo deposita ahí el texto incluyendo en las primeras cinco palabras los datos de la estación, la red y el puerto de orígen, como también el puerto de arribo y el tamaño del texto respectivamente, el mensaje es depositado a partir de la sexta palabra como se indica en el gráfico 9-6.

El formato de la instrucción es el siguiente:

QIO$C IO.ACR,lun,[efn],[pri],[isb],[ast],<buffer,size,puerto>

IO.ACR = Código de la función solicitada
buffer = Dirección virtual del inicio del buffer

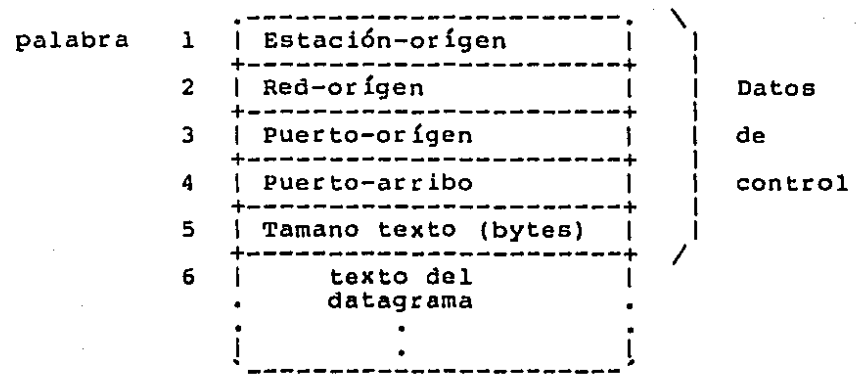
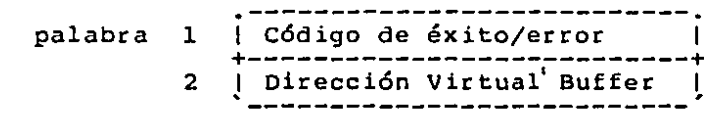


Figura 9-6: Formato de la entrega de un Datagrama

size = Tamaño del buffer en bytes (no menor a 20 bytes)
puerto = Identificador del puerto por el cual se espera la llegada del datagrama.
..... los parámetros omitidos son estándares

La directiva QIO emitida, quedará pendiente del evento de llegada de la carta, y entonces informará al usuario por medio de los mecanismos de banderas o rutinas asíncronas, y depositará el resultado de la operación en el bloque de I/O estatus cuyo formato es el siguiente:



Los posibles códigos de I/O estatus son los siguientes:

1 = Exito, la carta se recibió corectamente
-6 = Error, buffer ilegal,fuera del espacio de direcciones de la tarea.
-14 = Error, identificador del puerto de recepción es inválido.

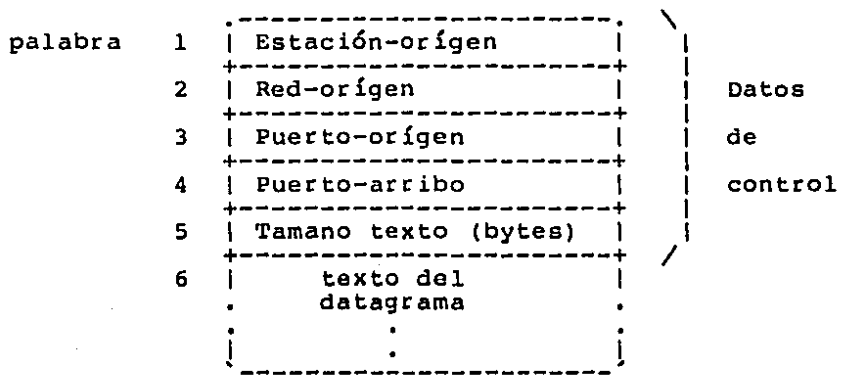
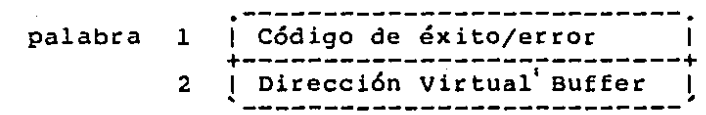


