UNIDAD ACADEMICA DE LOS CICLOS
PROFESIONAL Y DE POSGRADO
DEL
COLEGIO DE CIENCIAS Y HUMANIDADES

MAESTRIA EN CIENCIAS
DE LA
C O M P U T A C I O N

INSTITUTO DE INVESTIGACIONES
EN
MATEMATICAS APLICADAS Y EN SISTEMAS

STAI SISTEMA DE TRANSFERENCIA DE ARCHIVOS
Y
API ACTIVADOR DE PROCESOS
PARA Red-IIMAS

T E S I S

que para obtener el grado de

Maestro en Ciencias de la Computación

P R E S E N T A

Ing. Pedro Rafael Márquez Gutiérrez

UNAM – Dirección General de Bibliotecas Tesis Digitales Restricciones de uso

a mis padres

Pedro Márquez

y

Ana María Gutiérrez

...Sediento de saber lo que Dios sabe,
Judá León se dio a las permutaciones
de letras y complejas variaciones
y al fin pronunció el Nombre que es la Clave,

la Puerta, el Eco, el Huesped y el Palacio,
sobre un muñeco que con torpes manos
labró, para enseñarle los arcanos
de las Letras, del Tiempo y del Espacio.

El simulacro alzó los soñolientos
párpados y vió formas y colores
que no entendió, perdidos en rumores.
y ensayó temerosos movimientos.

...

El rabí lo miraba con ternura
y con algún horror. ?Cómo (se dijo)
pude engendrar este penoso hijo
y la inacción dejé, que es la cordura?

?Por qué di en agregar a la infinita
serie un símbolo más? ?Porque a la vana
madeja que en lo eterno se devana,
di otra causa, otro efecto y otra cuita?

En la hora de angustia y de luz vaga,
en su Golem los ojos detenía.
?Quién nos dirá las cosas que sentía
Dios, al mirar a su rabino en Praga?

El Golem. Jorge Luis Borges, 1958.

CONTENIDO

LISTA DE FIGURAS

PREFACIO

Muchas personas relacionadas con el procesamiento de datos han llegado a reconocer claramente (y en ocasiones con sufrimiento), la necesidad de contar con una arquitectura de sistemas de comunicación que facilite la distribución del procesamiento de la información. Una arquitectura de esta índole debe proporcionar protocolos de comunicación estándares entre diversos sistemas cuyas estructuras internas e interfaces de usuario pueden ser diferentes. El objetivo es facilitar una comunicación más general en sistemas de gran diversidad y rápido cambio. Además, la estructura debe permitir cambios futuros preservando parte de la inversión en el sistema y facilitando una rápida evolución de otras partes del mismo.

Un sistema experimental diseñado con esta ideología es Red-IIMAS. Red-IIMAS tiene su origen en el Proyecto REDLAC, Red Latinoamericana de Computadoras, proyecto CONACYT PIT/EE/OEA/79, patrocinado por la Organización de Estados Americanos OEA, que data desde 1979.

Red-IIMAS es una red local de computadoras tipo Ethernet que transmite a una velocidad de 10 megabits/seg, diseñada conjuntamente por los Departamentos de Computación y Diseño de Sistemas Digitales del Instituto de Investigaciones en Matemáticas Aplicadas y en Sistemas de la Universidad Nacional Autónoma de México.

En esta tesis se detalla el diseño y la implantación de la Capa de Aplicaciones de la Arquitectura de Red-IIMAS, conformada actualmente por el STA: Sistema de Transferencia de Archivos y el AP: Activador de Procesos.

La tesis se encuentra dividida en dos partes:

Parte I. Capítulos 1 y 2.

El Capítulo 1 es puramente introductorio y en él se ven algunos conceptos generales acerca de las redes de computadoras, sus objetivos y, particularmente, la arquitectura y servicios del Proyecto Red-IIMAS de Alta Velocidad. En el Capítulo 2 se hace una revisión de lo que son los High Level Languages HLL, High Level Protocols HLP y File Transfer Protocols FTP, finalizando con la presentación del STA de Red-IIMAS.

Parte II. Capítulos 3 al 8.

La segunda parte de la tesis está dedicada enteramente a la descripción del trabajo desarrollado en la Capa de Aplicaciones de Red-IIMAS. El Capítulo 3 muestra todos los conceptos manejados en el diseño y desarrollo de los módulos constitutivos del Sistema de Transferencia de Archivos y del Activador de Procesos. El Capítulo 4 trata de los aspectos relacionados con las interfaces de los sistemas. El Capítulo 5 habla acerca de los protocolos de comunicación de la Capa de Aplicaciones. El Capítulo 6 se dedica al Subsistema de Correo Electrónico. El Capítulo VII describe al Activador de Procesos y en el Capítulo VIII se encuentra el manual

del usuario. Finalmente, existen un apartado de CONCLUSIONES donde se hacen algunas consideraciones finales acerca del trabajo, un Apéndice A donde se muestran los distintos tipos de datos que manejan los sistemas, un Apéndice B que describe los formatos de archivos que soporta el STA, y un Apéndice C que es un glosario de términos técnicos.

AGRADECIMIENTOS

Deseo hacer patente mi agradecimiento al cuerpo de sinodales que participó en la revisión del escrito, así como en la presentación del examen de grado:

M.C. Raymundo Segovia Navarro	Director de Tesis.
Dr. Héctor Haro Guzmán	Ex-Jefe del DDSD y Ex-Coordinador del Area de Computación
Dr. Federico Kuhlmann Rodríguez	Director de la DEPFI y Miembro del Consejo Asesor de Cómputo
Dr. Alberto Tubilla Estefan	Jefe del Depto. de Computación
Dr. Felipe Bracho Carpizo	Ex-Jefe del Depto. Computación Asesor del Rector

Asimismo, deseo agradecer muy especialmente el interés y esfuerzo dedicado en la revisión del escrito por parte del Dr. Haro, así como de la M.C. Alma Rosa Vera Solís, y los consejos y (muchas) discusiones entabladas desde el inicio de este trabajo con el M.C. Oscar Cantú.

CAPITULO I

INTRODUCCION

UNA PERSPECTIVA

Se estima que las redes de computadoras vieron sus primeros días en la creación de sistemas operativos de tiempo compartido, cuyo principal interés era intentar optimizar el uso de la computadora evitando al máximo los tiempos muertos ocasionados por operaciones lentas de entrada y salida. Después se diseñaron sistemas para controlar el acceso a los medios de comunicación, manejar terminales, y sistemas interactivos. Las industrias de computación y comunicaciones, crearon protocolos de línea asíncronos ('start/stop') y síncronos (binario síncrono), y fueron surgiendo terminales de distintos tipos, controladores de terminales, procesadores frontales, concentradores de línea, y equipo relacionado. Todo ésto para poder acrecentar la disponibilidad del equipo de cómputo, que por entonces era demasiado costoso. Hubo así, un enorme crecimiento en casi todos los aspectos de las redes de comunicación y aunado a las reducciones en costo y mejoramiento en las características de velocidad de transmisión y calidad de los enlaces, formaron parte de los cimientos de una nueva revolución en el campo de la computación y las comunicaciones.

El factor clave que desencadenó la explosiva investigación y desarrollo de redes de computadoras en (virtualmente) todo el mundo, fue un proyecto patrocinado por el Departamento de Defensa de los Estados Unidos (DoD) bajo su Agencia de Proyectos de

Investigación Avanzada (ARPA), para la construcción de una red de recursos compartidos (ARPANET), cuyo contrato se otorgo a la firma Bolt, Beranek & Newman, Inc. (BBN) con asiento en Massachusetts, E.U. El gran mérito de este proyecto es que, además de ser pionero, sentó muchos de los conceptos sobre los cuales se fincaron investigaciones y desarrollos posteriores que han venido a culminar en el establecimiento de modelos de referencia y estándares internacionales para la creación de redes de computadoras [1,2,3]. Observemos, pues, la presencia de los ingredientes necesarios de toda buena fórmula: el caldo de cultivo, y el catalizador.

Esta perspectiva quedaría incompleta si no intentáramos predecir el futuro. Las tecnologías d computación y las comunicaciones están caminando hacia una coexistencia útil y compatible que está dando origen a cientos de redes con miles de terminales dispersas por todo el mundo. Será posible contar con redes que permitan la automatización de procesos industriales y administrativos, y que apoyen fuertemente a la educación en todos sus niveles. Las redes podrán ser utilizadas como un medio de comunicación de alta calidad para la transmisión de datos, voz e imagen. Se crearán redes de grande y muy grande alcance, que permitirán un mejor intercambio del conocimiento humano. Será factible acceder a bases de datos para obtener reservaciones en aviones, trenes, hoteles, etc. en cualquier parte del mundo y su confirmación inmediata. Se dispondrá de programas de uso público que ayude al hombre común a realizar operaciones bancarias y de impuestos desde su hogar. El periódico electrónico será una realidad, pudiendo el lector seleccionar la información de su interés. Asimismo, se tendrán

bibliotecas públicas asequibles desde el hogar. Es decir, la aplicación de las redes de computadoras solo se verá limitada por la imaginación.

OBJETIVOS DE LAS REDES DE COMPUTADORAS

Por ahora las redes de computadoras (redes, para abreviar) se encuentran en una fase de creciente desarrollo y aceptación, en espera de estándares maduros [4,5,6,7,8] que permitan el desarrollo armónico del campo mediante la aplicación coherente de esfuerzos. Existe, sin embargo, un cierto consenso general acerca de cuáles deben ser los objetivos primarios que se persiguen con el uso de las redes. A continuación describimos algunos que consideramos de los más importantes [9]:

* Compartir recursos (distantes) tales como información periféricos o procesadores. Este es quizá el más común de los objetivos.

* Proporcionar comunicación entre procesos, así como entre usuarios y procesadores. Para que se pueda dar ésto, es necesario que la red provea una comunicación (casi) libre de errores.

* Mejorar la confiabilidad de las redes a través de resguardo y redundancia.

* Distribuir las funciones de procesamiento.

* Permitir el control centralizado de un sistema geográficamente disperso.

* Compatibilizar equipo y 'software' disímil.

* Proporcionarle al usuario una máxima eficiencia a un costo mínimo.

* Crecimiento incremental de las capacidades de cómputo.

TAXONOMIA DE LAS REDES

Resulta sumamente difícil dar una clasificación de redes que contemple todos los aspectos relevantes de una red y que por ende, pueda describir cabalmente cualquier red dada. Esta dificultad se debe a la gran cantidad de aspectos de diseño importantes involucrados en el desarrollo de las redes, que además son utilizados generalmente como figuras de clasificación. En vez de intentar dar una nueva clasificación, o retomar alguna, daremos los criterios más empleados en la taxonomía de redes y la forma como se ven desde estas perspectivas.

Clasificación de las redes de computadoras de acuerdo:

a) Al diseño de la subred de comunicaciones [10]:

- Punto-a-punto.
- De difusión (broadcast).

b) A la tecnología del transporte de mensajes [9]:

- Conmutación de circuitos.
- Conmutación de mensajes.
- Conmutación de paquetes.

c) A la distancia entre procesadores [10]:

- Máquinas de flujo de datos.
- Multiprocesadores.
- Redes Locales.
- Redes de Largo Alcance.
- Interconexión de Redes de Largo Alcance.

d) A la topología de la red [10]:

- Estrella.
- Anillo.
- Arbol.
- Completa.
- Irregular.
- 'Bus'.

e) Al tipo de modulación empleada [11]:

- Banda base (baseband).
- Banda ancha (broadband).

f) Al tipo de control de acceso al medio de comunicación:

- Carrier Sense Multiple Acces with Collision Detect CSMA/CD
- CSMA with Collision Avoidance CSMA/CA
- Token-Passing
- Time Division Multiplexing TDM
- Frecuency Division Multiplexing FDM

Como se observa, existen muy diversas características, todas ellas importantes, que en forma única no son capaces de describir completamente a una red , por lo que generalmente se emplea un

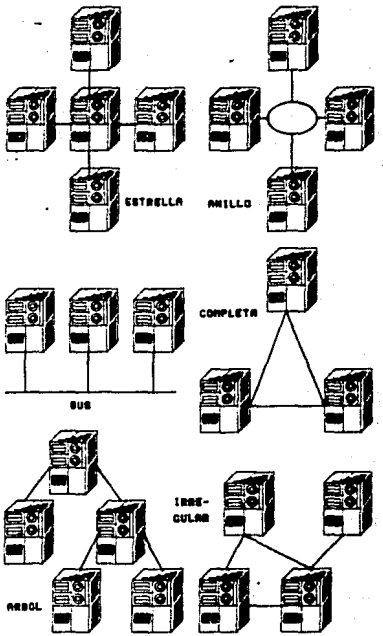


FIGURA 1.1
TOPOLOGIAS DE REDES

subconjunto de ellas para poder tener una mayor y mejor especificación de las redes. Han habido intentos por desarrollar esquemas de clasificación que permitan cubrir todos los aspectos importantes que matizan a las redes de computadoras, pero ha resultado infructuoso a la fecha por ser, tal vez, un aspecto más bien académico [12,13].

REDES DE AREA LOCAL (LAN)

En uno de los esquemas de clasificación dados anteriormente se menciona que las redes pueden ser de área local. Esta categoría presenta a su vez tantas características importantes, que se ha convertido en un término sumamente utilizado y en una de las maneras más comúnmente empleadas en la clasificación de redes. A pesar de ello, no existe una definición universalmente válida de lo que representa una red de área local, aunque sí una gran intuición al respecto. Resulta importante detenerse un poco en ésto y verlo con cierto detalle debido a que actualmente existe un enorme desarrollo y demanda por las que se han dado en denominar 'redes de área local', o simplemente 'redes locales'. De entre las definiciones más aceptadas se encuentra la que establece que una red de área local es aquella que satisface las siguientes propiedades básicas [14]:

1. Pertenecen usualmente a una sola organización.
2. Son generalmente locales, es decir, alcanzan distancias del orden de unos cuantos kilómetros.
3. Poseen algún tipo de tecnología de conmutación.

Existen, sin embargo, definiciones más holgadas [15] que permiten la exclusión de la propiedad número 1. En algo que la gran mayoría está totalmente de acuerdo (y que en gran parte dá origen a su nombre), es en que la distancia que puede abarcar una red local está restringida solo a unos cuantos kilómetros. Se encuentran remitidas generalmente a cubrir edificios o campuses. Debido a este rasgo, las redes locales mantienen varias propiedades que las distinguen de sus contrapartes, las redes largas, y que las hacen ser mucho muy atractivas. Dentro de estas propiedades se encuentran [16,17]:

1. Alta velocidad en la transmisión de datos.
2. Gran ancho de banda.
3. Comunicación en 'broadcast'.
4. Baja razón de errores.

La conjunción de todas estas virtudes hacen que las redes locales sean especialmente importantes en muchas aplicaciones cuyos requerimientos de rendimiento son fuertes. Así, por ejemplo, podemos verlas trabajar en ambientes de control de procesos industriales, de investigación, oficinas, comercio, etc [18].

A pesar de la gran cantidad de criterios que existen para la clasificación de las redes, y de que aún algunas de éstas no poseen una definición clara, existen métodos y modelos que nos permiten diseñar y analizar las redes de computadoras bajo un conjunto de premisas formales. Así, es posible tratar con las redes desde la perspectiva de su 'arquitectura' que trata de evitar caer en las ambiguedades comparativas a que llevan las

distintas formas de implantación.

ARQUITECTURAS DE REDES

Debido al creciente desarrollo que han experimentado las redes de computadoras desde sus inicios, y a la diversidad de sus implantaciones, fue necesario desarrollar una metodología que facilitara su diseño, análisis y enseñanza. Para ello se utilizó una técnica de ingeniería que permite atacar un problema complejo de manera más sencilla, simplificándose con ésto el cumplimiento de los tres principios anteriores. La ideología que sostiene esta técnica consiste en dividir el problema en varias partes con funciones bien definidas, para que puedan ser trabajadas en forma independiente [19].

Así fue que en 1977 la Organización Internacional de Estándares (ISO- International Standards Organization) decidió crear un subcomité (SC16) para la Interconexión de Sistemas Abiertos (OSI- Open System Interconnection), cuyo objetivo básico era estandarizar las reglas de interacción entre sistemas interconectados. De esta manera, solo el comportamiento externo de los Sistemas Abiertos debería ser ajustado al esquema de OSI, mientras que su organización y funcionamiento interno se encontrarían fuera del alcance de los estándares.

El término 'Abierto' se eligió con el fin de enfatizar el hecho de que si un sistema satisface los estándares internacionales, éste se encontrará abierto a todos aquellos que igualmente los obedezcan[20].

El método de partición mencionado, conocido comúnmente como subdivisión en capas, es una técnica de estructuramiento que permite que una red de sistema abiertos pueda verse como la composición lógica de una sucesión de capas, cada una de las cuales abarca los servicios de los inferiores y aisla a las superiores de los entrebajos de estos manejos. A este conjunto de capas, así como a los protocolos (reglas) de interacción entre capas de la misma jerarquía, y a las interfaces entre capas adyacentes, se le denomina la Arquitectura de la red. ISO determinó el conjunto de principios que deberían ser considerados en la definición del número específico de capas que debería contener la arquitectura OSI, y con base en ellos, se llegó al establecimiento de una configuración de siete capas. A continuación daremos una somera descripción de las funciones que deben realizar cada una de las siete capas, pudiendo encontrar el lector una descripción más detallada en [2,20,21].

MODELO OSI DE REFERENCIA

La arquitectura de OSI comprende siete capas que son [24]:

Capa Física

Proporciona características mecánicas, eléctricas, funcionales, y procedimentales para activar, mantener, y desactivar conexiones físicas para la transmisión de bits entre entidades de enlace de datos, posiblemente a través de sistemas intermedios relevando transmisiones entre capas físicas.

Capa de Enlace de Datos

Proporciona medios procedimentales y funcionales para establecer, mantener, y liberar conexiones de enlace de datos entre entidades de red. Una conexión de enlace de datos se construye sobre una o más conexiones físicas.
El objetivo de esta capa consiste en detectar y, posiblemente, corregir errores que puedan ocurrir en la capa física. Permite, además, que la Capa de Red controle la interconexión de circuitos de datos dentro de la Capa Física.

Capa de Red

Proporciona medios procedimentales y funcionales para intercambiar unidades de datos del servicio de red, entre dos entidades de transporte sobre una conexión de red. Les dá independencia a las entidades de transporte sobre consideraciones de ruteo y conmutación, asociadas con el establecimiento y operación de una conexión de red.

Capa de Transporte

Proporciona un servicio universal de transporte en conjunción con los servicios provistos por las capas subyacentes. Permite además una transferencia de datos transparente entre entidades de sesión y las libera de los detalles involucrados en una comunicación confiable (libre de errores). Optimiza el uso de los servicios de comunicación disponibles y proporciona medios efectivos para la transferencia de datos.

Capa de Sesión

Proporciona los medios para que las entidades de presentación (procesos) organicen y sincronicen su diálogo, y manejen su

intercambio de datos. Representa la interfaz de usuario hacia la red, y añade funciones orientadas a la aplicación a los servicios básicos de comunicación de la Capa de Transporte.

Capa de Presentación

Su propósito es presentar la información a las entidades de aplicación de manera que preserve su significado en tanto que resuelve diferencias sintácticas. Esta capa trata solo con el aspecto sintáctico de los datos y no con su semántica. Los servicios que proporciona son para el manejo de la entrada, intercambio, despliegue, y control de datos estructurados.

Capa de Aplicaciones

Proporciona un medio para que los procesos de aplicación puedan acceder al ambiente OSI. Su propósito es servir como una ventana entre procesos de aplicación en comuncación, usando el ambiente OSI para intercambiar información significativa.
Los protocolos de esta capa le proporcionan al usuario final servicios de información distribuida que, se encuentran a disposición de las aplicaciones, controlan la operación del sistema, y manipulan adecuadamente los datos.

ARQUITECTURA DE Red-IIMAS

Red-IIMAS [22,23] ha sido un proyecto de investigación sobre redes locales de computadoras, que ha permitido obtener muchos logros en diferentes aspectos. Su diseño e implantación ha exigido también la definición de una arquitectura suficientemente flexible como para poder alcanzar los objetivos pretendidos con su desarrollo.

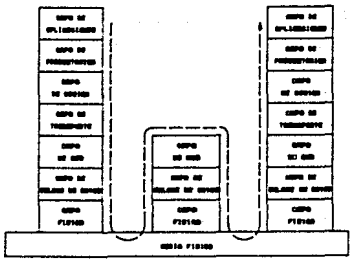


FIGURA 1.2
MODELO OSI DE REFERENCIA

La arquitectura de Red-IIMAS comprende cuatro capas básicas cuyos nombres y objetivos se resumen a continuación:

Capa de Comunicación de Datos

Se dedica al manejo de la disciplina de transmisión física de datos a través del canal de comunicación (cable coaxial), empleando un protocolo de acceso al medio CSMA/CD tipo Ethernet, con codificación en banda base, y a una velocidad de transmisión de 10 Mbps.

Capa Inter-red

Esta capa (no desarrollada aun) se encarga del enrutamiento, control de flujo, etc, entre anfitriones integrados a distintas redes.

Capa de Transporte

Esta capa proporciona un servicio fin-fin de comunicación virtual libre de errores de mensajes de tamaño variable, bajo un protocolo de circuitos virtuales, para los procesos de la capa de aplicación.

Capa de Aplicaciones

Proporciona los servicios últimos de la red a los usuarios (procesos de aplicación). Actualmente consta del Transferencia de Archivos, Correo Electrónico, y Activador de Procesos.

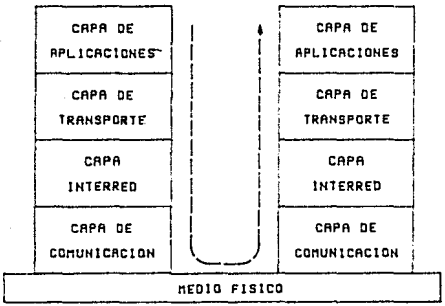


FIGURA 1.3
ARQUITECTURA DE Red-IIMAS

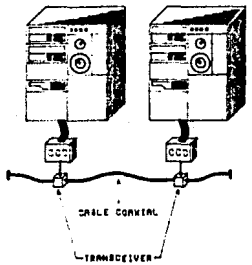


FIGURA 1.4
TOPOLOGIA DE Red-IIMAS

CAPA DE APLICACIONES DE Red-IIMAS

La arquitectura de Red-IIMAS se ha tratado de apegar en lo posible al modelo de referencia de ISO en la definición de los nombres de sus capas y las funciones que deben cumplir. Sin embargo, aún persisten ciertas diferencias de opinión al tratar de precisar la correcta localización de ciertos servicios de la red.

Protocolos de Transferencia de Archivos

Lo anterior es especialmente cierto cuando tratamos de dilucidar los servicios que pueden ser ofrecidos por la capa de aplicaciones. Así, por ejemplo, algunos autores sitúan a los protocolos de transferencia de archivos dentro de la capa de presentación [10], mientras que otros [8] los prefieren en la capa de aplicaciones, ésto de acuerdo al modelo OSI. Esta ambiguedad parece ser resultado de una confusión semántica en lo que encierra el siguiente concepto: de acuerdo a Harold C. Folts [2], 'la capa de aplicaciones sirve como una ventana para que los <<procesos de aplicación>> accedan al ambiente de interconexión de sistemas abiertos. Esta es la única capa que proporciona servicios directamente a los procesos de aplicación'. Esta definición hace pensar que existen aún ciertos procesos (de aplicación) que se encuentran fuera del modelo OSI, puesto que la capa de aplicaciones es la última capa considerada dentro de esta arquitectura. Añade además que, 'la definición avanzada en el Modelo de Referencia CCITT OSI identifica tres categorías de elementos de servicio: común, específicos de la aplicación, y específicos del usuario. Los elementos comunes proporcionan las

capacidades para una comunicación OSI independiente de la naturaleza de la aplicación específica. Los elementos específicos de la aplicación son requeridos por ciertas categorías de aplicaciones para apoyar las funciones del proceso de transferencia de información, tales como transferencia y manipulación de trabajos (jobs), transferencia de archivos, y terminales virtuales. Los elementos específicos del usuario se aplican a las funciones que son únicas al área de aplicación particular tales como reservaciones en aerolíneas y operaciones bancarias". Por otro lado, para A. S. Tanenbaum [10] tanto los protocolos de transferencia de archivos como los de terminales virtuales son servicios de la capa de presentación, y me atrevería a decir que seguramente para él los sistemas de reservaciones en aerolíneas y de operaciones bancarias, son materia de la capa de aplicaciones. Esta opinión se sustenta en que para él los 'procesos de aplicación' son aquellos que se encuentran dentro de la capa de aplicaciones, lo cual debería ser lógico pero, como apuntábamos anteriormente, no es del parecer de Folts. A pesar de todo , existe una luz que nos permite adoptar una postura válida en este problema conceptual. No se ha dicho en ninguna parte que un proceso de aplicación no pueda ser representado por un ser humano y, de hecho, de acuerdo a las dos tesis anteriores éste no queda excluído. Así pues, si un servicio de transferencia de archivos puede ser solicitado directamente por un ser humano, y éste se encuentra fuera de las consideraciones arquitecturales del modelo OSI, entonces este servicio debe ser ubicado en la última capa de cualquier arquitectura de redes que se trate, y por lo

mismo, en la de aplicaciones de acuerdo a OSI. Ahora bien, si existe además la posibilidad de que este servicio sea empleado por algún otro sistema, reservaciones en aerolíneas, por ejemplo, y éste a su vez ser empleado por seres humanos (es ilógico pensar en un sistema de este tipo sin aplicación para las personas) entonces no será posible que ambos sistemas coexistan al mismo nivel. Así, dependiendo de cuantas etapas existan entre el 'usuario final' y un servicio particular, es como podremos situar la capa correspondiente. Para el caso que nos atañe, el servicio de transferencia de archivos puede ser igualmente válida colocarlo dentro de la capa de aplicaciones o de presentación, dependiendo de quién es su usuario.

Correo Electrónico y Activador de Procesos

Otros de los servicios que presta la Capa de Aplicaciones de Red-IIMAS son el Correo Electrónico y el Activador de Procesos. Existe poca duda acerca de si es correcto colocar al Correo Electrónico en esta capa, pero sí puede haberla con respecto al Activador de Procesos.

En principio diremos que el Activador de Procesos AP es comúnmente referido como el 'logger', y es quien se encarga de coordinar la correcta activación de los servidores de la red. Lo mismo que el servicio de transferencia de archivos, el activador de procesos sufre de problemas de ubicación ya que A.S. Tanenbaum, en particular, lo describe tácitamente como una tarea al mismo nivel que la capa de transporte según el modelo OSI. Si especulamos un poco sobre el punto tendremos que hechar mano de un concepto

adicional: Existe una entidad activa por cada una de las capas de una arquitectura, en cada sistema involucrado en una interconexión. Estas entidades se comunican entre sí utilizando protocolos 'peer' que conllevan la información de control necesaria para apoyar las comunicación entre procesos de aplicación cooperantes. Esto no significa otra cosa sino que existe una comunicación real entre un par de capas que se encuentran al mismo nivel en la arquitectura en cuestión. Es por ésto que si, por ejemplo, el Sistema de Transferencia de Archivos STA se va a comunicar con el Activador de Procesos, y el STA se encuentra dentro de la Capa de Aplicaciones, y sólo puede haber comunicación real entre capas de la misma jerarquía, entonces el AP debe ser también un servicio de la Capa de Aplicaciones.

CONCLUSION

Todas estas ideas han sido esgrimidas como argumentos de validación en la definición del nombre y funciones de la capa superior de Red-IIMAS. Concretando, la arquitectura de Red-IIMAS contempla una capa que presenta protocolos de alto nivel para la consecución de servicios de administración de recursos de cómputo, denominada Capa de Aplicaciones, conformada en la actualidad por el Sistema de Transferencia de Archivos, el Correo Electrónico, y el Activador de Procesos.

