ARQUITECTURA DE APLICACIONES EN E-BUSINESS

INDICE

ARQUITECTURA DE APLICACIONES EN E-BUSINESS

(Versión Preliminar)

1. Introducción

La estructura física de la red Internet, el software que se ha provisto para operar sobre ella y las herramientas que existen para desarrollar aplicaciones e-business que funcionan en tal medio determinan la arquitectura de éstas. Por ello entregamos aquí un resumen de estos elementos determinantes y establecemos las arquitecturas típicas que puede tener una aplicación de tal tipo.

Nos centramos en la situación más compleja de aplicación e-Business, en la cual no sólo existe un sitio web que provee algún tipo de producto o servicio, sino que consideramos que también es necesario un "back office" que se preocupe de satisfacer los requerimientos de los usuarios –con posibles procesos de evaluación de clientes, evaluación de satisfacción, generación y/o entrega de los productos o servicios, etc.– y que es posible que se den relaciones complejas con proveedores y clientes del tipo de integración de la cadena de abastecimiento.

2. La Estructura de Internet

La estructura de Internet la podemos visualizar a diferentes grados de agregación para un mejor entendimiento. Empezamos con una visión general, que se muestra en la Figura 2.1. Allí se presentan, en primer lugar, los ISP (Internet Service Providers) que son la puerta de entrada a la red Internet. Ellos tienen los accesos y conexiones, los cuales se detallarán más adelante, que permiten a un usuario –persona o empresa– establecer comunicación con cualquier otro usuario en la red. Los usuarios pueden ser clientes –en el sentido de un equipo cliente individual– conectados por modem o en forma dedicada, habitualmente desde el hogar, con cable o RDSI, o empresas que tienen una red interna que se acopla a Internet.

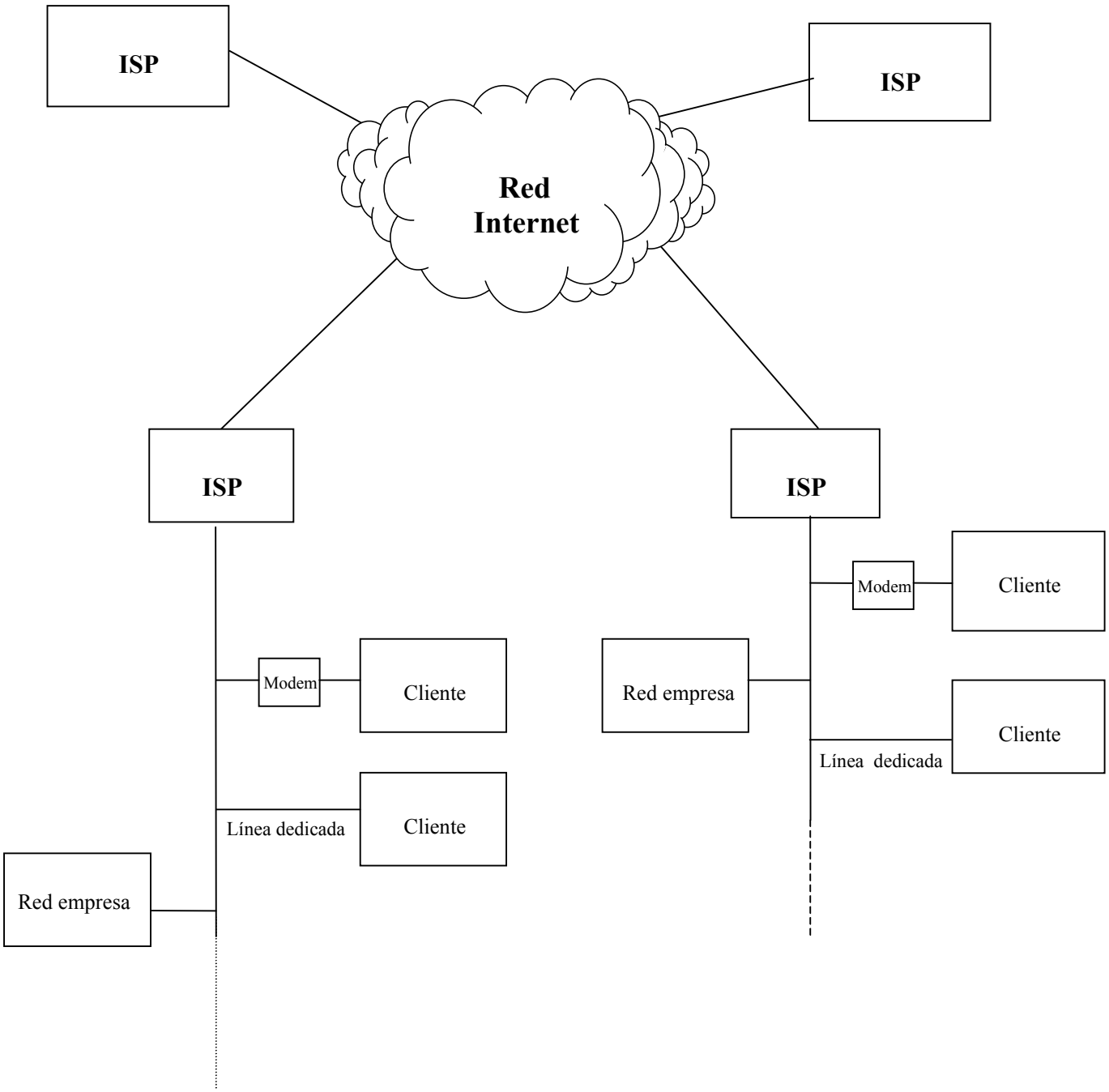


Figura 2.1. Estructura general Internet

Mirando más en detalle la red Internet, encontramos la estructura que se muestra en la Figura 2.2. La red que se muestra es una simplificación de la que une las ciudades más importantes de EE.UU. Para mostrar más detalle aún de esta estructura, consideremos un caso particular: la red vBNS (very-high-performance Backbone Network Service) que une varias instituciones académicas, del gobierno y de la industria en EE.UU [1]. Un nodo de acceso típico de tal red tiene la estructura que se muestra en la Figura 2.3. Esta red tiene en su nivel más bajo –nivel 1 de OSI– un cable de fibra óptica sobre el cual funciona una red óptica síncrona (SONET: Synchronous Optical Network). Sobre SONET funciona el protocolo ATM (Asynchronous Transfer Mode) y encima de éste el protocolo IP de Internet (nivel 3 de OSI). Es interesante hacer notar que esta red vBNS está, a su vez, montada sobre la red ATM comercial Hyperstream de la empresa MCI, un carrier de EE.UU. En la red MCI se define un conjunto de caminos virtuales (virtual paths) permanentes que transporta el tráfico vBNS. Cada camino, a su vez, transmite una agrupación de circuitos virtuales. Los switches dirigen el tráfico ATM sobre los caminos y los router IP enrutan los mensajes (paquetes) sobre los circuitos. Los dos tipos de router se explican por diferentes velocidades de enrutamiento: uno de ellos es para velocidad de super computadores. La plataforma de prueba de desempeño monitorea el tráfico y el desempeño para asegurar un nivel de servicio.

En todo el mundo existen redes que permiten el acceso a Internet, que son versiones menos avanzadas que vBNS –un modelo al cual tendería la tecnología–, pero que tienen componentes y funcionalidades similares.

Veamos, ahora, la estructura de la parte de Internet de más interés para la arquitectura de aplicaciones: las redes IP de las empresas. De nuevo tomamos el caso de mayor complejidad para identificar la mayor parte de los componentes que participan. Una estructura típica se muestra en la Figura 2.4.

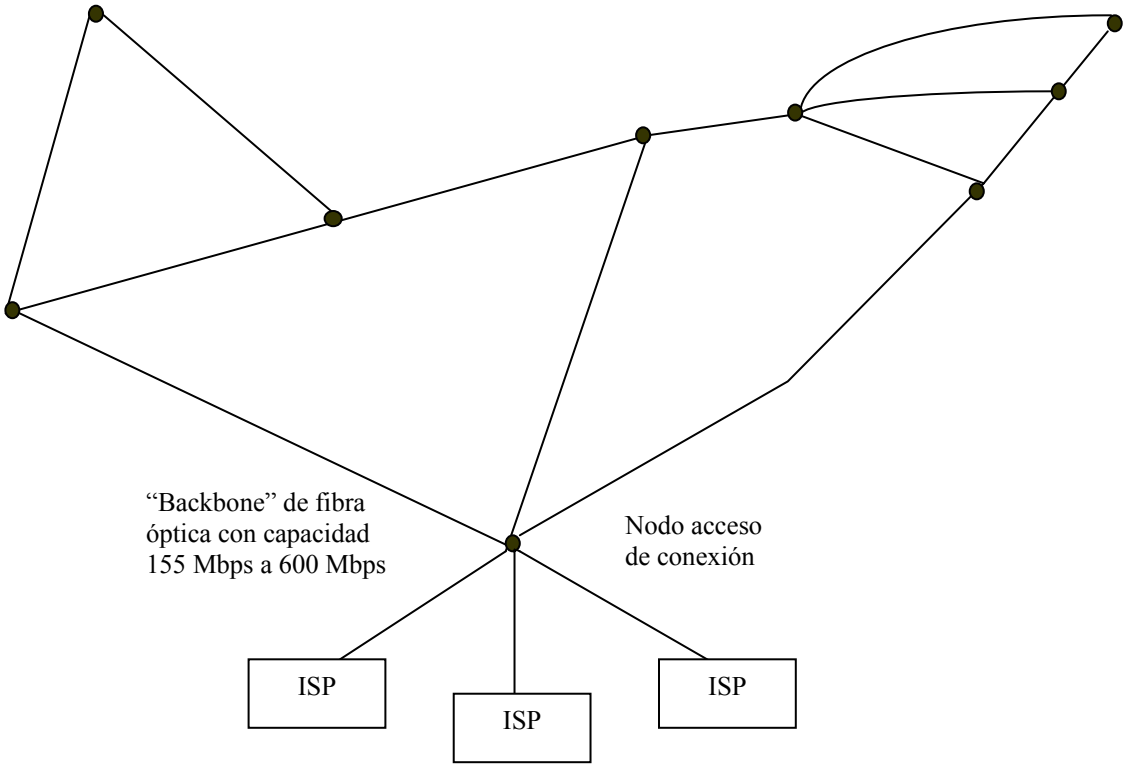


Figura 2.2. Estructura red Internet

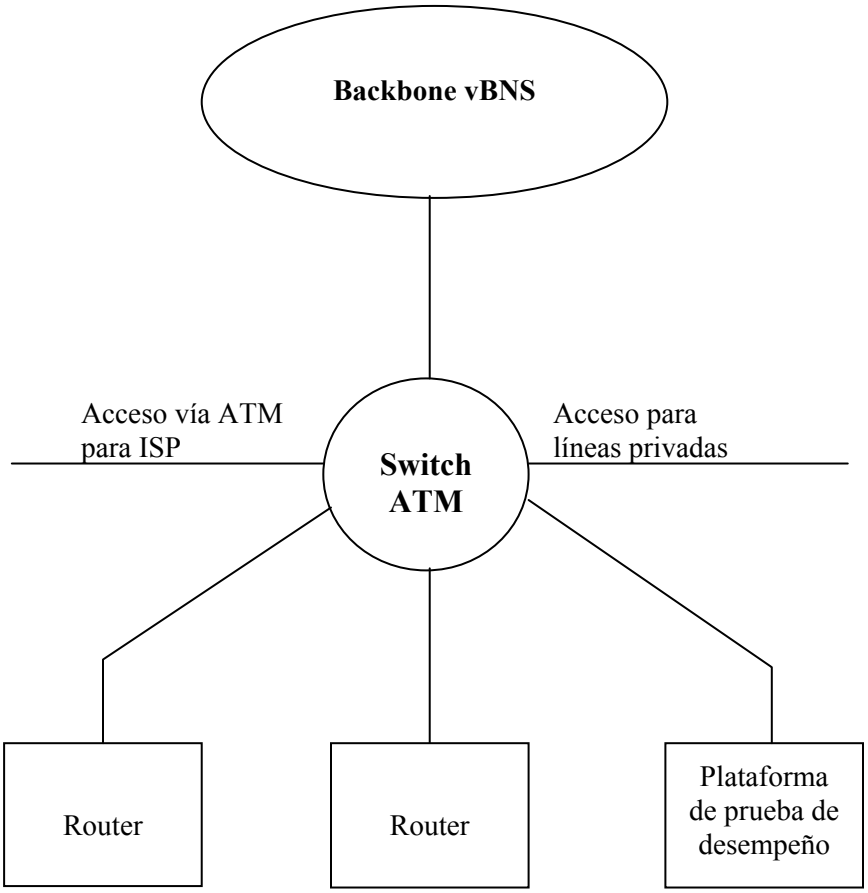


Figura 2.3. Nodo típico de acceso de la Red vBNS

El primer componente que se encuentra al ingresar a la red es un router, que realiza un filtraje primario y secundario de los mensajes, identificando aquellos que presentan potenciales problemas de seguridad, ensamblando paquetes y enrutándolos al destino apropiado dentro de la red de la empresa. Las empresas pueden tener servidores Web públicos, que permiten el acceso irrestricto de clientes que pasan por el router, a sitios que, fundamentalmente, proveen información. Cuando el cliente quiere realizar una transacción o acceder a información restringida, debe pasar a través de un conjunto de cortafuegos que permiten sólo ingreso autorizado. En el gráfico se incluye un cortafuego distribuidor de carga y dos distribuidos. Aquí la idea es que cada cortafuego desempeña una tarea específica; por ejemplo, ciertos tipos de autorización, autentificación y control de acceso. Los mensajes que pasan los cortafuegos tienen acceso a los servidores Web no públicos, los de lógica y de otro tipo –por ejemplo, correo, groupware, etc.– donde se entrega la funcionalidad solicitada por el cliente. Todas las comunicaciones entre servidores y con otros elementos son mediadas por el switch LAN. Nótese la existencia de un servidor de administración de sistemas, encargado de medir tráfico, identificar cuellos de botella, asignar recursos y otras tareas de administración de la red; de un servidor de autorización que controla los accesos –particularmente, a la base de datos– basados en reglas del negocio que señalan qué información está disponible para quién; y un servidor de bases de datos, que puede corresponder a información especialmente almacenada para apoyar los servicios Web o a sistemas "legacy".