Figura 9-6: Formato de la entrega de un Datagrama

size = Tamaño del buffer en bytes (no menor a 20 bytes)
puerto = Identificador del puerto por el cual se espera la llegada del datagrama.
..... los parámetros omitidos son estándares

La directiva QIO emitida, quedará pendiente del evento de llegada de la carta, y entonces informará al usuario por medio de los mecanismos de banderas o rutinas asíncronas, y depositará el resultado de la operación en el bloque de I/O estatus cuyo formato es el siguiente:



Los posibles códigos de I/O estatus son los siguientes:

1 = Exito, la carta se recibió corectamente
-6 = Error, buffer ilegal,fuera del espacio de direcciones de la tarea.
-14 = Error, identificador del puerto de recepción es inválido.

-15 = Error, el puerto de recepción pertenece a otra tarea
-16 = Error, el puerto de recepción está inactivo
-17 = Error, el tamaño del buffer de recepción es menor a 20 bytes
-18 = Overflow, el tamaño del buffer es insuficiente para recibir el texto del mensaje, sin embargo quedan registrados los primeros 5 datos de control.

I.8.6. Determinar los puertos de una tarea

Cuando un proceso de aplicación quiere comunicarse con otro localizado en la misma estación, deberá informarse previamente del puerto que este otro tiene asociado. No existe inconveniente si el otro proceso es una tarea especializada que utiliza un puerto fijo convenido de antemano, sin embargo, si se trata de una tarea normal que utiliza un puerto público, este puede ser diferente cada vez que se activa, tomando en cuenta su asignación dinámica, en cuyo caso es indispensable preguntar a HERMES, el o los puertos que la otra tarea tiene asociados.

Al formular una tarea esta pregunta, debe suministrar a HERMES un buffer cuyas primeras tres palabras contienen el nombre de la tarea objeto de la pregunta. Este nombre tendrá un máximo de seis caracteres ASCII, y si son menos se completarán con blancos. El espacio del buffer será suficiente para recibir en él, la respuesta que incluye, el número de puertos que la otra tarea tiene asociados, y el identificador de cada uno de ellos en el formato indicado en la figura 9-7.

El formato de la instrucción es el siguiente:

QIO$C IO.ACE,lun,[efn],[pri],[isb],[ast],<buffer,size>

IO.ACE = Código de la función solicitada
buffer = Dirección virtual del inicio del buffer en el cual se pasa el nombre de la tarea investigada y al mismo tiempo se recibe la respuesta.
size = Tamaño del buffer en bytes
..... Los parámetros omitidos son estándares.

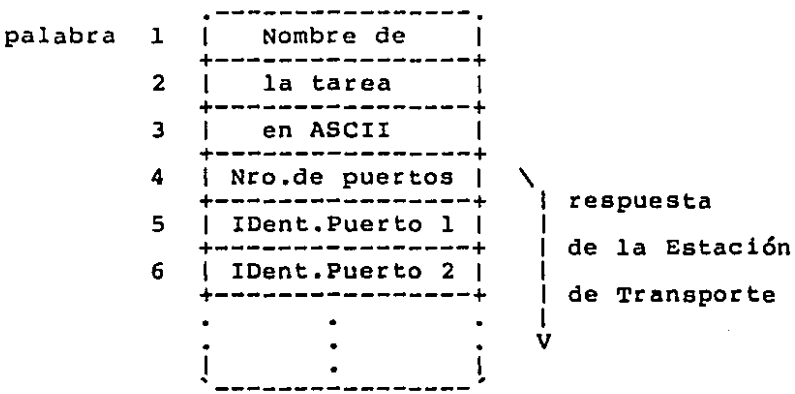
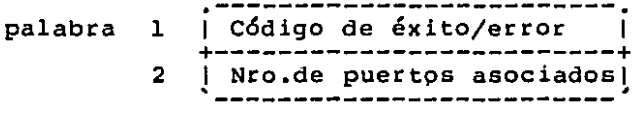


Figura 9-7:

El resultado de la operación queda registrado en el bloque de I/O estatus y su formato es el siguiente:



Los posibles códigos de I/O estatus son los siguientes:

1 = Exito, la operación tuvo éxito y la segunda palabra contiene el número de puertos de la otra tarea.
-6 = Error, buffer ilegal,fuera del espacio de direcciones de la tarea.
-18 = Overflow, el tamaño del buffer es insuficiente para

admitir los resultados, de todas maneras la segunda palabra contiene el número de puertos de la otra tarea.