BIBLIOGRAFIA

1. Padlipsky, M.A., 'A Perspective on the Arpanet Reference Model', IEEE Proceedings of INFOCOM 83, 1983, pags. 242-253.

2. Folts, Harold C., 'A Tutorial on the Open Systems Interconnection Reference Model'. Open Systems Data Transfer, number 1, June 1982, pages 2-2

3. Myers, Ward., 'Toward a Local Network Standard'. IEEE Micro, August 1982, pages 28-45.

4. Local Area Networks. Logical Link Control. ANSI/IEEE Std 802.2-1985 ISO/DIS 8802/2.

5. Local Area Networks. Token-Passing Bus Access Method. ANSI/IEEE Std 802.4-1985 ISO/DIS 8802/4.

6. Local Area Networks. Carrier Sense Multiple Access with Collision Detection. ANSI/IEEE Std 802.3-1985 ISO/DIS 8802/3.

7. Burrus, John., 'Features of the Transport and Session Protocols' The National Bureau of Standards Report Number ICST/HLNP-80-1, March 1980, pages 1-71.

8. Lewan,D. & Long,H.G., 'The OSI File Service', Proceedings of the IEEE, December 1983, pages 1414-1419.

9. Ahuja, Vijay., Design and Analysis of Computer Communication Networks. McGraw-Hill 1985.

10. Tanenbaum, A.S., Computer Networks. Prentice-Hall, Inc. 1981.

11. Charafas, Dimitris N., Designing and Implementing Local Area Networks McGraw-Hill, 1984.

12. Anderson, G.A. & Jensen, E.D., 'Computer Interconection Structures: Taxonomy, Characteristics, and Examples' Computing Surveys, Vol.7 No.4, diciembre 1975.

13. Sai,Inder M. & Aggarwal Krishnan K., 'A Review of Computer Communication Network Classification Schemes' IEEE Communications Magazine, Vol.19, No.2, marzo 1981, pp. 24-32.

14. Thurber, Kenneth J. & Freeman, Harvey A., 'Architecture Considerations for Local Computer Networks' Proceedings, 1st International Conference on Distributed Computing Systems, 1979, pp. 131-142.

15. Clark, David D.; Pogran, Kenneth T. & Reed, David P., 'An Introduction to Local Area Networks' Proceedings of the IEEE, noviembre 1978, pp. 1497-1517.

16. Cotton, Ira W., 'Technologies for Local Area Computer Networks' Computer Networks, Vol. 4, noviembre 1980, pp. 197-208. North-Holland.

17. Stallings, William, 'Local Networks' ACM Computing Surveys, Vol. 16, No. 1, marzo 1984. pp. 3-41.

18. Licklider, J.C.R. & Vezza, Albert, 'Applications of Information Networks' Proceedings of the IEEE, Vol. 66, No. 11, noviembre 1978, pp. 1330-1346.

19. Green, Paul E., 'An Introduction to Network Architecture and Protocols' IEEE Transactions on Communications, Vol.Com-28, No.4, abril 1980, pp. 413-424.

20. Zimmermann, Hubert, 'OSI Reference Model-The ISO Model of Architeture for Open Systems Interconnection' IEEE Transactions on Communications Vol. Com-28, No.4, abril 1980, pp. 425-432.

21. Tanenbaum, Andrew S., 'Network Protocols' Computing Surveys, Vol. 13, No.4, diciembre 1981, pp.453-489.

22. Hernández, Jaime; Cantú, Oscar; Márquez, Pedro; Nava, Francisco, 'Descrición de la Red Local IIMAS de Alta Velocidad' Comunicaciones Técnicas, Serie Naranja: Investigaciones, No.367, 1984, IIMAS-UNAM, México, D.F.

23. Cantú, Oscar; Márquez, Pedro; Nava, Francisco. 'Red-IIMAS de Alta Velocidad' Memorias, Primera Conferencia Nacional DECUS México, noviembre 1986.

24. Konangi, V., Dhas, C.R., 'An Introduction to Network Architectures' IEEE Communications Magazine, Vol. 21, No. 7, octubre 1983.

CAPITULO II

PROTOCOLOS DE ALTO NIVEL

LAS CAPAS SUPERIORES DEL MODELO OSI

Si observamos al modelo OSI de referencia desde una perspectiva más amplia, estaremos en condiciones de realizar una simplificación conceptual que nos permitirá abordarlo de manera más directa.

Las funciones de cada una de las de esta arquitectura van sufriendo un desacoplamiento gradual de los aspectos relacionados con el control de los mecanismos de comunicación, conforme subimos en la jerarquía. Es fácil darse cuenta que las capas íntimamente relacionadas con este control son la capa física, la de enlace de datos, y la de red. Debido a esta homogeneidad de funciones es posible abstraer estas tres capas en una sola y asignarle el objetivo genérico de resolver todos los aspectos relacionados con el control de la comunicación física entre sistemas abiertos. Si ahora desplazamos nuestro punto de visión y observamos al modelo desde su nivel superior, encontraremos que las funciones de algunas capas están más orientadas a la prestación de servicios a los usuarios y poco, o nada, tienen que ver con aspectos de comunicación. Este comportamiento lo encontramos en las tres capas superiores, o sean, las capas de aplicación, presentación, y sesión. Igual que procedimos con las capas inferiores, agruparemos estas últimas tres capas en una sola, y le encomendaremos la tarea general de controlar el acceso a los recursos de cómputo, su

manejo y prestación de servicios.

Si hubiésemos tenido la intención de crear una arquitectura de red seguramente habríamos llegado a la identificación de este mismo par de meta-capas. Sin embargo, es fácil advertir que este esquema es sumamente rígido puesto que todas las aplicaciones de la capa superior deben efectuar una comunicación con la inferior, ocasionando que se incremente grandemente su complejidad al tener que contar con mecanismos de control y coordinación que permitan llevar a cabo esta comunicación ordenadamente. De aquí se desprende la conveniencia de interponer un nivel adicional entre las dos capas, que realice todas las tareas de coordinación, ganándose así una simplificación en el diseño de las dos capas adyacentes, al disponerse de una interfaz única para todos los procesos de aplicación residentes y otra para la capa inferior. Todo este planteamiento nos lleva a considerar que una arquitectura mínima deseable para cualquier red de computadoras debe constar de tres capas básicas. Si el lector ha seguido el desarrollo de estas ideas, le resultará fácil identificar a la capa intermedia como la capa de transporte del modelo OSI, la capa de la que faltaba hablar. Si prosiguiésemos el análisis con la misma orientación, seguramente llegaríamos a desarrollar una arquitectura muy semejante al modelo OSI, lo cual nos demuestra en cierta forma la validez del modelo.

Como comentario adicional a este apartado diremos que el modelo OSI posee una estructura jerárquica justa al asignar tres capas para funciones orientadas a la comunicación, tres para funciones orientadas al usuario y una intermedia que concilia intereses.

CATEGORIZACION DE CAPAS

Con base en la simplificación del modelo OSI que hemos realizado es aún posible conceptualizarlo como dos grandes grupos [1]: las capas superiores, y las inferiores.

Las ideas en que se fundamenta el diseño de las capas inferiores resultan bastante familiares debido al gran trabajo previo que se ha hecho en el campo de las comunicaciones, y a decir de P.F. Linington [1], las capas superiores son más complejas. La característica distintiva de estas últimas es que guardan una relación más sutil y cercana hacia la computación que hacia la comunicación, y es práctica común incluir dentro de esta categoría a las tres capas superiores del modelo OSI, denominando como inferiores a las restantes. Tal vez pueda subsistir cierta reticencia en incluir a la capa de transporte dentro de la categoría de las capas inferiores, pero podemos aclararlo recordando que el principal objetivo de la capa de transporte es proporcionar el medio mediante el cual se pueda intercambiar información entre procesos, es decir, realiza una función de comunicación. Ahora bien, es conveniente aclarar que el calificativo de capas inferiores se dá en función de su ubicación dentro de la arquitectura, y por su connotación funcional de manejo de 'bajo nivel', y no es en forma alguna un término despectivo.

PROTOCOLOS DE ALTO NIVEL

Como consecuencia directa de la categorización del modelo OSI en capas superiores e inferiores, se ha acuñado la expresión

'protocolos de alto nivel' para denominar al conjunto de protocolos avocados al control de los procesos de computación involucrados en una aplicación [2]. Estos protocolos representan los lenguajes de alto nivel en un ambiente de computación distribuida, y a diferencia de los protocolos de las demás capas de una arquitectura, es necesario diseñar tantos protocolos como servicios se piense soportar; es decir, cada servicio de la red instaura el suyo propio. Debido a la necesidad de apoyar usos muy variados es necesario mantener un enfoque más flexible en la capa de aplicaciones, lo que provoca un incremento de su complejidad.

Los protocolos de alto nivel juegan un papel muy importante, ya que son quienes dan significado a las secuencias de bits que se envían a través de la red. Sin ellos, sería prácticamente imposible desarrollar aplicaciones con sentido. Estos protocolos realizan dos grandes labores: a) controlan los servicios ofrecidos por los recursos de una red, y b) enlazan en forma simétrica dos procesos que forman parte de una misma aplicación.

COMPONENTES DE UN PROTOCOLO DE ALTO NIVEL

Un protocolo de alto nivel, en inglés HLP (High Level Protocol), debe resolver una gama de aspectos que van desde los formatos de los mensajes que son intercambiados entre los procesos de aplicación, hasta los relacionados directamente con las particularidades propias del servicio. Podemos, sin embargo, distinguir tres partes importantes en cualquier HLP:

a) Lenguaje: Describe cual es el propósito funcional de los

intercambios de información entre procesos. Se debe anticipar la manera en la cual otro proceso utilizará el servicio, y proporcionar dentro del lenguaje los medios para que estos procesos pidan y reciban el servicio que necesitan. A este componente se le conoce más comunmente como protocolo.

b) Codificación: Toda comunicación que se dé entre un par de procesos deberá poseer una estructura capaz de ser generada e interpretada por ellos, para poder imponer un orden adecuado en su diálogo. A esta estructura se le denomina también como formato.

c) Transporte: La tercer y última parte del HLP es el mecanismo de recolección y entrega de bits de un proceso a otro. Para el caso en el que los procesos residen dentro de un mismo anfitrión este mecanismo lo proporciona el sistema operativo, y si se encuentran en máquinas distintas, la red. Aclaro que se utiliza el término 'red' en su más amplia connotación, es decir, implica no solo al 'hardware', sino a las demás capas de la arquitectura. Es así como se seguirá utilizando esta voz a lo largo del escrito, a menos que se especifique otra cosa.
Lo que en realidad debe importarnos del transporte son sus propiedades, como por ejemplo, velocidad, retardo, razón de errores, manejo de 'buffers', etc., pero no los metodos mismos de comunicación. Debido a esta consideración, podemos asumir que los HLP no pueden depender de procesos que comparten otros recursos, como memoria o archivos, esto es, el protocolo debe proporcionar todo lo necesario para la comunicación entre procesos.

Es importante separar estos tres aspectos de la comunicación entre

procesos cuando se diseñan y describen HLP's. Estos componentes no son independientes, por el contrario, tienen interacción y ésto debe ser tomado en cuenta en el momento del diseño, ya que las decisiones tomadas para cada uno de ellos puede llegar a impactar fuertemente la realización de los restantes.

DISEÑO Y DESARROLLO DE PROTOCOLOS DE ALTO NIVEL

Daremos a continuación algunos de los lineamientos generales que deben ser observados en el diseño y desarrollo de los HLP's, tomando como base las tres partes constitutivas tratadas en la sección anterior.

Lenguaje

La parte más importante del diseño de un HLP es claramente su lenguaje. Su principal reto es tratar de que sea independiente del dispositivo y, además, estándar. Sin embargo, es difícil abstraer todas las propiedades comunes a una gama de dispositivos y aplicaciones, en un diseño que permanezca sencillo. En muchas ocasiones resulta imposible evitar que peculiaridades de los sistemas operativos o 'hardware' especial, destruyan la sencillez de un protocolo. La red no es un contribuidor extrínseco al problema, ya que sólo añade unas cuantas dificultades a las intrínsecas de un sistema aislado que involucra múltiples procesos asíncronos en comunicación. No obstante, ya que la presencia de la red hace posible la comunicación entre diversos equipos de cómputo y dispositivos, de aquí se origina mucha de la presión en tratar de alcanzar la independencia del dispositivo y

la estandarización.

Independencia sobre el dispositivo: Los diseños que tratan de alcanzar este objetivo mantienen un forcejeo constante entre la generalidad y la estandarización. Conforme se añaden más capacidades a los dispositivos o se posee una gama más amplia de aplicaciones, la estandarización se vuelve más difícil.

La generalidad puede llevar también a una mayor complejidad, además de que los diseños generales son difíciles de desarrollar, pueden llevar demasiado tiempo avizorarlos, y requieren usualmente más recursos de procesamiento. Al otro extremo se encuentran los protocolos que son sumamente sencillos y que, aún así, pueden proporcionar un buen servicio.

Independientemente de que los protocolos sean generales o sencillos, siempre mantendrán un cierto grado de independencia del dispositivo y, dependiendiendo de su tipo, será posible aplicar en su diseño alguna de las dos técnicas siguientes.

1) Técnica del dispositivo virtual. Bajo esta técnica se diseña un estándar que especifica el conjunto de operaciones que deben realizar los procesos. En este caso puede llegar a ser necesario simular las operaciones que no soportan algunos de los procesos.

2) Enfoque parametrico. Esta técnica consiste en la sustentación de un diálogo entre las partes en comunicación para poder definir las capacidades de ambos, y poder establecer el 'universo de discurso'.

Los métodos empleados en alcanzar la independencia del dispositivo pueden influir profundamente en la estructura de los programas de

aplicación que emplean el HLP.

Combinación de Protocolos

Otro aspecto que no debe perderse de vista es que, aunque una aplicación haga uso de varias HLP's frecuentemente independientes, en la conexión de sus partes constitutivas, y éstos empleen trayectorias lógicas de comunicación distintas, pueden llegar a competir, sin querer, por ciertos recursos.

Robustez

Un HLP proporciona un enlace de comunicación vital en una aplicación distribuida sobre varias computadoras, enlace cuya falla puede ser costosa. El proporcionar confiabilidad para poder prevenir estas fallas requiere atención a todos los niveles. El programa de aplicación puede guardar posiblemente el estado operativo de tal manera que en la eventualidad de una catástrofe, sea posible regresar a un punto cercano restituible y continuar con la operación normal.

Especificación

El desarrollo de los lenguajes de los HLP's dependen normalmente de la interpretación adecuada de especificaciones detalladas. Este nivel de detalle resulta más de la necesidad que tienen todas las implantaciones de un HLP de comunicarse a través de un lenguaje idéntico, que de la complejidad misma del protocolo. Lo que se busca es una solución simple y elegante que ofrezca los servcios necesarios y mantenga una realización directa.

Es difícil escribir una especificación precisa para un HLP y,

aunque es similar a la especificación de una interfaz en 'software', difiere en dos aspectos. Primero, debe ser entendida por todos los realizadores acostumbrados a distintos lenguajes de programación, sistemas operativos, convenciones, y nomenclatura. Segundo, la especificación debe proporcionar suficiente guía para su desarrollo ya que es posible que el realizador no este familiarizado con el HLP en cuestión.

División en Capas

Esta técnica ayuda a reducir la complejidad de los HLP. Un HLP puede ser desarrollado encima de otro, pero a veces ésto es impedido por un mal diseño o por malas consideraciones.

Codificación

La codificación puede ser un tema sencillo o complejo. Es simple cuando sólo se quieren transmitir símbolos de un vocabulario bien definido. Los problemas en la codificación surgen cuando tratamos de representar objetos que poseen distintas interpretaciones en diferentes máquinas, por ejemplo, números de punto flotante, conjunto de caracteres, etc. Aunque es posible caracterizar con precisión los rangos de aceptación, es extremadamente difícil diseñar un protocolo que trabaje apropiadamente cuando un transmisor desea enviar un objeto que no puede ser interpretado por el receptor. Estos problemas se manejan generalmente, garantizando interpretaciones exclusivamente en subconjuntos restringidos, por ejemplo, caracteres ASCII.

Transporte

Del trabajo en HLP's ha emergido una lección muy importante acerca de los sistemas de transporte. No existe una vista abstracta única del mecanismo de transporte que se aplique a todos los HLP's; distintos protocolos necesitan acceder a distintos tipos de transportes.

Desarrollo de Protocolos de Alto Nivel

Para que un HLP resulte útil en una red de recursos compartidos, se deben observar varios desarrollos para muchas de las computadoras de la red. Una de las principales metas del diseño de los HLP's es que sean realizados fácilmente. Aunque la mayoría de los HLP's podrían ser desarrollados sin demasiada dificultad, a veces los complican los ambientes de programación de muchas computadoras. Muchos sistemas operativos incorporan acceso a una red en forma tardía y por lo tanto proporcionan mecanismos rebuscados. Algunos otros tienen un apoyo muy pobre para poder efectuar comunicación entre procesos. Finalmente, las herramientas para depurar procesos asíncronos son igualmente deficientes. En un ambiente de red, este problema es exacerbado por la incapacidad de examinar el estado de los procesos cooperantes o repetir las condiciones que ocasionaron un error. Aún el detener un proceso que se encuentra bajo prueba puede ser muy difícil: ¡el proceso contraparte puede detectar inactividad prolongada y abortarse!.
Se pueden ayudar a las tareas de desarrollo y depuración si se toman buenas medidas en el diseño del HLP. Por ejemplo, es recomendable contar con comandos que lleven a su estado inicial a

los procesos, y medios para reportar errores precisos. Algunas veces se pueden interconstruir ayudas explícitas para depuración dentro de un HLP. Dividir los HLP en capas también facilita su implantación. Si se construye un HLP encima de un sistema de transporte obligado a transmitir datos en forma confiable, se pueden eliminar muchas de las posibilidades de errores relacionados con el manejo del tiempo.

PROTOCOLOS Y LENGUAJES DE ALTO NIVEL

Tanto los lenguajes de programación de alto nivel, como los protocolos, son definiciones abstractas, pero precisas, de una computación o comunicación, que se pueden traducir en operaciones primitivas proporcionadas por una computadora o un sistema de comunicación. Los lenguajes de programación de alto nivel nacieron del deseo de poder expresar algoritmos en un lenguaje natural para la aplicación. Similarmente, un HLP se expresa más claramente en un lenguaje que se relaciona íntimamente con la aplicación; los detalles de codificación y transporte no son de interés primario para el diseñador y realizador de HLP's.
Una idea poderosa en los lenguajes de alto nivel, en inglés HLL (High Level Language), es el concepto de 'type', que es un método utilizado en la definición de objetos abstractos. Un compilador emplea las definiciones de <tipo> para elegir la manera de representar los objetos en la memoria física de la computadora. Con esta ayuda, se libera al programador de los detalles de codificar los objetos en memoria, y se concentra en su empleo para facilitar la construcción de su programa.

El concepto de tipo también es importante en los HLP. Generalmente un mensaje se divide en 'campos', que son secuencias de bits a los que se les asigna un significado particular. Pero para el diseñador de HLP's, estos campos son simples codificaciones de tipos empleados en el programa de aplicación. Así, las estructuras de datos juegan un papel clave en ambos ambientes, ya que estructuran la memoria para los HLL, y los mensajes para los HLP.
Los HLP comúnmente transmiten registros que consisten de un comando seguido por los valores de los argumentos. De esta forma, los HLP pueden ser expresados en términos de un 'protocolo de llamadas a procedimientos'. Esta clase de protocolos es muy frecuente cuando un solo programa de aplicación de varios módulos se distribuye en una red. Sin embargo, no todos los HLP se pueden expresar convenientemente con metodología de este estilo, y aún en estos casos resulta de gran valor estructurar las comunicaciones en registros mediante el empleo de declaraciones de tipo apropiadas. Este tipo de declaraciones deben ser compartidas, desde luego, tanto por el transmisor como por el receptor para asegurar que ambas partes apliquen la misma interpretación a los mensajes.
Las técnicas de 'data abstraction' en los HLL alientan métodos para el diseño de sistemas en software que son similares a las abstracciones requeridas para lograr la independencia del dispositivo o para definir estándares. Finalmente, los HLL's se encuentra muy relacionados con los métodos para especificar con precisión la interfaz con los módulos en software, en parte, para que el empleo que hagan otros programas de estos módulos pueda ser

verificado cuidadosamente.

A pesar de esta similitud entre los HLL y los HLP, existen ciertas ideas que se deben desarrollar aún más.

1) Definiciones de tipo: Actualmente, los HLL emplean definiciones de tipo que son muy 'débiles' y dependen de detalles de la computadora en la cual se ejecutan.

2) Comunicación entre procesos: Muchos lenguajes nuevos empiezan a proporcionar este tipo de capacidades, empleando algunos, la idea de transmitir tipo de registros como los mencionados en párrafos anteriores. Sin embargo, ninguno proporciona automáticamente el enlace vital entre un registro variable y la llamada al procedimiento.

3) Ligado: Idealmente, un proceso que ofrece un servicio de HLP puede distribuir una definición del protocolo para recibir el servicio. Esta definición puede ser comparada contra la definición del protocolo de un padre potencial al momento de iniciar la conexión.

4) Asincronía: Necesitamos mejores métodos para documentar secuencias e intercambios de mensajes. Como en cualquier sistema de multiprocesamiento, debemos preocuparnos de 'deadlocks' y aspectos de sincronización. Esto puede ser verdaderamente problemático si empleamos un mecanismo de transporte que a veces entrega mensajes fuera de secuencia u ocasionalmente los pierde.

5) Documentación: Es también importante contar con una definición más funcional. Debe ser claro que un programa de computadora

expresado, tal vez, como las operaciones abstractas de un proceso que lleva a cabo el protocolo, es mejor herramienta de documentación que la prosa.

Concluyendo: Romper el problema en aspectos de lenguaje, codificación, y transporte, es sólo una ayuda. Aún más importante, se debe entender perfectamente la aplicación en ambientes sencillos, libre de restricciones de estandarización antes de que se intente el diseño de un protocolo estándar.

PROTOCOLOS DE TRANSFERENCIA DE ARCHIVOS

Sabemos que una de los objetivos principales de las redes de computadoras es poder compartir la mayor cantidad de recursos de cómputo disponibles en la red. Una manera de cumplir esta tarea es a través del acceso remoto a todos los 'hosts' enlazados a la red mediante cualquier terminal que tenga este privilegio. De hecho, generalmente es la forma primaria de utilización de cualquier red. Para resolver los conflictos de manejo de los distintos tipos de terminales se diseñan protocolos, denominados de terminales virtuales, en inglés VTP (Virtual Terminal Protocol). Sin embargo, aunque de gran valía, resulta insuficiente este único servicio para explotar a su máxima capacidad las redes de computadoras. Otro recurso de cómputo sumamente importante que es necesario poner a disposición de la mayor cantidad posible de usuarios que lo requieren, es la información almacenada en los anfitriones. Es indudable que este servicio constituye la principal tarea de muchas redes, e inclusive existen alguna diseñadas exclusivamente para hacer uso de la información generada en los puntos

geográficos donde se encuentran localizadas las computadoras. Además, resulta difícil imaginar una red de computadoras de valor en la cual no se tenga acceso remoto a la información almacenada en las computadoras de la red. Aclaramos que el término información se utiliza en su sentido más general, implicándose cualquier colección de 'bytes' que pueden representar tanto datos como programas.

Al servicio de red que se encarga de satisfacer el compartimiento de información se le conoce comúnmente como protocolo de transferencia de archivos, en inglés FTP (File Transfer Protocol). Podemos decir entonces, que un FTP es un estándar que permite transferir, manejar y tener acceso a la información almacenada en, o movida entre, sistemas.

Se pudiera pensar que contar con un mecanismo de esta índole es añadir demasiada complejidad y sofisticación, y que sería suficiente enviar la información por el canal de comunicación y recibirla al otro extremo para ser procesada, máxime si se cuenta con un método confiable de transmisión de datos. Pero esta idea es demasiado ingenua, como veremos a continuación.

Hasta este momento hemos estado hablando de que es muy importante poder compartir la información almacenada en archivos dispersos por la red, y que para realizar esta tarea adecuadamente es conveniente contar con un FTP. Sin embargo, no hemos dicho porque es importante transferirlos. Los archivos se transfieren básicamente para tres cosas:

a) Para almacenarlo y poder recuperarlo posteriormente.

b) Para imprimirlos.

c) Para ejecutarlo como programa o utilizarlo como datos.

Es claro que estas funciones no representan mayor complejidad cuando se realizan dentro de una computadora, ya que su ejecución se encomienda al sistema de manejo de archivos local, controlado continuamente por el sistema operativo residente. Pero cuando operamos en un ambiente de red surgen un sinfín de complicaciones que deben ser resueltas. Estas dificultades tienen poco que ver con la necesidad de tener que utilizar un canal externo de comunicación, y enfrentarse por ende, con aspectos relacionados, ya que es responsabilidad de las capas inferiores resolver estas vicisitudes. La verdadera complejidad no estriba, pues, en la diferencia de ambiente operativo, sino en aspectos propios de su naturaleza.

Analicemos mejor cada uno de los tres propósitos para transferir archivos mencionados anteriormente.

Si un archivo va a ser transferido para ser almacenado en un anfitrión remoto, el requerimiento esencial es que cuando se recupere se obtenga una copia exacta bit por bit, del original. Si esta recuperación siempre la efectuara quien la envía, entonces la tarea resulta más sencilla puesto que la información puede ser guardada exactamente como se recibe, sin ninguna transformación. Pero si la intención es ponerla a disposición de distintos usuarios, será necesario encontrar una transformación que nos preserve la semántica de la información de modo tal, que cualquier anfitrión de la red pueda interpretarla y efectuar las operaciones necesarias para adecuarla a su forma nativa. Esta función de transformación puede requerir, por ejemplo, convertir entre distintos códigos (ASCII vs. EBCDIC), caracteres de control de

carro (CR, LF, FF, VT), registros (longitud fija vs. longitud variable), y si tratamos con máquinas de distintas longitud de palabra puede ser necesario diseñar algoritmos de relleno ('padding').

Muchas de las consideraciones anteriores son directamente aplicables al caso en que se desea imprimir un archivo, ya que es necesario observar el tipo de convención utilizada para el control de carro, que por ejemplo, puede ser que el archivo posea control de carro estilo Fortran en la columna 1, o convención ASCII. Y aún cuando el conjunto de caracteres sea el mismo, e.g. ASCII, pueden llegar a ocurrir divergencias en la acción generada por un mismo caracter. Si a ésto se añade la posibilidad de imprimir gráficos la tarea puede volverse sumamente complicada.

En caso de que la información transferida represente datos o un programa es imprescindible un diseño más sofisticado. Si se trata de datos los problemas surgen cuando existen diferencias en la representación de la información, y bajo este evento puede no ser simple la tarea de moverla. Por ejemplo, si se tratan de transmitir enteros binarios y una de las computadoras los representa en complemento a 1's y la otra a 2's, o si difieren en longitud de palabra, en cualquiera de los casos se requerirá realizar una transformación. De cualquier forma se puede encontrar alguna función de transformación que nos permita trabajar en la mayoría de las situaciones, pero, si estamos tratando con programas, tantas cosas pueden ir mal que todo puede empeorar considerablemente y llegar a ser intratable.

Otro aspecto igualmente importante, es el mecanismo de

identificación de archivos. Esto es un gran problema dada la gran diversidad de convenciones de especificación existentes que van desde las estructuras jerárquicas, hasta las planares. Este es, tal vez, el problema de computación distribuida atacado menos satisfactoriamente, que ha dado como resultado soluciones menos estándares. El tratamiento usual es identificar al sistema destino y emplear sus propias convenciones.

Además de todo ésto, es necesario contar con mecanismos adecuados de sicronización, transferencia de datos, y recuperación de errores, que son aspectos importantes relacionados con el envío y recepción de la información.

OBJETIVOS DE LOS PROTOCOLOS DE TRANSFERENCIA DE ARCHIVOS

Los objetivos que se persiguen al diseñar los protocolos de transferencia de archivos son diversos, y generalmente cada desarrollo posee características muy particulares. No obstante, es posible englobarlos dentro de uno general: poder transferir, manejar, y tener acceso a la información almacenada, o movida, entre sistemas. Este objetivo puede ser visto como un intento por extender a nivel de la red, los servicios de manejo de archivos de una computadora. Existen, sin embargo, distintos grados de alcance en el diseño y pueden ir desde el desarrollo de servidores de archivos (file servers) hasta un sistema de manejo de archivos distribuido. En el primer caso, lo que se persigue es proporcionar un medio de almacenamiento remoto y centralizado para estaciones de trabajo conectadas a él vía una red de comunicaciones, facilitando el compartimiento de datos entre las estaciones de

carro (CR, LF, FF, VT), registros (longitud fija vs. longitud variable), y si tratamos con máquinas de distintas longitud de palabra puede ser necesario diseñar algoritmos de relleno ('padding').