Por último, existe una red corporativa interna que también puede tener acceso a través de una Intranet a la información en los servidores, para lo cual también hay un router y cortafuego interno que rutean y controlan los accesos. El propósito de esta red interna puede ser apoyar los procesos necesarios para satisfacer los requerimientos de los clientes, ya mencionados anteriormente, que tienen que ver con generar el producto o servicio, realizar actividades logísticas necesarias y asegurar la satisfacción del cliente. Por lo tanto, si detalláramos tal red aparecerían servidores Web, de lógica y otros elementos orientados a apoyar clientes internos.

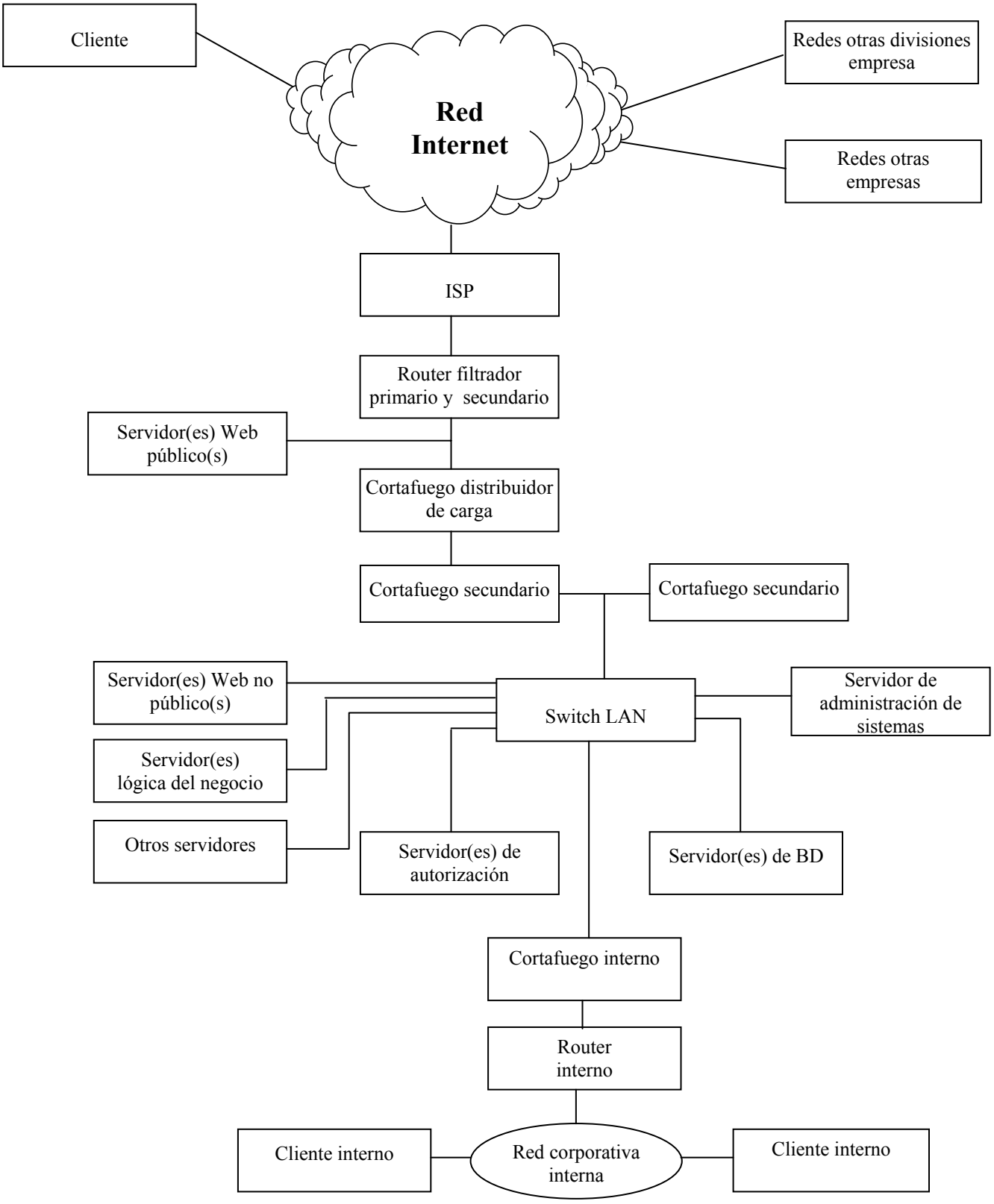


Figura 2.4. Red IP típica de una empresa

La red bosquejada estará típicamente montada sobre un medio físico como Ethernet. Alternativas relevantes son ATM y "frame relay". La disponibilidad de Gigabet Ethernet la hace hoy día la alternativa más popular.

Damos un caso real de red IP conocida que ilustra las ideas anteriores. Se trata de la empresa Sigma Aldrich, que ganó un premio por tener una buena solución e-business [2], la cual provee insumos químicos para laboratorios de investigación. La red se resume en la Figura 2.5, junto con las funcionalidades que provee a los clientes. Nótese que contiene un subconjunto de la red típica de la Figura 2.4, debido a que la red corporativa interna de satisfacción de pedidos no está detallada.

Es importante destacar que la red de la Figura 2.4. se puede replicar en otras divisiones de la misma empresa y en otras empresas con la cual la red en cuestión interactúa. Esto significa que relaciones complejas con proveedores o con empresas a las cuales se distribuyen los productos –al estilo de una cadena de abastecimiento extendida– posiblemente implicarían colaboración entre un cliente interno en la red en cuestión y un cliente interno de otra división o empresa a través de todos los elementos de la misma, como se ejemplificará más adelante.

3. Cliente/Servidor en Internet

Sobre la estructura de Internet bosquejada en el punto anterior se pueden montar variadas arquitecturas de aplicaciones para desarrollar ciertas funciones dentro de un negocio o grupo de negocios. La más simple de estas arquitecturas corresponde a la tradicional solución Cliente/Servidor (C/S), que históricamente se dio en otro tipo de plataformas. Examinamos, a continuación, las diferentes variedades de C/S que pueden implementarse en Internet.

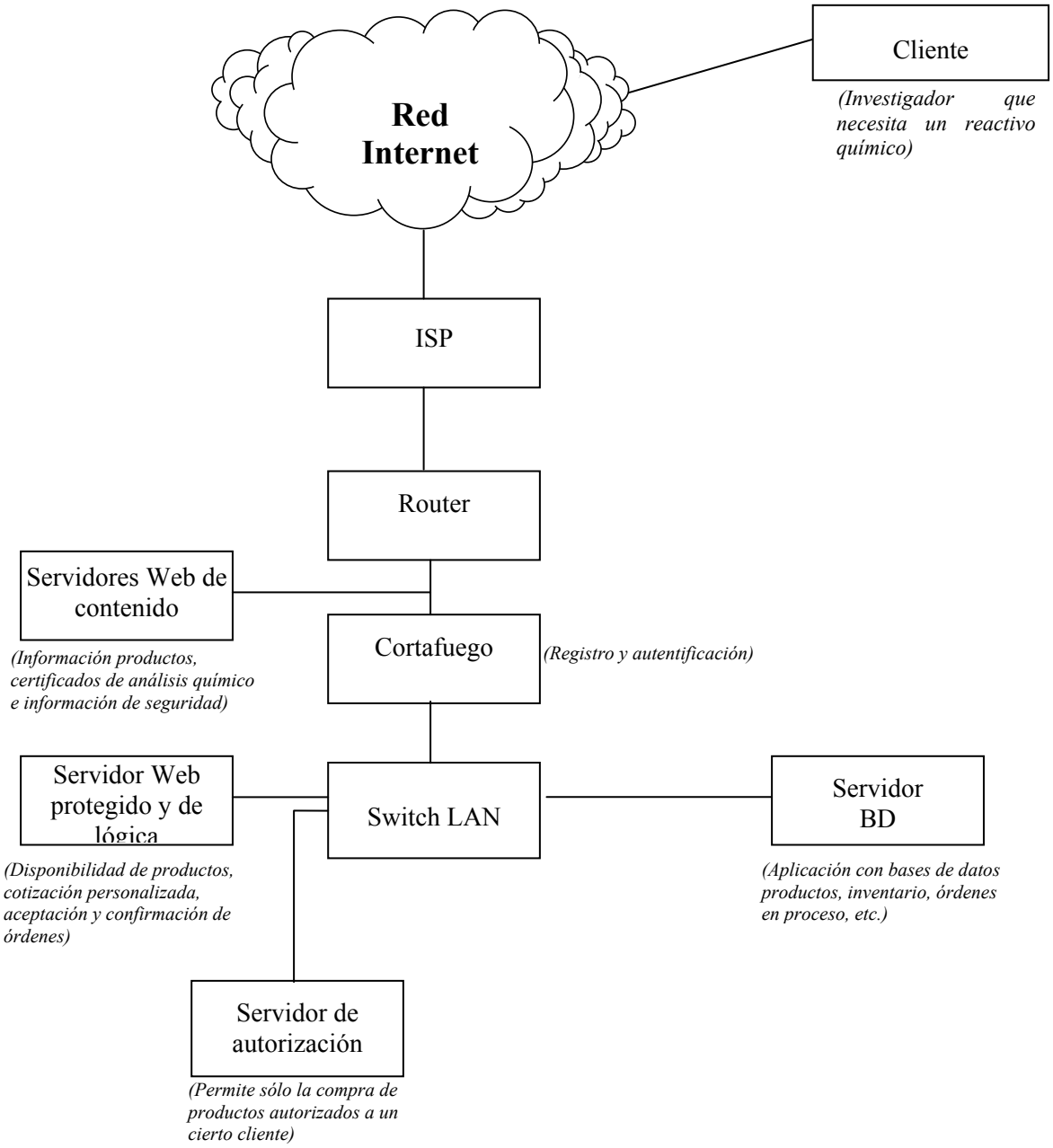


Figura 2.5.Red IP de Sigma Aldrich

3.1. Cliente/Servidor de dos capas

La más simple de las aplicaciones C/S en Internet es la dos capas. Ella se puede implementar en versiones de Extranet e Intranet.

La versión Extranet, que es la más popular en comercio electrónico B2C simple, consiste en un cliente, que corresponde al equipo desde el cual un usuario externo a la empresa requiere algún tipo de servicio, y un servidor Web que lo satisface integralmente. Gráficamente, la situación es la que se muestra en la Figura 3.1. En ella se ha incorporado, para el caso más general, los elementos intermedios que permiten la conexión entre cliente y servidor.

La funcionalidad que puede proveer una solución de este tipo es baja, ya que se reduce a lo que se puede ejecutar con las páginas estáticas y dinámicas que se puedan ofrecer a través del sitio. Al nivel más elemental, se pueden ofrecer páginas que contengan el catálogo de productos y/o servicios de la empresa, de tal manera de dar a conocer la oferta de la empresa. Con algún esfuerzo, utilizando páginas dinámicas, se podría ofrecer una funcionalidad de cotización, siempre que la información de precios pudiera mantenerse en tales páginas. La posibilidad de transacciones es remota, excepto en casos en que el producto o servicio es trivial o se transfiere la transacción a un "back office". Por ejemplo, un sitio de información –noticias, artículos técnicos, enciclopedia, etc.– que a través de un buscador permite encontrar un ítem de interés; un crédito de consumo que se cotiza en un sitio web financiero y que es transferido a un banco que continúa su procesamiento; o una compra de un producto a través del sitio de una empresa, que después de cotizado y pagado con tarjeta de crédito, se transfiere a procesamiento manual para confirmación de disponibilidad y entrega.

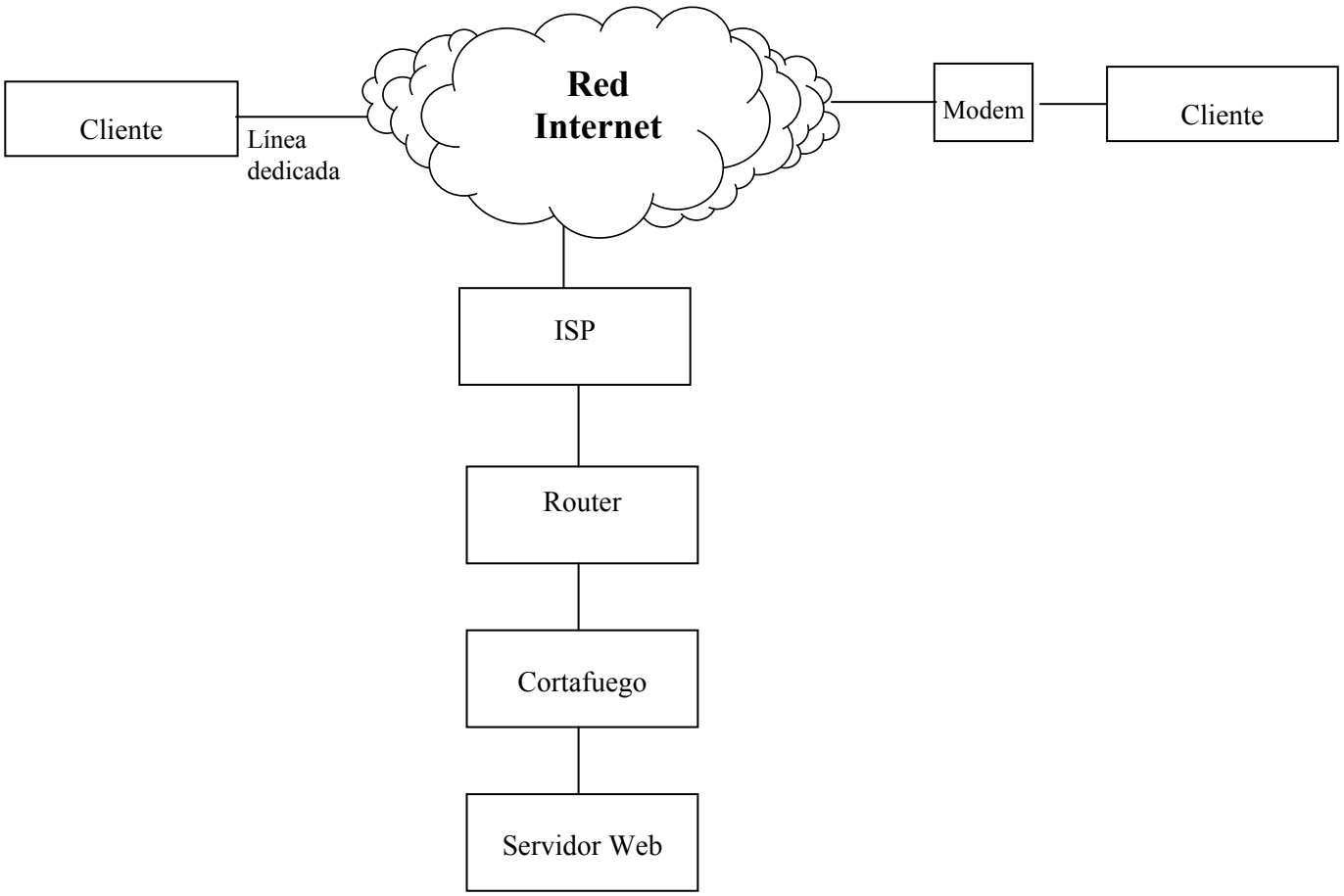


Figura 3.1. C/S dos capas tipo Extranet

Este mismo tipo de solución se da en una modalidad Intranet, donde el servidor Web existe para proveer servicios internos a los empleados de una empresa. La representación gráfica es la que se muestra en la Figura 3.2. En esta arquitectura los servicios están orientados sólo al personal de la empresa, que puede acceder directamente al Servidor Web desde una red en la matriz de la empresa o por Internet, desde la red de una división de la empresa que se encuentra en otra ubicación física. Se incluyen routers y cortafuegos para el caso más general de este tipo de solución.