Esta función es válida, tanto para investigar los puertos fijos, como los puertros públicos de la otra tarea, dentro de la misma estación. Sin embargo, el investigar los puertos de una tarea localizada en otra estación, requiere de los servicios de una tarea especial, el LOGGER a discutirse en la próxima sección.

I.9. Servicios adicionales proporcionados por el "LOGGER"

Como una extensión de los servicios de la Estación de Transporte, se pueden considerar los ofrecidos por una tarea privilegiada llamada "LOGGER", la cual está siempre activa, mientras lo está la Estación de Transporte y tiene asociado el puerto fijo número 1. Los servicios se solicitan ahora por medio de cartas que los procesos de aplicación envían al puerto 1 de la estación en la cual estos servicios interesan, y los resultados los devuelve el LOGGER igualmente a través de cartas.

Estos servicios se reducen por el momento a dos y en los párrafos siguientes se explicarán para cada caso el formato de la carta con el pedido y el de la respuesta.

I.9.1. Activar otra tarea

Si un proceso de aplicación necesita activar un segundo proceso, puede solicitarlo a través de una carta dirigida al LOGGER de la "estación remota" donde reside el proceso requerido. El texto del pedido (figura 9-8) contiene en la primera palabra

el código de la solicitud (125 octal = 85 decimal), y en las tres palabras siguientes los seis caracteres ASCII del nombre de la tarea, completándose con blancos si la longitud de éste es menor a seis.

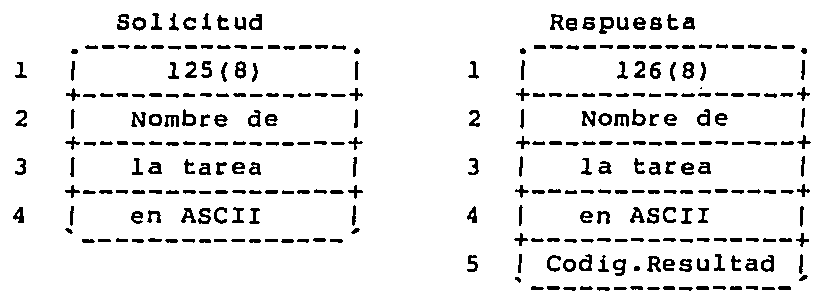


Figura 9-8: Activar una tarea - Formatos de Solicitud y Respuesta

El LOGGER al identificar la solicitud, extrae el nombre de la tarea y lo traduce a código RAD50, para utilizarlo en la directiva RQST$S que se encarga de activar la tarea, y prepara luego una carta con el resultado de la operación, cuyo texto contiene en la primera palabra el código de respuesta (126 octal = 86 decimal), en las tres palabras siguientes el nombre original de la tarea y en la última palabra, el código de resultado que corresponde al contenido del DSW (Directive Status Word), al término de la ejecución de la directiva RQST.

Los códigos de resultado, junto con sus nombres simbólicos son los siguientes:

IS.SUC = 1 Exito, la tarea fue activada

IE.UPN = -1 Error, no hay memoria dinámica suficiente
IE.INS = -2 Error, la tarea solicitada no está instalada
IE.ACT = -7 Error, la tarea ya se encontraba activa

I.9.2. Averiguar los puertos de otra tarea

Este servicio es de utilidad, si la "otra tarea" corre en una "estación remota", en cuyo caso la tarea solicitante dirige una carta al LOGGER de esa estación, incluyendo en el texto del pedido, el código de la solicitud (135 octal = 93 decimal), y en las tres palabras siguientes los seis caracteres ASCII con el nombre de la tarea investigada, completando este con blancos, si el número de caracteres es menor a seis. Ver figura 9-9