Muchas de las consideraciones anteriores son directamente aplicables al caso en que se desea imprimir un archivo, ya que es necesario observar el tipo de convención utilizada para el control de carro, que por ejemplo, puedu ser que el archivo posea control de carro estilo Fortran en la columna 1, o convención ASCII. Y aún cuando el conjunto de caracteres sea el mismo, e.g. ASCII, pueden llegar a ocurrir divergencias en la acción generada por un mismo caracter. Si a ésto se añade la posibilidad de imprimir gráficas la tarea puede volverse sumamente complicada.

En caso de que la información transferida represente datos o un programa es imprescindible un diseño más sofisticado. Si se trata de datos los problemas surgen cuando existen diferencias en la representación de la información, y bajo este evento puede no ser simple la tarea de moverla. Por ejemplo, si se tratan de transmitir enteros binarios y una de las computadoras los representa en complemento a 1's y la otra a 2's, o si difieren en longitud de palabra, en cualquiera de los casos se requerirá realizar una transformación. De cualquier forma se puede encontrar alguna función de transformación que nos permita trabajar en la mayoría de las situaciones, pero, si estamos tratando con programas, tantas cosas pueden ir mal que todo puede empeorar considerablemente y llegar a ser intratable.

Otro aspecto igualmente importante, es el mecanismo de

identificación de archivos. Este es un gran problema dada la gran diversidad de convenciones de especificación existentes que van desde las estructuras jerárquicas, hasta las planares. Este es, tal vez, el problema de computación distribuida atacado menos satisfactoriamente, que ha dado como resultado soluciones menos estándares. El tratamiento usual es identificar al sistema destino y emplear sus propias convenciones.

Además de todo ésto, es necesario contar con mecanismos adecuados de sicronización, transferencia de datos, y recuperación de errores, que son aspectos importantes relacionados con el envío y recepción de la información.

OBJETIVOS DE LOS PROTOCOLOS DE TRANSFERENCIA DE ARCHIVOS

Los objetivos que se persiguen al diseñar los protocolos de transferencia de archivos son diversos, y generalmente cada desarrollo posee características muy particulares. No obstante, es posible englobarlos dentro de uno general: poder transferir, manejar, y tener acceso a la información almacenada, o movida, entre sistemas. Este objetivo puede ser vista como un intento por extender a nivel de la red, los servicios de manejo de archivos de una computadora. Existen, sin embargo, distintos grados de alcance en el diseño y pueden ir desde el desarrollo de servidores de archivos (file servers) hasta un sistema de manejo de archivos distribuido. En el primer caso, lo que se persigue es proporcionar un medio de almacenamiento remoto y centralizado para estaciones de trabajo conectadas a él vía una red de comunicaciones, facilitando el compartimiento de datos entre las estaciones de

trabajo, y el apoyo a estaciones de trabajo no costosas que no poseen almacenamiento secundario o es limitado [5]. Además, apoya también:

a) Respaldo y recuperación automática de archivos.

b) Mobilidad del usuario.

c) Estaciones de trabajo sin disco de almacenamiento.

d) Desarrollo de otros servidores como, por ejemplo, de impresión (printer server), y de nombres (name server).

Como vemos, la diferencia principal entre un servidor de archivos y un protocolo de transferencia de archivos, es que en el primero existe un equipo dedicado exclusivamente al manejo de archivos, i.e. es centralizado, mientras que en el segundo éste no es restrictivo, es decir, tanto la información como el manejo es distribuido. Pero aún en el caso de los servidores de archivos existen grados, que van desde el bajo, en el que el servidor representa para el usuario un dispositivo de almacenamiento, un disco virtual; hasta el alto, en el que el servidor proporciona un sistema de archivos completo.

Servicios de FTP

No existe una definición estricta del tipo de servicios que debe prestar los FTP, ya que cada desarrollo impone condiciones particulares dependientes de su ambiente operativo. No obstante, existe un consenso acerca del conjunto de funciones que cualquier FTP puede tener:

a) Creación de archivos.

b) Borrador

c) Adición.

d) Renombrado.

e) Intercambio.

f) Impresión.

g) Copiado.

h) Control de acceso.

i) Manipulación de datos.

En pocas palabras, un FTP debe proporcionar los mecanismos adecuados para trabajar con los atributos, estructuras, y operaciones de archivos de los anfitriones de la red.

Modelo General de un Protocolo de Transferencia de Archivos

Actualmente, la gran mayoría de los FTPs en uso se apegan a un mismo modelo. Este modelo se puede considerar que está compuesto de dos elementos:

a) Puntos de servicio, que proporcionan acceso a los servicios locales de manejo de archivos.

b) Puntos de control, que monitorean y sincronizan la operación de los puntos de servicio.

En este modelo las operaciones de manejo de archivos, denominadas también transacciones, toman lugar entre un punto de control y uno, o dos, puntos de servicio. Sobre estos puntos se encuentran los procesos de control y de servicio, respectivamente. En el caso de una transferencia de archivos se involucran un controlador y dos servidores:

a) Un controlador en el punto de control, e.g. donde se localiza

el usuario.

b) Un servidor en el punto de servicio donde reside el archivo fuente: el productor.

c) Un servidor en el punto de servicio hacia donde se transferirá el archivo: el consumidor.

En este modelo existe también un diálogo entre los servidores. Un servidor opera solo sobre un archivo a la vez, y un controlador y un servidor cooperan sólo en una transacción a la vez. Sin embargo, pueden existir varias transacciones en paralelo involucrando los mismos puntos de control y de servicio, considerándose en este caso que son ejecutadas por distintas instancias de los procesos de control y de servicio.

Medio de Comunicación

El diálogo establecido entre el controlador y el (los) servidor(es) se lleva a cabo mediante el servicio de transporte, proporcionado localmente por la estación de transporte. Este servicio puede ser considerado como un mecanismo de comunicación entre procesos, el cual establece ligas entre los puertos asignados a éstos. Las estaciones de transporte proporcionan control de errores y de flujo sobre estas ligas. A la liga establecida entre un controlador y un servidor se le denomina Liga de Control, mientras que la establecida entre dos servidores, a través de la cual se transfieren los datos reales se llama Liga de Datos.

Modelo Simplificado de FTP

El modelo general propuesto anteriormente involucra tres ligas cuando se aplica a una transferencia de archivos. Se establece una liga de control entre el controlador y el servidor productor, una liga de control entre el controlador y el servidor consumidor, y una liga de datos entre los servidores productor y consumidor. Podemos derivar un modelo simplificado a partir de éste, en el cual se agrupen funciones de control y servicio en un mismo sistema. Este enfoque no impacta los mecanismos básicos del protocolo, sólo simplifica los problemas de sincronización y reinicio (en caso de que se rompa una liga) entre sitios distantes. De esta manera solo se envuelven dos puntos distantes y solo dos sistemas deben ser controlados (en vez de tres) durante una transacción.

Además, dado que los formatos de los comandos y respuestas intercambiadas sobre la liga de control son distintos de los formatos empleados para transferir datos sobre la liga de datos, es fácil distinguirlos, de tal manera que se pueden multiplexar datos y control sobre la misma liga, simplificándose de nuevo los problemas de sincronización y reinicio entre ambos extremos. Como puede observarse, el movimiento hacia el modelo general donde las funciones de transferencia de control y datos se encuentra bien aisladas, tendrá mayor impacto en la realización y estructura de los distintos módulos que comprenden los procesos controlador y servidor, que en las primitivas reales del protocolo.

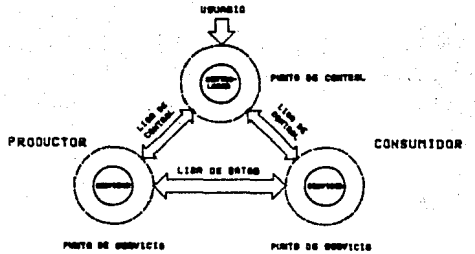


FIGURA 2.1

MODELO GENERAL DE UN FTP

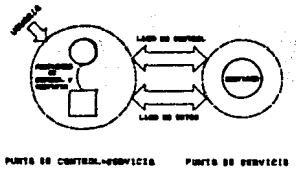


FIGURA 2.2

MODELO SIMPLIFICADO DE UN FTP

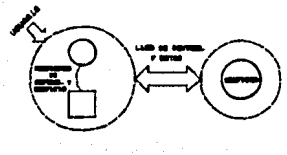


FIGURA 2.3

MODELO MINIMO DE UN FTP

EL SISTEMA DE TRANSFERENCIA DE ARCHIVOS DE Red-IIMAS

El protocolo de transferencia de Archivos, denominado Sistema de Transferencia de Archivos (STA) de Red-IIMAS contempla un modelo que se asemeja mucho al modelo mínimo de FTP descrito anteriormente. El STA se divide, igualmente, en dos tipos de procesos llamados de Usuario y Servidor, cuyas funciones son las de control y servicio para el Usuario, y exclusivamente servicio para el Servidor. La filosofía utilizada en la elección de este esquema es básicamente la misma que la del modelo, ésto es, reducir los problemas de sincronización y reinicio. Este sistema es el objetivo del presente trabajo y será analizado con gran detalle en los capítulos restantes.

BIBLIOGRAFIA

1. Linington,P.F., 'The Upper Layers of the ISO Reference Model', Pathways to the Information Society! Proceedings of the Sixth International Conference on Computer Communications, septiembre 1982, pags. 849-853.

2. Sproull, Robert F. & Cohen, Dan, 'High Level Protocols', Proceedings of the IEEE, vol.66 no.11, noviembre 1978, pags. 1371-1386.

3. Bartoli, Paul D., 'The Application and Presentation Layers of the Reference Model for Open Systems Interconnection', The Proceedings of INFOCOM 83, 1983, pags. 196-201.

4. Lewan, Douglas & Long, H.Garrett, 'The OSI File Service' Proceedings of the IEEE, vol. 71, No.12, diciembre 1983, pags. 1414-1419.

5. Svobodova, Liba, 'File Servers for Network-Based Distributed Systems' Computing Surveys, vol.16,No.4, diciembre 1984, pags.353-398.

6. Shoch, John, 'A File Transfer Protocol Using the BSP -- 4th edition' Xerox Palo Alto Research Center, 1979.

7. Gien, M., 'A File Transfer Protocol (FTP)' Computer Network Protocols, Symposium Universite Liege, 1978.

8. Postel, J.,'File Transfer Protocol' IEN 149, RFC 765, Arpanet, junio 1980.

9. Solling, N.R., 'The TFTP Protocol (Revision 2)', NWG RFC 783, Arpanet, junio 1981.

10. Horak, W., 'Concepts of the Document Interchange Protocol for the Telematic Services- CCITT Draft Recommendation S.a' Computer Networks, vol.8, 1984, pags. 175-185.

11. Pouzin, Louis & Zimmermann, Hubert, 'A Tutorial on Protocols' Proceedings of the IEEE, vol.66, No.11, noviembre 1978, pags. 1346-1370.

12. Butscher, B. Heinze, W., 'A File Transfer Protocol and Implementation' Hahn-Meitner-Institut fuer Kernforschung Berlin GmbH Department of Computer Science and Electronics, 1979.

13. Barber, D.L.A., 'Protocols for Intelligent Terminals' European Informatics Network, 1979.

14. Srinivasan,B. & Ananda A.L., 'A Simple File Server for Terminal Support Network Environment' Computer Networks and ISDN Systems, vol 11. No.5, 1986.

CAPITULO III

SISTEMA DE TRANSFERENCIA DE ARCHIVOS
PARTE I: LOS CONCEPTOS

El Sistema de Transferencia de Archivos STA es el primer protocolo de alto nivel desarrollado para Red-IIMAS. El STA implanta el protocolo de transferencia de archivos (FTP) de la red incluyéndo, además, un Subsistema de Correo Electrónico. Este último sistema será objeto de un capitulo posterior. Describiremos en lo que sigue cuales han sido los aspectos tomados en cuenta en el diseño y construcción del STA.

OBJETIVOS DEL STA

Ya que el STA es la realización de un protocolo de transferencia de archivos, sus objetivos generales son semejantes a cualquier FTP, y sobre ésto ya se ha comentado en el capitulo anterior. Sin embargo, debida a que el STA se encuentra inmerso en un ambiente de trabajo particular, mantiene un conjunto de características singulares que lo distinguen y que forman un segundo grupo de objetivos, representados por todas aquellas restricciones de diseño que debe satisfacer, y que impone su ambiente operativo exclusivo. Habiendo identificado los objetivos propios del STA, revisaremos en primer término su modelo para luego pasar a discutir todas las restricciones de diseño a las que hemos hecho mención.

MODELO DEL STA

El Sistema de Transferencia de Archivos se encuentra dentro de las Capa de Aplicaciones de la Arquitectura de Red-IIMAS. Por ésto, constituye un proceso de aplicación de la red que está orientado principalmente a ser utilizado por personas aunque seria factible solicitar sus servicios desde programas de usuario.

El STA está conformado por dos subsistemas: el Subsistema Usuario SU y el Subsistema Servidor SS. Estos subsistemas son procesos individuales que pueden radicar en anfitriones separados, o en uno mismo. Los servicios de transferencia de archivos que proporciona el STA se realizan mediante la cooperación de los subsistemas SU y SS, independientemente de su ubicación geográfica. El SU proporciona funciones de control que permiten mantener la sincronía de ambos procesos y dan acceso a las distintas operaciones que soporta el STA. Las funciones de servicio responden a las peticiones explícitas del SU para ejecutar conjuntamente la tarea deseada. La figura 3.1 muestra la esquematización de este modelo.

De acuerdo a este modelo, el usuario (proceso o persona) mantiene una comunicación directamente con el SU, y por su conducto pide la consecución de un servicio. Es decir, que aunque se tuviera acceso directo al Subsistema Servidor vía un VTP (Virtual Terminal Protocol), no sería de ninguna utilidad, ya que este subsistema carece de una interfaz que le permita relacionarse con un ente externo al STA.

Subsistema Usuario

Existe una gran simetria de funciones entre los subsistemas Usuario y Servidor concentrada básicamente en la sección de servicio. La gran diferencia entre ambos subsistemas radica, como sabemos, en que el SU cuenta además con una sección de control destinada a interactuar con el usuario. Estas operaciones de control definen una interfaz con el usuario, siendo la caracteristica principal que distingue al Subsistema Usuario del Servidor. Podemos decir entonces con gran exactitud, que al SU lo componen una Interfaz de Usuario y un SS.

Interfaz de Usuario

La Interfaz de Usuario IU proporciona todos los mecanismos necesarios para que un sistema pueda solicitar las funciones del STA bajo diferentes modos de operación. Estas modalidades le dan al sistema gran flexibilidad de manejo, de modo que usuarios con distintos grados de experiencia y necesidades puedan hacer uso del mismo fácil y rápidamente. Para poder lograr ésto existen tres modos básicos de acceso al STA que corresponden univocamente a otros tantos módulos de la IU; estos módulos son: Módulo Novato MN, Módulo Experto ME, y Módulo Batch MB.

Subsistema Servidor

El Subsistema Servidor SS es un ente pasivo (sin iniciativa) que trabaja en coordinación con el sistema de manejo de archivos de la computadora local para satisfacer todas las demandas de servicio

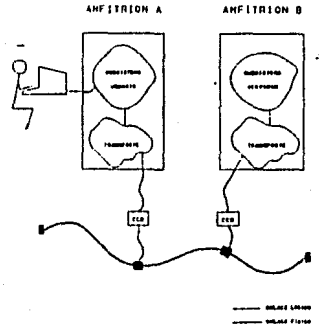


FIGURA 3.1

MODELO GENERAL DEL STA

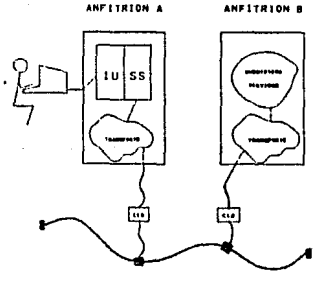


FIGURA 3.2

SU = IU + SS

solicitadas por el SU. Es a través del SS que un usuario tiene acceso a la información almacenada en una computadora remota y que es manejada sólo mediante los servicios de manejo de archivos SMA, controlados por el sistema operativo de la computadora anfitriona. De esta forma, el SS queda compuesta por los SMA, el Sistema de Manejo de archivos de la Red SMAR, y las Funciones de Control y Supervisión del Sistema Operativo FCSSO, indispensables en la coordinación de la ejecución de los programas.

El SMAR se encarga de mantener la comunicación entre el SU y el SS mediante un protocolo "peer-to-peer" y de realizar las funciones de transferencia propias del STA. Esta comunicación se inicia al momento de crear una liga SU-SS, y se mantiene apoyada por los servicios de la estación de transporte de la red: el Sistema de Control de Transmisión de Circuitos Virtuales SCT/CV. Por tanto, el SMAR se compone en realidad del Módulo de Servicios de Archivos MSA, y el Módulo de Comunicaciones MC.

Si trasladásemos al STA hacia el diagrama de la arquitectura de Red-IIMAS se vería de la siguiente forma:

Estructura Relacional del STA

Es necesario aclarar que en los diagramas anteriores no debe suponerse que las proporciones de los módulos reflejan su tamaño relativo en comparación de unos con otros, ni que sus fronteras marcan funciones de relación entre ellos. En estos diagramas se esquematiza exclusivamente el contenido y la cantidad de módulos constitutivos del STA. Un diagrama que muestre la relación entre

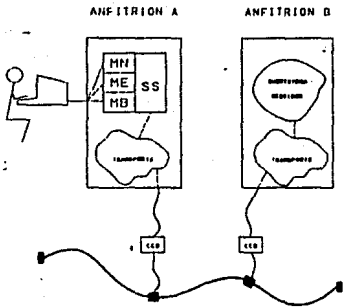


FIGURA 3.3
IU = MN + ME + MB

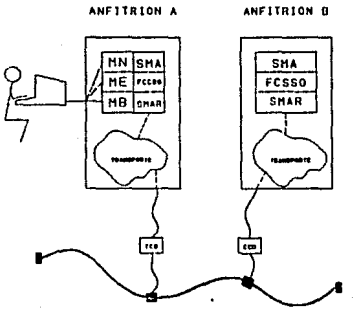


FIGURA 3.4
SS = SMA + FCCSO + SMAR

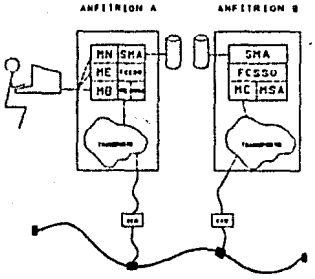


FIGURA 3.5
SMAR = MC + MSA

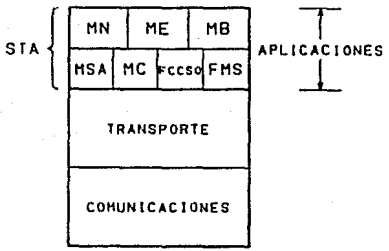


FIGURA 3.6
STA EN LA ARQUITECTURA DE Red-TIMAS

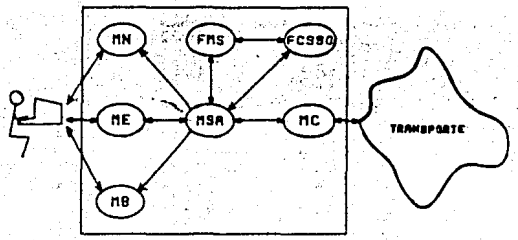
módulos se muestra en la figura 3.7.

Las relaciones que existen entre los módulos son representadas por flechas de doble punta que muestran el sentido del flujo de información. Si deseáramos utilizar el diagrama a bloques de los módulos del STA y quisiéramos respetar las funciones de relación que existen entre ellos, entonces se debería hacer que las fronteras comunes representaran la interrelación de los módulos, como se muestra en la figura 3.8. Este diagrama tiene la ventaja adicional de que nos facilita ubicarlo dentro de la arquitectura de la red, mostrándonos su relación con los demás sistemas de la misma.

Además, este diagrama permite hacer una abstracción importante que utilizaremos como base de referencia cuando entremos en la descripción detallada del STA.
Si incluimos dentro del MSA a los módulos SMA y FCCSO podremos hacer que el STA quede confinado a tres grandes partes principales: a) los tres módulos superiores que conforman a la Interfaz de Usuario IU, b) el Módulo de Servicios de Archivos MSA, y c) el Módulo de Comunicaciones MC que representa la Interfaz de Transporte IT.

ASPECTOS DE DISEÑO DEL STA

Además de respetar la división lógica tratada, el STA debe satisfacer una serie de requerimientos o restricciones de diseño, para que se pueda lograr una implantación eficiente. Estas restricciones están relacionadas con aspectos de tipo estructural



STA

FIGURA 3.7

DIAGRAMA RELACIONAL DE LOS MODULOS DEL STA

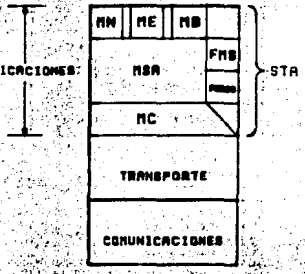


FIGURA 3.8

DIAGRAMA A BLOQUES RELACIONAL DEL STA EN Red-IIMAS

y funcional. Las restricciones estructurales son todas aquellas impuestas por el ambiente operativo y de desarrollo, es decir, provienen de las limitaciones en los recursos de cómputo. Las restricciones funcionales son todas aquellas características deseables que debieran poseer todo buen sistema, para que sea flexible y amistoso con el usuario, sin que pierda eficacia.

RESTRICCIONES FUNCIONALES

Ya que la intención del STA es satisfacer diversas aplicaciones de transferencia de archivos, debe poseer los mecanismos que le permitan acomodarse fácilmente a las diferentes necesidades. Esto es, debe ser posible acceder a todas las funciones del STA desde distintos ambientes de operación sin que ésto conduzca hacia una degradación seria en las capacidades y eficiencia del sistema. En el caso del STA se han considerado básicamente tres tipos de escenarios:

Modo Novato

Para poder alentar el uso del STA es requisito que presente sencillez en su manejo. Esto significa que pueda ser utilizado desde un principio a casi toda su capacidad. Un requerimiento aún más fuerte, es que no se exija del usuario una gran experiencia en computación, sino que casi cualquier persona que desee utilizarlo pueda hacerlo. Esto puede satisfacerse en gran medida si se esconde la mayor cantidad posible de aspectos técnicos y si se cuentan con los medios adecuados de interacción hombre-máquina que la faciliten. Si además de ésto es posible otorgar ayuda sobre

casi cualquier aspecto que pudiera resultar oscuro y esta ayuda es lo más expedita y clara posible, entonces podemos cumplir satisfactoriamente nuestro objetivo primordial: sencillez.

El STA cuenta con un modo de operación, denominado Modo Novato, que pretende cumplir con todas las expectativas planteadas, mediante un uso exhaustivo de menues que proporcionan un ambiente interactivo en el cual el usuario es casi literalmente 'llevado de la mano' a realizar cualquiera de las funciones que proporciona el sistema. Asimismo, el usuario puede obtener ayuda sobre casi cualquier tópico que le pueda interesar acerca del mismo, y en casi cualquier instante. Se ha cuidado también que la forma y el contenido de los menues sea atractiva y den una clara idea del tipo de datos que el usuario debe proporcionar al sistema cuando éste lo solicite y el tipo de operaciones validas accesibles desde un menú en particular.

Modo Experto

Pero, también hay que tomar en consideración al tipo de usuario que ya posee una cierta formación en computación y/o que ha tenido la oportunidad de trabajar con el STA en modo novato y le gustaría realizar sus tareas con mayor rapidez, evitando tener que entablar un diálogo más prolongado con el sistema. Es evidente que para ésto el usuario requiere tener una idea precisa de la mecánica de las funciones y sus datos para poder lograr sus objetivos. La idea básica es entonces, contar con un lenguaje claro y conciso que permita comunicarle al STA nuestros deseos sin ambiguedad. La forma en la cual el STA cumple este objetivo es mediante la

utilización de un lenguaje de comandos diseñado exprofeso y que contempla igualmente todas las variantes posibles de ejecución que se tienen desde los menues. Pero, aunque la utilización del Modo Experto supone un buen conocimiento de la sintaxis del lenguaje, y de la semántica de las oraciones válidas que puede construir, a veces se llegará a necesitar cierta ayuda sobre el lenguaje y los tipos de datos que puede manipular. Por ésto, también el modo experto prevee esta eventualidad y posee su propio marco de ayuda.

Modo Batch

En muchas otras ocasiones, y dependiendo del tipo de usuario, puede ser que nuestra preocupación o responsabilidad sea realizar varias veces el mismo tipo de operación o conjunto de operaciones para el mismo tipo o conjunto de datos. Sería de gran utilidad, entonces, disponer de un mecanismo que permitiese proporcionarle al STA una serie de peticiones (comandos) para ejecutar una tarea y que se encargara de cumplirlas en un cierto orden. Esto tendría la ventaja adicional de que podría asignarse a una persona con poco conocimiento del sistema para que realizara esta tarea. Lo único que se requeriría, en todo caso, sería saber como activar al STA y darle el conjunto de mandatos.

El STA resuelve este problema a través del modo batch de operación, mediante el cual es posible pasarle una lista de comandos almacenada en un archivo, para su ejecución.

RESTRICCIONES ESTRUCTURALES

El STA se desarrolló en una computadora PDP-11/34 con sistema

operativo RSX-11M, siendo este ambiente una de las dos fuentes de las restricciones estructurales más importantes. La otra es el ambiente de red para el cual fue diseñado.

Restricciones debido al Sistema Operativo

El sistema operativo RSX-11M impone una limitación muy fuerte a todas las tareas de usuario. Siendo la PDP-11/34 una máquina con una longitud de palabra de 16 bits, las tareas solo pueden tener acceso a un espacio de direcciones de 64Kb. Esto hace que el STA pueda ocupar como máximo una área de memoria no mayor a los 64Kb, aunque en realidad es aún menor, como veremos más delante. Es importante pues, eficientizar al máximo el código generado observando siempre buenas técnicas de programación de manera que podamos obtener un buen tiempo de respuesta del sistema y un código compacto.

Otro aspecto de igual importancia es que no podemos realizar ninguna operación sobre los archivos que no sea soportada directamente por el sistema de archivos nativo. En RSX-11M este sistema está representado por los servicios de control de archivos FCS (File Control Services), siendo quién determina las posibilidades. FCS permite ejecutar operaciones de entrada/salida orientadas a registros y orientadas a bloques, y funciones adicionales de control de archivos. Para invocar esta funciones el usuario emite macro-llamadas especificando las operaciones deseadas. Las macros de FCS se llaman al tiempo de ensamble para generar el código para las funciones y operaciones especificadas. FCS es básicamente un conjunto de rutinas que se ligan con el

programa de usuario al momento de construcción de la tarea (task-build) desde una biblioteca residente del sistema. Estas rutinas proporcionan una interfaz entre el usuario y el sistema de archivos, y permiten leer y escribir archivos en dispositivos estructurados como archivos y procesar archivos en términos de registros lógicos.

El hecho de que las rutinas se liguen al programa de usuario hace que se reduzca aún más el espacio disponible para la tarea de usuario, ya de por sí limitada.

Restricciones debidas a Red-IIMAS

No hay que olvidar que el STA se encuentra inserso en un ambiente de red y que debe respetar las políticas de diseño previstas para ella. El STA posee dos interfaces principales a través de las cuales interactúa con sus vecinos. La interfaz visible del STA es la de usuario, siendo ésta el medio de acceso a las funciones del STA, como ya quedó explicado. La segunda interfaz se encuentra en la frontera con la capa inferior de la arquitectura de Red-IIMAS; la capa de transporte. Es esta interfaz la que define el conjunto de servicios de transporte disponibles en Red-IIMAS, y la forma en que pueden ser invocados. De manera parecida a lo que ocurre con FCS, las operaciones de transporte son básicamente un conjunto de rutinas que se ligan al programa de aplicación al momento de construcción de la tarea, desde una biblioteca residente del sistema. En este momento las rutinas se encuentran localizadas en un único archivo, lo que no facilita una mejor utilización de la memoria. Esta biblioteca ocupa un espacio aproximado de 4Kw, que

deben ser restados del total disponible para el STA.

Algo que también es necesario destacar es, que los protocolos de transferencia están fuertemente influídos por el tipo de servicio de transporte de que se disponga y por la manera en que éste opera.