La funcionalidad que una arquitectura de este tipo puede proveer es de información interna susceptible de ser publicada en páginas Web. Por ejemplo, catálogos de productos, especificaciones técnicas, normas y procedimientos, beneficios ofrecidos al personal, noticias, oportunidades laborales dentro de la empresa, etc.

Las dos soluciones bosquejadas se pueden implementar en modalidad cliente "delgado" y cliente "grueso". En el primer caso, el cliente requiere sólo un browser a través del cual se solicitan servicios y toda la funcionalidad es computada y provista por el servidor, y el browser sólo despliega las páginas. En el segundo caso el cliente pide servicios a través del mismo browser y recibe de vuelta una applet o algún otro programa que se ejecuta en el browser. Para ello, éste debe tener el software adecuado para la ejecución: una máquina virtual Java, en el caso de una applet.

Por supuesto, hay soluciones intermedias a las de clientes delgado y gordo, como, asimismo, se pueden dar mezclas de soluciones tipo Extranet e Intranet.

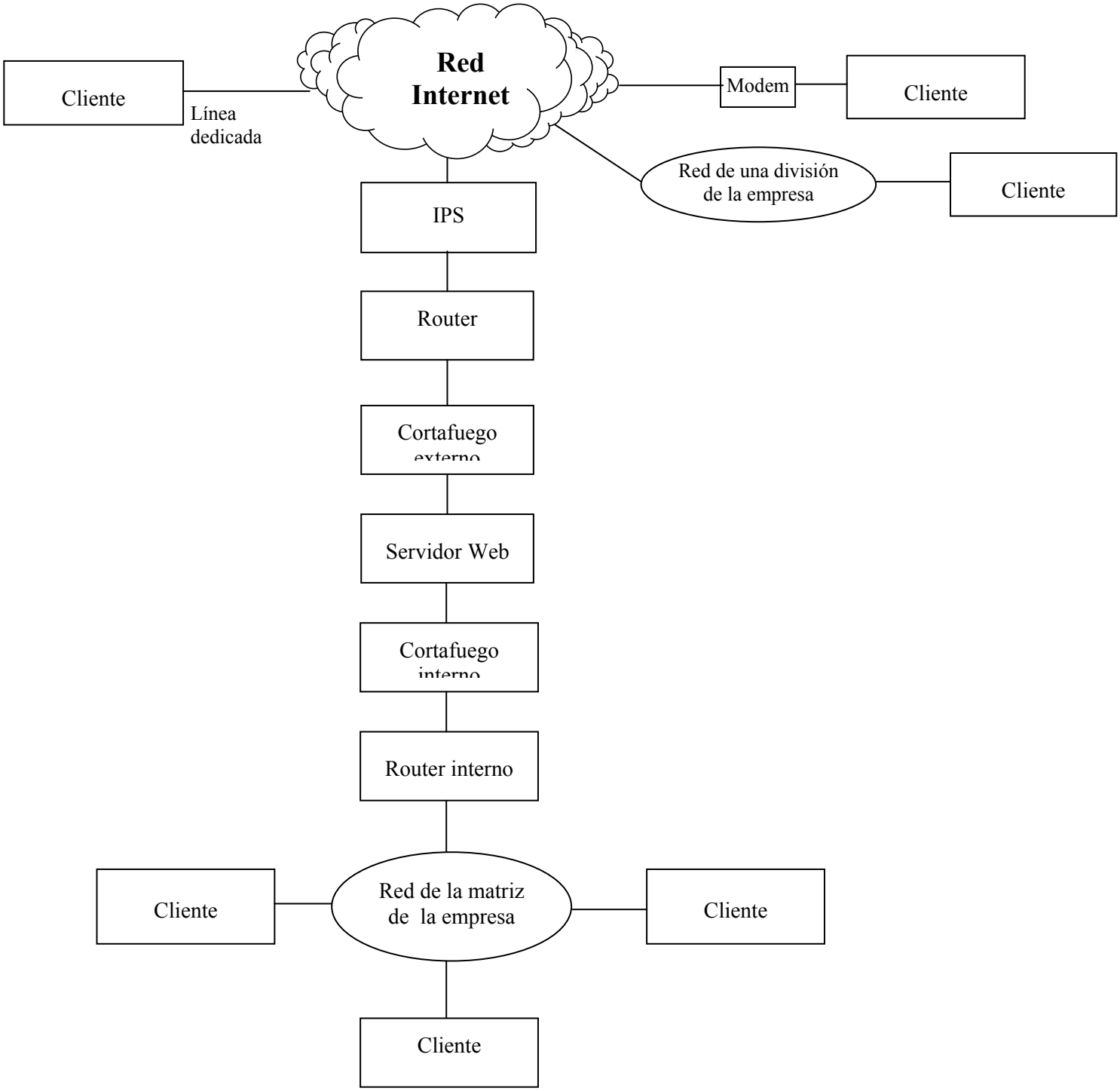


Figura 3.2. C/S dos capas tipo Intranet

3.2. Cliente/Servidor de tres capas

La arquitectura de tres capas nace de la necesidad de proveer mayor funcionalidad, sobre todo cuando se requiere procesar automáticamente las transacciones por productos o servicios solicitados por los clientes en una típica aplicación e-Business. Para ilustrar esta idea, consideremos el caso de Amazon.com. En la Figura 3.3 se muestra gráficamente la funcionalidad requerida, en forma simplificada. Lo relevante para nosotros es que las actividades 1 y 3 son totalmente automatizadas. Esto implica que el flujo "Mensaje entrega" lo genera un computador –que previamente ha determinado un punto de distribución, de entre los muchos que tiene esta empresa, que satisfará el pedido– por medio de encender luces en los lugares de la estantería del punto de distribución donde se encuentran los productos que solicita un cliente. Es evidente que la lógica a implementar, en este caso, para verificar disponibilidad en diferentes puntos de venta, elegir uno y, si no hay, buscar alternativas, es compleja. Además, se requieren datos tales como inventario de los productos en los diferentes puntos de distribución, costo de transporte desde diferentes puntos de distribución a un destino y varios más, los cuales, típicamente, deberían estar en bases de datos relacionales tradicionales –por sus características técnicas– y no en páginas. Por lo tanto, se requieren: un sitio web para las páginas que ofrecen los servicios, un servidor para ejecutar la lógica del negocio necesaria para satisfacer un servicio y un servidor de datos, lo cual conforma, en general, la arquitectura que se muestra en la Figura 3.4. La primera capa corresponde a los equipos clientes que, con un browser, acceden a los servicios de la aplicación. La segunda capa incluye el servidor que mantiene el sitio Web al cual acceden los clientes y el servidor de lógica que implementa las reglas del negocio que están detrás del sitio. Por último, la tercera capa corresponde al servidor de datos, desde donde se obtiene la información para implementar la lógica.