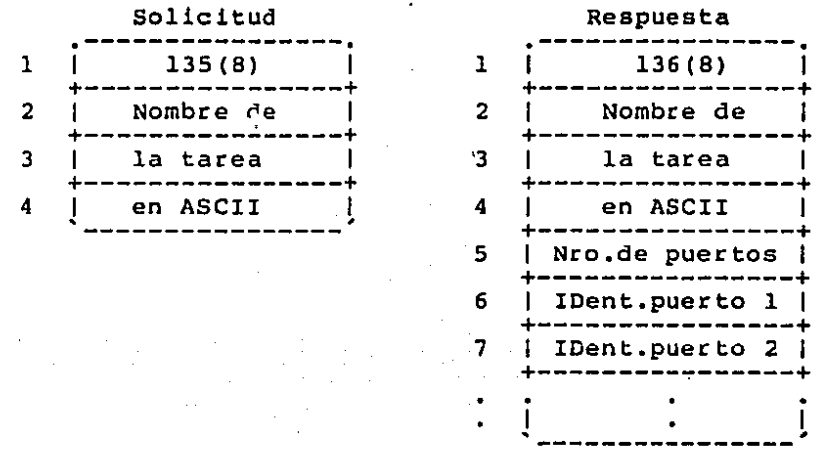


Figura 9-9: Puertos de otra tarea - Texto de Solicitud/Respuesta

El LOGGER toma el pedido y lo traduce en una solicitud a la Estación de Transporte Local, por medio de una directiva QIO, como se discutió en I.8.6, al término de la cual, prepara un

mensaje de respuesta, cuyo texto contiene en la primera palabra el código de respuesta (136 octal = 94 decimal) , el nombre de la tarea objeto de la solicitud en las tres palabras siguientes, y a continuación el número de puertos que dicha tarea tiene asociados, y los identificadores de éstos como se muestra en el esquema 9-9.

APENDICE II

HERMES: SISTEMA DE TRANSPORTE PROCESO-A-PROCESO DE DATAGRAMAS CONFIABLES PARA LA RED LOCAL DEL IIMAS

Manual del usuario en el sistema RT-11

RED LOCAL Nro. 1 - ESTACION Nro. 2

Gursharan S. Sidhu
José Villagómez C.

IIMAS, UNAM, México

Junio de 1982

II.1. RESUMEN

Este documento contiene el instructivo para el uso de los servicios de HERMES, para el transporte de datagramas confiables en el sistema RT-11, implementado en la máquina PDP 11/10 de la Maestría en Ciencias de la Computación.

II.2. Introducción

La interface entre el usuario y la Estación de Transporte en el sistema RT-11, lo constituyen un juego de rutinas en lenguaje ensamblador, las cuales son los comandos del usuario para solicitar los servicios básicos para la comunicación entre procesos por el sistema de datagramas confiables, denominado HERMES. Dichos comandos corresponden a las rutinas:

- ACTPTF: Activar un puerto fijo
- LIBPTF: Liberar un puerto fijo
- ENVCAR: Enviar un datagrama
- DAMECA: Recibir un datagrama

Sin embargo, el servicio de comunicación entre procesos, no solo implica que HERMES se limite a aceptar comandos, sino también a devolver información al proceso solicitante, indicando el éxito o la causa del fracaso de sus solicitudes, o notificarle el arribo de mensajes, y con tal propósito existen dos rutinas adicionales que se encargan de:

- SYNCET: Establecer la sincronía del diálogo entre HERMES y el usuario.
- QUEES: Examinar la información que HERMES entrega al usuario.

Además, puesto que los mensajes que HERMES entrega al usuario, pueden arribar en cualquier momento, estos se almacenan en una cola y por medio de una bandera:

- QFLAG: el usuario será advertido de la presencia de estos, cuando su valor es diferente de cero.