ESTRUCTURA DE DESARROLLO DEL STA

Ahora veremos como es que fue construido el STA a partir de su modelo funcional y de las restricciones de diseño descritas. Nos limitaremos a explicar la estructura global del sistema, para dejar sus detalles internos para el siguiente capítulo.

ESTRUCTURA DEL STA

El mecanismo primario de direccionamiento de las PDP-11 es la palabra de 16 bits. El máximo espacio de direcciones físicas que puede referenciar la PDP-11 es función de esta palabra. Ya que la computadora es una máquina direccionable por bytes, la palabra de 16 bits le permite direccionar en cualquier momento hasta 65,535 bytes (64Kb) del espacio de direcciones físicas. Se denomina espacio de direcciones virtuales al tamaño del espacio de direcciones que puede referenciar una máquina en un momento dado. La PDP-11/34 es un sistema mapeado, lo cual significa que puede poseer una mayor capacidad de memoria física y, por lo tanto, no coincidir los espacios de direcciones físicas y virtuales. Cuando se ligan los módulos objetos relocalizables de una tarea que va a correr en un sistema mapeado, se le asignan direcciones de 16 bits a la imagen de la tarea y, por ésto mismo, no puede direccionar

más allá de 64 Kb (32Kw). La figura 3.10 muestra la relación entre la memoria física y el espacio de direcciones virtuales en un sistema mapeado.

El STA es un sistema complejo que ocupa una gran cantidad de memoria y que, por lo tanto, es necesario utilizar mecanismos que permitan reducir su espacio de direcciones físicas y virtuales para que pueda ser alojado en un espacio máximo de 32 Kw. Sin embargo, este espacio debe ser compartido también por la biblioteca de funciones de transporte que se ligan a todos los procesos de aplicación de la red. Esta biblioteca ocupa un espacio de 4Kw por lo que el espacio máximo disponible para el STA se reduce a 28Kw. Dentro de estos 28Kw debe acomodarse, además, la biblioteca de funciones del lenguaje de programación que se emplee. El STA fue desarrollado en su mayor parte en el lenguaje C, que sobreimpone un costo adicional aproximado de 8 Kw. El resto del sistema se desarrolló en el ensamblador nativo de la PDP-11: MACRO-11. Aún este lenguaje añade requerimientos de almacenamiento que deben ser obtenidos del total restante, pero consideraremos que ésto no es representativo. Así, después de revisar el costo total de memoria indispensable de acuerdo al ambiente operativo previsto, observamos que disponemos de un espacio de direcciones virtuales de 20 Kw, aproximadamente. Este espacio es sumamente reducido y es por que ésto fue necesario utilizar una metodología de desarrollo que redujese los requerimientos de almacenamiento del STA.

Una estrategia comúnmente utilizada en el tratamiento de esta problemática es construir la tarea en base a "overlays", que

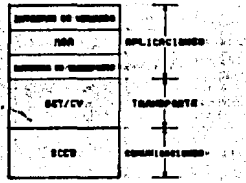


FIGURA 2.9

DIAGRAMA LOGICO DEL STA EN RSX-11M

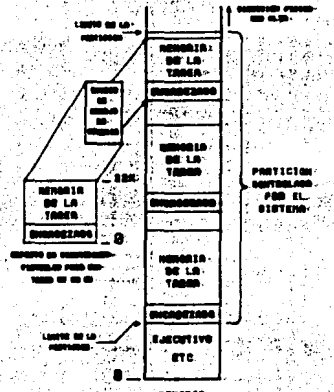


FIGURA 2.10

MAPA DE MEMORIA DE RSX-11M

permite disminuir los requerimientos de espacio de direcciones virtuales y/o memoria.

El STA posee una estructura de overlays tipo árbol, de acuerdo a la filosofía de la computadora, formada por un conjunto de segmentos consistentes de:

a) Un único segmento raíz, que siempre reside en memoria y,

b) Varios segmentos de overlay, que residen en disco y comparten entre ellos el espacio de direcciones virtuales y la memoria física.

Los segmentos pueden consistir de uno o varios módulos objeto, y para que puedan trasloparse entre sí deben ser lógicamente independientes, es decir, los componentes de un segmento no pueden referenciar los componentes de otro segmento con el cual comparte un mismo espacio de direcciones virtuales. Además de esta independencia lógica, es necesario observar detalladamente el flujo general de control dentro del STA ya que éste también influye en el diseño de los segmentos de overlay.

Existen varios tipos de overlay que pueden colocarse en una cierta posición dentro de la estructura global. Si se emplean overlays que residen en disco se puede ahorrar espacio físico, pero se introduce un 'overhead' al tener que cargarlos desde disco cada vez que son requeridos, y no se encuentran residentes en memoria en ese momento. Por otro lado, los overlays que residen en memoria se cargan de disco sólo la primera vez que son referenciados mediante remapeo. El costo en que se incurre bajo la adopción de esta alternativa es que no se ahorra espacio físico. Otra pequeña desventaja de utilizar cualquier tipo de overlays es el almacenamiento adicional necesario para las estructuras de control

empleados en el manejo de los mismos.

El STA fue construido en base a overlays residentes en disco, dado que se trata de un sistema interactivo en el que el retardo por acceso a un disco es despreciable. Esta opción también fue la adecuada debido a la buena estructuración del STA, que permitió lograr un muy buen tiempo de respuesta.

El Arbol de Overlay

El orden de los segmentos de overlay dentro del espacio de direcciones virtuales de una tarea se puede representar esquemáticamente como una estructura en forma de árbol. Cada rama del árbol representa a un segmento. Las ramas paralelas denotan segmentos que se traslapan unos a otros y poseen, por lo tanto, las mismas direcciones virtuales; estos segmentos deber ser lógicamente independientes. Las ramas conectadas de punta a punta representan segmentos que no comparten espacio de direcciones virtuales entre sí; estos segmentos no necesitan ser lógicamente independientes.

Un árbol posee tantas trayecorias como hojas tiene. Es importante saber las propiedades del árbol y sus trayectorias para entender los mecanismos de carga de overlays y la resolución de los símbolos globales.

Mecanismo de Carga

Los módulos pueden llamar a otros módulos que existen sobre una misma trayectoria. Cuando un módulo en un segmento overlay llama a

un módulo en otro segmento, el segmento llamado debe estar en memoria y mapeado, o debe ser traído a memoria.

Resolución de Símbolos Globales

El alcance (scope) de los símbolos globales es alterado por la estructura de overlay. En una tarea de un solo segmento, cualquier módulo puede referir a cualquier definición global. En cambio en una tarea multisegmento un módulo solo puede referir a un símbolo global que se encuentre definido sobre una trayectoria que pasa a través del segmento llamado. Todo ésto posee implicaciones fuertes en el tratamiento de símbolos múltiplemente definidos y sobre el procedimiento de resolución de símbolos globales que provienen de bibliotecas específicas y de "default".

Existen todavía consideraciones importantes que deben ser tomadas muy en cuenta cuando se está diseñando una tarea estructurada en overlays, pero con el afán de simplificar esta presentación preferimos omitirlas.

Todas estas ideas fueron aplicadas en el diseño de la estructura del STA, la cual se muestra en las figuras 3.11 y 3.12.

Como puede destacarse en las figuras 3.11a y b, la verdadera conformación del STA consiste de dos estructuras arbóreas denominadas co-árboles. Los árboles son idénticos excepto porque el segmento raíz del árbol principal es cargado por el Ejecutivo cuando se activa la tarea, mientras que los segmentos dentro de cada coárbol se cargan a través de llamadas específicas a las rutinas que manejan los overlays al momento de ejecución. La principal propiedad de una estructura que contiene más de un árbol

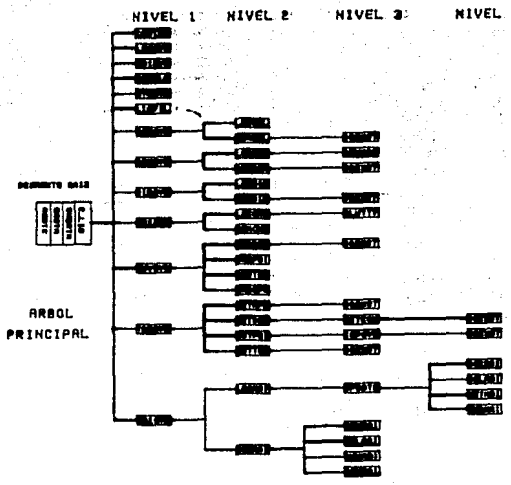


FIGURA 3.11A
ARBOL DE OVERLAY DEL STA

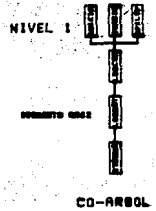


FIGURA 3.11B
CO-ARBOL DE OVERLAY DEL STA

es que no se comparte el almacenamiento entre los árboles. Cualquier segmento en un árbol puede ser referido desde otro árbol sin desplazar segmentos del árbol que llama. Por ejemplo, las rutinas que se llaman desde varios segmentos del árbol principal se pueden traslapar entre sí en un coárbol.

Módulo	Significado
ROOTC	Programa principal y funciones de alto nivel
ROOTM	Primitivas en Macro-11 de manejo de pantallas y overlays
ENDATM	Interfaz de Transporte
C_LIB	Biliotaca de funciones de C
MENUSC	Sistema de menues
RTCOMC	Parser de Comandos
LEEBAT	Procesador de Batch
LEECOM	Lector de Comandos
INICIA	Fase de Inicialización del Modo Experto
BVARID	Ayuda,Hola,Login,Lista,Adios,Chisme,Directorio,Conecta
BFILTP	Trae,Envia,Borra,Renombra
CONTRA	Protocolo Conecta
LOGOVR	Protocolo Login
ADIOVR	Protocolo Adios
DOHOLA	Protocolo Hola
AYUOVR	Ayudas
LISFIL	Función Lista
BOROVR	Programa principal de función Borra
LOCDEL	Borrado Local
REMDEL	Protocolo Borra
DOEXST	Protocolo Existe
DIROVR	Programa principal de función Directorio
LOCDIR	Directorio Local
REMDIR	Protocolo Directorio
MAIOVR	Programa principal del Subsistema de Correo
LOCMAI	Correo Local
UPDATE	Mantenimiento de buzones locales
BCKMAI	Carta anterior
DELMAI	Borrar carta actual
NXTMAI	Carta siguiente
SNDMAI	Enviar carta
REMMAI	Programa principal correo remoto
RBKMAI	Protocolo Carta anterior
RDLMAI	Protocolo Borrar carta
RNXMAI	Protocolo Carta siguiente
RSNMAI	Protocolo Enviar carta
RENOVR	Programa principal función Renombra
LOCREN	Renombrado Local
RENMAC	Primitivas en Macro-11 de función renombra
REMREN	Protocolo Renombra
TRAOVR	Programa principal función Trae
GETAPN	Protocolo Trae para añadir

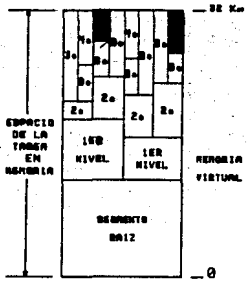


FIGURA 3.12

DISTRIBUCION DE LOS MODULOS EN MEMORIA

GETDSK	Protocolo Trae a disco
GETCON	Protocolo Trae a disco archivo contiguo
GETPRI	Protocolo Trae para imprimir
IMPOVR	Primitivas de impresión
GETTER	Protocolo Trae para desplegar
CHIOVR	Programa principal de función Chisme
LOCCHS	Chisme Local
ALUTTY	Primitivas de manejo del enlace de terminales
REMCHS	Protocolo Chisme
ENVOVR	Programa Principal de función Envía
SNDDSK	Protocolo Envía a disco
SNDPRI	Protocolo Envía a impresora
SNDTER	Protocolo Envía a terminal
SNDAPN	Protocolo Envía para agregar

Figura 3.13 Lista de Overlays del STA

BIBLIOGRAFIA

1. RSX-11M/M-PLUS Task Builder Manual, Order No. AA-H266A-TC Digital Equipment Corporation, Maynard, Massachusettes, 1979.

2. IAS/RSX-11 I/O Operations Reference Manual, Order No. AA-2515D-TC, Digital Equipment Corporation, Maynard, Massachusettes, 1979.

3. RSX-11 Utilities Manual, Order No. AA-H268A-TC, Digital Equipment Corporation, Maynard, Massachusettes, 1979.

4. RSX-11M/M-PLUS Executive Reference Manual, Order No. AA-H265A-TC, Digital Equipment Corporation, Maynard, Massachusettes, 1979.

5. RMS-11 User's Guide, Order No. AA-D538A-TC, Digital Equipment Corporation, Maynard, Massachusettes, 1979.

6. Introduction to RMS-11, Order No. AA-0001A-TC, Digital Equipment Corporation, Maynard, Massachusettes, 1979.

7. RSX-11M RMS-11 V1.5 Utilities User's Guide Updated for V3.1, Order No. AA-D083A-TC, Digital Equipment Corporation, Maynard, Massachusettes, 1979.

8. RSX-11M RMS-11K Installation Guide, Order No. AA-5296B-TC, Digital Equipment Corporation, Maynard, Massachusettes, 1979.

9. RMS-11 Installation Guide AA-H235A-TC, Digital Equipment Corporation, Maynard, Massachusettes, 1979.

10. RMS-11 MACRO-11 Reference Manual, Order No. AA-H683A-TC, Digital Equipment Corporation, Maynard, Massachusettes, 1979.

CAPITULO IV

SISTEMA DE TRANSFERENCIA
DE ARCHIVOS
PARTE II: LOS DETALLES
SECCION A: LAS INTERFACES

En este capitulo analizaremos la conformación interna del STA tomando como modelo de referencia la esquematización de la figura 3.9. Empezaremos con la Interfaz de Usuario IU y terminaremos el capitulo tratando la Interfaz de Transporte IT.

INTERFAZ DE USUARIO

La Interfaz de Usuario IU es el medio mediante el cual un usuario puede establecer una sesión interactiva o no-interactiva con el STA, para realizar tareas de transferencia de archivos entre anfitriones enlazados a Red-IIMAS. La IU proporciona el acceso a las funciones del STA desde diversos ambientes de operación que le dan al usuario una gran flexibilidad en el manejo del sistema, cubriendo distintos tipos de aplicaciones y niveles de experiencia del usuario. La IU soporta tres modos básicos de operación: Modo Novato, Modo Experto, y Modo Batch. El STA le permite elegir al usuario el modo de operación bajo el cual desea trabajar, al momento de activar el sistema. El usuario puede elegir entre cualquiera de los tres modos de la IU al inicio de una sesión, pero no puede conmutar entre ellos a mitad de la sesión. Si se

quisiera hacer ésto se debería finalizar la sesión en tránsito e iniciar otra bajo el nuevo modo de operación deseado.
Ya que los módulos que implantan la IU deben referir a todos los segmentos de overlay de los servicios de transferencia, es necesario que se encuentren en el segmento raíz del árbol de overlays.

MODO NOVATO DE OPERACION

Este modo permite acceder a los servicios del STA a través de una sesión interactiva usuario-STA bajo control de un sistema de menues que muestran en todo momento, los campos de información que debe llenar el usuario para poder ejecutar la función actual y/o las operaciones válidas alternativas.

Manejo de Terminales

El diseño del modo novato de operación plantea un problema muy interesante: Cómo lograr que el esquema de menues trabaje en diferentes terminales? La respuesta a esta interrogante tiene distintos grados de exigencia que pueden variar desde la realización de versiones individuales del STA para cada tipo de terminal prevista, hasta un diseño totalmente general donde se pueda soportar cualquier tipo de terminal. Obviamente que la opción más atractiva es la última, pero es igualmente la más complicada de obtener. Por ello es necesario elegir una marca adecuada dentro de esta acotación. La solución al límite superior sería el disponer de un protocolo de terminales virtuales VTP, cuyo objetivo es, precisamente, resolver los conflictos de manejo

y representación de datos en terminales con acceso a la red. Desafortunadamente, no se cuenta con este servicio actualmente en Red-IIMAS, de modo que fue conveniente idear un mecanismo adecuado y no costoso que pudiera ser incluido en el diseño del STA y que permitiese utilizar un conjunto básico de terminales. Como no era deseable incrementar demasiado la complejidad y requerimientos del Modo Novato, la Interfaz de Usuario del STA empleó un método sencillo para resolver el problema aludido: El usuario podía seleccionar el tipo de terminal que más se asemejara a la que estaba empleando. El criterio de definición de este conjunto de terminales fue la población principal de terminales que se encontraban en el Instituto. Este conjunto está conformado por tres tipos de terminales:

a) Heathkit

b) Datamedia

c) Tipo teletipo

Los dos primeros tipos poseen capacidades de graficación, mientras que el último ha sido incluído pensando en terminales que no cuentan con ninguna capacidad gráfica, de modo que si un usuario no está sentado frente a una terminal con características similares a los tipos a y b, puede definir su terminal como tipo c, y trabajar sin ninguna dificultad.

Aún cuando es un número reducido de terminales, el diseño debe ser flexible para que facilite la inclusión/exclusión de tipos de terminales. Es claro que, dependiendo del número de terminales distintas que se espera soportar, un cierto método de manejo puede dejar de ser el adecuado. Teniendo muy presente esta consideración

para el método de manejo de terminales, se optó por el siguiente diseño: Mantener una interfaz estándar de manipulación de pantallas, independiente del tipo de terminal en uso, manteniendo en disco tablas de control de graficación para cada tipo de terminal.

Manejo de Pantallas

La satisfacción del enfoque anterior permite que se pueda cargar en cualquier momento la Tabla de Definición de la Terminal TDT que se desea emular. Esto hace factible que se puedan guardar un buen número de TDTs en disco, sin mucha dificultad. Este enfoque es bueno siempre y cuando no crezca demasiado la población de clases, ya que haría impráctico el mantenimiento de estos archivos. Una alternativa podría ser mantener un solo archivo que contuviera todas las TDT, pero ésto añadiría la necesidad de desarrollar las herramientas indispensables para crear, mantener, y manipular esta información. Es fácil darse cuenta que entre más crece la necesidad de soportar un mayor número de tipos de terminales más complejo se vuelve el diseño y más se tiende hacia un VTP.

Las tablas de definición TDT son archivos objeto (binarios) mantenidos en disco que solo contienen información acerca de los caracteres de graficación que son de interés para la IU. Estas tablas poseen la estructura mostrada en la figura 4.1.

Si alguna secuencia de control no tiene equivalencia en algún tipo de terminal se sustituye por un caracter, o cadena de caracteres, que no afecten la pantalla, es decir, son 'nop's". Actualmente sólo se cuenta con la TDT para los terminales Heathkit.

Funciones de Manejo de Pantallas

Para lograr la independencia sobre el tipo de terminal es necesario contar con funciones de alto nivel que permitan utilizar un número variable de argumentos que indiquen el tipo de control gráfico deseado. Es, en cierta forma, muy similar a una función 'printf' de C, o 'write' de Pascal, pero muy enfocada al manejo de pantallas y, en particular a las del STA. A pesar de esta última intención, la función diseñada mantiene un buen grado de generalidad que, con el afán de reducir el código total del sistema, le permite ser utilizada aún en el Modo Experto de operación, del cual trataremos más adelante.
La función que realiza el manejo de pantallas posee la siguiente sintaxis:

```
video(mod,fmt,arg1,arg2,...,arg15)
donde:
     mod es tipo entero
     fmt es un apuntador a una cadena
     arg1,...arg15 son tipo apuntador a una cadena
```

Mod es una variable que nos permite elegir la modalidad de operación de la función. Sólo puede adquirir 2 valores: ModExp y ModMen; modo experto y modo menú, respectivamente. La elección de ModMen indicará al sistema que debe utilizar las capacidades de graficación de la terminal en uso.

Fmt nos permite pasar una cadena de caracteres a la función para

indicarle el tipo de manejo de pantalla que se desea efectuar. En este sentido, su objetivo es el mismo que el campo de formato de las instrucciones tipo write de los lenguajes de programación de alto nivel.

Arg1,...,arg15 son los argumentos sobre los cuales se operará, y son interpretados por posición, es decir, no existe validación de los mismos. Dado que estos argumentos son del tipo apuntador a strings, es necesario efectuar las conversiones adecuadas hacia el tipo de argumento esperado.

Los formatos que se manejan en fmt son los siguientes:

%a Pregunta. Escribe en la pantalla la cadena "ES/NO?" y regresa TRUE o FALSE dependiendo de si el usuario contestó S o cualquier otra cosa, respectivamente. No posee argumentos.

%b Borra la línea sobre la que se encuentra el cursor. No posee argumentos.

%c Suena la campana. Sin argumentos.

%d Decimal. Escribe la cadena ascii que representa al valor binario pasado como argumento.

%f Apaga el cursor.

%g Pone a la terminal en modo gráfico. Sin argumentos.

%h Saca a la terminal del modo gráfico. Sin argumentos.

%i Pone a la terminal en modo de video inverso. Sin argumentos.

%j Saca a la terminal del modo de video inverso. Sin argumentos.

%l Escribe l (ele) blancos. l (ele) se pasa como argumento.

%mn Borra una ventana de n renglones de longitud m, a partir de

donde se encuentre el cursor. Posee dos argumentos: m es el ancho de la ventana, n es el número de renglones.

%n Línea nueva. Imprime un LF. La n puede ser precedida por un entero que indique el número de LF's deseados. Se pueden especificar cero o un argumento que indicará el número de LF's.

%o Prende el cursor. Sin argumentos.

%p Borra la ventana de trabajo (renglones 8 - 22). Sin argumentos.

%r Regreso de carro. Imprime un CR. La r puede ser precedida por un entero que indique el número de CR's deseados. Se pueden especificar cero o un argumento que indica el número de CR's.

%s Escribe una cadena de caracteres. El apuntador al inicio de la cadena se pasa como argumento. 0/1 argumentos.

%t Tabulador. Imprime un Tab. La t puede ser precedida por un entero que indique el número de Tab's deseados. Se pueden especificar cero o un argumento.

%u Habilita la línea 25 de la pantalla.

%v Borra las líneas 24 y 25 de la pantalla.

%w Escribe la cadena pasada como argumento en la línea 24 de la pantalla. Pide oprimir la tecla de return para continuar. Por último borra las líneas 24 y 25.

%xy Mueve el cursor a la posición (x,y) de la pantalla. Las coordenadas se pasan como argumentos. Dos argumentos: 'x' es el renglón, 'y' la columna.

%z Borra la pantalla. Sin argumentos.

%B Borra la pantalla partiendo de los extremos superior e

inferior hacia el centro. Sin argumentos.

%I Escribe una cadena de caracteres en video inverso a partir de la posición actual de cursor, si es que estamos en ModMen. Un argumento: apuntador a la cadena.

%N Escribe una cadena de caracteres en video normal a partir de la posición actual del cursor, si es que estamos manejando menues.

Una característica muy importante de esta función es que las operaciones primitivas que interactúan con las terminales emiten peticiones directamente al 'handler' de terminales, lográndose con ésto el mejor tiempo de respuesta posible y el control total de las mismas, evitándose, además, los problemas de entrada/salida propios del lenguaje de alto nivel o del código generado por su compilador. Para poder lograr ésto fue necesario escribir código en MACRO-11 para las primitivas de manejo de terminales e interfazarlo con C.

Manejo de Menues

El principal atractivo del modo novato es la sencillez de uso de las funciones del STA proporcionada por su esquema de menues que facilitan el diálogo usuario-sistema. Estos menues intentan que la distribución de la información en la pantalla sea legible haciendo una clara división de las áreas de datos y de funciones, y empleando las capacidades de intensidad de imágen de las terminales para resaltar información importante.

Una pantalla del STA está dividida básicamente en tres ventanas

principales:

a) Encabezado: Esta ventana se ubica en la parte superior de la pantalla abarcando los renglones 7,8 y 9 y se utiliza para indicar el tipo de operación actualmente en ejecución. El contenido de la ventana se actualiza conforme cambiamos de menú.

b) Diálogo: Corresponde a la parte central y mayor de la pantalla, abarcando los renglones 10 al 22, y en ella se efectúa el diálogo entre usuario y STA. También se utiliza para mostrar cierto tipo de información acerca de la dinámica de una operación y otros mensajes.

c) Mensajes: Es la parte inferior de la pantalla ocupada por los renglones 24 y 25, dedicada al despliegue de mensajes concernientes al estado de finalización de una operación, las funciones válidas para esa pantalla, y otros mensajes del sistema.

Un aspecto importante de este esquema de menues es que se ha cuidado que la ejecución de todas las funciones del STA se realicen con un mínimo de teclas, mediante el uso extensivo de las capacidades de manejo de teclas de función de las terminales.
La figura 4.2 muestra la esquematización de las pantallas de menues.

Arbol de Menues

El manejo de todas las pantallas de menues de la IU puede ser visto como una estructura arbórea, donde los nodos localizados al mismo nivel en la jerarquía son mutuamente exclusivos, es decir, solo son asequibles desde un nivel superior por el cual se pueda

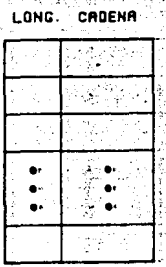


FIGURA 4.1

TABLA DE DEFINICION DE TERMINALES

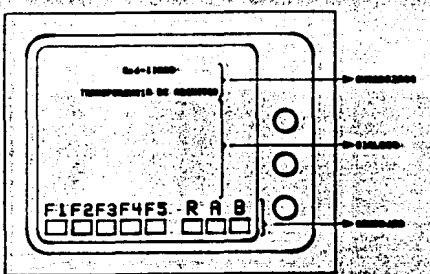


FIGURA 4.2

APARIENCIA DE UNA PANTALLA DEL STA

establecer una trayectoria descendente hasta él. Asimismo, cada nodo de la jerarquía representa un menú diferente para el usuario. La figura 4.3 muestra esta estructura.

Menues y Funciones

El STA cuenta con tres menues principales a partir de los cuales son accesibles todas las funciones del sistema. Cuando un usuario selecciona alguna de las funciones mostradas en cualquiera de los menues, se pasa al conjunto de menues correspondientes a la función elegida, pudiendo ser de varios niveles.

En este apartado solo daremos un resumen de los menues y funciones que posee el STA, ya que el capitulo 7 se dedica al manual del usuario y en él se detallan cada uno de los menues y sus funciones, así como al Lenguaje de Comandos del Modo Experto.

Primer Menú Principal

Función BORR Borrar archivos
Función DIRE Directorio de archivos
Función HOLA Verifica versión del sistema y línea viva
Función TRAE Trae archivos remotos
Función CHIS Comunicación en línea entre terminales

Segundo Menú Principal

Función LOGI Entrar en sesión en anfitrión remoto
Función MAIL Subsistema de correo electrónico
Función RENM Renombrar archivos
Función MNDA Enviar archivos hacia anfitrión remoto
Función CONE Conexión con el Sistema Servidor del STA

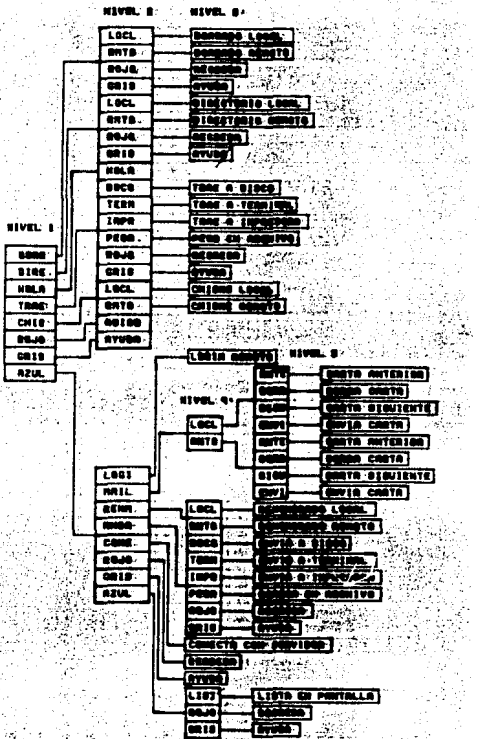


FIGURA 4.3
ESTRUCTURA ARBOREA DEL ESQUEMA DE MENUES

Tercer Menu Principal

Función LIST Listar un archivo en la terminal del usuario

MODO EXPERTO DE OPERACION

Este modo permite acceder a los servicios del STA en una sesión interactiva usuario-STA, mediante la utilización de un lenguaje de comandos. Este modo de operación fue ideado para todos aquellos usuarios con experiencia en sistemas de cómputo y/o el STA. Su intencion es reducir el diálogo necesario para realizar las tareas de transferencia en Red-IIMAS.