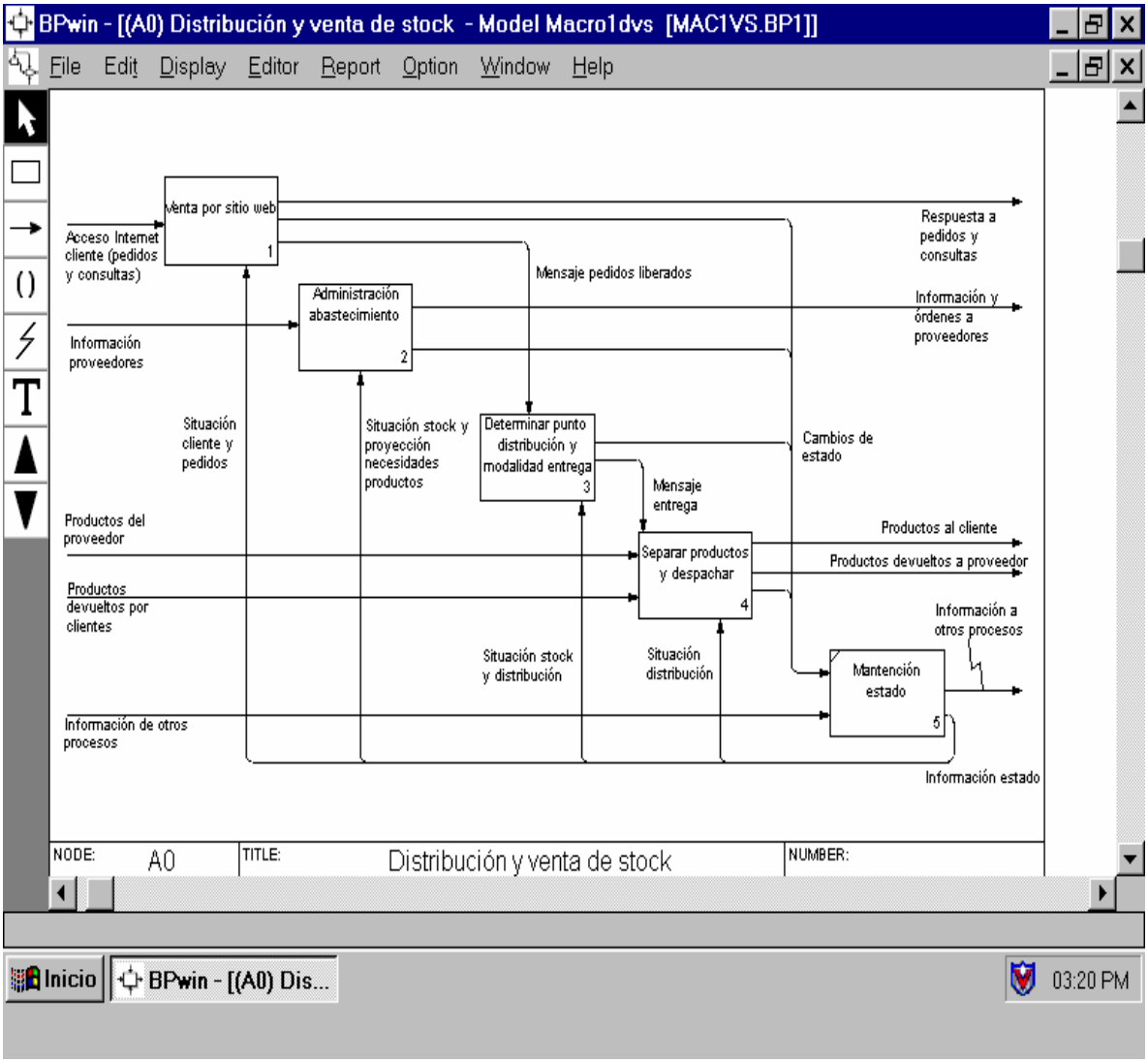


Figura 3.3. Modelo simplificado funcionamiento Amazon.com

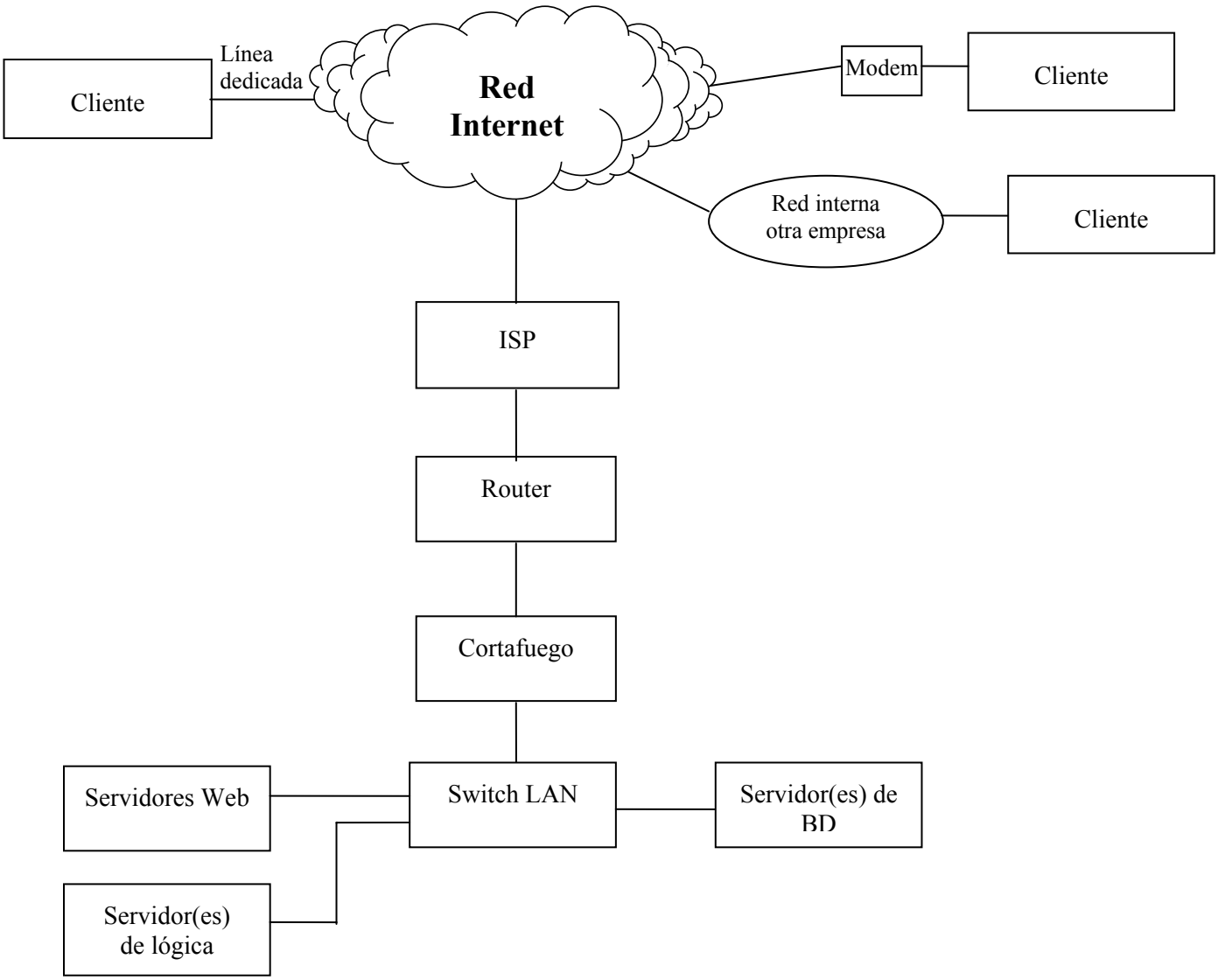


Figura 3.4. C/S tres niveles en Internet

Una aplicación interesante de la arquitectura de 3 niveles se da en la relación B2B entre dos empresas: una demandante y otra proveedora. El manejo moderno de esta relación –en la idea de integración de la cadena de abastecimiento mediada por Internet– apunta a que el demandante explicita sus necesidades por productos, insumos u otro –por medio de pronósticos de consumo, planes de producción a alguna otra información que permita conocer anticipadamente la demanda– y que el proveedor asuma un rol activo en la provisión "just in time" de lo que el demandante requiere. Esto hace necesaria la integración de las redes de ambos participantes a través de Internet, de la manera que se muestra en la Figura 3.5. La idea fundamental de esta arquitectura es que el personal autorizado del proveedor –cliente en la red del proveedor– tenga acceso –con la debida seguridad, apoyada en un servidor de autorización– a páginas en el servidor Web del demandante y a las rutinas del de lógica –alimentada con accesos al servidor de Base de Datos– que le provean la información acerca de las necesidades proyectadas de productos, insumos u otro. Con esta información se gatilla la acción de varias actividades en el proveedor –planificación de producción, procesamiento de órdenes, despacho, logística, etc.–, representadas como cliente en la red del proveedor, con el fin de satisfacer tales necesidades. Por supuesto, los servidores del proveedor apoyan con páginas, lógica y datos tal acción. Asimismo, a través de tales servidores e Internet se comunican las acciones relevantes –despacho, por ejemplo– al demandante, el cual, a través de sus servidores, toma acciones internas en respuesta a lo hecho por el proveedor –recepción de productos por ejemplo. Por otro lado, los clientes del demandante, que representan –en este caso– las diferentes actividades usuarios de los servidores del proveedor, tienen la responsabilidad de mantener actualizada, en el Servidor de Base de Datos, toda la información que requiere el proveedor para proyectar las necesidades del demandante. Asimismo, toman acciones –que también son apoyadas por la red interna y los correspondientes servidores– para recepcionar y mover internamente los productos, insumos u otro entregados por el proveedor y proceder a su uso en la operación de la empresa.

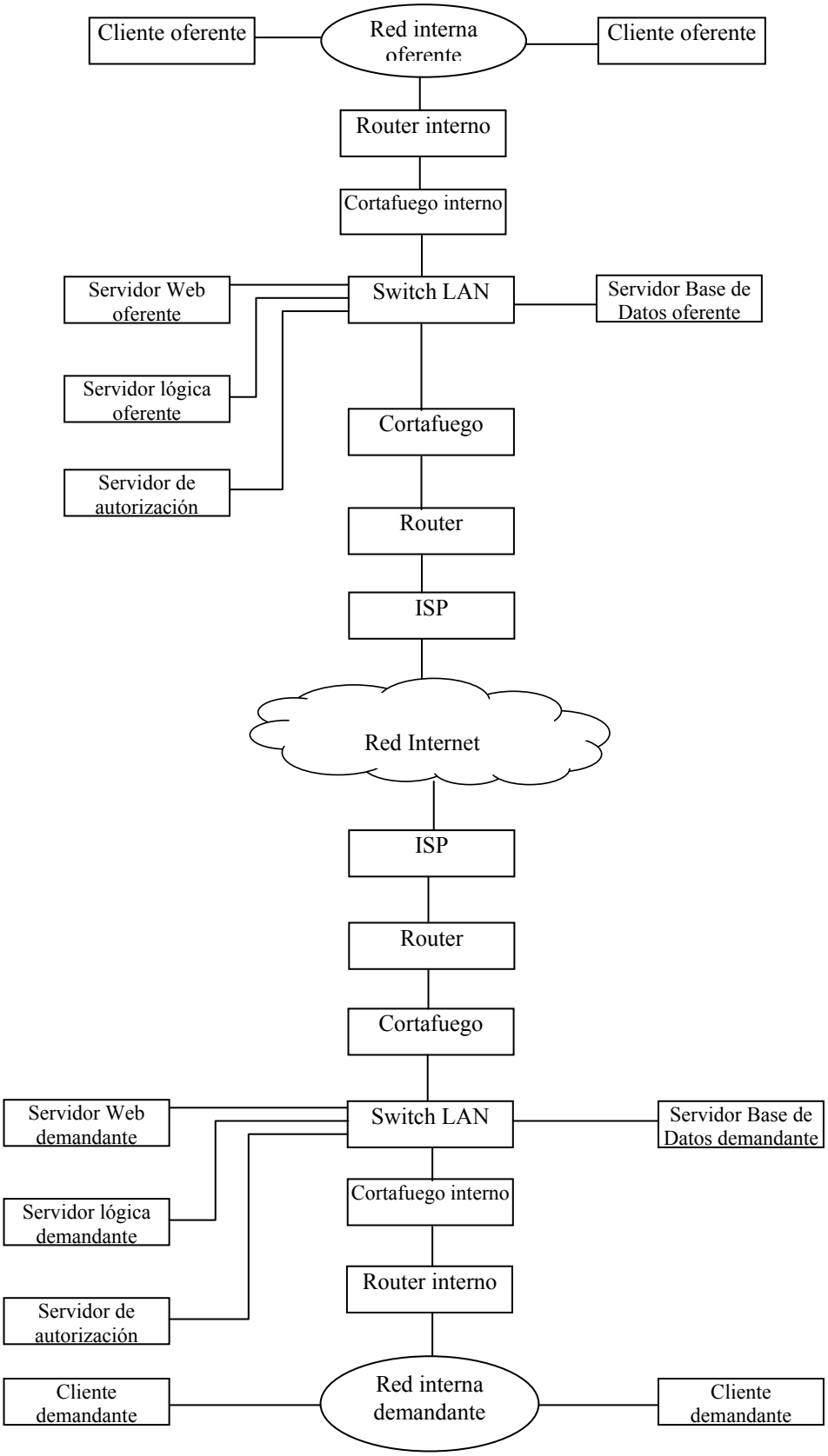


Figura 3.5. Arquitectura B2B

4. Componentes de la Arquitectura Moderna de Aplicaciones E-business: n Capas

4.1. Infraestructura de redes

4.1.1. TCP/IP

En la base de las redes Internet está el protocolo IP. Este provee un esquema de direccionamiento –direcciones IP– independiente de las direcciones físicas locales, que permite identificar cualquier equipo que participan en la red. Es un protocolo que no requiere conexión –como la que se realiza en telefonía– entre el enviador de un mensaje y el receptor del mismo. Por lo mismo no es totalmente confiable, ya que, al no haber comunicación directa, no asegura que los mensajes se transmiten entre origen y destino, aunque provee mecanismos, que explicaremos, para verificar la entrega de un mensaje.

IP se basa en un direccionamiento lógico compuesto de 32 bits, dividido en 4 octetos que se organizan de diferentes maneras para identificar un equipo (host).

La estructura más básica de identificación se basa en segmentar los 32 bits en dos partes, como se muestra en la Figura 4.1: una para especificar la red en que el equipo reside y la otra el particular equipo en tal red.

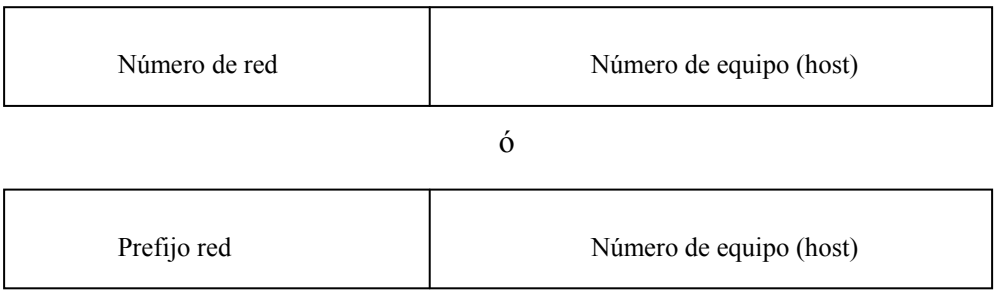


Figura 4.1. Segmentación o jerarquía básica de la estructura de direcciones IP

La terminología que predomina actualmente es la de "prefijo red" para la primera parte. Todos los equipos (host) de una determinada red, en una cierta organización, deben tener exactamente el mismo prefijo y los equipos de redes de diferentes organizaciones no pueden compartir el mismo prefijo.

La estructura antigua de direcciones se basa en la idea de usar el prefijo para, además de identificar una red en particular, asignarla a una de tres clases: A, B y C. Cada clase tiene un diferente largo, en bits, del prefijo, como se muestra en la Figura 3.2. Los bits 0, en el primer caso, 10, en el segundo y 110, en el tercero son fijos y señalan clases las A, B y C respectivamente. También, por convención, indican cuántos bits adicionales identifican una red particular: 7 en la clase A, 14 en la clase B y 21 en la clase C.

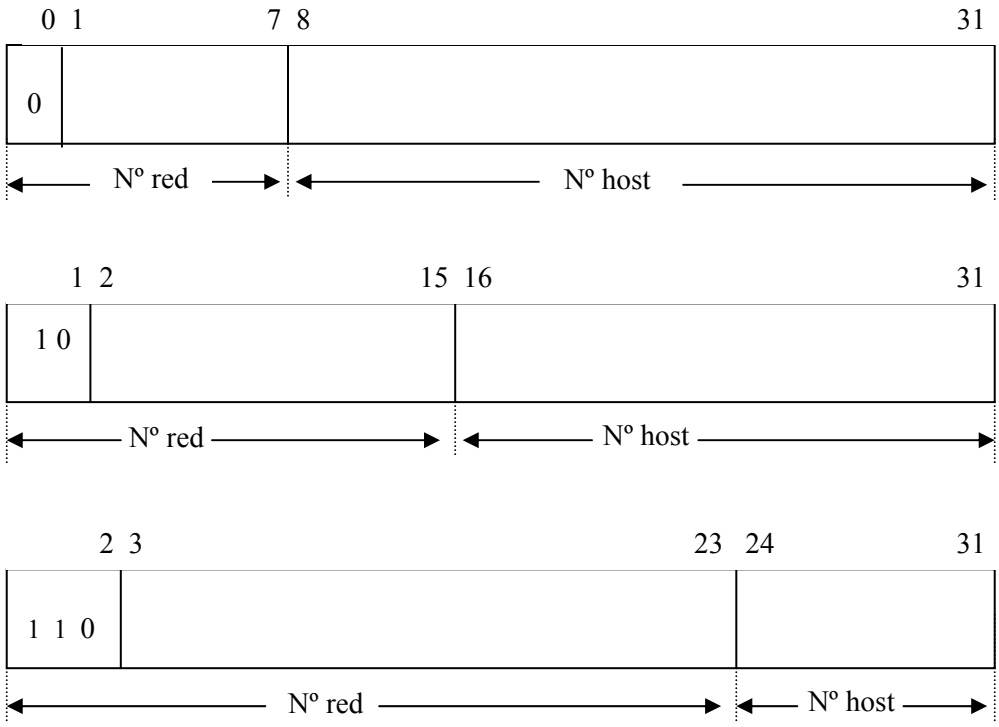


Figura 4.2. Formatos de direcciones, usando clases

Tomando los bits de división entre las partes de una dirección –8 para la A, 16 para la B y 24 para la C–, modernamente se habla de clases /8s, /16s y /24s para referirse a A, B y C.

La clase /8s permite definir 126 (2^7-2) redes, donde se restan 2 para uso especial, y 16.777.214 (2^{24}-2) equipos (host) por red, donde también se restan 2 para fines específicos: sólo ceros para "esta red" y sólo unos para "broadcast". De la misma manera la clase /16s permite 16.384 redes y hasta 65.534 equipos por red, y la clase /24s, 2.096.152 redes y 254 equipos por red.

En la práctica se usan direcciones en formato decimal para facilitar la lectura de las direcciones, como se muestra en la Figura 4.3. El ejemplo identifica la red tipo /8s (primer dígito cero) número 10 y el host 119.204.35.

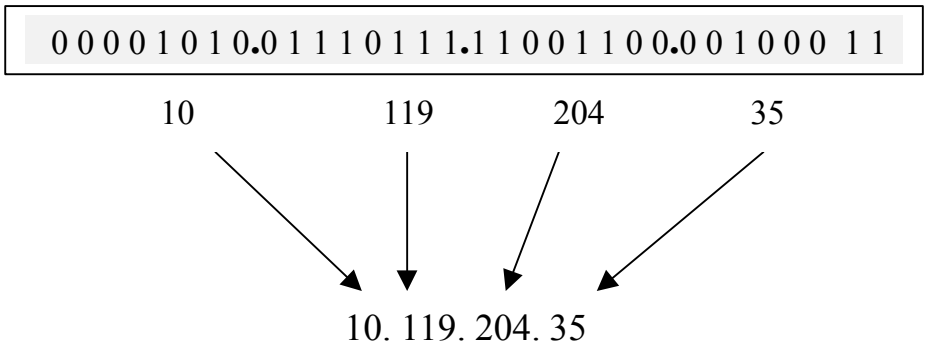


Figura 4.3. Notación decimal separada por puntos

Con el fin de aumentar el número de equipos que se pueden direccionar y la eficiencia del enrutamiento de los mensajes en el formato de clases, se desarrolló la idea de subred. Esta consiste en subdividir el segmento de número de equipo (host) en dos: un número de subred y un número de equipo (host) propiamente tal. La idea se grafica en la Figura 4.4.

Al tener un número de subred, que se maneja internamente dentro de una organización, se puede estructurar una red grande en muchas más pequeñas, con lo cual se puede hacer un enrutamiento más eficiente. En efecto, los mensajes entre los equipos (host) de una red interna pueden ser ruteados directamente sin necesidad de que salgan de ella, en base al número de subred. A nivel de Internet como un todo, las subredes de una organización son transparentes, ya que todas ellas tienen como dirección el prefijo de la red. Por lo tanto, las tablas de enrutamiento de los routers de los backbones de Internet –que explicaremos más adelante– no contienen las direcciones de las subredes y, consecuentemente, son más pequeñas y realizan un procesamiento más eficiente. Como se muestra en la Figura 4.5 todas las subredes de la derecha comparten la misma dirección tipo /16, 130.5.0.0 y cada una de ellas tiene un número característico: 32, 64, 96, 128. Estas subredes pueden ser también lógicas en vez de físicas.