Todos los nombres mencionados deben identificarse como externos al programa del usuario, como se indica en la siguiente instrucción de MACRO-11:

```
.GLOBL ACTPTF, LIBPTF, ENVCAR, DAMECA, QFLAG, QUEES, SYNCET
```

Los programas fuentes de estas rutinas, están almacenados en el módulo INTET.MAC y los objetos en INTET.OBJ y deberán tomarse en cuenta a la hora del ligado del programa del usuario como se ilustra a continuación:

```
.RUN LINK
*userprog=userprog,..,INTET
*^C
.
```

Siendo RT-11 un sistema para un solo usuario, es indispensable que este último pueda operar la Estación de Transporte en el momento que la necesite, y con tal motivo se incluyen aquí la secuencia de comandos necesarios:

```
.LOAD RX              ;Cargar el manejador de recepción
.LOAD TX              ;Cargar el manejador de transmisión

.FRUN ESTTPT          ;Arrancar la Estación en Foreground
F>
Estación de Transporte lista !!
<CTR>B
B>
.RUN userprog         ;Arrancar el programa de aplicación
                      ;en Background
```

II.3. Rutinas de la interface HERMES-Usuario

Estas rutinas están escritas en el lenguaje ensamblador MACRO-11, auxiliadas por los PROGRAMMED REQUESTS del sistema y macros de alto nivel, y su descripción estará orientada al uso de éstas por programas de usuario escritos también en ensamblador. Los argumentos se pasan a través de registros, ya sea conteniendo directamente el valor, o como apuntadores de areas donde se depositan los argumentos tanto de entrada como de salida. Los registros no mencionados en la llamada a las rutinas no sufren alteración.

Cabe mencionar que dichas rutinas solicitan o reciben los servicios de HERMES mediante los Programmed Requests .SDAT y .RCVDC que permiten pasar bloques de información entre los programas en background y foreground. Además tienen acceso al area común de buffers en foreground para el envío o recepción de datagramas. El modo de operación de estas rutinas es algo similar a una directiva QIOW en el sistema RSX-11M, puesto que al pasar la solicitud a HERMES, el programa del usuario es suspendido, reanudando su ejecución al recibir respuesta a su solicitud. Este tipo de sincronización se logra con los Programmed Requests .SPND y .RSUM

Los mensajes que HERMES entrega al usuario, para informarle del arribo de un datagrama, o la llegada de un ACK o el aborto de un envío, son almacenados por la interfase en una cola circular, y se informa al programa del usuario a través de la bandera QFLAG,

si el diálogo es síncrono, o pasando el control a una rutina previamente especificada por el usuario, si el modo de recepción elegido es asíncrono.

II.3.1. Activar un puerto fijo

En el sistema RT-11 HERMES dispone de un conjunto de veinte (20) puertos públicos cuyo rango de numeración varía de 1 a 20, y cualquiera de ellos puede ser asociado por una tarea de usuario que desea comunicarse con un proceso de otra máquina.

El usuario proporcionará el identificador del puerto deseado en el registro R1, y llamará a la rutina con la instrucción JSR PC,ACTPTF. Se devuelve como resultado de la operación un código almacenado en una palabra, a la cual apunta el registro R2, y corresponden a:

0 = Exito, el puerto fue activado
1 = Error, el identificador del puerto es inválido
2 = Error, el puerto ya se encontraba activo

II.3.2. Liberar un puerto fijo

Este comando es el inverso del anterior, y con el se deshabilita la comunicación con otros procesos a través de este puerto, y cualquier paquete dirigido a este puerto, que arribe a la estación será ignorado, sin que el usuario se percate de ello.

Igualmente en este caso, el identificador del puerto en cuestión lo proporciona el usuario en el registro R1, y llamará a

la rutina con la instrucción JSR PC,LIBPTF. Al retornar el control al programa del usuario, el registro R2 apunta a una palabra que contiene el resultado de la operación y corresponde a:

0 = Exito, el puerto fue liberado
1 = Error, el identificador del puerto es inválido
2 = Error, el puerto ya se encontraba libre