Lenguaje de Comandos

El modo experto se soporta enteramente en la definición de un lenguaje de comandos diseñado exprofeso. El diseño preveía el desarrollo de una gramática clara y compacta que no representara dificultad al usuario. Existían básicamente dos alternativas de diseño:

a) Diseñar una gramática nueva.

b) Apegarse a una existente.

Bajo cualquiera de las dos alternativas el trabajo de desarrollo es prácticamente el mismo, con la salvedad de que en el primer caso se añade la fase de diseño. En cambio, si elegimos apegarnos a una gramática existente, simplificamos la fase de diseño y podemos ganar en el arranque del uso del sistema debido a la posible experiencia previa con el lenguaje al cual se asemeja. La primer elección pasaría a primer plano siempre y cuando fuese un

lenguaje mucho más sencillo que el modelo. Para el caso del STA esta última figura es difícil de avaluar debido a la sencillez de las funciones que debe expresar. Por todo ésto, se decidió apegarse a la sintaxis de un lenguaje de comandos que se comportara de manera similar a nuestro caso. Un factor determinante en esta decisión fue la experiencia propia con este tipo de lenguajes y el equipo en que se hizo el desarrollo. La PDP-11/34 cuenta con sistema de utilería denominado PIP (Peripheral Interchange Program) avocado al manejo de archivos equivalente, al FMS del modelo general de una red, con la excepción de que no puede manejar los registros de los archivos. Fue la sintaxis de PIP la que se tomó como referencia en el diseño del Lenguaje de Comandos del Modo Experto y, en mucho, debido a que serían las PDP las primeras computadoras que se enlazarían a la red, y a que PIP posee una sintaxis compacta y legible.

Sintaxis del Lenguaje de Comandos

Describiremos a continuación la sintaxis del lenguaje de comandos LC y la función de cada uno de los comandos, postergando su tratamiento detallado hasta el momento en que revisemos el MSA. Los comandos válidos son los siguientes:

ADI

Termina una sesión de STA.

AYU

Proporciona ayuda acerca de la sintaxis y semántica del lenguaje.

?

Misma que ATU.

CON <host>

Solicita la conexión con el Sistema Servidor del STA.

TER <host>ttn:

Solicita el establecimiento de un enlace entre terminales.

LOG <host>[grp,mem]/passw

Solicita una sesión de STA en un afitrión remoto.

TFR <host>ddn:[grp,mem]nam.extiver/DE
/DR
/LD
/LI
/RE

Dependiendo de la opción elegida borra, lista el directorio, lista el directorio para borrar archivos bajo confirmación, despliega, y renombra archivos.

TFR <hi>di:[g1,m1]n1.e1;v1=<h2>d2:[g2,m2]n2.e2;v2/IM
/CP
/AG
/SO

Dependiendo de la opción elegida transfiere archivos modo imagen, copia, agrega y copia sobreescribiendo.

Parser de Comandos

Como podrá haberse dado cuenta el lector, la sintaxis del Lenguaje de Comandos es muy adecuada para anfitriones que poseen reglas de especificación de archivos similar a la de RSX-11M. Pero, como esta especificación no es en forma alguna un esquema estándar, su alcance de aplicación queda restringido. Para ayudar en la solución de este problema se desarrolló una herramienta que facilita la creación y manejo de lenguajes de comandos adecuados a las diversas formas de especificación de los anfitriones. Esta parte de la idea de apegarse a la idiosincracia propia de cada máquina, en vez de pensar en un método universal de especificación de archivos (recursos) que, como ya ha quedado asentado, no es una tarea trivial ni exclusiva de los FTP.

El lenguaje de comandos se apoya enteramente en una función de 'parser' sumamente poderosa que facilitó su desarrollo y que permite construir parsers para cualquier tipo de lenguaje de comandos debido a su naturaleza general. Esta función hace que la tarea de mapear un autómata de estados finitos a código sea casi directa, ya que cuenta con argumentos flexibles que resuelven todas las necesidades de este tipo de lenguajes. La función tiene la siguiente sintaxis:

trans(fmt,state,module,action,bits,mask,arg1,arg2,arg3,arg4,arg5)

donde:

fmt

Es una cadena de caracteres que controlan la ejecución del parser y especifican los formatos de los datos. El patrón codificado en fmt es el que se busca en la línea de entrada, es decir, es el que

se espera recibir.

fmt soporta los siguientes formatos:

%b aparea cualquier número de blancos incluyendo espacios y tabulador (TAB).

%nc aparea n caracteres alfabéticos. n puede ser cualquier número entero. Si se omite toma el valor de 1 por default.

%nd aparea n dígitos. n puede ser cualquier número entero. Si se omite toma el valor de 1.

%s aparea una cadena de caracteres alfabéticos de cualquier longitud.

%na aparea n caracteres alfanuméricos. n puede ser cualquier número entero. Si se omite toma el valor de 1.

%nx aparea n caracteres alfanuméricos o blancos. n puede ser cualquier número entero. Si se omite toma el valor de 1.

\n aparea un caracter 'newline'.

\0 aparea un caracter EOL.

LAMBDA es un formato especial que indica la condición de no-condición, o incondicional. Este formato es útil cuando se requiere efectuar ciertas acciones independientemente de las

condiciones de entrada.

state

Indica el estado del autómata al cual se saltará en caso de que se hubieren satisfecho las condiciones de brinco. Los estados 0 y 1 han sido reservados para propósitos específicos: El valor 0 indica que, si la instrucción trans que se está ejecutando actualmente no satisface el formato fmt, se pasará el control a la instrucción trans inmediatamente contigua, para dar oportunidad a que se pueda seguir analizando la línea de entrada. El valor 1 indica que, si encontramos en la línea de entrada una construcción que definitivamente no tiene sentido dentro del lenguaje, se concluya no sólo el análisis actual, sino toda la operación del parser.

file

Si el sistema que hace uso de la función trans se encuentra construido en overlays, file indica el módulo que debe ser cargado en memoria y al cual se le pase el control. Este argumento está íntimamente relacionado con el siguiente.

action

Es el nombre de una rutina de acción a la cual se le pasa el control, siempre y cuando se hayan satisfecho las condiciones especificadas en fmt. Junto con el argumento anterior forman el punto de entrada para una acción al especificarse que rutina de que módulo desea ejecutarse.

bits

Es una máscara de bits de propósito general que puede ser

utilizada para ir registrando las ocurrencias de cierto tipo de eventos, como por ejemplo, la presencia de ciertas variables en un comando.

mask

Es la palabra máscara (registro de banderas) sobre la cual se afectarán los bits indicados por el argumento bits.

arg1,...,arg5

San argumentos de propósito general que representan variables de interes a las cuales se les desea asignar un valor en el caso de que haya sido satisfecha la condición especificada en fmt. Estos argumentos pueden ser de distintos tipos.

Ejemplo:

Supongamos que deseamos analizar el comando que establece un enlace de transporte con el anfitrión remoto número 5. El comando es

CON <5>

y la instrucción de parser que nos verificará si es una construcción válida (tomada del código del STA) es:

TRANS('CONXb<Xd>\n',1,'bvario',Btrans,0,0,&sta1)_

Existen varias cosas muy importantes que debemos notar. Primero, vemos que la función está escrita en mayúsculas y que ha sido terminada con el caracter _. Esto significa que no estamos usando exactamente la función trans que hemos explicado, sino una modificación de la misma. La modificación es muy sencilla. En

realidad lo que hemos escrito equivale, en lenguaje C, a:

if (trans('CONXb<Xd>\n',1,'bvario',Btrans,0,0,&sta1)) break;

Esta y otras compactaciones similares permiten tener un código más limpio y claro, que redunda en la simplificación de la fase de depuración, sobre todo en grandes sistemas, como el STA.

La interpretación de la construcción es la siguiente: Se verificará, letra por letra, que la línea de entrada contenga los caracteres C,O,N, en ese orden y en mayúsculas. Después debe aparecer cualquier número de blancos, el caracter <, un número entero, el caracter >, y debe finalizar con un newline. En caso de que la línea de entrada haya satisfecho este patrón, se habrá asignado a la variable sta1 el valor decimal encontrado en la línea, que en nuestro ejemplo es un 5 y, finalmente, se ejecutará la rutina Btrans localizada en el overlay bvario. En caso que de que la línea de entrada no haya satisfecho el patrón de comparación, ésta no habrá sufrido ninguna modificación, de modo que pueda ser revisada por alguna otra instrucción del parser. La presencia de los ceros indican que no son parámetros de interés en esta construcción y que, por lo tanto, no se tomarán en cuenta en la evaluación de la expresión. Esta es la forma de indicarlo en una instrucción de trans. La presencia del 1 en el campo state ya ha sido explicada anteriormente.

Una ventaja muy grande de la función trans, con respecto a otros métodos de construcción de parsers, es la posibilidad de manejar formatos complejos, de modo que podemos crear los patrones máximos adecuados que deben ser analizados, reduciendo enormemente la cantidad de estados que se necesitarían para construir un parser

bajo otro tipo de herramientas. El autor ha podido experimentar especialmente con TPARS (Table Driven Parser) de Digital, y comparar la velocidad de diseño y desarrollo para un mismo lenguaje con la obtenida mediante TRANS (transición), y las pruebas han sido sumamente favorables para ésta. La gran dificultad estriba en la complejidad misma de esta función que debe permitir, entre otras cosas, cargar módulos de overlay, ejecutar rutinas, manejar formatos complejos, y soportar distintos tipos de variables.

Autómata de Estados Finitos del Parser de Comandos

Con el interés de mantener la completez de este trabajo, mostraremos a continuación el diagrama del autómata del parser, esperando que el lector no tenga dificultad para interpretarlo, ya que no se dará explicación alguna. También se transcribe un fragmento del código fuente del parser con la única intención de mostrar cuan directamente se reflejan entre sí.

Transiciones del Parser de Comandos		Estados	
1	LOG%b / iniLOG	A	LOG
2	<%d> / stai	B	---
3	L	C	UIC
4	[%d,%d] / g1^m1	D	PAS
5	:%1c%a / Blog^paswE0]^paswE1]	E	===
6	TER%b / iniTER	F	STC
7	<%d> / stai	G	---
8	TT%d:\n / Bcon^ter	H	===
9	HOL\n /Bhola	I	===
10	TRA\n / Btrans	J	===
11	ADI\n / Badios	K	===
12	.\n / Badios	L	===
13	AYU\n / Bhelp	M	===
14	?\n / Bhelp	N	ST1
15	MAI\n / Bmail	O	op1

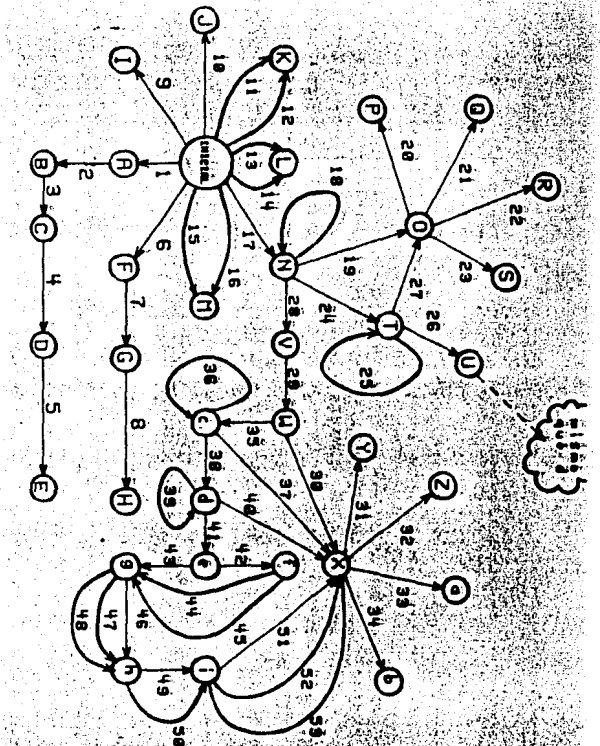


FIGURA 4.4
AUTOMATA DEL PARSER DE COMANDOS

```
16    MAI%b<%d>\n / Bmail^sta1                P    ===
17    TFRXb /iniFL1                           Q    ===
18    <%d> / sta1                             R    ===
19    /                                       S    ===
20    LI\n / Bfiles                           T    DV1
21    LD\n / Bftp^F_LD                        U    UC1
22    DE\n / Bftp^F_DE                        V    ST2
23    RE\n / Bftp^F_RE                        W    ---
24    L                                       X    op2
25    %2c%d / devi^uni                        Y    ===
26    L                                       Z    ===
27    /                                       a    ===
28    = / iniFL2                              b    ===
29    <%d> / sta2                             c    DV2
30    /                                       d    UC2
31    CP\n / Bftp^F_CP                        e    nm2
32    AD\n / Bftp^F_AD                        f    NM22
33    SD\n / Bftp^F_SD                        g    ex2
34    IM\n / Bftp^F_IM                        h    ex22
35    L                                       i    vr2
36    %2c%d / dev2^un2
37    /
38    L
39    [%d,%d] / g2^m2
40    /
41    L
42    /
43    L
44    %a / nom2[1]
45    L
46    .*
47    .%ic / ext2[0]
48    L
49    L
50    %2a / ext2[1]
51    /
52    ;#/ / ver2
53    ;%d/ / ver2


parser(cmd,est)

BEGIN

  case INI:
                TRANS('.\n',1,'bvaria',Badios,0,0)_
                TRANS('ADI\n',1,'bvaria,Badios,0,0)_
                TRANS('HLP\n',1,'bvaria',Bhelp,0,0)_
                TRANS('?\n',1,'bvaria',Bhelp,0,0)_
                TRANS('HOL\n',1,'bvaria',Bhola,0,0)_
                TRANS('CONXb<%d>\n',1,'bvaria',Btrans,0,0,&sta1)_
                TRANS('TERXb',STC,'inicia',iniCON,0,0)_
                TRANS('MAIXb<%d>\n',1,'inicia'iniMAI,0,0)_
```



```
                TRANS('MAI\n',1,'inicia',iniLOG,0,0)_
                TRANS('LOGXb',LOG,'inicia',iniLOG,0,0)_
                TRANS('XFRXb',BTi,'inicia',iniFL1,0,0)_
                XT

  case STC:
                TRANS('<%d>',0,0,0,0,0,&sta1)_
                TRANS('TTXdi\n',1,'bvaria',Bter,0,0,&ter)_
                XT

    o
    o
    o

END
```

Figura 4.5 Fragmento del Código del Parser de Comandos

MODO BATCH DE OPERACION

La IU presenta un tercer y último modo de operación que resuelve otro tipo de aplicaciones. Bajo esta modalidad es posible iniciar una sesión de STA, efectuar tareas de transferencia, y terminar la sesión, todo con un mínimo de comandos y, lo que es más atractivo, poder ejecutar la misma secuencia cualquier número de veces sin tener que reteclearla. La principal diferencia con las modalidades anteriores es que el tipo de sesión establecida es no-interactiva, lo que significa que (al momento) el usuario no tiene manera de poder supervizar la operación del STA.

Este modo se apoya totalmente en el lenguaje de comandos del modo experto para realizar sus funciones. Para que el usuario pueda hacer uso del modo batch lo único que necesita es crear un archivo con todas las instrucciones de STA necesarias para resolver su aplicación, empleando para ello cualquier tipo de editor de texto

que tenga a la mano y pasando el nombre de este archivo cuando así lo solicite el sistema. Como vemos, la única diferencia habida entre el modo batch y el experto es que la entrada de comandos al STA se realiza vía un archivo, en vez de darse a través del teclado. A pesar de ésto, puede ser que el sistema solicite información al usuario, dependiendo del grado de especificación de archivos que proporcionó o del tipo de operación pedida. Recordemos que el cumplimiento de ciertas funciones requieren la confirmación del usuario. Por todo lo demás, el modo batch se comporta idénticamente que el experto, de hecho comparten una gran parte de código.

INTERFAZ DE TRANSPORTE

El STA posee una segunda interfaz que en conjunto con la IU forman sus fronteras. Esta interfaz interactúa con la capa inferior de la arquitectura de la red, la Capa de Transporte. Es a través de esta interfaz que puede establecerse la comunicación entre STA's, y más propiamente, entre Capas de Aplicaciones. La capa de transporte proporciona un servicio de control de transmisión de circuitos virtuales, lo cual significa que provee los mecanismos de control de errores de transmisión y secuenciamiento de mensajes de tal suerte que el STA puede suponer que contará con una transmisión libre de errores, pudiéndose dedicar a controlar las operaciones propias de su naturaleza.

Funciones de Transporte

En el caso de la Interfaz de Transporte IT, es difícil ubicarla en

un sólo lugar dentro del código del sistema, ya que no es propiamente parte de él. Decimos que no existe propiamente una relación de pertenencia porque, aunque se liga parte del código de transporte con el del STA y comparte el mismo espacio de direcciones virtuales y físicas, la única inclusión visible de esta interfaz en el fuente del STA son las llamadas a las distintas funciones de biblioteca que proporcionan los servicios de transporte. Por esta razón, no existe un lugar específico dentro del fuente donde esté la IT, sino que se encuentra dispersa en muchos de los overlays. A pesar de esta dispersión de llamadas, existe código en el segmento raíz de la estructura de overlays que es compartido por todos los demás segmentos. Este código corresponde a un par de funciones de alto nivel, diseñadas explícitamente para facilitar el desarrollo del STA. Estas funciones se desglosan en varias primitivas de transporte, que son exactamente las que proporciona el SCT/CV de la estación de transporte. Las funciones de alto nivel a que hacemos referencia se utilizan únicamente para el envío de mensajes a través de la red. De nuevo, la idea surgió del interés por reducir código y crear herramientas de diseño.

Este fue un excelente desarrollo que probó ser conveniente incluirlo como una facilidad más de las funciones de transporte. La idea básica es poder conformar mensajes de envío sencilla y rápidamente. Pero, el problema son la gran cantidad de formatos que pueden poseer los mensajes, lo cual podremos apreciarlo posteriormente y, si deseamos que ésto sirva para la gran mayoría de las aplicaciones, el método debe ser suficientemente general

para que pueda manejar un buen ndsero de tipos de datos.

Las funciones son las siguientes:

manda(puerto,formato,a1,a2,a3,a4,a5,a6,a7)

puerto

Es el puerto de transporte asignado por la estación de transporte a la instancia actual del STA. Equivale al identificador de la tarea. A través de este puerto se realiza el diálogo entre STA's.

formato

Es una cadena de caracteres que indican el tipo, la secuencia, y la cantidad de espacio alojado, de los datos que deberán conformar al mensaje de envío. Los formatos válidos son los siguientes:

%c agrega un caracter (byte) al mensaje, sin importar su naturaleza.

%d agrega un entero (palabra) al mensaje, sin importar su naturaleza.

%n agrega n bytes al mensaje. Este formato requiere dos argumentos: la cantidad de bytes y la dirección de inicio de donde los leerá.

%s agrega una cadena de caracteres (bytes), sin importar su naturaleza.

Como vemos, no se modifica en absoluto el tipo de información que conformará al mensaje, por lo que puede ser incluso binaria.

a1,...,a7 _

Representan los argumentos que aparean uno a uno, los patrones habidos en el formato.

La función manda se comporta de acuerdo al siguiente algoritmo:

1) Pide un 'buffer' de envío al SCT/CV.

2) Conforma el mensaje de envío en este buffer de acuerdo al formato especificado.

3) Pide el envío del mensaje al SCT/CV. En caso de existir un error de transmisión irrecuperable para la estación de transporte, se pide el cierre del puerto de transporte y se rompe el circuito virtual establecido. Hasta este momento todos los errores de transmisión son considerados como fatales.

La segunda función de envío de alto nivel es:

sndbyt(puerto,byte)

que se utiliza en el caso de envío de un único byte de información. La intención es poderse ahorrar algunos 'teclazos' y hacer más nítido el código.

De acuerdo a los párrafos anteriores, podemos percatarnos que la función manda se compone de varias primitivas. Estas primitivas comprenden propiamente la IT, y son:

abrept(numpto,nuburc,taburc,nubutr,tabutr)

Pide a la estación de transporte ET, que se asigne un puerto cuyo valor será regresado en numpto. El puerto cero se reserva para el 'logger' de la red; los demás se asignan dinámicamente. Además, se pide que se alojen nuburc buffers de envío de tamaño taburc, y

nubutr buffers de recepción de tamaño tmbutr.

crrqpt(numpto)

Pide a la ET se descontinúe el uso del puerto numpto, de modo que quede disponible.

actrec(numpto)

Pide a la ET active la recepción de mensajes para el puerto numpto.

dameBf(numpto)

Pide a la ET un buffer para el envío de mensajes. La ET regresa la dirección de inicio del buffer.

libraBf(ptrbuf)

Pide a la ET libere el buffer de envío o recepción cuya dirección de inicio es ptrbuf.

edopt(numpto,ptrbuf)

Regresa información de estado del puerto numpto, y la almacena en el buffer apuntado por ptrbuf.

conect(numpto,estdst,ptodst)

Pide a la ET que establezca un circuito virtual entre el puerto local numpto, y el puerto ptodst localizado en la estación estdst.

descon(numpto)

Pide a la ET la terminación normal del circuito virtual mantenido por el puerto numpto.

espcon(numpto,estrem,ptorem,timout)

Pide a la ET que espere a que se termine de establecer la conexión entre el puerto local numpto, y el puerto ptorem de la estación remota estrem. Este evento no debe excederse de timout segundos.

reini(numpto)

Pide la reinicialización del circuito virtual mantenido por el puerto local numpto. Se pierde todo el 'status' actual del puerto.

envmsg(numpto,ptrbuf,numbyt,modo,prdad)

Pide el envío del mensaje de tamaño numbyt, alojado a partir de la dirección ptrbuf, a través del puerto numpto, con prioridad prdad (urgente o normal), y en modo síncrono o asíncrono.

espenv(numpto,ptrbuf)

Pide que se detenga la ejecución del programa hasta que se haya terminado el envío del mensaje ptrbuf que se envió por el puerto numpto. Esta función es de utilidad cuando se realizó un envío asíncrono.

rcbmsg(numpto,ptrbuf,numbyt,rutacc,modo,timout)

Pide a la ET la recepción de un mensaje por el puerto numtpo, que se ejecute la rutina de acción rutacc inmediatamente después de recibir el mensaje en el buffer direccionado por ptrbuf, y que además no se excedan timout segundos en espera de la recepción. La cantidad de bytes recibidos se guardará en la variable numbyt.

De todas las primitivas de transporte, solo la que muestra el estado de los puertos no se utilizó en el STA.

CAPITULO V

SISTEMA DE TRANSFERENCIA DE ARCHIVOS
PARTE III: LOS DETALLES
SECCION B: LOS PROTOCOLOS

Este capitulo está dedicado a la presentación de las funciones que actualmente soporta el STA. Explicaremos cual es el objetivo de cada una de ellas, acompañando la explicación con el diagrama del protocolo que respeta. Solo se hará el análisis del comportamiento del primero de los protocolos debido a que son numerosos y a que su lectura es sencilla.

DIAGRAMAS DE PROTOCOLOS

Empezaremos por dar la descripción de los símbolos que se emplean en los diagramas de los protocolos con el fin de que el lector pueda interpretarlos adecuadamente.
Los diagramas hacen uso de los siguientes símbolos:

Sección U

Todos los diagramas de protocolos se dividen en tres áreas básicas: sección izquierda, sección central y sección derecha. La sección izquierda, identificada por la letra U de tipo mayor, representa al sistema usuario. Su margen lo constituyen los bloques identificados como W y R.

Sección S

Es la sección que se encuentra en la parte extrema derecha del diagrama representa al sistema servidor. Empieza con los módulos identificados como W y R, y se extiende hasta el extremo derecho.

Canal

El canal lógico de comunicación (sin identificación explícita) se localiza en la parte central del diagrama, entre las secciones U y S. Esquematiza a la conexión de transporte habida entre el sistema usuario y el sistema servidor, y muestra en su interior el sentido del flujo de la información, así como los tipos de datos que la conforman.

Bloque W

Los bloques identificados con la letra W mayúscula representan estados complejos, compuestos por estados primitivos de transporte que efectúan "escrituras" de mensajes de transferencia sobre el canal. Este bloque equivale a las funciones de envío de alto nivel mandq y sndbyt.

Bloque R

Este bloque esquematiza a la función de recepción de la estación de transporte en un sentido más general. Equivale más bien a un estado de recepción, más que al evento mismo. Este bloque, junto con el anterior, constituyen la interfaz con los servicios de transporte. En este sentido, todo el control de lo que ocurre más allá de sus límites diagramáticos es responsabilidad directa de los bloques W y R, es decir, de la estación de transporte SCT/CV.

Bloques de decisión

Los bloques en forma de rombo poseen la misma cualidad que en los diagramas de flujo de programas. La diferencia sustancial es más de grado que de fondo. En los diagramas de los protocolos mantienen un significado más amplio, en el sentido de que son condicionales complejos, esto es, su cualidad no representa necesariamente a una sola variable de decisión.

EXIT

Representa un estado complejo de salida. Indica la finalización del protocolo en forma normal; exitosa en términos del protocolo. En él se llevan al cabo todas las operaciones necesarias para terminar grácilmente.

ERROR

Esquematizado por el símbolo tradicional de impresión con la leyenda en su interior, muestra también un estado complejo de salida en el mismo sentido anterior. La diferencia estriba en que es un estado al cual se llega cuando falla la operación del protocolo, es decir, no se cumplen las expectativas del comando (función) al cual pertenece.

flechas

Muestran el sentido de operación del protocolo. Asimismo, representan estados de operación del protocolo en los que se ejecutan todas las tareas internas necesarias para el buen funcionamiento del mismo. Esta cualidad nos permite reducir sustancialmente los diagramas al no tener que emplear otro símbolo para esta eventualidad. Esto no restringe, de cualquier forma, a

que puedan llegar a ser adecuados para resaltar alguna operación especialmente importante, en algún momento dado.

Lectura de los Diagramas de Protocolos

En vez de tratar de dar una explicación más o menos abstracta de como se leen estos diagramas, estimamos como mejor estrategia hacerlo con un ejemplo sencillo y, para avanzar en el tema, lo haremos con el primero de los protocolos del STA, que además respeta el calificativo.

FUNCIONES Y PROTOCOLOS DEL STA

En una sesión de STA podemos identificar tres fases de operación: de asociación, transferencia de datos, y disociación. Haremos uso de este comportamiento para la clasificación de las funciones y protocolos del STA, y tomaremos este orden en la siguiente descripción.

FASE DE ASOCIACION

Realiza las tareas de identificación de los derechos de acceso al anfitrión remoto y establecimiento del enlace entre los sistemas usuario y servidor. El STA posee las siguientes funciones y protocolos para esta fase:

CONECTA

Un usuario puede hacer uso de los servicios del STA de Red-IIMAS entrando primero en sesión normal en un anfitrión enlazado a la

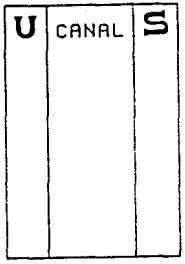


FIGURA 5.1
SECCIONES DE UN DIAGRAMA DE PROTOCOLO

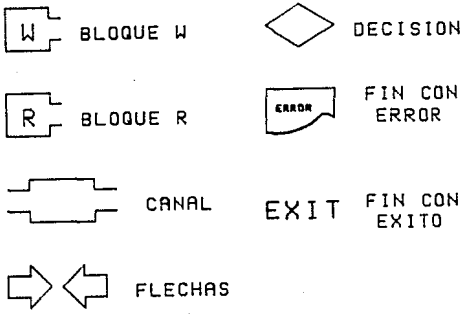


FIGURA 5.2
SIMBOLOGIA DE LOS DIAGRAMAS DE PROTOCOLO

red, y pidiendo la activación de la tarea STA, denominada FTP. Esta es una tarea privilegiada que ha sido instalada en el sistema como pública, de tal forma que cualquier usuario puede activarla independientemente de sus privilegios. En caso de que el usuario sea no-privilegiado, la tarea asume automáticamente sus privilegios. Después de haber activado el STA, el usuario se encuentra en sesión con el sistema y puede hacer uso de sus funciones. Casi todas las funciones del STA poseen dos modos de operación: Modo Local y Modo Remoto.