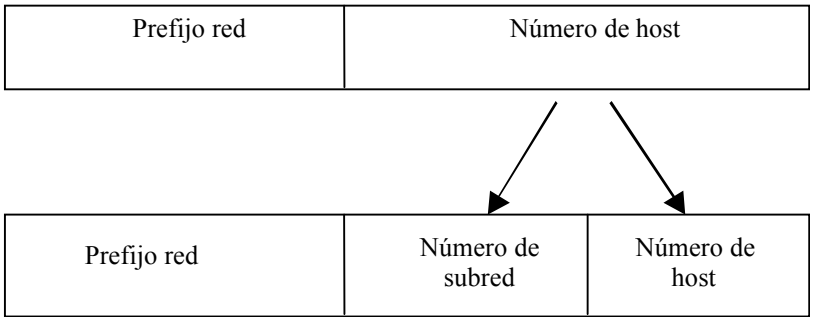


Figura 4.4. Esquema de direccionamiento con subred

El enrutamiento dentro de una organización se hace en base al prefijo de la red extendida, que une el número de prefijo con el número de subred. Asociado a un prefijo de la red extendida, se define una máscara de subred para efectos de enrutamiento de mensajes, que consiste en el número más grande, expresado en notación decimal con puntos, que se podría definir con el largo, en bits, del prefijo de la red extendida. Por ejemplo, la máscara para las subredes de la Figura 5.5 es 255.255.255.0, donde 255 es el número más alto que se puede expresar con 8 bits (número binario de ocho bits de valor uno).

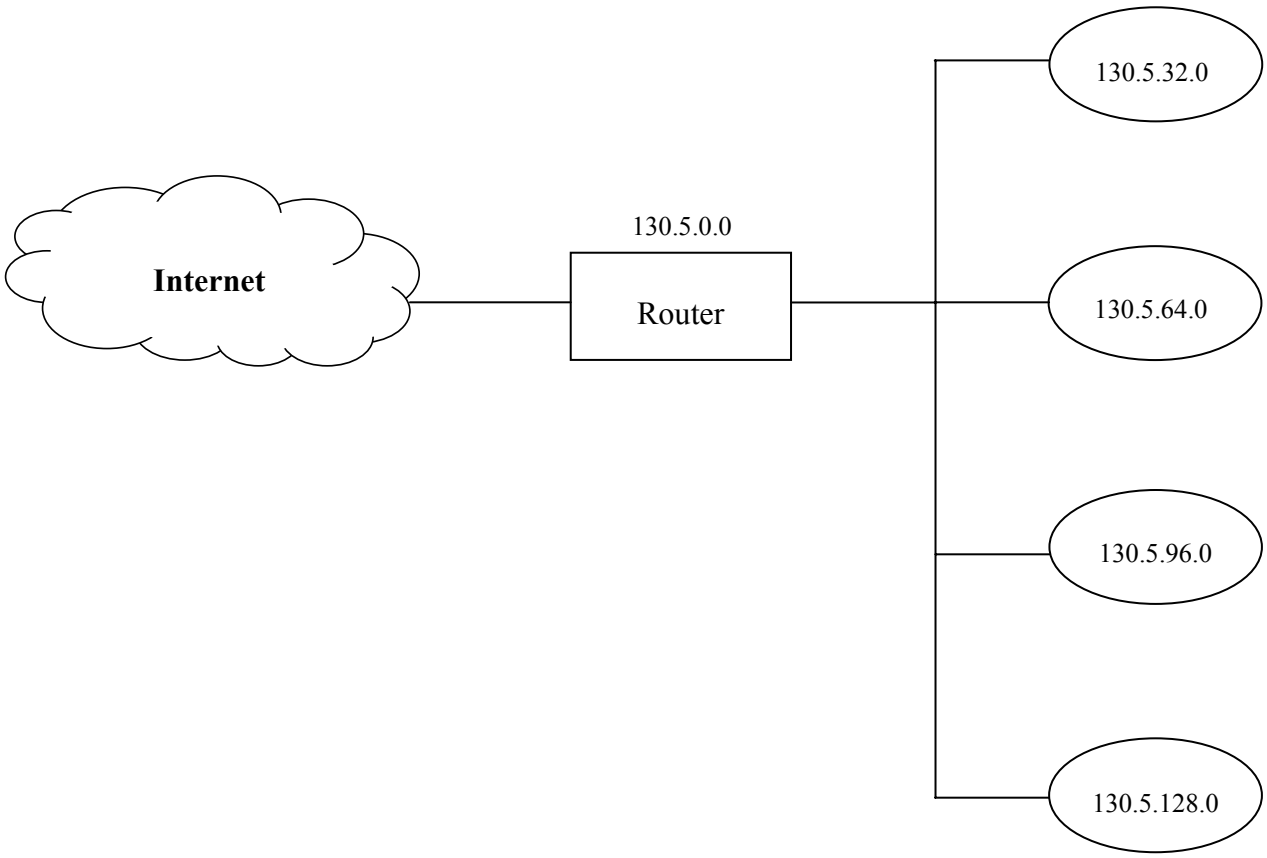


Figura 4.5. Subredes de una organización

En terminología moderna se usa el largo en bits del prefijo de la red extendida en vez de la máscara, el cual se agrega después de un "slash" al final de una dirección. Así la dirección 130.5.5.25/24, tiene un largo de prefijo de la red extendida de 24 bits y corresponde a una máscara 255.255.255.0.

Los prefijos de la red extendida permiten a una organización definir internamente largos arbitrarios para ésta, dependiendo de las necesidades de ella. Así, por ejemplo, una organización que tiene una red /24 (clase C) puede definir un prefijo de la red extendida de 27, con lo cual una dirección cualquiera dentro de esta estructura sería la que se muestra en la Figura 4.6 en la última línea.

El ejemplo de la Figura 4.6 deja 5 bits para asignar direcciones a equipos específicos pertenecientes a una subred dentro de la organización. En el mismo se muestra una dirección específica tanto en notación decimal (con /27) como binaria.

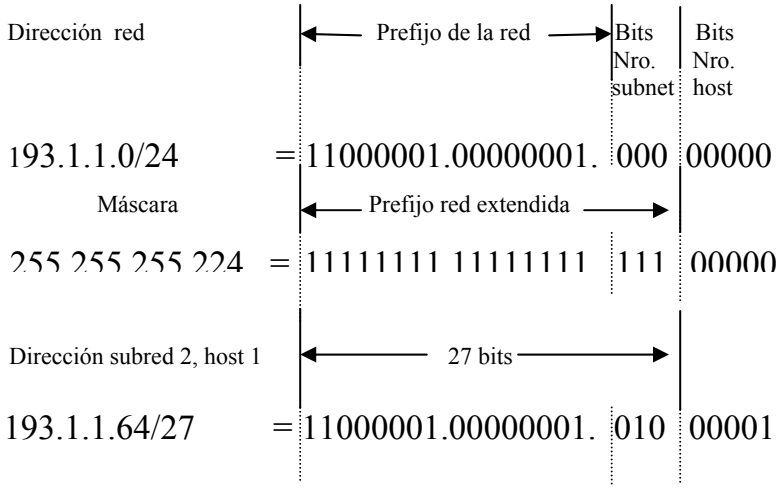


Figura 4.6. Ejemplo de prefijo red extendida

Es evidente que una organización debe diseñar su estructura de subredes y de prefijo, teniendo libertad para definir el largo del prefijo de la red extendida, de acuerdo a los requerimientos de cantidades de direcciones (equipos) necesarios y el flujo esperado de mensajes entre ellas, tema que no abordaremos aquí.

El crecimiento de Internet, que ha producido un paulatino agotamiento del espacio de direcciones IP y un aumento en la complejidad de enrutamiento en los backbones, llevó a reemplazar el sistema de clases por un esquema más general, no incompatible con éstas. Para ello se introdujo el concepto de prefijo de red generalizado, con un largo arbitrario en bits, que es el que usan los routers actualmente para enrutar los mensajes en Internet. Por lo tanto, se elimina el uso de los bits iniciales –por lo menos desde el punto de vista de enrutamiento en los backbones– que denotan la clase, y, consecuentemente, el concepto de clase. Cada prefijo de red es identificado por una máscara de bits o el largo del prefijo. Por ejemplo, en una red de prefijo de largo 20, los 20 primeros bits representan la dirección de la red y la máscara son 20 unos clasificados en octetos o fracciones de éstos, que también pueden representarse en forma decimal. Lo interesante es que una red que tiene un prefijo /20 puede también corresponder a una clase que primitivamente fue A, B o C. Pero los routers de Internet no computan esta información contenida en los primeros dígitos de la dirección.

En una red de prefijo /20, existen 2^{12} o 4.096 direcciones posibles de máquinas en la red (host).

Los largos más comunes de prefijos de red son los que se indican en la Figura 4.7, donde también se señala su equivalencia a redes con dirección tradicional de clases.

Largo prefijo	Máscara en décimal	Cantidad de direcciones individuales	Cantidad de redes definidas en clases
/13	255.248.0.0	512K	8B ó 2048C
/14	255.252.0.0	256K	4B ó 1024C
/15	255.254.0.0	128K	2B ó 512C
/16	255.255.0.0	64K	1B ó 256C
/17	255.255.128.0	32K	128C
/18	255.255.192.0	16K	64C
/19	255.255.224.0	8K	32C
/20	255.255.246.0	4K	16C
/21	255.255.248.0	2K	8C
/22	255.255.252.0	1K	4C
/23	255.255.254.0	512	2C
/24	255.255.255.0	256	1C
/25	255.255.255.128	128	1/2C
/26	255.255.255.192	64	1/4C
/27	255.255.255.224	32	1/8C

Figura 4.7. Largos de prefijos de red

Por ejemplo una red de prefijo 14 tiene una máscara

11111111.11111100.00000000.00000000 = 255.252.0.0

y 2^{10} = 256.000 posibles direcciones de host. También puede hacerse una correspondencia a redes tradicionales de la siguiente manera. Como una red tipo B tiene un prefijo de 16 bits, la diferencia a 14 bits (2 bits) permite definir 4 números de redes de tipo B. De la misma manera, como una red C tiene un prefijo de 16 bits, se pueden definir 2^{10} = 1024 redes tipo C.

Repetimos que para los routers de Internet la clasificación A, B, C es transparente, pero internamente, dentro de una organización, se pueden procesar los bits iniciales correspondiente a tal clasificación y utilizar el largo total de máscara A, B o C para rutear los mensajes.

Los IPS reciben bloques de direcciones de largo arbitrario y pueden, a su vez, asegurar a sus clientes bloques de largo también arbitrario para uso interno. Sin embargo, desde el punto de vista del mundo Internet, la dirección y máscara que se usa para el enrutamiento es la asignada al IPS. O sea, todos los mensajes de clientes de un IPS llegan a él y de allí se enrutan internamente a quien corresponda. Si el receptor es una empresa, se produce un enrutamiento adicional dentro de las subredes y hosts dentro de éstas.

Si bien el uso de prefijos de largo arbitrario ha incrementado la eficiencia de uso del espacio de direcciones IP y generado, por lo tanto, más direcciones libres, el problema de agotamiento de direcciones persiste. Por ello está en implementación un esquema que alarga la dirección a 128 bits, e introduce una serie de mejoras en el manejo de los mensajes. Este nuevo esquema se llama IPv6, en contraposición al actual que se conoce como IPv4.

Junto con el esquema de direccionamiento IP existe el protocolo de comunicaciones TCP (Transmision Control Protocol). Este divide los mensajes en paquetes que se transmiten

y enrutan en forma independiente en Internet, ensamblándose de nuevo en el destino. En el esquema IPv4, cada paquete tiene un encabezamiento de un largo mínimo de 160 bits, el cual contiene, entre otros, 4 bits para la versión del protocolo –IPv4 ó IPv6–, 16 bits para indicar el largo total del paquete, 32 bits para la dirección IP de destino y 32 bits para la dirección IP de origen. Tras el encabezamiento vienen los datos asociados al paquete, cuyo tamaño está limitado al hecho de que el largo total del paquete no puede sobrepasar 65.535 octetos de bits.

TCP usa un principio llamado extremo a extremo, que señala que sólo los extremos son confiables y que la transmisión en sí es insegura. Los extremos son, en este caso, las direcciones IP de los equipos (host) origen y destino.

Basado en lo anterior, TCP usa una técnica llamada de confirmación y retransmisión para asegurar la confiabilidad de la transmisión de los mensajes. Al enviar un mensaje, el equipo origen espera un tiempo predefinido por la recepción de confirmación de recepción en el destino y, si no la recibe, retransmite el paquete no confirmado. En el destino se confirman sólo los paquetes que llegan sin error y, descartan los que tienen problemas. Debido a que la confirmación puede llegar al origen después de que en éste retransmita el mensaje, en el destino pueden haber paquete duplicados. También los paquetes de un mensaje pueden haber llegado en un orden diferente al de envío, por lo cual el host de destino debe eliminar duplicados y secuenciarlos de la manera correcta.

TCP enumera secuencialmente cada byte en los datos que acompañan un encabezamiento y usa estos números para verificar la confirmación de los datos recibidos en el host de destino. Para iniciar la conexión y establecer el número inicial de la secuencia, el host origen envía un paquete “sync” al host destino. Este responde con un “sync” de confirmación que también contiene el tamaño inicial de ventana y, opcionalmente, el tamaño máximo de paquete. El tamaño de ventana señala la cantidad máximo de datos que se puede enviar antes de recibir una confirmación. De esta manera el host de destino controla el flujo de información.

El mensaje de confirmación contiene el más alto octeto (byte) consecutivo recibido correctamente y el nuevo tamaño de ventana. El host de destino no confirma paquetes fuera de orden hasta no recibir los segmentos que faltan en la secuencia. Después del período de espera de confirmación, el host origen retransmite la parte no confirmada de la ventana.

Una vez recibidos los segmentos, el host de destino pasa los confirmados al número de puerto designado dentro del equipo, en la forma de una secuencia de octetos. La aplicación en el host tiene la responsabilidad de transformar tal secuencia al formato requerido por ella.