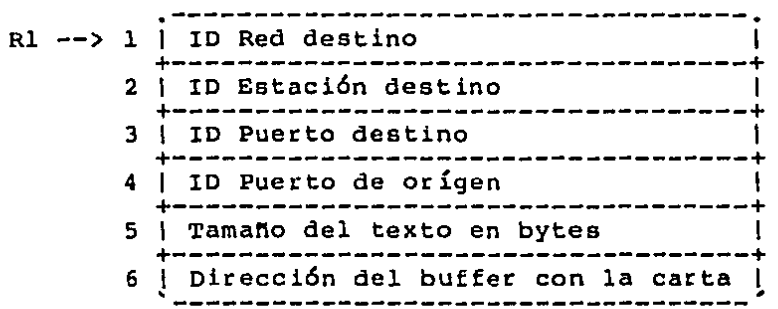
Se advierte al usuario, que HERMES en esta estación, no toma providencias, para el caso en que una tarea al concluir no liberó los puertos que asoció. Sin embargo es posible que otra tarea los libere, ya que siendo el sistema accesible por un solo usuario, los puertos no contienen ninguna información que los ligue específicamente al usuario que los activó.

II.3.3. Envío de un datagrama

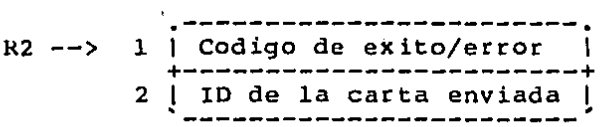
Cuando un programa de aplicación somete una carta a envío, HERMES la retransmite en tanto no reciba el reconocimiento de su entrega, o se agota un cierto límite de retransmisiones. En cualquier caso el usuario es informado del éxito o fracaso de la operación.

Los argumentos para el envío de una carta, los proporciona el usuario en un bloque de seis palabras apuntado por R1, en el que se incluye la dirección del destino, y la del buffer con el texto de la carta, cuya extensión no sobrepasará los 538 bytes. El comando se invoca con la instrucción JSR PC,ENVCAR y el formato

del bloque de parámetros se muestra en el esquema:



La respuesta a esta solicitud si es exitosa, se refiere únicamente a que la carta fue aceptada por HERMES y está ahora en la cola de transmisión, mas no que la carta llegó a su destino. Esta respuesta se almacena en un bloque de dos palabras apuntado por R2 con el siguiente formato:



Los posibles códigos de respuesta corresponden a:

0 = Exito, la carta esta siendo transmitida

1 = Error en el identificador del puerto local

2 = Error, el puerto local esta inactivo

3 = Error en el identificador de la Red-destino

5 = Error en el identificador de la Estación-destino

4 = Error, el buffer está inactivo

7 = Error, no se pudo asignar identificador al datagrama

10= Error, no hay buffer libre de envío

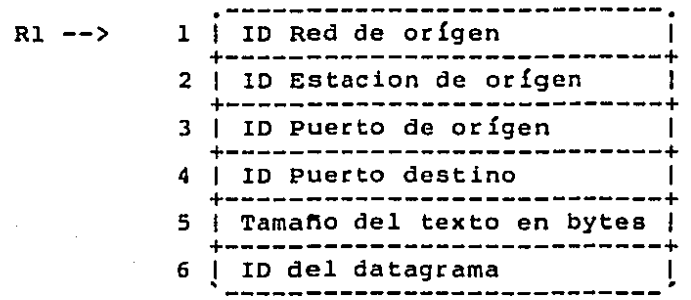
Si hubo éxito, la segunda palabra contiene el identificador que transporte asignó a este datagrama, y es de utilidad, puesto que HERMES hará referencia a este, para informar al usuario si el mensaje recibió ACK o se abortó.

II.3.4. Recibir un datagrama

Cuando el usuario, ha sido notificado de la llegada de un datagrama, debe gestionar su entrega, proporcionando un buffer para que la Estación deposite ahí el texto, junto con cierta información de control adicional. Dicha gestión es el objeto de este comando expresado por la instrucción JSR PC,DAMECA. Los parámetros de entrada a esta rutina, se pasan en registros y corresponden a:

- R0 = Apuntador a un bloque de dos palabras, el cual contiene parte de la información con la cual HERMES notificó al usuario el arribo del datagrama, correspondiendo la primera palabra a la dirección del buffer con el datagrama en la estación de transporte, y la segunda al número del canal de recepción.
- R1 = Dirección de un bloque de seis palabras, para recibir la información de control.
- R2 = Dirección de un buffer del usuario, donde se recibirá el texto.