Modo Local de Operación

Un usuario puede trabajar con el STA en este modo dándole un ambiente local de manejo de archivos, es decir, no es necesario que se conecte a un anfitrión remoto para interactuar con el STA en este modo. Aunque ésto pueda parecer paradójico, en realidad no lo es ya que el STA posee una serie de funciones que lo pueden hacer atractivo fuera del ambiente de red.

Modo Remoto de Operación

Esta forma de operar el STA es la más natural al sistema. Es el modo bajo el cual opera un usuario cuando realiza transferencias de archivos a través de la red, es decir, hace uso de los servicios de transporte. Esta aclaración viene al caso puesto que un mismo anfitrión puedeo ser tanto el origen como el destino de una transferencia.

Para que un usuario pueda realizar una transferencia de archivos vía Red-IIMAS, es necesario que pida el enlace del sistema usuario

local con el sistema servidor del anfitrión remoto. Este enlace no se realiza directamente, sino que interviene en él un tercer sistema vital de la red: el Activador de Procesos AP. Bástenos saber, por el momento, que es a él a quién se pide la activación de los servidores de Red-IIMAS (el capítulo 6 se dedica a él). La petición de enlace se ejecuta a través de la función CONECTA del STA. CONECTA lleva a cabo toda una negociación con la estación de transporte SCT/CV de la red para poder satisfacer la demanda. El único parámetro que se le solicita al usuario es la identificación del anfitrión destino con el cual desea enlazarse. STA no conoce el nombre bajo el cual vive el SS en el anfitrión destino, de modo que solicita su activación bajo el nombre universal designado para este sistema: SERVER. La figura 5.3 muestra el protocolo de conexión del SU con el SS.

El protocolo funciona de la siguiente manera:
El estado original del SU y del SS son de escritura y lectura respectivamente, indicado por los bloques superiores W y R. La operación de ambos sistemas es concurrente. SU pide al Activador de Procesos AP activar el sistema SERVER, mediante el mensaje: fACP SERVER. SU pasa inmediatamente después al estado de recepción R, para esperar el resultado de su petición. Mientras tanto, el mensaje viaja por la red. El AP al momento de recibir el mensaje pide al sistema operativo la activación del SS y, dependiendo si tuvo éxito o no, le comunicará este resultado al SU mediante un mensaje fACK pto (acKnowledge) o fNAK err (negative acKnowledge), respectivamente, y termina su operación. Si tuvo éxito la operación el mensaje fACK posee un segundo argumento en el cual se

indica el número de puerto asignado al SS por el SCT/CV remoto, para que después el SU pueda enlazarse con él. En caso de haber fallado la petición, el mensaje fNAK lleva el tipo de error en que se incurrió para, tal vez, poder remediarlo. Finalmente, el SU recibe la respuesta y actúa en consecuencia, terminando su operación.
Los demás diagramas, aunque más complicados, son igualmente fáciles de interpretar.

LOGIN

Una vez que el SU ha establecido el enlace con el SS remoto, el usuario debe entrar en sesión virtual con él. Esto se hace con el fin de verificar los derechos de acceso del usuario en el SS. Si un usuario no tiene cuenta en el anfitrión remoto, no podrá realizar ninguna operación de transferencia. Esta cuenta es exactamente la misma que utilizaría en caso de que estuviera sentado directamente a la computadora, y es la que determina la asignación de sus privilegios. El SS impone la restricción de que sólo se pueden utilizar los archivos que están almacenados bajo la cuenta de acceso, aun cuando sea un usuario privilegiado. La información que debe proporcionar el usuario es la cuenta de acceso (UIC), y la contraseña. La Figura 5.4 muestra el protocolo de login.

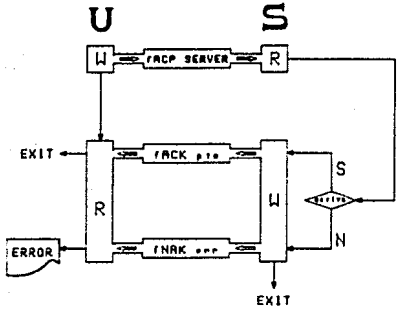


FIGURA 3.3
PROTOCOLO DE CONEXION

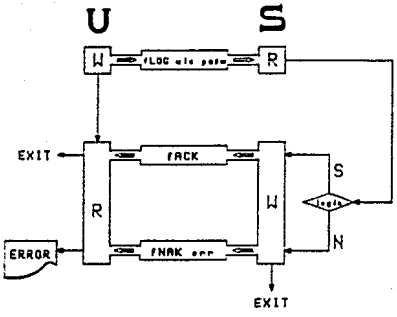


FIGURA 3.4
PROTOCOLO DE LOGIN

FASE DE TRANSFERENCIA DE DATOS

Esta fase incluye las operaciones locales sobre archivos, identidad y atributos de los archivos, definicion de la direccion de la transferencia, manejo de las estructuras de los archivos y datos, especificacion de operaciones adicionales sobre el archivo transferido, la tansferencia de datos y el dialogo administrativo necesario para regularla. Las funciones son las siguientes:

BORRA

Esta funcion se pide cuando se desean borrar archivos. Puede ser invocada en modo local o remoto. El usuario proporciona la especificacion de los archivos que desea borrar y, bajo cualquiera de las modalidades, se puede precisar si se desea ser interrogado para confirmar la ejecucion o no. La especificacion de los archivos es la nativa del anfitrión. La figura 3.5 muestra el protocolo.

CHISME

Uno de los atractivos que poseen las redes de computadoras es su capacidad de poder establecer comunicación entre usuarios enlazados a la red. Una manera de hacerlo es estableciendo una comunicacion entre terminales, permitiendo efectuar un diálogo interactivo. Esta posibilidad se dá en el STA mediante la función chisme. El usuario debe proporcionar la identificación de la terminal con la cual desea que se establezca la liga. El usuario sentado al otro lado de la 'linea' puede rechazar o aceptar el enlace. Esta función no tiene forma, actualmente, de poder indagar quienes están en sesión en algun anfitrión de la red. Aún así es

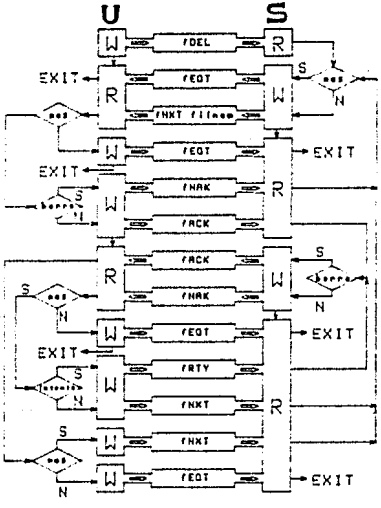


FIGURA 5.5
PROTOCOLO DE BORRADO

de bastante utilidad. Existen diferencias menores entre los modos local y remoto de operación debido, principalmente, a que en modo local es posible explotar mejores características de diseño que en el modo remoto. A pesar de ésto, la invocación es la misma. La figura 5.6 muestra el protocolo.

DIRECTORIO

Es posible tener un listado en pantalla o papel, de los archivos disponibles a un usuario bajo la cuenta de acceso en que se encuentra. El usuario debe suministrar la especificación de archivos, y la modalidad de operación: local o remoto. La figura 5.7 muestra el protocolo.

ENVIA

Puede ser que hasta ahora el lector tenga un sentimiento de incredulidad acerca del porque se insiste en el término de transferencia de archivos; la función presente aliviará esta impresion.

No iteraremos de nuevo acerca de las bondades que puede presentar el transferir informacion entre anfitriones, digamos que, cuando se desee, se puede utilizar la función ENVIA. Esta función solo se soporta actualmente en modo remoto, y puede actuar de tres formas:

Envia para Añadir

Un archivo puede ser transmitido para que sea agregado al final de otro, y solo admite archivos no-contiguos. La figura 5.8 muestra el protocolo.

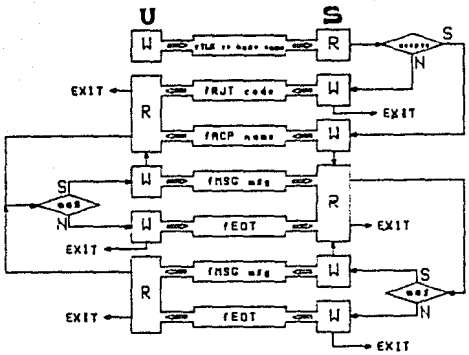


FIGURA 5.6

PROTOCOLO CHISME

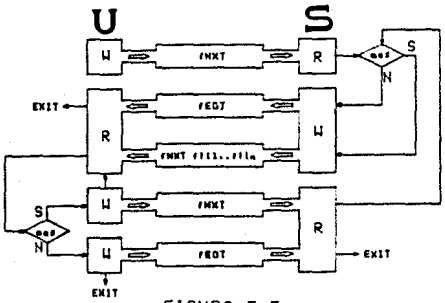


FIGURA 5.7

PROTOCOLO DIRECTORIO

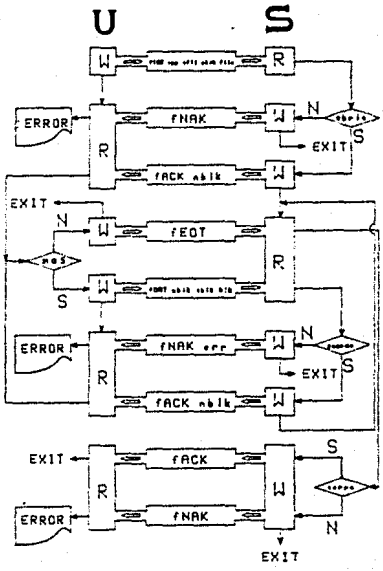


FIGURA 5.8

PROTOCOLO ENVIA PARA AÑADIR

Envia para Imprimir

Otra posibilidad es que se desee utilizar alguna impresora remota con buena calidad de impresión. No existe restricción acerca del tipo de archivo que se imprimirá, ya que no existe validación del mismo. Sin embargo, el archivo se manejo como si fuese de texto y, aunque existe la posibilidad de enviar archivos binarios puede ser que no se impriman correctamente, o no se impriman en absoluto. De nuevo, el envio es solo remoto. La figura 5.9 muestra el protocolo.

Evia para Desplegar

La ultima alternativa posible es que se envie el archivo para ser desplegado en alguna terminal. Aquí se aplican las mismas consideraciones del párrafo anterior para el caso de las terminales, agregándose a esto una fase inicial de autorización para utilizar una terminal que está en uso. La figura 5.10 muestra el protocolo.

LISTA

En ciertas ocasiones puede surgir la duda de si el archivo que se desea enviar es el que uno supone, o si el que ha sido traido ha sido el adecuado. Para salir de dudas podemos utilizar la función lista y desplegar en la pantalla el contenido del archivo. Lo unico que debe hacer el usuario es proporcionar la especificación del archivo. Esta función solo opera en modo local y, por lo tanto, no posee protocolo.

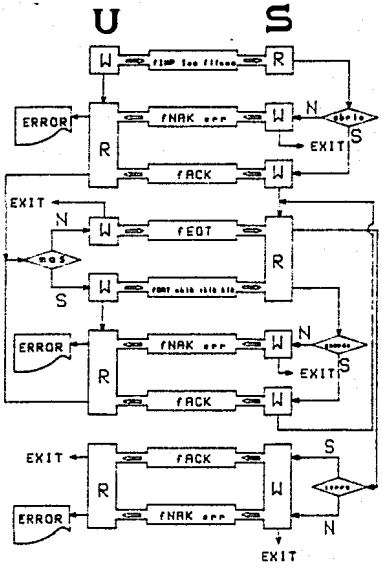


FIGURA 5.9
PROTOCOLO ENVIA PARA IMPRIMIR

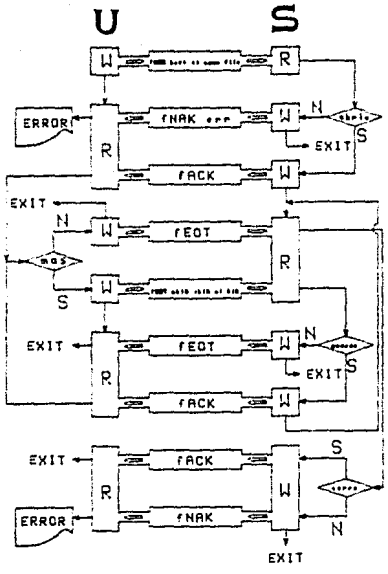


FIGURA 5.10
PROTOCOLO ENVIA PARA DESPLEGAR

CORREO

Ver capitulo VI.

RECIBE

Esta funcion es la contraparte exacta de ENVIA. Todas las anotaciones hechas para aquella son totalmente aplicables en este caso, observando los cambios pertinentes de vocablos. La figura 5.11 muestra el protocolo para Recibe para Agregar. la 5.12 corresponde al de Recibe para Imprimir, y la 5.13 al de Recibe para Desplegar.

RENOMBRA

Esta es la ultima de las funciones que pertenecen a la fase de transferencia de datos del STA. Su intención es poder cambiar el nombre a un archivo, o conjunto de archivos. Esto puede llegar a requerirse en casos en que existen conflictos en la denominación de los archivos, o que simplemente se le quiere rebautizar de manera más significativa. Para cualquiera que sea la idea es posible efectuar esta operación, y en los modos local o remoto. La operación de renombrado posee ciertas diferencias minimas en su comportamiento si trabajamos en modo Experto o en modo Novato. Esta divergencia proviene de las capacidades mismas de estos esquemas, y es que utilizando el lenguaje de comandos es posible especificar un nombre de archivo origen y uno destino, y efectuar la operacion de golpe. Para el caso de nombres genericos de archivos, el comportamiento es el mismo: la operación se lleva a cabo en forma interactiva, teniendo el usuario que proporcionar un nombre a la vez conforme se le va indicando. La figura 5.14

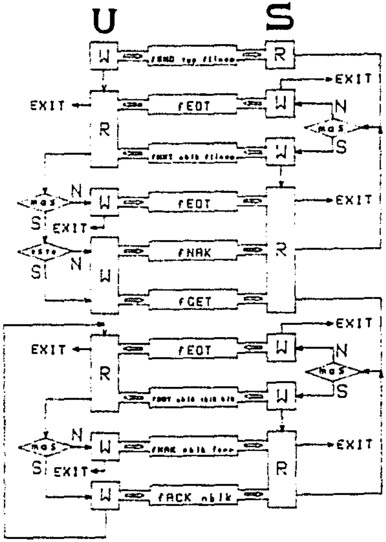


FIGURA 5.11
PROTOCOLO RECIBE PARA AGREGAR

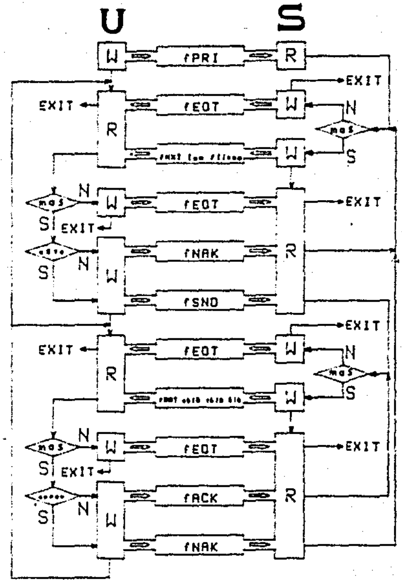


FIGURA 5.12
PROTOCOLO RECIBE PARA IMPRIMIR

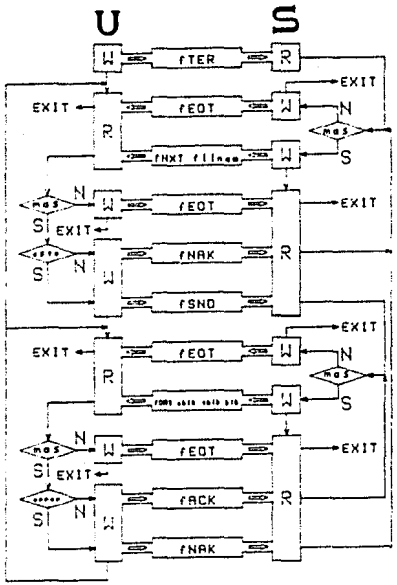


FIGURA 5.13
PROTOCOLO RECIBE PARA DESPLEGAR

muestra el protocolo.

FASE DE DISOCIACION

Esta fase termina las operaciones locales sobre los archivos, cierra la conexión, e indica el estado final de la transferencia. Esta fase cuenta únicamente con una función: ADIOS. La figura 5.15 muestra su protocolo.

FUNCIONES COMPLEMENTARIAS

Existen varias funciones que por el tipo de operación que realizan no caben propiamente en alguna de las fases del STA, o son tareas de apoyo. Estas funciones son las siguientes:

AYUDA

Algo que no debe perderse de vista cuando se diseñan grandes sistemas que deben interactuar con una persona, es la posibilidad de poder brindarle toda la ayuda que sea posible, con el fin de apoyarlo en la consecución de sus tareas. El STA mantiene dos sistemas de ayuda, uno para el modo novato y otro para el experto. En ambos casos es posible tener ayuda sobre casi cualquier cosa que pueda resultar oscura en cierto momento, sobre el significado de alguna función o comando, los tipos de datos, o la sintaxis. Siendo esta una facilidad local, es claro que no exista un protocolo para ella.

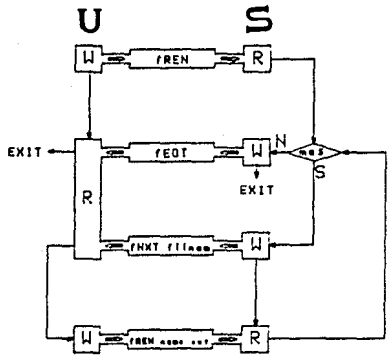


FIGURA 5.14
PROTOCOLO RENOMBRA

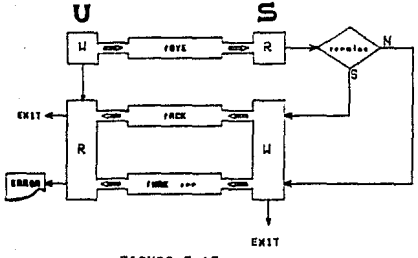


FIGURA 5.15
PROTOCOLO ADIOS

HOLA

Aunque esto no es aplicable a la situación actual, puede llegar un momento en que puedan haber divergencias entre un sistema y otro, debido principalmente a que corresponden a distintas versiones y que, por tanto, no pueda darse un pleno entendimiento entre ellos. Con el fin de poder determinar si nos encontramos en esta situación o no, se ha diseñado una función que posee un doble proposito. Podemos utilizarlo para verificar el numero de versión de los sistemas en juego, y anticipar el posible resultado de nuestras acciones futuras. El segundo proposito es poder emplearla como mecanismo de detección de fallas, explícitamente, verificar si aún 'vive' el SS ya que (siendo totalmente factible) puede ocurrir una 'caída' del anfitrion remoto. Si emitimos este comando y no obtenemos respuesta en un tiempo perentorio, podemos imaginar la causa. La figura 5.16 muestra el protocolo que respeta.

EXISTE

EXISTE es una función que sirve de apoyo en muchos de los protocolos de la fase de transferencia de datos y tiene encomendada la tarea de verificar la existencia y/o validez de la especificación de los archivos de red. Además de ésto, facilita la manipulación de archivos remotos al permitir reducir el número de parametros necesarios en la escritura del código del SS, al no tener que llevar y traer esta especificación por todo el programa. El supuesto es que, si se ha pedido la verificación de existencia y/o validación de un archivo, es porque se pretende trabajar con este y, entonces, es posible asignar esta especificación a una

variable global accesible desde cualquier punto del programa.
El protocolo de EXISTE es muy sencillo, y es mostrado en la figura 5.17.

Para finalizar la discusión acerca de los protocolos del STA, daremos una tabla en la que se clasifican todas sus funciones del STA de acuerdo a las distintas fases a las que pertenecen, así como las complementarias.

CRITERIO	COMANDOS
FASE DE ASOCIACION	CONECTA LOGIN
FASE DE TRANSFERENCIA	BORRA CHISME DIRECTORIO ENVIA para Agregar para Imprimir para Desplegar LISTA CORREO Enviar Carta Carta Siguiente Carta Anterior Borrar Carta RECIBE para Agregar para Imprimir para Desplegar RENOMBRA
FASE DE DISOCIACION	ADIOS
COMPLEMENTARIAS	AYUDA EXISTE HOLA

Figura 5.18 Fases y Funciones del STA

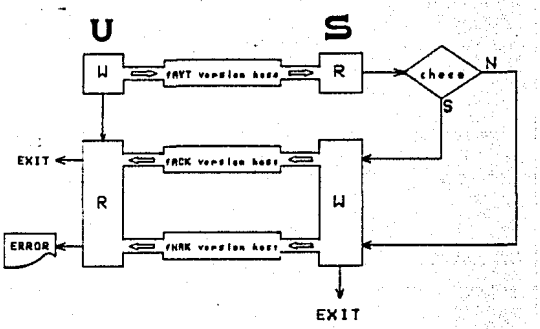


FIGURA 5.16
PROTOCOLO HOLA

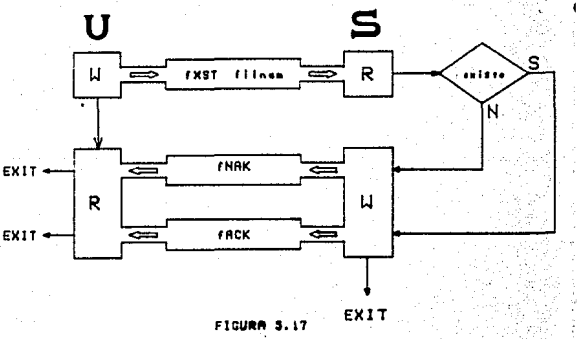


FIGURA 5.17
PROTOCOLO EXISTE

ANOTACION FINAL

No hemos querido engrosar aún más el número de páginas de este capítulo por lo que pedimos al lector interesado referirse al apéndice A para una descripción del significado, tipo y valor, de los parámetros empleados en los protocolos mostrados a lo largo de este capítulo, y en el siguiente.

CAPITULO VI

SUBSISTEMA DE CORREO ELECTRONICO

Otro gran atractivo que ofrece el STA es el Subsistema de Correo Electrónico SCE. Esta facilidad es otra forma de alentar la comunicación entre los usuarios de Red-IIMAS.

Cuando no es posible, o no se desea, entablar una comunicación en línea con un usuario, se puede hacer uso del servicio de correo electrónico del STA (que pretende convertirse en el estándar de la red) y dejar mensajes en el buzón del usuario destino. El porque el STA contempló al SCE como subsistema es que éste puede verse como un tipo especial de FTP.

FUNCIONES DEL SCE

El SCE permite realizar tareas básicas de correo electrónico y, dependiendo de su evolución, podrá ser más conveniente erigirlo fuera del STA conforme se vaya sofisticando.

El subsistema de correo electrónico proporciona sus servicios tanto en modo local como en modo remoto. Esto significa que el STA podrá serle sumamente útil incluso a un usuario que posea exclusivamente una cuenta de acceso a una computadora de la red, a través del SCE. Esto le permitirá enviar cartas a usuarios locales y recibir correo desde cualquier usuario de la red que tenga acceso a su computadora.

El subsistema de correo contempla las siguientes operaciones:

Enviar una Carta

Un usuario puede conformar una carta, y enviarla hacia cualquier buzón a que tenga derecho. Las cartas son archivos de texto que se construyen bajo la tutoría del sistema de correo, quien controla todas las operaciones involucradas. A cada usuario de la red se le asigna un buzon bajo su cuenta normal, y en forma automática. Es decir, no existen funciones especiales para la creación de buzones, sino que, una vez que a un usuario se le ha autorizado enviar una carta, el sistema verifica si el destinatario posee buzón y, en caso de no contar con el, se lo crea automáticamente. Este buzon está integrado por el par de archivos MAIL.DIR y MAIL.TXT, de modo que el usuario destino no debe tener ninguna información almacenada bajo estos nombres pues, de lo contrario, no podrá participar en el SCE. Mail.dir es un archivo que mantiene todo el control de la ubicación de las cartas almacenadas en mail.txt. Posee una tabla de apuntadores que se actualiza acordemente a las operaciones de manejo de cartas que realiza el dueño del buzon. Particularmente, el dueño de un buzon es aquel que posee acceso a la cuenta de usuario bajo la cual radica. Un usuario puede enviar un mensaje a cualquiera otro de la red, y la única posibilidad de acceso remoto que se tiene cuando no se ha efectuado login remoto, es la propia computadora. Esta es la única operación que no necesita saber la clave de seguridad de una cuenta, el único requisito es conocer la identificación del usuario destino; su UIC. La información de acceso pedida al usuario incluye el anfitrión y usuario destino.

La figura 6.1 muestra la estructura del buzón, la 6.2 el

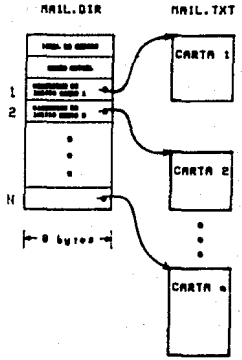


FIGURA 6.1
ESTRUCTURA DE UN BUZON

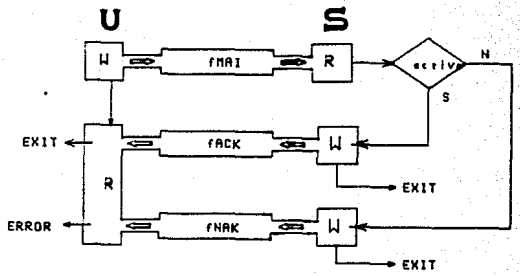


FIGURA 6.2
PROTOCOLO DE ENLACE CON CORREO REMOTO

protocolo de enlace con el subsistema remoto, y la 6.3 el protocolo de envio.

Carta Siguiente

El subsistema de correo posee un conjunto básico de manejo de cartas. Una forma de viajar a traves del buzon es via la función que nos muestra el contenido de la carta siguiente. Solo podemos viajar por el buzon a un paso a la vez, es decir, carta por carta. Las operaciones de viaje funcionan idénticamente en modo local que en remoto. La figura 6.4 muestra el protocolo de carta siguiente.

Carta Anterior

Esta es la segunda posibilidad para el despliegue de cartas. Su operación es inversa a la anterior. La figura 6.5 muestra su protocolo.

Borrar Carta Actual

Una vez que hemos leído las cartas y que ya no es importante el seguir almacenándolas, podemos borrarlas del buzon y mantener una estructura integra dentro del mismo. Para poder eliminar estas cartas es necesario que nos 'posicionemos' sobre la carta deseada y borrarla mediante la ejecucion de la función prevista. En la figura 6.6 se muestra el protocolo de borrado de carta.