De la misma manera que la conexión requiere una confirmación en tres pasos, el cierre también se basa en esta idea. Una vez que el host origen recibe el requerimiento de cierre por parte de la aplicación, envía un mensaje de cierre al host destino con una numeración de segmentos como la ya explicada. Para cerrar la conexión, el host origen debe recibir la confirmación de todos los segmentos enviados y del cierre por parte del host destino. El origen confirma entonces el cierre e informa a la aplicación.

El método explicado minimiza el tráfico IP sobre la conexión. Además, no depende de la confiabilidad del portador de los mensajes. Al contrario, un portador que gasta mucho tiempo tratando de asegurar confiabilidad –verificando y corrigiendo segmentos, por ejemplo– causa que los segmentos se retransmitan por agotamiento del tiempo de confirmación. Un protocolo de transmisión adecuado para TCP/IP es uno que elimina, en vez de intentar corregir, datagramas dañados.

La manera en que se utiliza TCP para la conexión entre un cliente y un servidor Web es muy simple. Una vez conectado el cliente con el servidor, éste empaqueta los contenidos de su página principal en una serie de paquetes TCP y los envía al primero. El cliente recibe los paquetes y los ensambla para visualizar la página. La conexión se termina al recibirse la página. Este proceso se repite varias veces para cada petición que realiza el usuario en el cliente.

4.1.2. Otros Protocolos en Internet

TCP/IP es parte de un conjunto de protocolos desarrollados para Internet, los cuales aplican a diferentes niveles dentro del modelo OSI. Tales protocolos se muestran en la Figura 4.8. Explicamos brevemente, a continuación, los complementarios a TCP/IP.

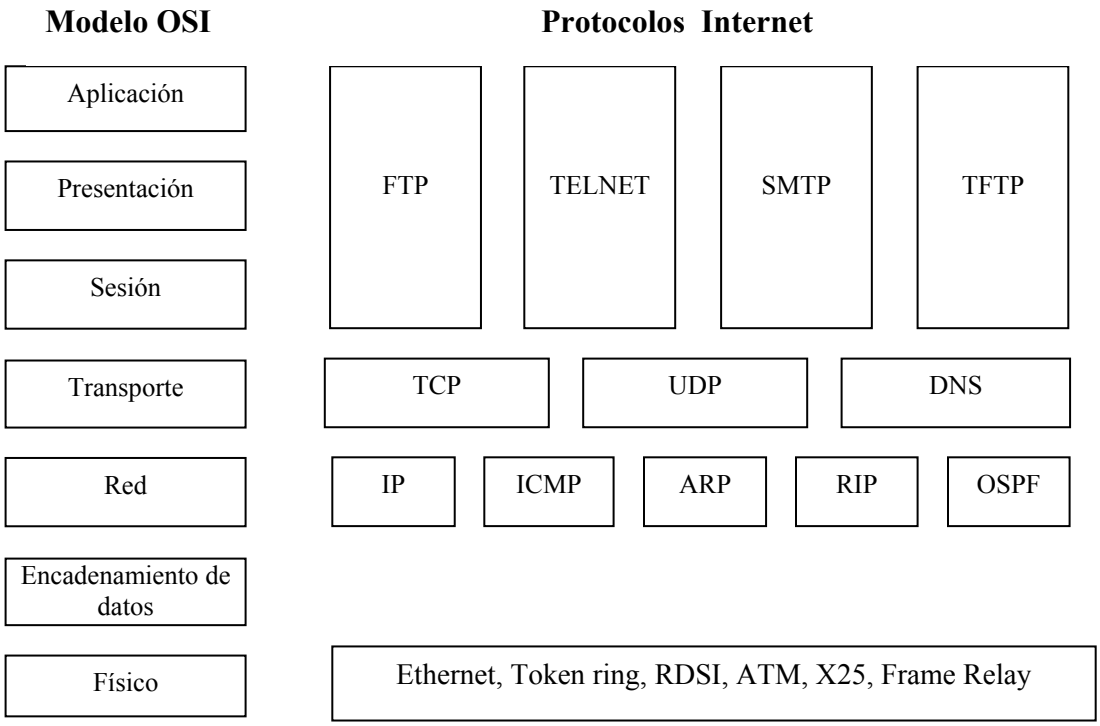


Figura 4.8. Internet y el modelo OSI

- ICPM (Internet Control Message Protocol) funciona con IP para proveer información de error y de otro tipo. Como ya lo explicamos, TCP no opera en base a una conexión y no puede, por lo tanto, detectar la situación de la red; por ejemplo, congestión o de falla de algún camino. ICPM se usa para notificar al protocolo IP y otros de las capas superiores del modelo OSI, de errores en la red y problemas de control de flujo. Por ejemplo, uno de los servicios que provee este protocolo es enviar un mensaje al host origen cuando una red, un equipo, un protocolo o un puerto al cual éste envió un mensaje, no están disponibles.

- ARP (Address Resolution Protocol) combina la dirección lógica del nivel red de OSI (dirección IP) con la dirección física del nivel de encadenamiento de datos para crear una dirección Internet completa. Adicionalmente, asigna nombres lógicos de nodo para hacer más amistoso el direccionamiento. Se aplica, fundamentalmente, al nivel de una red local para pasar de direcciones lógicas a físicas.

- RIP (Routing Information Protocol) es un protocolo de detección de rutas basado en un vector distancia, que disemina periódicamente RIT (Route Information Tables) en la red. Estas tablas de enrutamiento son las que usan los routers para encaminar los mensajes. Debido a que este protocolo no funciona bien en redes grandes interconectadas, se ha desarrollado una alternativa más poderosa: OSPF.

- OSPF (Open Short Path First) es un protocolo de detección de rutas basado en un "estado de caminos", que tiene la habilidad de describir las topologías específicas de una red. Genera tablas de enrutamiento que se actualizan periódicamente en base a la disponibilidad de caminos en la red y las distancias más cortas entre puntos.

- UDP (User Datagram Protocol) esté cercanamente relacionado con TCP en cuanto a que provee servicios de nivel de transporte de OSI. Sin embargo, UDP no está orientado a la conexión y no reconoce la recepción de datos.

Simplemente, acepta y transmite datagramas. Al no tener que realizar toda la burocracia del nivel de transporte que hace TCP, puede transmitir datos mucho más rápido, pero con menor confiabilidad. Se puede caracterizar como una alternativa más rápida que TCP cuando no se necesita seguridad en la transmisión.

- DNS (Domain Name System) es una base de datos distribuida que realiza traducción de nombres a direcciones a pedido de otras aplicaciones. Se origina en el hecho de que, por razones de facilidad de recordación de direcciones, todas las direcciones IP tienen un nombre, que es el que utilizan los usuarios en las aplicaciones, que permite comunicarse en Internet. Los servidores DNS mantienen una estructura jerárquica de nombres, de tal manera que cualquier host origen pueda acceder al adecuado para determinar la dirección IP correspondiente en un nombre entregado por un usuario. Daremos algunos detalles acerca de DNS cuando expliquemos el enrutamiento en Internet.

- FTP (File Transfer Protocol) permite a un usuario transferir archivos entre dos computadores en la red. También provee una serie de otros servicios, como login, búsqueda en directorios, manipulación de archivos, ejecución de comandos y otros de alto nivel. Además FTP permite mover archivos entre equipos de diferentes sistemas operativos usando TCP.

- TELNET es el protocolo de emulación de terminales remotos. Permite acceso de los usuarios a aplicaciones en equipos (host) centralizados, actuando los computadores personales como terminales tontos. En su forma más simple, el software TELNET permite a un equipo cliente emular un terminal, proveyendo también conectividad entre equipos con sistemas operativos disímiles.

- SMTP (Simple Mail Transfer Protocol) es un estándar de enrutamiento de correo electrónico que usa TCP/IP para encaminar mensajes entre los equipos (host) de una red. No provee, eso sí, una interfaz para correo local.

- TFTP (Trivial File Transfer Protocol), lo de trivial no se refiere al protocolo, sino que a la pequeña cantidad de código necesaria para implementarlo. TFTP no incluye autentificación, por lo cual se limita a leer y escribir archivos públicamente disponibles; por lo tanto es riesgoso y no debe ser usado para transferir datos a través de un cortafuego.

4.1.3. Enrutamiento de paquetes

Tal como se ha indicado anteriormente, el enrutamiento de los paquetes TCP se hace a través de los routers utilizando las direcciones IP. Al generarse un paquete a partir de un mensaje –tal como se explicó en el punto anterior– en una cierta subred local, lo primero que se hace es determinar si él va dirigido a algún otro host de la misma subred. Para esto el router utiliza la máscara de la subred, realizando las operaciones que se indican en el ejemplo de la Figura 4.9. La primera operación es un XOR (OR exclusivo: bit resultado es uno sólo si uno de los correspondientes bits de las direcciones en cálculo es uno, sino es cero) aplicado a las direcciones IP origen y destino. Si el resultado fuera bits ceros en los tres primeros octetos, la dirección de destino estaría en la misma subred de la dirección origen, ya que se trata de una red /24 –la dirección de la subred está en los primeros 24 bits– y al tener la misma dirección el XOR produce ceros. La manera en que el router automatiza este cálculo es haciendo una operación binaria AND (bit resultado es uno sólo si ambos bits en cálculo son uno, sino es cero) entre el resultado XOR y la máscara. Si la dirección destino estuviera en la subred, el resultado de esta operación son sólo ceros, ya que los primeros 24 bits del resultado XOR serían ceros y los primeros 24 bits de la máscara, unos. Los últimos ocho bits son siempre cero, ya que la máscara tiene ceros en esas posiciones. En el caso del ejemplo de la Figura 4.9 el resultado OR no es cero, por lo cual la dirección destino no está en la misma subred.

Si la dirección destino estuviera en la misma subred, se usa el protocolo ARP, ya explicado, para pasar de la dirección lógica (IP) a la dirección física del equipo al cual está dirigido el mensaje y determinar dónde enviar el paquete.

Si la dirección destino está en otra red, el router busca en su tabla de enrutamiento si existe en ella una entrada que establece un camino más corto y disponible para llegar a la red destino. Por ejemplo, la tabla de enrutamiento puede tener una entrada que establece que para la red destino con dirección IP 10.1.1.0, con máscara 255.255.255.000, la dirección 10.1.2.1 señala el router al que deben ser enviados los mensajes para llegar a tal destino.

Si no hay una entrada en la tabla de enrutamiento para la red destino, el router manda el paquete a un enrutador designado por defecto, que repite las operaciones ya explicadas.

Dirección IP origen :10.119.204.35 00001010.01110111.11001100.00100011
Dirección IP destino:10.119.15.217 00001010.01110111.00001111.11011001
Resultado XOR 00000000.00000000.11000011.11111010
Máscara subred 255.255.255.0 11111111.11111111.11111111.00000000
Resultado AND 00000000.00000000.11000011.00000000

Figura 4.9. Cálculo binarios para determinar ruta

Los enrutadores conforman una jerarquía, como se muestra en la Figura 4.10. Por lo tanto, un paquete viaja de router en router dentro de la jerarquía hasta encontrar una entrada en una tabla de enrutamiento que lo dirige a su destino.

Como en el uso diario es difícil recordar direcciones IP binarias o decimales, se ha desarrollado un sistema de nombres nemotécnico que reemplaza los números. Este sistema, llamado DNS (Domain Name Service), se basa también en una jerarquía. El nivel superior de ésta es una raíz, a la cual pertenecen, tal como se muestra en la Figura 4.11, una serie de dominios o grupos de nombres que comparten una característica común. Por ejemplo, el dominio "edu" reúne a todas las instituciones educacionales de EEUU y de otros países que decidan inscribirse dentro de tal dominio. Una institución educacional en particular tiene, entonces, como nombre su nombre particular seguido de un punto y el sufijo "edu"; por ejemplo: mit.edu.

Otro dominio popular es el comercial (com), donde están las organizaciones de EEUU y otros países que persiguen fines de lucro; por ejemplo: Amazon.com.

Los países tienen dominios particulares. Así, Chile tiene el dominio "cl" e Inglaterra el dominio "uk".

Existen servidores DNS ubicados estratégicamente, que mantienen las tablas que ligan direcciones IP a nombre DNS y permiten realizar la conversión de uno a otro. Típicamente, cada ISP debe tener un servidor DNS o acceso a uno. También, una organización en particular puede tener su propio servidor DNS. Además de estos lugares, existen servidores DNS en toda la red Internet que "se copian uno a otros" para mantenerse actualizados, de tal manera que cualquier usuario que requiera el servicio de traducción lo obtenga rápidamente. En particular, la organización Internic –que es responsable de asignar nombres de dominio a los que lo solicitan y de que no se produzcan duplicaciones– mantiene una base de datos centralizada de nombres.