Al término de la ejecución de este comando, el usuario dispondrá del texto del mensaje, y los seis valores de la información de control devueltos corresponden a:



II.3.5. Iniciar el diálogo con HERMES

Este comando es el primero que el usuario debe lanzar, para poder utilizar los servicios de transporte de HERMES. El programa de usuario que corre en "background" y el de la estación en "foreground", comparten una area común de buffers localizada en foreground, y el objetivo de este comando es informar a HERMES, que hay un usuario interesado en sus servicios, para que éste a su vez, le informe la dirección del area común, donde él podrá depositar o recibir sus mensajes.

Además el usuario puede establecer la política de recepción de mensajes, pudiendo especificar un modo de recepción síncrono o asíncrono, siendo en este último caso indispensable, proporcionar la dirección de una rutina asíncrona, a la cual se transferirá el control, cada vez que arriba un mensaje desde HERMES.

El formato de la instrucción es JSR PC,SYNCET y los parámetros de entrada a esta rutina se almacenan en los registros:

R0 = Modo de recepción solicitado.

(síncrono = 0 ; asíncrono = 1)

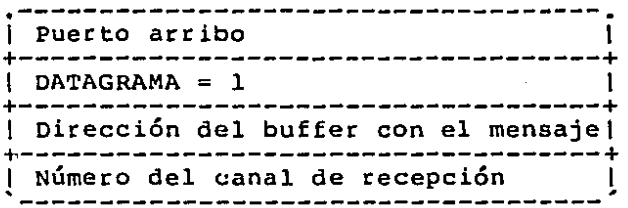
R1 = Dirección de una rutina del usuario que atenderá la llegada de mensajes, si el modo de recepción es asíncrono.

II.3.6. Examinar el primer elemento de la cola de mensajes

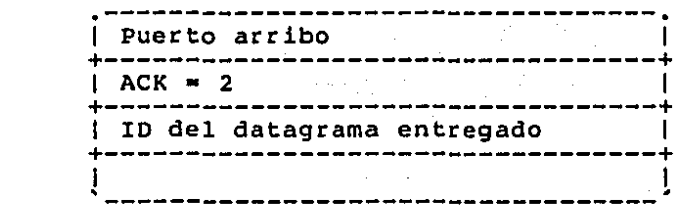
Los eventos de la llegada de un datagrama, o el arribo de un ACK a un puerto de esta estación, o el aborto del envío de un datagrama, son comunicados al usuario a través de mensajes captados por las rutinas de la interfase y almacenados en una cola circular, de donde los toma el usuario auxiliado por la rutina QUEES.

Esta instrucción cuyo formato es JSR PC,QUEES extrae el primer elemento de la cola y lo deposita en un bloque de cuatro palabras, cuya dirección la proporciona el usuario, como parámetro de entrada en el registro R1. El formato del bloque devuelto depende del tipo de mensaje recibido como se ilustra a continuación:

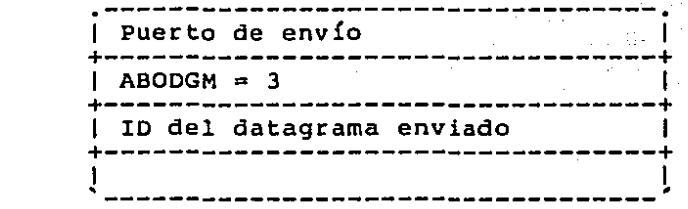
- Arribo de un datagrama



Arribo de un ACK



- Aborto de un envío



Si la rutina QUEES es invocada, cuando la cola está vacía, el código que identifica el tipo de mensaje es devuelto con el valor -10.