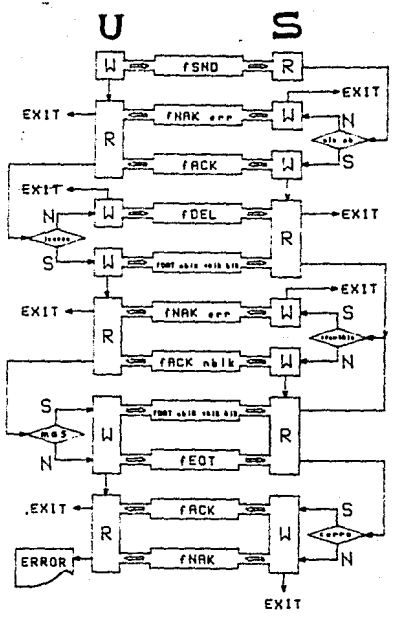


FIGURA 6.3
PROTOCOLO DE ENVIO DE UNA CARTA

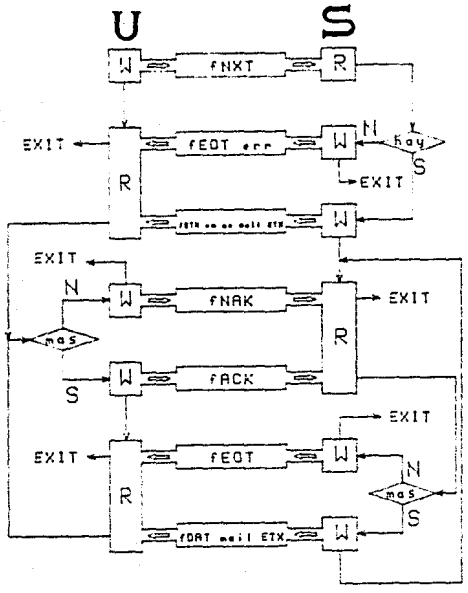


FIGURA 6.4
PROTOCOLO CARTA SIGUIENTE

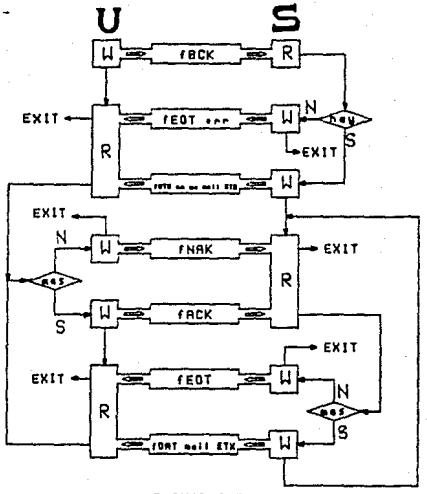


FIGURA 6.5
PROTOCOLO CARTA ANTERIOR

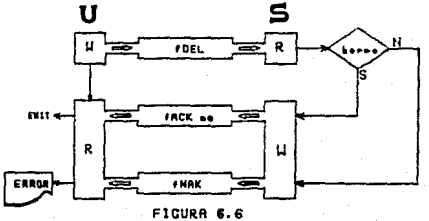


FIGURA 6.6
PROTOCOLO BORRAR CARTA ACTUAL

GUIA DEL USUARIO

El SCE puede ser activado desde el STA a través de menues o comandos. En el primer caso es necesario que el usuario elija el segundo menú principal y seleccione la función MAIL. Si el usuario no estado trabajando en modo experto, deberá ejecutar el comando MAI. En cualquiera de las dos situaciones se habrá activado el subsistema de correo y se estará en disposición de solicitar sus servicios, ya sean locales o remotos, con la salvedad de que para poder trabajar en modo remoto es prerrequisito haber efectuado login en algún anfitrión de la red, ya que se ha seleccionado el modo de operación deseado, el usuario podrá ejecutar cualquiera de las cuatro operaciones de manejo de cartas de que dispone.
A continuación se muestra una sesión típica de correo.

```
STA> MAI                    correo local
     o
STA> MAI <host>             correo remoto con host
MAI> BCK                    despliega el contenido de la carta anterior
     o
MAI> NXT                    despliega el contenido de la carta siguiente
     o
MAI> DEL                    borra la carta actual
     o
MAI> SND [cpy,eeml/cosu(asunto)    envia carta
termina con <cr><cr>
     -------
     -------
<cr><cr>
MAI> .
STA>
```


CAPITULO VII

ACTIVADOR DE PROCESOS

Este capítulo lo dedicamos a un sistema que juega un papel fundamental en la operación de Red-IIMAS: el Activador de Procesos AP. A este se le conoce comunmente en la literatura como logger, debido a que sus funciones son en cierta forma, similares a las que realiza un usuario para acceder a los servicios de una computadora: login.

PARTE I: LOS CONCEPTOS

MANEJO DE DIRECCIONES Y ESTABLECIMIETO DE CONEXIONES

En una red de computadoras los anfitriones poseen recursos que pueden ser accedidos desde otros anfitriones. En la siguiente discusión nos enfocaremos hacia un tipo de recurso: el servicio genérico. Esta clase de servicio puede realizar diversos tipos de acciones para sus usuarios, como por ejemplo, hora del día, compilar en Pascal, etc., etc. Cada uno de estos servicios tiene asociado un proceso, conocido generalmente como servidor, cuya tarea es proporcionar el servicio a cualquier proceso autorizado que lo solicite.
Una manera de poder otorgar estos servicios es hacer que cada proceso servidor ejecute una llamada de 'escucha' (ESPCON del SCT/CV) sobre alguna dirección de transporte asignada. Para poder

emplear el servicio. El usuario efectúa entonces una conexión, (CONNECT del SCT/CV) especificando la dirección de transporte remota apropiada. Aquí surge entonces una pregunta, ¿cómo es que un usuario puede saber qué servidor escucha sobre qué dirección de transporte?. Además, ¿cómo sabe un servidor sobre qué dirección de transporte escuchar?.

Aquí es importante resaltar la diferencia entre un 'nombre', esto es, ¿cuál es el servicio que desea un usuario, y una 'dirección', es decir, sobre qué puerto de transporte debe escuchar el servidor. Un nombre es generalmente una cadena de caracteres cuya intención es que sea utilizada por personas en vez de máquinas. Consecuentemente, es necesario efectuar un mapeo entre el nombre y la dirección. El mapeo puede ser llevado al cabo consultando un **servidor de nombres**, análogo al directorio telefónico. En ciertos casos los servidores pueden escuchar en las, así llamadas, **direcciones bien conocidas**, o domicilio conocido, que son direcciones que todos conocen y que rara vez se cambian, de modo que es necesario que estén perfectamente bien publicadas. La dirección del servidor de nombres nunca debe ser cambiada, ya que éste no puede ser utilizado para localizarse a sí mismo.

En cualquier situación, una vez que ha sido mapeado un nombre hacia una dirección, la estación de transporte SCT/CV debe establecer una conexión con el proceso que está escuchando sobre ella. Aunque no existen reglas generales sobre cómo se deben nombrar los servicios, siendo posiblemente local a cada computadora o específico de cada aplicación, sí existen un par de estrategias para alojar direcciones de transporte: direcciones jerárquicas y direcciones planas. Red-IIMAS posee un esquema

jerárquico de direcciones.

Generalmente, los usuarios acceden a los servicios genéricos por nombre, no por dirección, necesitándose, por ende, realizar el mapeo descrito. Existe todavía otro problema, que puede llegar a ser muy importante. Muchas computadoras poseen recursos de cómputo muy restringidos, y en particular pueden ser de memoria. Esto significa que no pueden coexistir muchos procesos activos a la vez. Hemos dicho que un proceso servidor requiere efectuar una operación de escucha para que pueda ser establecida una conexión, esto implica necesariamente, que debe permanecer activo para poder atender la petición en cualquier momento que ocurra. Otro aspecto relevante también es, que no necesariamente puede haber una sola instancia de cada servicio. Este problema de espacio limitado se tuvo en la PDP-11/34, y hubo que encontrársele solución.

'Afortunadamente', este mismo problema surgió en los primeros días de ARPANET, y fue su experiencia la que vino a proporcionar una manera elegante de resolución. Red-IIMAS posee un protocolo similar al de ARPANET, denominado Protocolo de Conexión Inicial PCI.

PROTOCOLO DE CONEXION INICIAL

En vez de que los servidores se encuentren siempre escuchando sobre un domicilio conocido, cada anfitrión que ofrece servicios a usuarios remotos posee un servidor de procesos especial (o logger), a través del cual deben ser pedidos todos los servicios. En Red-IIMAS se conoce a este logger como Activador de Procesos AP. Cuando el AP se encuentra ocioso escucha sobre su domicilio

conocido, identificado por el puerto 0 de transporte en todos los anfitriones. Este puerto es reservado y su asignación es controlada por la estación de transporte, y no puede ser utilizado por ningún otro servidor. Para poder establecer una conexión entre usuario y servidor, el usuario debe conectarse primero con el AP especificando el puerto cero del anfitrión remoto. Una vez que ha sido establecida esta conexión, el usuario envía al AP un mensaje indicándole cual es el servidor que requiere sea activado. El AP remoto activa entonces al servidor pedido, y este le comunica la dirección de transporte que le ha asignado la ET. Finalmente, el AP retransmite al usuario esta dirección, ambos terminan su conexión, el AP se pone en modo escucha y el usuario puede ahora conectarse con su servidor a la dirección que ya conoce ahora. La figura 7.1 muestra la esquematización de este protocolo.

PARTE II: LOS DETALLES

Ahora explicaremos cuales son exactamente todas las operaciones internas que realiza el AP, empezando con la descripción de su estructura de implantación, para finalizar con la del protocolo PCI.

ESTRUCTURA DE IMPLANTACION

El Activador de Procesos AP posee una estructura de implantación muy especial, distinta de las del STA y el CE. Debido a que se trata de un sistema pequeño, en comparación con los dos anteriores, su código está escrito en un solo módulo en lenguaje C

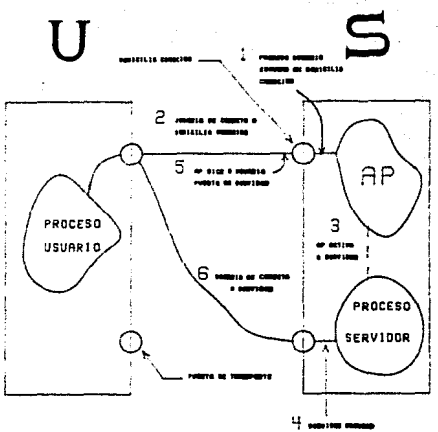


FIGURA 7.1
PROTOCOLO DE CONEXION INICIAL

y MACRO-11. Sin embargo, no es esta la principal distinción del AP, sino el tipo de interfaces que posee.

Ya que el AP no sostiene interacción con un usuario (persona), carece de la IU preservando, únicamente, una IT puesto que debe dialogar con procesos de aplicación a través de Red-IIMAS. Esta IT contiene siendo la misma de siempre. Por otro lado, posee una interfaz adicional única que le permite comunicarse con los servidores, cuya activación controla, denominada Interfaz de Comunicación Tarea-a-Tarea.

Interfaz de Comunicación Tarea-Tarea

En la figura 7.2 se muestra la estructura lógica del AP dentro de la arquitectura de Red-IIMAS y en la 7.3 la forma en que participan las interfaces IT e ICTT en la operación del AP.

La razón que justifica la existencia de una segunda interfaz se hará evidente si consideramos el siguiente escenario.

Una vez que un proceso usuario remoto ha establecido una conexion con el AP, y le ha solicitado la activación de un cierto servidor, el AP deberá realizarlo de alguna forma. Una condición esencial para que pueda ser satisfecha esta acción es, que la tarea correspondiente a ese servidor se encuentre 'instalada' en el sistema, ésto es, que exista una entrada en las tablas del sistema operativo correspondiente a ella. El identificador de la tarea puede ser una cadena de caracteres idéntica al nombre genérico empleado en la red para referirse a este servidor pero, debido a restricciones propias de la computadora, ésto no sucede

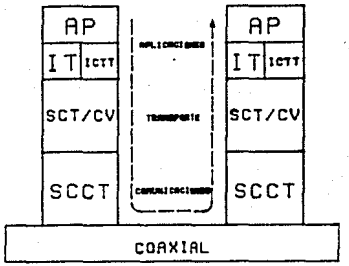


FIGURA 7.2
UBICACION DEL AP DENTRO DE Red-IIMAS

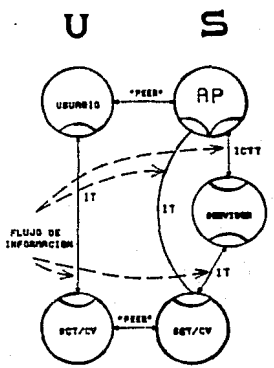


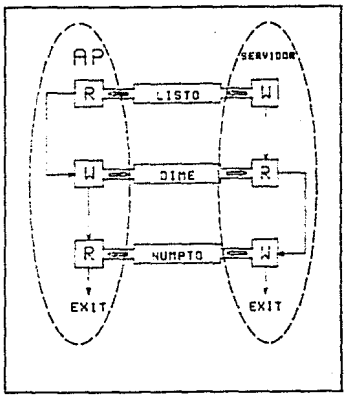
FIGURA 7.3
COMUNICACION ENTRE AP Y USUARIO

generalmente. Por ello, el AP debe mantener una tabla de mapeo que ayude en la conversión de los nombres de red a identificadores utilizados por el anfitrión local.

La manera usual de poner en estado activo a una tarea instalada en una computadora, es a través de llamadas al ejecutivo debidamente parametrizadas. En RSX-11M es necesario, entre otras cosas, especificar el identificador de la tarea instalada para que el sistema operativo se encargue de su activación, quién a su vez, le regresará el resultado de la operación a la tarea que emitió la solicitud.

Una vez activo el servidor este deberá solicitar a su estación de transporte que le asigne un puerto de comunicación hacia la red, y una vez otorgado, deberá decírselo al AP para que él a su vez se lo retransmita al proceso usuario que solicitó la activación del servidor. Pues bien, toda la comunicación que se suscita entre el AP y el SS se dá bajo un ambiente distinto al de red, denominado como comunicación entre tareas, controlada por el sistema operativo del anfitrión local. Aunque el tipo de directivas empleadas difieren sustancialmente de las de transporte, existe todo un protocolo de comunicación entre AP y SS denominado Protocolo Interno de Comunicación PIC, el cual se muestra en la figura 7.4.

Este protocolo debe ser respetado por todos los servidores de la red pues forma parte de su especificación formal. El conjunto de directivas que se emplean en la consecución del mismo conforman la ICTT. Estas directivas forman parte del ejecutivo, y son accesibles a las tareas mediante macro-llamadas que, al momento de



ANFITRION

FIGURA 7.4
PROTOCOLO INTERNO DE COMUNICACION

ensamblar la tarea, incluyen el código correspondiente. Esto, sin embargo, no implica ningún inconveniente para los diseñadores de procesos servidores, ya que se ha diseñado una función de alto nivel que ejecuta todo el PIC, de modo que sólo debe incluir el algoritmo de la figura 7.5 al principio de su código.

1. Solicitar un puerto de transporte.
2. Transmitirlo al AP ejecutando la función
   SndLog(numpto)
3. Activar las recepciones por numpto.
4. Esperar por la conexión en numpto.

Figura 7.5 Fase de Inicialización de un Servidor

A partir de este punto, el diseñador sólo deberá preocuparse por su propio código. Existe, sin embargo, una pequeña restricción actual, con respecto a la utilización de la función SndLog, y es que está contemplada para que sea usada solamente desde el lenguaje C, aunque esto se puede resolver fácilmente para cualquier otro lenguaje.

PROTOCOLO INTERNO DE COMUNICACION

Ahora tornaremos nuestra atención hacia el PIC. En esta discusión haremos total referencia al diagrama de la figura 7.6, que muestra la sección del PIC correspondiente al AP.

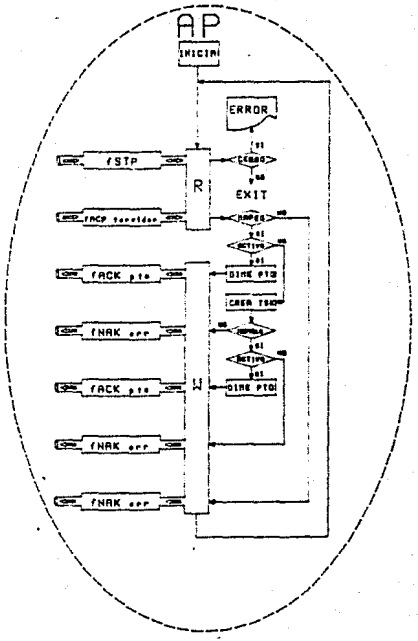


FIGURA 7.6
PCI DEL AP

El AP siempre que se encuentra ocioso permanece en estado de lectura R, permitiendo que cualquier proceso usuario remoto que desee conectarse pueda hacerlo. Para que un proceso usuario pueda solicitar la activación de un servidor es necesario que envíe un mensaje al AP con el formato especificado en la figura 7.7.

Este mensaje consta solo de los campos correspondientes al tipo de función y al nombre del servidor deseado. El nombre del servidor es único, es decir, no pueden coexistir dos servidores en una red con una misma denominación. Pero, ya que no existe (actualmente) un verdadero control sobre la asignación de nombres, puede ser que alguna otra aplicación haya bautizado a su servidor con un nombre ya existente, lo cual, llevaría a un conflicto. La solución actual es documentarlos para que se encuentren a disposición de los diseñadores. Pero, aún cuando se hiciera caso omiso de la prohibición de emplear nombres iguales, existen salvaguardas alternas que garantizan la supervivencia de la red: incompatibilidad de protocolos. Es decir, que si se llegase a establecer una conexión con algún servidor homónimo, es casi imposible que se llegaran a entender y esto se traduciría en una aborción eventual del enlace. Esta consideración permitio diseñar un PCI sencillo.

El AP utiliza el segundo campo para el mapeo del nombre generico, utilizando para ello la tabla de transformacion mostrada en la figura 7.8.

Cada uno de los renglones de la tabla posee tres campos que especifican la longitud máxima reconocible del nombre generico del servidor, el identificador de la tarea con el cual se encuentra

FIGURA 7.7
FORMATO DEL MENSAJE DE ACTIVACION

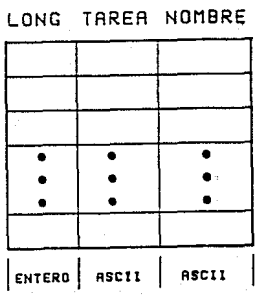


FIGURA 7.8
TABLA DE TRANSFORMACION

instalado el servidor en el anfitrión y el nombre genérico del servidor como se lo conoce en la red. Este arreglo permite soportar nombres genéricos de cualquier longitud. La búsqueda a través de la tabla es secuencial. Esta estructura de datos facilita grandemente la labor de implantación, aunque puede llegar a ser inoperable cuando se cuente con una gran cantidad de sistemas disponibles en la red.

Después del mapeo original es necesario efectuar algunas transformaciones adicionales para satisfacer las restricciones del sistema operativo.

A continuación vendría la activación de la tarea y en este punto podrían ocurrir dos cosas, que sea la primera vez que se va activar el servidor o, que ya existan una o más instancias activas de éste. En el primer caso se prosigue con la petición al ejecutivo para que cargue la tarea en memoria y la ejecute. Si la solicitud se cumplió exitosamente, se arranca el PIC para que el AP conozca el número de puerto que se asignó, éste se lo comunique al proceso usuario, y regrese a su estado inicial.

En caso de que hubiesen existido varias instancias activas, el AP crearía, entonces, un nombre nuevo bajo el cual pueda instalar otra instancia del servidor en las tablas del sistema operativo. Después habría solicitado la instalación de la tarea, solicitado su activación y, si tuvo éxito, arrancado el PIC para continuar la ejecución como en el caso anterior.

CAPITULO VIII

MANUAL DE USUARIO

En este capítulo describiremos la secuencia de pasos que puede tomar un usuario en su interacción con el STA, para efectuar tareas de transferencia de archivos en Red-IIMAS.

CONCEPTOS BASICOS

Empezaremos por presentar un conjunto de definiciones que nos ayudarán a entender cuáles son los tipos y sintaxis de datos con los que trata el STA.

archivo-de-red

Un archivo de red es una especificación formal de los archivos que pueden ser manipulados mediante el STA. Esta especificación guarda una estrecha relación con la especificación de archivos de RSX-11M. De hecho, su sintaxis sólo difiere en un campo adicional que maneja el STA y que no prevee RSX-11M: el anfitrión. La sintaxis de un archivo de red es la siguiente:

<anfitrión>[usuario]archivo

En la anterior especificación los signos <,>,:,[y] son obligatorios, es decir, forman parte de la sintaxis.

<anfitrión>

Todas las computadoras que se encuentran enlazadas a Red-IIMAS poseen un identificador que les permite ser reconocidas como miembros de la red. Este identificador es un entero decimal que se

le asigna a cada anfitrión al momento de instalación del 'software' de red y es único para cada computadora.

sintaxis: <entero-decimal>

ejemplo: 25

dispositivo:

Es la unidad de almacenamiento en la cual residen los archivos de red. Su especificación depende del RSX-11M local.

sintaxis: ddn:

donde: d es alfabética y n octal

ejemplo: SM0:

[usuario]

En todas las operaciones del STA se debe especificar el usuario de red con el cual se desea realizar una operación de transferencia de archivos. Este usuario mantiene una identificación o clave, cuya sintaxis es la misma que en RSX-11M. De hecho, un usuario de red debe serlo también de alguno de los anfitriones de Red-IIMAS, es decir, debe poseer una cuenta en la computadora. Ahora bien, ya que el STA permite ser utilizado en forma local, un usuario puede hacer uso de algunas de sus capacidades (correo electrónico, por ejemplo) pero, es claro que no podrá disfrutar de todas sus potencialidades. Para ello es necesario que mantenga también una clave en cada uno de los anfitriones con los cuales piensa establecer enlaces con sus servidores. Las cuentas en RSX-11M poseen la siguiente sintaxis:

[grupo,miembro]

donde: grupo y miembro son enteros octales o '*'

ejemplo: [2,7], [2,*], [*,*]

archivo

Este campo da la especificación de los archivos RSX-11M que se desean manipular.

sintaxis: nombre.ext;ver

donde: nombre y ext son alfanuméricos, o '*'

y ver es octal, nulo, 0, -1, o '*'

nulo: implica el más reciente

0: el más nuevo

-1: el más viejo

Así, una especificación total de un archivo de red podría verse de la siguiente manera:

IS:[DR0:[121,93STA.C;32

Valores Asumidos por Omisión

Los siguientes valores son asumidos cuando el usuario no ha especificado alguno de los parámetros:

PARAMETRO		VALOR
<anfitrión>	->	HOST (computadora local)
dispositivo:	->	dispositivo de la cuenta de usuario
grupo	->	grupo dado en HELLO (RSX-11M)

miembro	-	miembro dado en HELLO (RSX-11M)
nombre	-	*
ext	-	*
ver	-	*

ACTIVACION DEL SISTEMA DE TRANSFERENCIA DE ARCHIVOS

El STA es una tarea residente en la computadora, característica que la convierte en pública y que permite que cualquier usuario pueda solicitar su activación. El usuario debe entrar primero en sesión en la computadora a través del comando HELLO de RSX-11M, para luego activar el STA, por ejemplo:

>HELLO 2,7/PMG<cr>

donde > es el 'prompt' de la computadora,
2,7 es la cuenta del usuario,
PMG es el 'password' y
<cr> es 'return' (el cual ya no se volverá a especificar).

Después de haber efectuado este paso, el usuario se encuentra en posibilidad de iniciar una sesión de transferencia de archivos en Red-IIMAS. La manera de solicitar la activación del STA es:

>STA

El STA responderá de la siguiente manera:

Bienvenido a IIMAS-STA V1.0 (c) PM julio 1986
STA> EN-LINEA o BATCH [E/B]: _

La primer línea dá la bienvenida al usuario desplegando el número de versión del sistema y la fecha de actualización del mismo. PM

significa Pedro Márquez.

La segunda línea muestra el 'prompt' del STA y le pide al usuario elegir entre una sesion interactiva o no-interactiva, respectivamente.

Opción Batch

Si el usuario selecciona la opcion BATCH (B), el sistema responde:

STA> NOMBRE DEL ARCHIVO BATCH: _

y entra en un ciclo hasta que el usuario proporcione el nombre de un archivo válido, o hasta que se cumplan cinco intentos infructuosos y en tal caso se termina la ejecución total del sistema. Si se dió un nombre válido el STA procederá a procesar cada una de las instrucciones guardadas en él, hasta que se encuentra el comando de terminación de la sesión, (terminación normal) o llega al final del archivo y se termina la sesión anormalmente. En cualquiera de los casos se notifica al usuario de la condición de terminación en un archivo que registra la bitácora de la sesion en batch, denominado con el mismo nombre del archivo batch pero con extensión .LOG. En caso de que el STA no haya podido abrir este archivo no se llevará la bitácora de la sesión, pero si se realizará la sesion no-interactiva.

Opción En-Línea

Si el usuario elige, por el contrario, la opción de en-línea, el sistema responderá:

STA> USUARIO EXPERTO o NOVATO (E/N): _

preguntándole al usuario si se considera un usuario experto y por lo tanto desea realizar la sesión a través de comandos o, en caso de responder novato, preferir hacerlo mediante menues.

USUARIO NOVATO

Si el usuario elige novato, se iniciará una sesion interactiva apoyada en un esquema de menues, y pedirá al usuario que le indique el tipo de terminal que está empleando para cargar las secuencias de control adecuadas y hacer uso de las capacidades de graficación que posee esa terminal. Así, el sistema desplegará el siguiente menú:

STA> TERMINAL TIPO:

1. Heathkit
2. Datamedia
3. Teletipo

Opcion: _

y el usuario deberá elegir alguna de las tres opciones. La terminal tipo teletipo significa que la terminal no posee en absoluto ninguna capacidad de graficación o, simplemente, que no es ninguna de las anteriores y, que aún en este caso, se desea trabajar con menues.

PRIMER MENU DE FUNCIONES

Despues de elegido el tipo de terminal, el sistema cambia de

pantalla dando de nuevo la bienvenida al usuario, pero en esta ocasión bajo un nuevo modo de operacion y mostrando el primer menú de funciones. En todas las pantallas las únicas teclas de función que poseen denominación constante son las de colores (Heathkit), y su operación es la siguiente:

AZUL

Se utiliza para pedir la pantalla siguiente. Su operación es contextual, es decir, depende de la función actualmente en operación. Si el usuario oprime esta tecla estando en el último de los menues, sonará la campana indicando que ya no existen más pantallas que mostrar.

ROJO

Se emplea para pedir que se muestre la pantalla anterior. Su operación, lo mismo que en la azul, es contextual. Si el usuario oprime esta tecla estando en el primer menú, el sistema efectuará automáticamente la terminación de la sesión, desligándose del servidor al cual estaba conectado, si fuera el caso.

GRIS

Esta tecla permite solicitar ayuda acerca de la función actualmente en operacion o del menú mostrado.

FUNCIONES DEL STA

En todos los menues que maneja el STA se han reservado las teclas de función f1, f2, f3, f4 y f5 para las funciones y subfunciones de transferencia de archivos. El primer menú muestra las

funciones BORR, DIRE, HOLA, TRAE y CHIS.

Función BORR

El usuario debe seleccionar esta opción cuando desea BORRar archivos. Esta función posee dos modos básicos de operación: LOCL (LOCaL) y RMTO (RemoTO) accesibles a través de las teclas f1 y f2, respectivamente. En cualquiera de los casos, se pedirá al usuario que proporcione el nombre del archivo de red que desea borrar.

Función DIRE

Esta opción se emplea cuando se desea desplegar el DIREctorio de archivos de acuerdo a la especificación proporcionada por el usuario. El usuario puede elegir entre el directorio local o remoto.

Función HOLA

Esta función tiene dos propósitos: Como un mecanismo de ayuda al usuario que le permita inspeccionar el estado de operación del servidor, para casos en que sospeche que existe algo extraño en el comportamiento de la red (porque el tiempo de respuesta se haya degradado demasiado), y como verificación de que los números de versión de los sistemas usuario y servidor concuerdan, y se pueda estar seguro de que no existirán incompatibilidades que trastornen el funcionamiento de una sesión de transferencia de archivos. En esta función no se le pide información al usuario.

Función TRAE

Esta opción se utiliza cuando se desea TRAEr un archivo de red remoto, es decir, no hay manera de utilizar la función en modo local. Este archivo puede ser traído para ser guardado en disco, desplegado en pantalla, impreso o añadido a otro archivo. Esta última opción sólo opera cuando la especificación de archivo de red dada se refiere a un sólo archivo. El usuario elige la modalidad oprimiendo alguna de las teclas DSCO, TERM, IMPR o PEGA, respectivamente.

Función CHIS

Esta función permite enlazar dos terminales, de manera que sus usuarios pueden mantener una conversación (CHISme) en línea. El usuario puede elegir entre enlazarse a una terminal local o a una conectada en un anfitrión remoto.

SEGUNDO MENU DE FUNCIONES

Si el usuario hubiese oprimido la función azul estando en el primer menú de funciones, hubiésemos pasado al segundo donde es posible seleccionar las opciones LOGI, MAIL, RENM, MNDA y CONE.

Función LOGI

Esta función se utiliza para realizar LOGIn a Red-IIMAS, lo cual significa que estableceremos un enlace con el servidor de un anfitrión remoto. La idea es exactamente la misma cuando efectuamos 'login' en alguna computadora, es decir, validamos nuestro derecho de acceso a la computadora y establecemos los

privilegios del usuario. Pero, ademas de esto, la función de login del STA activa los mecanismos de transporte de la red para que se efectúe el enlace virtual entre un par de anfitriones. En forma similar a lo que ocurre con el comando HELLO de RSX-11M, la función LOGin del STA pide al usuario la identificación del anfitrion remoto, su numero de cuenta, y el 'password' correspondiente.