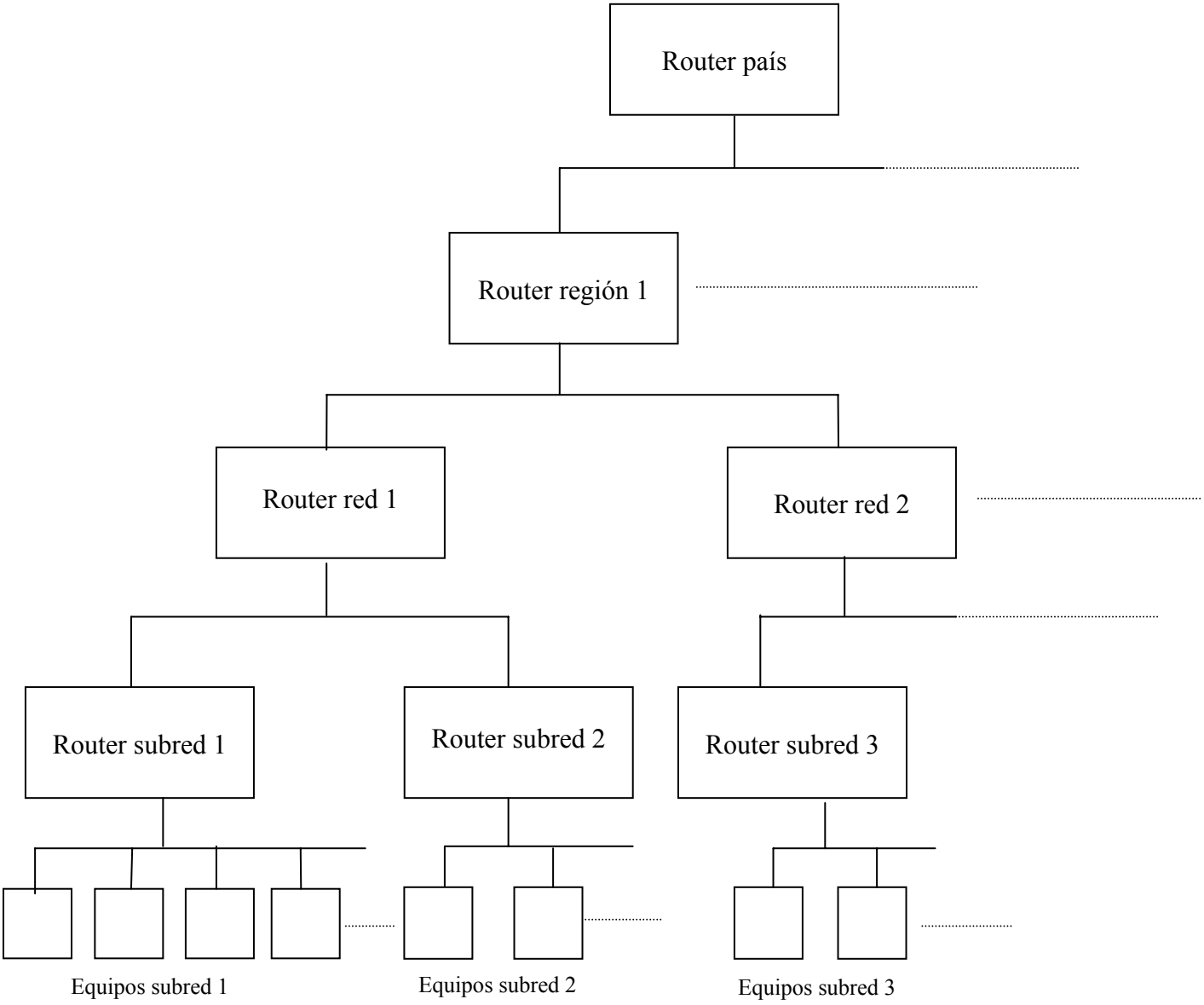


Figura 4.10. Enrutamiento jerárquico en Internet

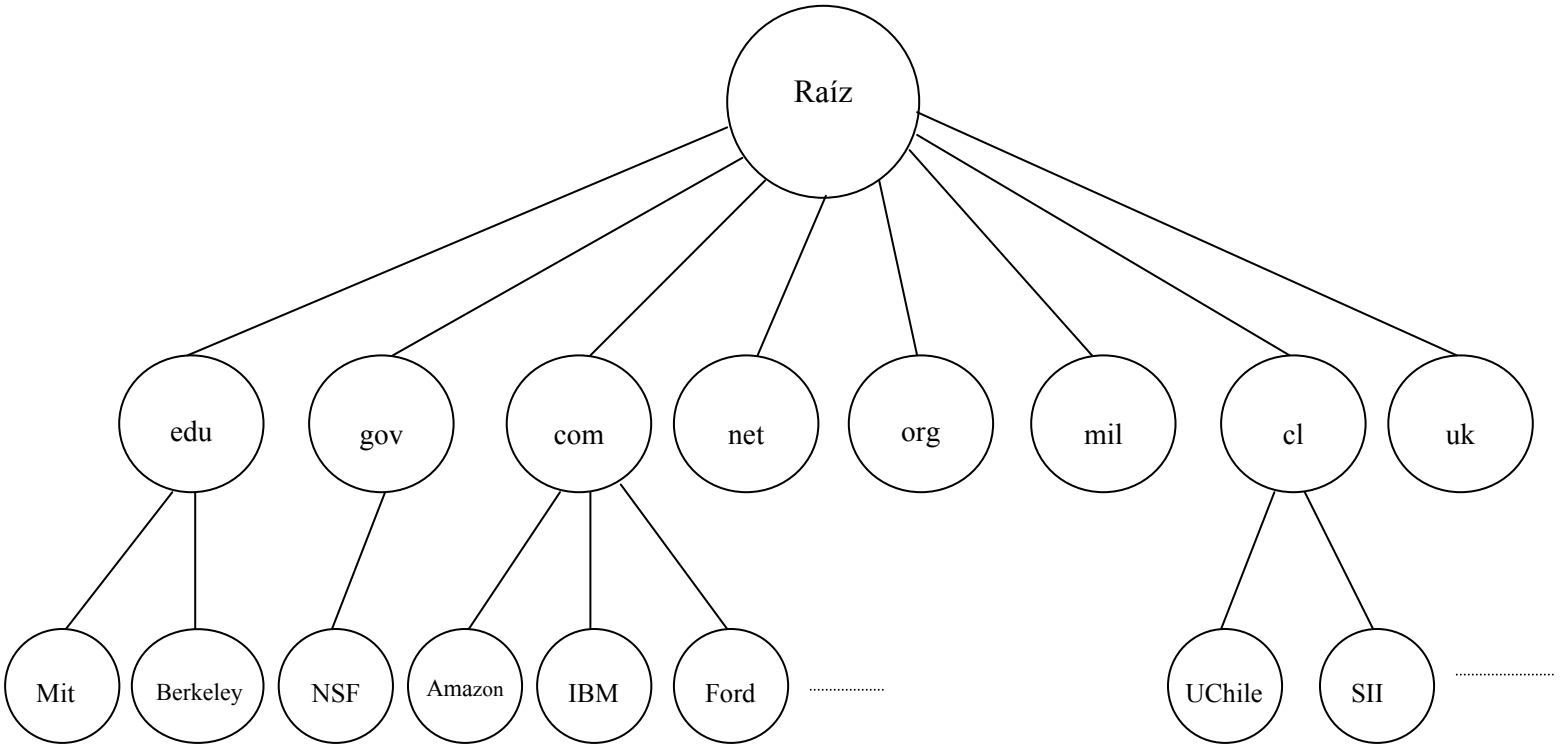


Figura 4.11. Jerarquía de dominios

Las direcciones de correo electrónico son una variante del nombre DNS: a él se le antepone el nombre de la persona u organización destinataria, separada por una arroba (@); por ejemplo, obarros@uchile.cl

4.1.4. La World Wide Web

La World Wide Web (Web) aprovecha la infraestructura de Internet ya definida para facilitar la construcción de sitios que publican páginas o documentos del tipo hipertexto. Tales páginas tienen direcciones o enlaces del tipo hipertexto que permiten cambiar a una página en el mismo sitio o en otro, por el mero hecho de seleccionar un enlace.

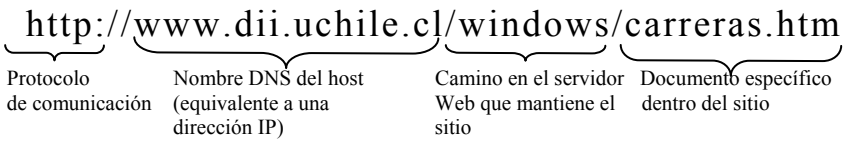
Los sitios Web son, entonces, conjuntos de páginas interrelacionadas por medio de tales enlaces que permiten "navegar" en ellas de acuerdo a los intereses de los usuarios del sitio. Un caso simple de uso de tal idea es una enciclopedia electrónica, donde un usuario podría ir a una página específica en un sitio, que le entrega información respecto a una cierta materia y desde allí, por medio de enlaces, a múltiples otras páginas que explican más detalladamente conceptos que aparecen en el texto asociado a tal materia. Este proceso se puede repetir para estas páginas y, también, para las nuevas páginas accesadas, y así en forma indefinida según lo que el usuario determine. Esta misma idea está detrás de cualquier sitio que exista sólo para proveer información; por ejemplo, noticias, datos financieros, vuelos, etc.

La misma idea anterior puede utilizarse para sitios orientados al comercio electrónico, donde las páginas –empezando con una "home page" o página principal– dan a conocer la oferta de productos o servicios de una empresa. Un sitio de este tipo detalla progresivamente tal oferta –de acuerdo a los intereses de un usuario– por medio de éste elija los enlaces para llegar a lo que le interesa. Tales sitios incluyen, además, la posibilidad de efectuar la transacción comercial para adquirir un producto o servicio seleccionado.

La manera en que se identifican los sitios Web y las páginas en ellos es mediante una dirección URL (Universal Resource Locator). La estructura general de un URL es la siguiente:

<<http:>><<//>> máquina [<<:>> puerto] camino

Por ejemplo, la URL que se señala a continuación tiene el significado que se indica:



HTTP (Hyper Text Transmission Protocol) es un protocolo de comunicación rápido y eficiente que controla las operaciones que ocurren entre un host que requiere información de otro que mantiene un sitio; entre otras, iniciar y controlar la conexión y la transmisión de información entre ambos, y, en versiones más avanzadas, verificar el inicio de la sesión y transmisión encriptada de la sesión. Este protocolo funciona sobre TCP.

La interacción entre un host que requiere información y el sitio que la provee se puede considerar, como ya se indicó anteriormente, como una relación C/S, en que el cliente es el host que pide la información de una cierta página y el servidor, el host que mantiene un sitio que contiene tal página.

Las páginas de los sitios tienen que estar desarrolladas en un formato estándar, que incluye enlaces, para poder ser reconocidas y transmitidas por HTTP. Este formato es generado con el lenguaje HTML (Hiper Text Manipulation Language), el cual permite construir las páginas de un sitio.

Para facilitar la relación C/S entre un host que solicita páginas HTML y un host con un sitio que las provee a través de una conexión HTTP, existen los navegadores (browsers). Estos son paquetes de software que permiten al usuario en el cliente entregar

direcciones URL a las cuales quiere acceder, y dejar todos los detalles de búsqueda de las páginas, conexión y transferencia de ellas y presentación al browser. El procedimiento típico que se ejecuta al usar un browser para bajar una página desde un servidor Web a un host cliente, es la que se indica en la Figura 4.12. Por razones de claridad, se han dibujado flujos directos entre los elementos que intervienen. Sin embargo, obviamente, estos ocurren a través de la infraestructura de Internet ya explicada y usando los protocolos TCP/IP también detallados.

El intermediario que aparece en la Figura 4.12 se refiere a un cortafuego u otro mecanismo de filtrado, reformateo o traducción de mensajes. El filtrado es por razones de seguridad y puede basarse en la dirección IP del solicitante y las páginas y tipo de servicio que solicita.

La conexión HTTP termina apenas enviada la página –estructurada en paquetes, como se explicó anteriormente– al cliente. Si el usuario en el cliente invoca algún enlace en la página recibida, se inicia una nueva conexión. La única diferencia es que no se invoca al servidor DNS, ya que se conoce la dirección IP del servidor Web.

Las conexiones HTTP entre los clientes y los servidores Web se hacen por medio de mensajes, que son empaquetados por TCP, como cualquier otro. Los mensajes más comunes son GET, que se usa para obtener páginas del servidor identificado por el URL; HEAD, para conseguir información sólo del encabezamiento –título, descripción, relación con otros documentos del nodo y el URL– del documento; y POST, que le señala a un servidor que debe tomar nota de un recurso existente, enviar un mensaje, enviar un formulario de datos para su procesamiento, o añadir información a una base de datos.

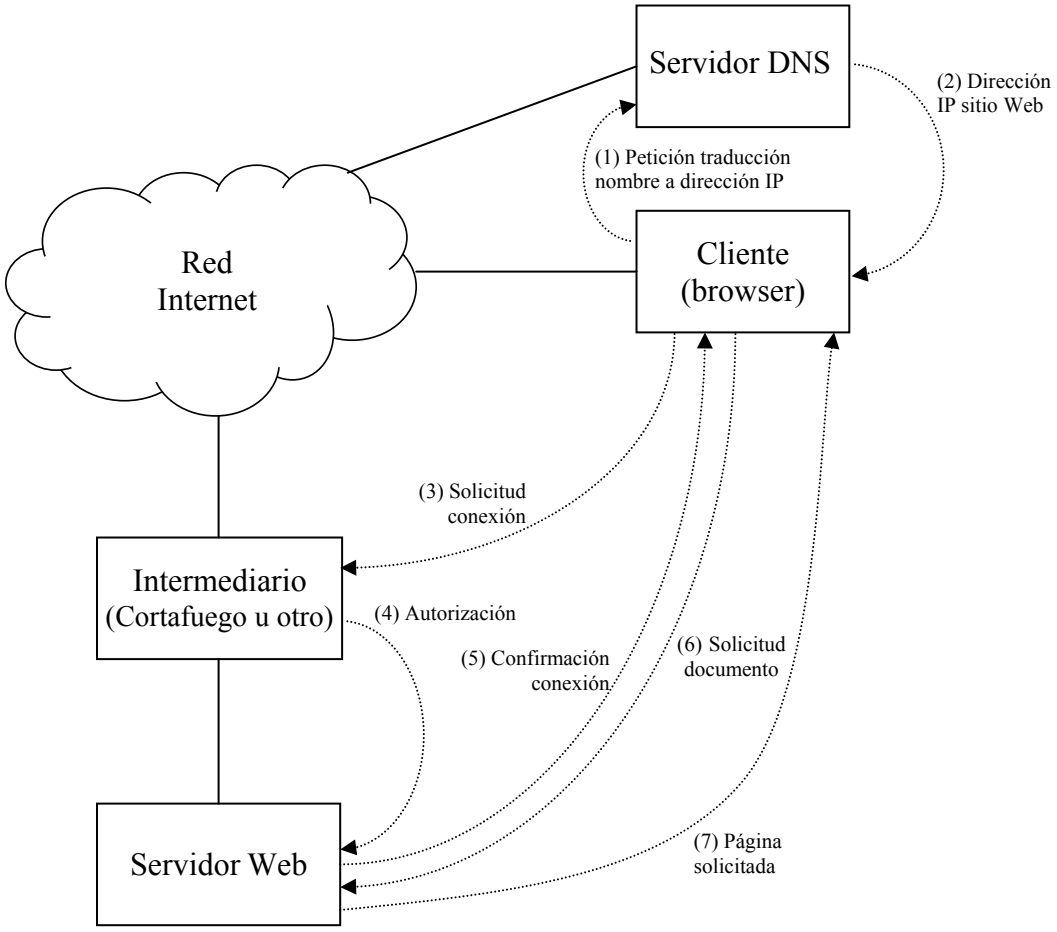


Figura 4.12. Procedimiento de acceso a páginas en la Web

4.1.5. Seguridad en Internet

Los primeros elementos relativos a la seguridad en Internet ya aparecieron en el Punto 2. Remitiéndonos a una red típica, como la que se muestra en la Figura 2.4, existe, en primer lugar, un router filtrador de paquetes. Este examina las direcciones IP de origen y destino de cada uno de los paquetes que entran a la red y puede bloquear a algunos de ellos, impidiéndoles el ingreso. También puede impedir que algunos paquetes salgan de la red. Los criterios de filtración pueden ser, por ejemplo, que sólo los mensajes provenientes de empresas conocidas sean aceptados y que mensajes a sólo ciertas direcciones están permitidos para un determinado tipo de funcionarios de la empresa. Dado que individuos con malas intensiones pueden simular que vienen de direcciones autorizadas, en posible incluir en los mensajes y verificar una identificación encriptada que asegura que un mensaje realmente viene de alguien autorizado.