II.3.7. Bandera QFLAG

Sea cual fuere el modo de recepción de mensajes seleccionado por el usuario, siempre está a su disposición una bandera QFLAG, manipulada por las rutinas de la interfase, que toma un valor diferente de cero si hay mensajes en la cola recibidos desde HERMES, y es igual a cero si la cola está vacía. Su utilidad es clara en el modo síncrono de recepción, pues verificando ésta, el usuario puede proceder a examinar los mensajes recibidos.

II.4. Utilería

Entre los programas de aplicación desarrollados para las pruebas de esta estación, cabe mencionar a NLS, que permite al usuario enviar el listado de un archivo a la "impresora Printronix" de la estación 1 (PDP 11/34), y su manipulación requiere de las instrucciones:

```
.RUN NLS
Nombre del archivo:  xxxxxx      ;archivo a listar
```

Las actividades principales de NLS corresponden a:

- Verificar el archivo a listarse.
- Activar un puerto de transporte.
- Solicitar al LOGGER de la estación 1 activar el listador NETLIS, el cual debe encontrarse previamente instalado.
- Proceder al envío del archivo en pedazos de 512 bytes, que corresponden al tamaño de un bloque de disco.
- Enviar un mensaje de "fin de archivo", y desactivar NETLIS.
- Liberar el puerto de transporte.

Los dos programas NLS y NETLIS cooperan a través de un protocolo de alto nivel ad-hoc, que controla la secuencia del flujo y evita duplicados.

BIBLIOGRAFIA

[1] V.Cerf, A.McKenzie, R.Scantlebury, H.Zimmermann.
Proposal for an Internetwork End-to-End Transport Protocol.
General Note 96.1, International Federation for Internetwork Processing Working Group 6.1, February, 1978.

[2] RSX-11M Executive Reference Manual
Digital Equipment Corporation, Maynard,Massachusetts, 1979.

[3] RSX-11M Guide to Writing an I/O Driver
Digital Equipment Corporation, Maynard,Massachusetts, 1979.

[4] C.Loyo, G.S.Sidhu, R.Segovia.
Breve análisis retrospectivo de las redes de computadoras.
Technical Report, Grupo ECO, Departamento de Ciencias de la Computación, IIMAS-UNAM, 1979.

[5] L.Mochán.
Macros de Alto Nivel para PDP-11.
Reporte ECO 80-2 Serie Naranja: Investigaciones No.237, IIMAS-UNAM, Julio, 1980.

[6] PDP-11 Peripheral Handbook
Digital Equipment Corporation, 1978.

[7] G.S.Sidhu and J.Villagómez.
NARAD: A System of Byte Pipes for IIMASNET.
Specification and Maintenance Manual, IIMAS-UNAM, Mexico, Julio, 1982.

[8] Louis Pouzin and Hubert Zimmermann.
A Tutorial on Protocols.
In Proceedings of the IEEE, pages 1346-1370. Vol.66, No.11, November, 1978.

[9] PDP-11 Processor Handbook
Digital Equipment Corporation, 1978.

[10] RT-11 Software Support Manual
Digital Equipment Corporation, 1975.

[11] G.S.Sidhu.
Descripción Preliminar de la Red Local IIMAS.
Technical Report REDLAC Doc,No. 10, IIMAS-UNAM, Octubre 1, 1979.

[12] G.S.Sidhu.
Compuertas (Gateways) para la Interconección de Redes de

Computadoras.
Technical Report REDLAC Doc,No. 15, IIMAS-UNAM, Julio 18, 1979.

[13] G.S.Sidhu y A.Siperstein.
Una Arquitectura de Protocolos Propuesta para REDLAC.
Technical Report REDLAC Doc,No. 18, IIMAS-UNAM, Octubre 1, 1979.

[14] G.S.Sidhu y J.Hernandez R.
A High Speed Broadcast Local Network for Multi-User Machines.
In Proceedings of the Fifteenth International Symposium on Minicomputers and Microcomputers. MIMI Mexico City, April, 1981.

[15] G.S.Sidhu y R.Segovia.
Redes de Computadoras -- Una Introducción.
In Memorias de IEEE Mexicon 77. Mexico, 1977.

[16] Andrew S.Tanenbaum.
Computer Networks.
Prentice Hall, 1981.