Función MAIL

Esta función nos permite activar el correo electrónico del STA. El correo puede operar tanto en modo LOCL como RMTO y, en cualquiera de los casos, posee las mismas capacidades. Estas capacidades permiten al usuario revisar la carta ANTErior, la SIGUiente, BORRar la carta actual o ENVIar una carta. El sistema de correo electrónico recuerda siempre cual fue la última carta revisada, de modo que las operaciones de revisión puedan tomar efecto. En el caso de envío de correspondencia se pide al usuario la identificación del usuario destino, junto con el 'password' de la cuenta asociada; como dispositivo de seguridad.

Función RENM

Esta función se utiliza para RENombrar archivos locales o remotos. El sistema pide al usuario la especificación del archivo de red.

Función MNDA

Esta función opera solo en modo RMTO y se utiliza para enviar

(MANDAr) archivos a través de la red. El envío puede ser hacia DISCO, TERMinal, IMPResora, o para ser PEGAdo a un archivo existente. En este último caso solo se admite que la especificación del archivo de red implique sólo uno, es decir, no se admiten 'wildcards'.

Función CONE

Cuando un usuario desea establecer el enlace con un servidor remoto de Fwd-UNAS debe realizar una CONExión con este. El STA pedirá entonces que identifique a la estación destino (anfitrión) donde reside el servidor deseado, e intenta establecer el enlace y la activación del servidor. Si todo funciona correctamente, el usuario estará en posibilidades de entrar en sesión con el anfitrión remoto.

TERCER MENU DE FUNCIONES

Existe un último menú principal de funciones que contiene únicamente una opción:

Función LIST

Esta función permite LISTar un archivo de red en la terminal del usuario. El sistema pide la especificación del archivo.

USUARIO EXPERTO

Cuando un usuario ya ha ganado suficiente experiencia con el STA y desea realizar su trabajo con mayor prontitud, puede elegir,

entonces, una sesión en-línea con el STA bajo la modalidad de usuario experto que le permitirá interactuar con el sistema solo a traves de comandos. Estos comandos facilitan la emisión de peticiones al sistema y, si el usuario posee cierta experiencia con RSX-11M, su aprendizaje será casi inmediato debido a la estrecha semejanza que existe con la sintaxis de la utileria PIP de RSX-11M. Bajo esta modalidad son igualmente explotables las capacidades de transferencia de archivos que posee el STA mediante la ejecucion de los siguientes comandos:

Comando ADI

El comando ADIos se emplea para terminar la sesión de tranferencia de archivos. Si el usuario habia establecido un enlace con un servidor remoto, la ejecución de este comando permite terminar normalmente el enlace y la sesión.

sintaxis: ADI
. (punto)

Comando AYU

Este comando nos permite solicitar AYUda acerca de la sintaxis de los comandos. A diferencia del esquema de ayuda que se tiene bajo la modalidad de menues, esta ayuda no pretende ser exhaustiva puesto que supone experto al usuario.

sintaxis: AYU
?

Comando HOL

Este comando es el equivalente exacto de la opción HOLA del esquema de menues.

sintaxis: HOL

Comando CON

Se emplea el comado de CONexion para establecer el enlace con un anfitrion remoto. Es el equivalente exacto de la opción CONE del esquema de menues.

sintaxis: CON <anfitrión>

Comando CHS

Este comando es el equivalente exacto de la opción CHIS del esquema de menues. Se utiliza para entablar un diálogo entre un par de terminales conectadas a computadoras de Red-IIMAS.

sintaxis: CHS <anfitrión>tty:

Comando LOG

Este comando se emplea cuando se desea realizar LOGin en un anfitrión remoto. Es el equivalente exacto de la opción LOGI del esquema de menues.

sintaxis: LOG <anfitrión>[grupo,miembro]/contraseña

Comando MAI

Este comando se utiliza cuando se desea activar el Subsistema de Correo Electronico (ver capitulo VI).

Comando TFR

Este comando se utiliza para efectuar todas las operaciones de Transferencia de archivos, aparca las modalidades local y remoto del esquema de nombres y posee dos sintaxis alternativas. La funcion que se desea ejecutar se especifica mediante opciones. Las sintaxis y funciones son las siguientes:

Sintaxis #1:

TFR <anfitrión>[grupo,miembro]nombre.extiver/opción

opcion /DE

Borra el archivo de red especificado, sin pedir confirmación.

opción /DR

Lista el directorio de red especificado.

opcion /LD

Borra el archivo de red especificado, con confirmación.

opción /LI

Lista en pantalla el archivo de red especificado.

opción /RE

Renombra el archivo de red especificado. El nombre al cual se desea cambiar es pedido posteriormente por el sistema.

Sintaxis #2:

TFR <a1>d1![g1,m1]n1.x1!v1=<a2>d2![g2,m2]n2.x2!v2/opción

opción /IM

Transfiere un archivo de red en modo IMagen (contiguo). El sistema verifica que las especificaciones sean semánticamente correctas para que tome lugar la operacion, es decir, sólo puede emplearse este comando cuando los archivos origen y destino definen archivos en disco. En caso de que ya exista el archivo destino se pregunta al usuario si desea sobreescribirlo.

opcion /CP

CoPia un archivo de red a otro archivo de red. De acuerdo a las especificaciones del archivo destino, la transferencia puede ser hacia una terminal, impresora o disco, exclusivamente. En caso de ya exista el archivo destino se pregunta al usuario si desea sobreescribirlo.

opción /AG

AGrega un archivo de red al final de otra archivo de red. En este comando también se realiza una verificación semántica de las especificaciones. Si no existe el archivo destino este es creado.

opción /SO

Copia un archivo de red a otra archivo de red SObreescribiéndolo, es decir, no verifica si existe ya un archivo destino bajo esta nombre.

Sumario de Comandos

ADI adiós (también se puede teclear .)

AYU ayuda (también se puede teclear ?)

```
HOL  hola
CON <estación>  conectarse con un anfitrion
CHS <estación>:tn:  enlazarse con una terminal
LOG <estación>:[grupo,miembro]/contraseña  login remoto
MAI activar el Subsistema de Correo Electronico
TFR <estación>:dd1:[grupo,miembro]nombre.extiver/opción
      donde /opcion puede ser:
           /DE  borrar sin confirmación
           /DR  directorio
           /LD  borra con confirmación
           /LI  listar
           /RE  renombrar
TFR <e1>:d1:[g1,m1]n1.x1;v1=<e2>:d2:[g2,m2]n2.x2;v2/opción
      donde /opción puede ser:
           /IM  copiar modo imágen
           /CP  copiar
           /AG  agregar
           /SO  copiar sobreescribiendo
```


CONCLUSIONES

La realizacion de esta tesis ha necesitado abordar diversos campos de la computación, que abarcan desde aspectos relacionados con la transmision de datos, hasta lenguajes de programación. En este sentido, fue un trabajo completo que requirio del conocimiento de diversas áreas y del dominio de otras.

Las áreas mayormente involucradas en el desarrollo de este trabajo fueron:

- Redes de Computadoras. Requerida en la aplicación de las técnicas de diseño de arquitecturas de redes, y en el diseño de los servicios, sistemas y funciones de la Capa de Aplicaciones.

- Programación de Sistemas. Utilizada en la realización de los 'handlers' de comunicación, empleados en las fases de prueba de los sistemas, antes de su integración a Red-IIMAS.

- Sistemas Operativos. Empleada en el desarrollo de los mecanismos de comunicación entre procesos , acceso a estructuras de datos, manipulación de archivos y definiciones del sistema.

- Compiladores. Utilizada en el diseño e implantación del 'parser' de comandos del STA y SCE.

- Procesamiento Distribuido. Empleado en el diseño e implantación de los protocolos de transferencia y mecanismos de sicronización entre procesos concurrentes.

- Programación Estructurada. Requerida para implantar un software

compacto y legible, que redujera sus requerimientos de memoria, incrementara su confiabilidad y facilitara la tarea de depuracion y modificaciones futuras.

- Especificacion formal, necesaria en la especificacion de los protocolos diseñados, de tal manera que permita su analisis e implantacion exacta, eliminado problemas de ambiguedad.

- Lenguajes de programacion, requerida en el diseño del lenguaje de comandos, y en la seleccion de un lenguaje de programacion adecuado para la implantacion de los sistemas.

Así, es válido considerar que uno de los beneficios obtenidos con la realizacion de un trabajo de está índole, es la posibilidad de conjuntar distintas disciplinas de la computacion en una sola, convirtiendola en una área completa, y que permite obtener diversas habilidades a las personas que incursionan en ella.

Sin embargo, este beneficio es sólo personal, y lo que puede ser realmente importante son las caracteristicas de los sistemas, que pueden ser vistas como logros obtenidos. Dentro de estas caracteristicas podemos destacar las siguientes:

El STA de Red-IIMAS:

- Extiende a nivel de la red, los servicios de manejo de archivos de los anfitriones, pudiéndose realizar entre anfitriones casi cualquier funcion soportada por éstos.

- Es un sistema compacto que ocupa 56 kb de memoria principal, lo cual no impone requerimientos severos de espacio en memoria.

- Posee distintos modos de operacion que le permite ser empleado por usuarios novatos o expertos.

- Mantiene interfaces hombre-maquina atractivas y de fácil manejo.

- Se encuentra estructurado en overlays, lo cual le permite extender fácilmente el numero de funciones y/o su sofisticación, sin incrementar (posiblemente) sus requerimientos de memoria.

- Permite realizar tareas adicionales a las de manejo de archivos, tales como, comunicación entre usuarios, sesiones remotas, conexión con servidores, verificación de estado y version, activación del correo electronico, y ayuda.

El SCE de Red-IIMAS:

- Permite transferir archivos-carta entre usuarios abonados a la red, proporcionando un servicio de correo electrónico.

- Es un sistema accesible desde el STA, lo cual le permite utilizar las facilidades que éste provee.

- Permitè realizar tareas de envio, despliegue y borrado de cartas para usuarios abonados en el anfitrion local o en uno remoto.

- Mantiene interfaces hombre-máquina atractivas y de fácil manejo.

- Se encuentra estructurado en overlays, lo cual le permite extender sus funciones fácilmente y sin aumento apreciable en sus requerimientos de memoria.

El AP de Red-IIMAS:

- Permite activar los servidores de la red y enlazarlos con los sistemas usuario.

- Proporciona funciones de alto nivel para el manejo de los protocolos de comunicación con los sistemas servidores de la red, la cual facilita su tarea de implantación a los diseñadores de servidores de red.

- Es el único sistema que requiere estar activo en todo momento, de modo que permite utilizar de forma más eficiente la memoria disponible.

Logros Adicionales:

- Se han desarrollado herramientas para el diseño de parsers de lenguajes de comandos, que pueden ser utilizados para distintos propósitos.

- Se han desarrollado funciones de alto nivel que permiten la conformación de mensajes de envío en base a primitivas de transporte.

- Se han desarrollado funciones de alto nivel para el manejo de terminales con capacidad de graficación.

Concluyendo, el diseño e implantación de los sistemas STA, SCE y AP de la Capa de Aplicaciones de Red-IIMAS ha permitido conjuntar distintas disciplinas de la computación, y se han obtenido avances importantes en el desarrollo de sistemas que presentan buenos características de servicio, tamaño, flexibilidad, completez y eficiencia, que han cumplido

ampliamente con los objetivos generales planteados para esta capa y para el Proyecto Red-IIMAS de Alta Velocidad. Además, estos desarrollos pueden permitir la creación de tecnología propia en el campo de las Redes de Computadoras y otras disciplinas que las emplean como herramienta de trabajo, como por ejemplo: Automatización de Oficinas, Procesamiento Distribuido, Bases de Datos Distribuidas, Control de Procesos, Correo Electrónico y Redes de Datos de Sistemas Integrados.

APENDICE A

TIPOS DE DATOS DE LOS PROTOCOLOS

TABLA DE FUNCIONES

FUNCION	VALOR	BYTES	DESCRIPCION
FABO	001	1	ABORT
FACK	002	1	ACKNOWLEDGE
FACP	003	1	ACTIVATE PROCESS
FAPN	004	1	APPEND
FASK	005	1	ASK
FAYT	006	1	ARE YOU THERE
FBCK	007	1	BACK
FBLK	010	1	BLOCK
FBYE	011	1	BYE
FCLS	012	1	CLOSE
FDAT	013	1	DATA
FDEL	014	1	DELETE
FEOT	015	1	END OF TRANSMISSION
FGET	016	1	GET
FINP	017	1	INITIATE PRINTING
FINS	020	1	INITIATE SENDING
FLOG	021	1	LOGIN
FMAI	022	1	MAIL
FMSG	023	1	MESSAGE
FNAK	024	1	NEGATIVE ACKNOWLEDGE
FPRI	025	1	PRINT

FREN	026	1	RENAME
FRJT	027	1	REJECT
FRTY	030	1	RETRY
FSND	031	1	SEND
FSTP	032	1	STOP
FSTX	033	1	START OF TEXT
FTER	034	1	TERMINAL
FTLK	035	1	TALK
FXST	036	1	EXIST
FNXT	037	1	NEXT

CAMPOS DE DATOS

CAMPO	BYTES	DESCRIPCION	TIPO
SERVER	6	nombre del servidor	alfanum
pto	2	número de puerto	entero
err	2	código de error	entero
uic	6	cuenta del usuario	alfanum
pasw	6	contraseña	alfanum
filnam	-	nombre RSX de archivo	alfanum
tt	2	número de terminal	octal
host	2	ident. anfitrión	entero
name	-	nombre del usuario.	alfanum
code	2	código de rechazo	entero
msg	-	mensaje	alfanum
filn	-	nombre RSX de archivo	alfanum

typ	1	tipo de archivo	byte
tfil	2	tamaño del archivo	entero
nblk	2	numero de bloque	entero
file	-	nombre RSX de archivo	alfanum
tblk	2	tamaño del bloque	entero
Iaa	-	anfitrión local	alfanum
blk	tblk	bloque de datos	byte
nl	2	numero de líneas	entero
nm	2	numero de mail	entero
ma	2	mail address	entero
mail	-	datos de la carta	alfanum
ETX	1	End of Text	alfanum
ferr	2	file error	entero
fname	-8	nombre arch. sin ext.	alfanum
fext	-3	extensión de arch.	alfanum
version	-	número de version	alfanum
Lhost	-	nombre del anfitrión	alfanum

OBSERVACIONES

1) Todos los campos tipo alfanum son delimitados al final por un caracter NULL (cero ascii).

2) Los campos tipo entero son enteros con signo.

3) Los campos tipo byte son campos sin contexto, es decir, pueden contener cualquier conformación de bits.

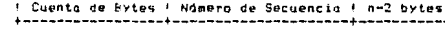
APENDICE B

FORMATOS DE ARCHIVOS

Los datos se transfieren entre dispositivos periféricos y memoria en bloques. Un archivo de datos consiste de bloques virtuales, cada uno de los cuales puede contener uno o más registros lógicos creados por un programa de usuario. En términos de FCS, un bloque virtual en un archivo consiste de 512 bytes. El tamaño de los registros lógicos en los bloques virtuales se encuentra bajo control del programa de usuario que escribió originalmente los registros.

Cuando se crea un nuevo archivo, un programa de usuario puede especificar que los registros en el archivo no necesitan ser todos del mismo tamaño. Estos registros se conocen como registros de longitud variable. Inversamente, si el programa de usuario indica que todos los registros en el nuevo archivo serán del mismo tamaño, se dice que son de longitud fija.

Existen dos tipos de registros de longitud variable: secuenciados y no secuenciados. Ambos deben estar alineados por palabras. Los registros secuenciados de longitud variable son precedidos por un encabezados de dos palabras. La primera palabra contiene la longitud del registro y la segunda el valor del numero de secuencia:



typ	1	tipo de archivo	byte
tfil	2	tamaño del archivo	entero
nblk	2	número de bloque	entero
file	-	nombre RSX de archivo	alfanum
tblk	2	tamaño del bloque	entero
lom	-	anfitrion local	alfanum
blk	tblk	bloque de datos	byte
nl	2	número de líneas	entero
nm	2	número de mail	entero
ma	2	mail address	entero
mail	-	datos de la carta	alfanum
ETX	1	End of Text	alfanum
ferr	2	file error	entero
fname	- 8	nombre arch. sin ext.	alfanum
fext	- 3	extensión de arch.	alfanum
version	-	número de version	alfanum
lhost	-	nombre del anfitrión	alfanum

OBSERVACIONES

1) Todos los campos tipo alfanum son delimitados al final por un caracter NULL (cero ascii).

2) Los campos tipo entero son enteros con signo.

3) Los campos tipo byte son campos sin contenido, es decir, pueden contener cualquier conformación de bits.

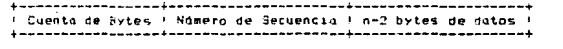
APENDICE B

FORMATOS DE ARCHIVOS

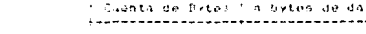
Los datos se transfieren entre dispositivos periféricos y memoria en bloques. Un archivo de datos consiste de bloques virtuales, cada uno de los cuales puede contener uno o más registros lógicos creados por un programa de usuario. En términos de FCS, un bloque virtual en un archivo consiste de 512 bytes. El tamaño de los registros lógicos en los bloques virtuales se encuentra bajo control del programa de usuario que escribió originalmente los registros.

Cuando se crea un nuevo archivo, un programa de usuario puede especificar que los registros en el archivo no necesitan ser todos del mismo tamaño. Estos registros se conocen como registros de longitud variable. Inversamente, si el programa de usuario indica que todos los registros en el nuevo archivo serán del mismo tamaño, se dice que son de longitud fija.

Existen dos tipos de registros de longitud variable: secuenciados y no secuenciados. Ambos deben estar alineados por palabras. Los registros secuenciados de longitud variable son precedidos por un encabezados de dos palabras. La primera palabra contiene la longitud del registro y la segunda el valor del número de secuencia:



Los registros no secuenciados de longitud variable son precedidos por un encabezado de registro de una sola palabra que contiene la longitud del registro:



Tanto los registros de longitud variable como fija se alinean en un límite de palabra. Cualquier byte extra que resulta de un registro de longitud impar simplemente se ignora. (El byte extra no necesariamente es cero).

Los bloques virtuales y los registros lógicos dentro de un archivo se numeran secuencialmente, cada uno comenzando en 1. Un número de bloque virtual es un valor relativo al archivo, mientras que un número de bloque lógico es un valor relativo al volumen. Ordinariamente, los registros pueden cruzar los límites de un bloque. Esto significa que el inicio de un registro puede llenar el final de un bloque mientras que el resto del registro ocupa el inicio del siguiente bloque.

Los bloques de los archivos se pueden asignar de dos formas: contiguos y no contiguos. Si el usuario especifica un número positivo de bloques al tiempo de creación del archivo, ese número de bloques serán alojados contiguamente y el archivo será contiguo. Si, por el contrario, especifica un número negativo, se alojarán el complemento a 2 del número especificado de bloques, no necesariamente contiguos, y el archivo será no contiguo. Si el usuario tiene una firme idea de la longitud deseada del archivo, es más eficiente alojar el número requerido de bloques al momento de creación del archivo, en lugar de que FCS extienda el archivo.

si es necesario, durante la escritura del mismo.

Si el usuario no especifica el parámetro de número de bloques entonces el archivo se crea como un archivo vacío, es decir, no se asigna ningún espacio dentro de él al momento de creación.

APENDICE C

GLOSARIO DE TERMINOS

ANFITRION. La colección de hardware y software que se enlaza a una red y la emplea para proporcionar comunicación entre procesos y servicios de usuario.

ANILLO. Una topología de red local en la cual las estaciones se enlazan a repetidores conectados en un lazo cerrado. Los datos son transmitidos en una dirección alrededor del anillo, y pueden ser leídos por todas las estaciones enlazadas.

ARBOL. Una topología de red local en la cual las estaciones se enlazan a un medio de transmisión compartido. El medio de transmisión es un cable ramificado que emana de una terminal cabeza, sin circuitos cerrados. Las transmisiones se propagan a través de todas las ramas del árbol y son recibidas por todas las estaciones.

ARQUITECTURA DE RED. El conjunto de capas y protocolos que respeta una red de computadoras. Ni la estructura interna de las capas ni las interfaces forman parte de la arquitectura.

ARQUITECTURA DE COMUNICACIONES. La estructura en hardware y software que implementa las funciones de comunicación.

BANDA BASE. Transmisión de señales sin modulación. En una red local de banda base las señales digitales se insertan directamente en el cable como pulsos de voltaje. La señal consume el espectro entero del cable. Este esquema no permite multiplexaje por división de frecuencia.

BANDA ANCHA. El uso de cable coaxial para proporcionar transferencia de datos mediante señales analógicas o de radio-frecuencia. Las señales digitales se pasan a través de un modem y se transmiten sobre una banda de frecuencia del cable.

BUS. Una topología de red local en la cual las estaciones se enlazan a un medio de transmisión compartido. El medio de transmisión es un cable lineal; las transmisiones se propagan por toda la longitud del medio y son recibidas por todas las estaciones.

CAPA DE APLICACIONES. Capa 7 del modelo OSI. Esta capa determina la interfaz del sistema con el usuario.

CAPA DE ENLACE DE DATOS. Capa 2 del modelo OSI. Convierte un canal de transmisión no confiable en uno confiable.

CAPA DE RED. Capa 3 del modelo OSI. Responsable del enrutamiento de los datos a través de una red de comunicaciones.

CAPA DE SESION. Capa 5 del modelo OSI. Maneja la conexión lógica (sesión) entre dos procesos o aplicaciones en comunicación.

CAPA FISICA. Capa 1 del modelo OSI. Relacionada con los aspectos eléctricos, mecánicos, y de tiempo de la transmisión de señales sobre un medio.

CAPA DE PRESENTACION. Capa 6 del modelo OSI. Relacionada con el formato y especificación de datos.

CAPA DE TRANSPORTE. Capa 4 del modelo OSI. Proporciona transferencia de datos confiable y transparente entre puntos terminales.

CIRCUITO VIRTUAL. Un servicio de conmutación de paquetes en la cual una conexión (circuito virtual) se establece entre dos estaciones al inicio de una transmisión. Todos los paquetes siguen la misma ruta, lo necesitan llevar una dirección completa, y arriban en secuencia.

CONMUTACION DE PAQUETES. Un método de transmitir mensajes a través de una red de comunicaciones en la cual los mensajes grandes se subdividen en paquetes cortos. Los paquetes son transmitidos entonces como en conmutación de mensajes. Generalmente, la conmutación de paquetes es más eficiente y rápida que la conmutación de mensajes.

CONMUTACION DE CIRCUITOS. Un método de comunicación en la cual se establece una trayectoria de comunicación dedicada entre dos dispositivos a través de uno o más nodos de conmutación. A diferencia de conmutación de paquetes, los datos digitales se envían como un flujo continuo de bits. Se garantiza el ancho de banda, y el retardo es esencialmente el tiempo de propagación.

CONMUTACION DE MENSAJES. Una técnica de conmutación empleando un sistema de mensajes guarda-reexpide. No se establece una trayectoria dedicada. Más bien, cada mensaje contiene una dirección destino y de la fuente al destino a través de nodos intermedios. En cada nodo, se recibe el mensaje entero, se almacena momentáneamente, y se pasa posteriormente al siguiente nodo.

CSMA. Carrier Sense Multiple Access. Una técnica de control de acceso al medio para un medio de transmisión de múltiples accesos. Una estación que desea transmitir observa primero el medio y transmite solo si éste se encuentra desocupado.

CSMA/CD. CSMA with Collision Detection. Un refinamiento de CSMA en la cual una estación cesa de transmitir si detecta una colisión.

DIFUSION. La transmisión simultánea de datos hacia todas las estaciones.

ENTIDAD. La instancia activa de una capa de la arquitectura.

ETHERNET. Una especificacion de red local de banda base de 10 Mbps desarrollada conjuntamente por Xerox, Intel, y Digital. Es la base del estandar IEEE 802.3 CSMA/CD.

FCCSO. Funciones de Control de Servicios del Sistema Operativo. Funciones que permiten solicitar al sistema operativo la ejecucion de diversas tareas de bajo nivel e interactuar estrechamente con el. La utilizacion de algunas de ellas puede requerir que el usuario sea privilegiado.

FTP. File Transfer Protocol. Protocolo de Transferencia de Archivos. Especificacion formal que permite la definicion, manejo y transmision de informacion conformada como archivos.

INTERFAZ. Especificacion formal de los mecanismos de enlace y comunicacion entre dos entes adyacentes.

ISO. International Standards Organization. Organizacion internacional dedicada a la generacion de estándares.

IU. Interfaz de Usuario. Conjunto de métodos y mecanismos de interacción del usuario con el STA para el uso de sus servicios. Incluye el Modo Novato, Modo Experto y Modo Batch.

MB. Modo Batch. Modalidad de operación del STA mediante la cual es posible proporcionarle un conjunto de comandos en un archivo, para la ejecución de una aplicación.

MSA. Módulo de Servicios de Archivos. Módulo del STA que se encarga de atender las solicitudes de manejo de archivos del sistema.

MC. Módulo de Comunicaciones. Módulo del STA que se encarga de la manipulación de la interfaz con la capa de transporte de Red-IIMAS.

ME. Modo Experto. Modalidad de operación del STA mediante la cual es posible solicitar sus servicios a través de comandos.

MEDIO DE TRANSMISION. Una trayectoria física entre transmisores y receptores en una red de comunicaciones.

MN. Modo Novato. Modalidad de operación del STA mediante la cual es posible solicitar sus servicios a través de menúes.

OSI. Open Systems Interconection. Interconexion de Sistemas Abiertos. Modelo de referencia para la creación de arquitecturas de redes.

PADDING. Técnica que permite rellenar con una cierta configuración de bits el espacio vacio de una unidad funcional de datos.

PAQUETE. Un grupo de bits que incluye datos mas direcciones fuente y destino. Se refiere generalmente a un protocolo de la capa de red.

PROTOCOLO. Un conjunto de reglas que gobiernan la operación de unidades funcionales para lograr la comunicación.

PROTOCOLO PEER-TO-PEER. Protocolo que permite la comunicación entre entidades de una misma jerarquía.

PUNTO-A-PUNTO. Una configuración en la cual dos estaciones comparten una trayectoria de transmision.

PUNTO DE ACCESO AL SERVICIO. Un medio de identificar un usuario de los servicios de una entidad de protocolo. Una entidad de protocolo proporciona uno o más puntos para que sean empleados por las entidades de capas superiores.

RED LOCAL. Una red de comunicaciones que proporciona interconexion de una variedad de dispositivos de comunicacion de datos dentro de un área pequeña.

RED DE AREA LOCAL. Una red local de proposito general que puede servir a una variedad de dispositivos. Empleada típicamente para terminales, microcomputadoras y minicomputadoras.

SCE. Subsistema de Correo Electrónico. Subsistema del STA que permite el envio, borrado y despliegue de cartas a través de Red-IIMAS.

SISTEMA ABIERTO. Dícese de la facultad que posee un sistema de comunicarse con cualquiera otro que respete los mismos principios de diseño.

SMA. Sistema de Manejo de Archivos. Sistema que se encarga de la administracion de los archivos de una computadora.

SMAR. Sistema de Manejo de Archivos de la Red. Sistema que se apoya en los SMA para administrar el manejo de los archivos a traves de Red-IIMAS. Extiende el concepto del SMA al nivel de red.

SS. Subsistema Servidor. Subsistema del STA que proporciona los servicios en conjunción del SU y bajo coordinacion expresa de este.

STA. Sistema de Transferencia de Archivos. Protocolo de transferencia de archivos de Red-IIMAS. Posee además un Subsistema de Correo Electronico.

SU. Subsistema Usuario. Subsistema del STA que interactua directamente con el usuario para proporcionar servicios. Incluye la IU y el SS.

TOPOLOGIA. La estructura, consistente de trayectorias y conmutadores, que proporciona la interconexión de comunicaciones entre nodos de una red.

TRANSMISION SINCRONA. Transmisión de datos en la cual el tiempo de

ocurrencia de cada señal que representa un bit se encuentra relacionado a un marco de tiempo fijo.

TRANSMISION ASINCRONA. Transmision en la cual cada caracter de informacion se sincroniza individualmente (generalmente mediante el empleo de elementos de inicio y terminación).

TRANSRECEPTOR. Un dispositivo que transmite y recibe.

VTP. Virtual Terminal Protocol. Protocolo de Terminales Virtuales. Permite la utilizacion de cualquier tipo de terminal para acceder a los servicios de una red. Existe una negociacion inicial para la definicion de las capacidades de las terminales.