Algunas organizaciones mantienen sitios Web de acceso libre, sin filtrado alguno, orientados a que el público en general acceda a información sobre la empresa y sus productos o servicios. Cuando esto es así, los servidores que quedan fuera de todo tipo de filtrado de mensajes se dice que están en una zona desmilitarizada.

Otro tipo de cortafuegos, que aparece en la Figura 2.5, es del nivel aplicación – cortafuego distribuidor de carga y cortafuego secundario. Los de este tipo intentan descubrir paquetes que contengan programas potencialmente peligrosos –que pueden incluir instrucciones ocultas que roben o destruyan datos– y darles un procesamiento especial. Por ejemplo, enviarlos a un servidor proxy que transfiere la información a un programa proxy que se puede correr en forma segura en la red.

Además, como también se muestra en la Figura 2.5, pueden existir cortafuegos para proteger los datos de los sistemas de la empresa –servidor de autentificación–, que dan el acceso a cierta información sólo a personal de la empresa o de otras empresas que tienen derecho a ver y/o modificar tal información.

Un mecanismo de seguridad adicional consiste en la encriptación de información contenida en los paquetes, transformándola de una manera compleja que la vuelve ininteligible a cualquier persona excepto al recipiente autorizado. Este debe contar con una clave que permite hacer la transformación en sentido contrario para poder leer el mensaje.

La encriptación de mensajes que contienen información confidencial o que, si se modifica, puede producir graves consecuencias a una empresa es necesaria, ya que es relativamente fácil capturar mensajes que se mueven en la red interna de una empresa o en la red del mundo Internet externa a la empresa. En relación a la red interna, conviene hacer notar que una fuente potencialmente peligrosa de uso inadecuado de la información de una organización, son las personas que trabajan en ella.

La ciencia que provee las herramientas para proteger los mensajes es la criptografía, la cual tiene origen matemático y nació de las necesidades de comunicaciones seguras que tienen los militares.

Una avance fundamental de la criptografía se produjo cuando dos profesores de la Universidad de Stanford inventaron en 1976 la encriptación basada en una clave pública[*]. Esta resolvió un problema fundamental: para encriptar se necesita una clave, la cual debe ser conocida por el que recibe el mensaje y, por lo tanto ser transmitida, con el consecuente riesgo de ser interceptada. La idea de clave pública evita tal problema, como lo explicaremos a continuación.

La criptografía basada en una clave pública permite que dos participantes se comuniquen de manera segura, sin necesidad de un medio para intercambiar claves. Se basa en dos claves que son diferentes pero complementarias. Cada clave descifra el mensaje que la otra clave encripta, pero el proceso no es reversible: la llave que se usa para encriptar un mensaje no puede utilizarse para descifrarlo. Así, una de las llaves puede diseminarse sin problemas (llave pública) y la otra (privada) es conocida sólo por su dueño. Así, por ejemplo,

[*] Schneier. B, Applied Criptography, John Wiley, 1996.

cuando Alicia quiere enviar un mensaje a Benito usa su llave pública para encriptarlo y este último empleará su llave privada para descrifrar la información.

El procedimiento para intercambiar la clave privada entre dos participantes, sin riesgo de ser interceptada y descifrada por otros, se describe en la Figura 4.13. La idea fundamental es que cada uno de ellos calcula un número basado en su número secreto (x ó y) y otro número apropiado g. Estos se intercambian y cada uno de ellos eleva tal número a una potencia igual a su número secreto, resultando en ambos casos g^{xy}, que es un número secreto que sólo ellos conocen. Como los mensajes de intercambio de la Figura 4.13 podrían ser interceptados, se transmite la siguiente transformación de g^x: g^x módulo p (igual al remanente de g^x dividido por el número primo p), haciendo lo mismo con g^y. Esto hace muy difícil recuperar x (ó y). Con un procedimiento matemático apropiado se puede generar, a partir de lo anterior, una clave privada (x para Alicia) y una pública, consistente de g, p y g^x módulo p.

Debido a que la implementación de la encriptación con una clave pública es relativamente lenta, en la práctica se usa mayoritariamente un procedimiento basado en claves simétricas –que sirven para encriptar y descifrar– que es mucho más eficiente. Sin embargo, el esquema de clave pública se usa en tal procedimiento para intercambiar las claves simétricas, de la manera que se explica a continuación.

Si Alicia quiere enviar un mensaje a Benito, encripta su mensaje usando una clave simétrica. Para enviar tal clave a Benito –que éste necesita para descifrar el mensaje– la encripta usando su clave pública y la acompaña al mensaje. Benito utiliza su clave privada para descifrar la clave simétrica y con ésta descifrar el resto del mensaje.

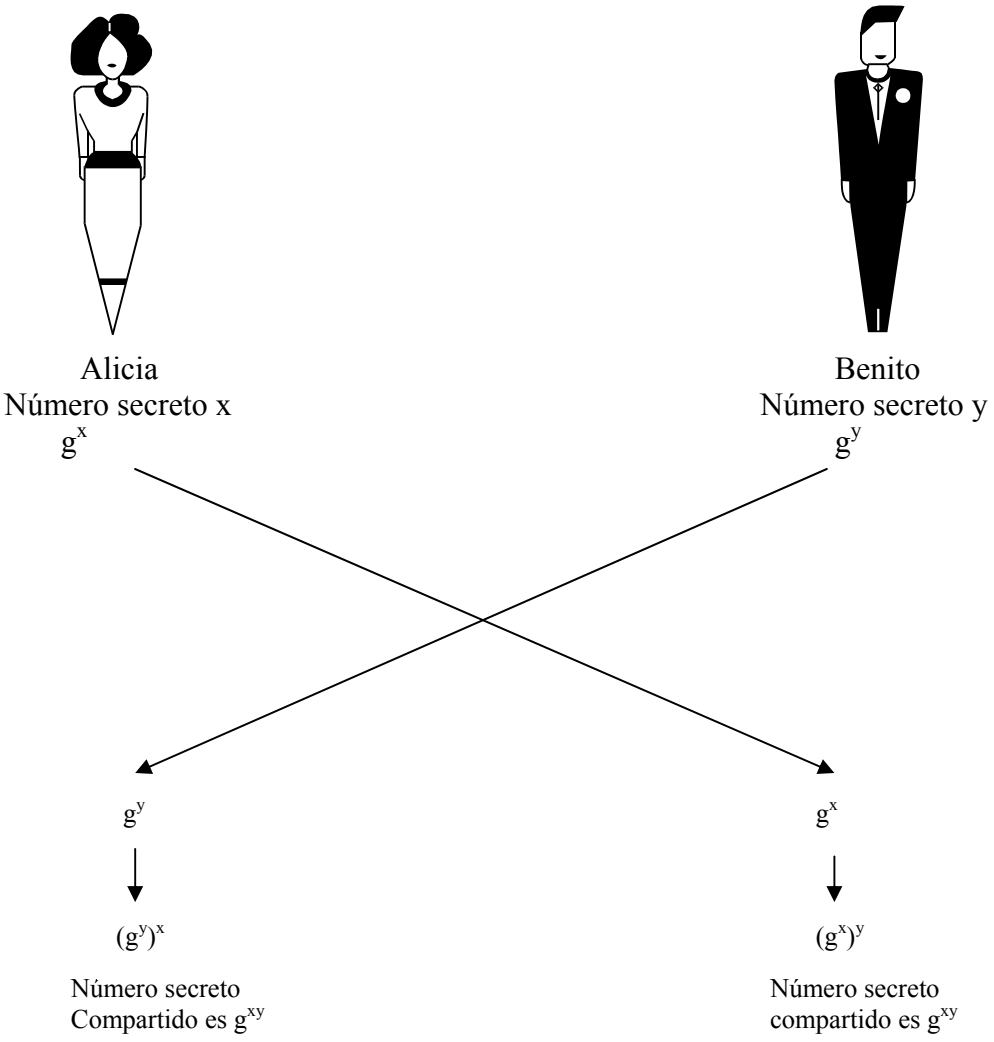


Figura 4.13. Generación de una clave secreta compartida

Para asegurarse que la que envía el mensaje es realmente Alicia se usa un procedimiento de autentificación. Este consiste en computar un "hash" del mensaje –que es un procedimiento matemático que condensa un texto de cualquier largo a un tamaño fijo, típicamente 160 bits– encriptarlo con su llave privada, pasando a constituir una especia de "firma electrónica". Esta se envía junto con el resto del mensaje encriptado. Benito recibe esta firma y la descifra con la clave pública de Alicia. Entonces puede comparar el "hash" descifrado con un "hash" que el mismo computa. Si son iguales el mensaje proviene de Alicia.

El procedimiento anterior puede ser complementado con un certificado digital para una mayor seguridad de que el mensaje realmente viene de la persona que se señala en él. Para ello existe la necesidad de tener una organización confiable que emita certificados digitales y sea aceptada por una comunidad de usuarios. Entonces, la persona que quiere enviar un mensaje –Alicia, por ejemplo– pide un certificado a la organización certificadora, a la cual debe estar afiliada, por medio de enviar su clave pública. Este solicita a Alicia antecedentes privados –previamente conocidos por la autoridad como parte de la afiliación– para autentificarla, y de ser esto satisfactorio emite el certificado que señala que la clave pública pertenece a Alicia. Junto con el certificado va la firma digital de la autoridad, que puede ser verificada por cualquiera que tenga su clave pública. Por otro lado, la autoridad divulga su llave pública a todos los afiliados.

Al recibir Benito el mensaje de Alicia de la manera anteriormente explicada más el certificado y la firma digital de la autoridad, procede a verificar tal firma con la llave pública de ésta. Si está conforme, el certificado es legítimo y la llave pública pertenece a Alicia, con lo cual puede proceder a descifrar la firma de Alicia y el mensaje como ya se explicó.

4.2. **Clientes**

Como ya se ha destacado, los clientes demandan servicios de los servidores de diversos tipos en Internet. Físicamente, los clientes pueden ser PC, computadores o artefactos de red (más simples que los PC), agendas, teléfonos, etc. El factor común de éstos es que los servicios se requieren usando las mismas tecnologías: HTTP, WAP, TCP/IP, HTML, HDML, Java y otros, a partir de un browser o equivalente.

La tendencia de los clientes es mantener la menor cantidad posible de código –en la idea de cliente delgado– demandando a los servidores los programas a ejecutarse para otorgar un determinado servicio a un usuario o pidiendo la ejecución del código en el mismo servidor. Una tecnología que facilita la primera posibilidad es la de las applets en Java, que explicamos más adelante. Alternativamente a Java, se pueden utilizar JavaScript, VBScript y otros lenguajes –que no describiremos aquí– para los mismos fines que persiguen la applets.

4.2.1. **Applets**

Las applets son programas escritos en el lenguaje Java que pueden insertarse dentro de páginas Web desarrolladas en HTML o ejecutarse en forma independiente. Las applets persiguen hacer dinámicas las páginas, en el sentido de permitir que un usuario interactúe con ellas. Por ejemplo, una típica página dinámica es una que permite que un usuario ingrese sus datos para realizar una determinada compra –RUT, nombre, dirección, número y emisor de tarjeta de crédito, etc.–, valida tal información y la ingresa a una base de datos.

Las applets se insertan en el código de una página HTML por medio de la siguiente directiva:

```
<APPLET></APPLET>
```

Esta señala la ejecución de un programa externo a la página escrito en Java. La directiva tiene diferentes parámetros dependiendo de su tipo. Por ejemplo, podría tener como parámetro la dirección o URL donde se encuentra le programa, el nombre del programa y otros más específicos de la aplicación.

Tanto la página Web como las applets asociadas a ella residen en uno o más servidores, siendo la página requerida por su URL por parte del usuario, invocando ésta las applets que contiene, por medio de sus direcciones.

Una applet no está en lenguaje fuente Java, sino que en la forma de un byte code que debe ser interpretado por el cliente. Este byte code es una especie de precompilado que es independiente de la plataforma de ejecución; vale decir el mismo byte code puede ser ejecutado en un PC con Windows o NT, un MAC, un equipo con UNIX o LINUX, etc.

Para que un cliente pueda ejecutar el byte code debe contar con una máquina virtual Java, la cual es una especie de intérprete del byte code, que es particular para cada tipo de equipo. Al emplearse la modalidad de interpretación, la ejecución del código puede ser lenta, por lo cual existe la alternativa de usar compiladores rápidos de Java en el cliente, que también son particulares de cada máquina.

Es evidente que la modalidad anterior provee una gran portabilidad y flexibilidad de uso para una applet, ya que puede desarrollarse una sola vez y mantenerse en un sitio central y distribuirse por la red a múltiples puntos de ejecución de la misma en los más variados tipos de computadores. Esto contribuye a disminuir los costos de desarrollo y mantención y a facilitar la administración de los clientes, particularmente de soporte y actualización del software, reduciendo el costo total de mantener tales clientes.

4.2.2. Cliente HTML

Un cliente HTML o delgado es uno que sólo despliega gráficamente las páginas que se generan en un servidor. Vale decir, toda la lógica de la interfaz con el usuario es generada en el servidor.

Esta es la modalidad más usada en Internet hoy día y la relación Cliente/Servidor es la que se describe en la Figura 4.12, entregando el servidor, como ya se indicó, sucesivas páginas que van satisfaciendo la necesidad de un cierto usuario en forma progresiva.

Conviene hacer notar que, en este caso, la lógica del negocio será típicamente ejecutada en el servidor. Por ejemplo, al acceder un cliente a una página de oferta de productos y entregar ciertos datos necesarios para adquirir un producto, el servidor recibirá este requerimiento; ejecutará toda una lógica de negocio para verificar la validez del cliente y de su medio de pago y la disponibilidad del producto; y generará la gráfica de una nueva página –utilizando una cierta lógica de interfaz– que se enviará al cliente para su ejecución. Esta página le informará al cliente el resultado de su petición de compra. Un resumen de esta operatoria se grafica en la Figura 4.14.

Esta manera de operar tiene la ventaja de poder correr una aplicación e-Business utilizando cualquier cliente que pueda ejecutar páginas HTML y, posiblemente, applets, JavaScript u otro. Además, el código que se ejecuta en el cliente es pequeño y se baja rápidamente. Por último, el código que se envía al cliente se puede ajustar basado en sus características.

Como desventaja, se puede mencionar la dificultad de crear aplicaciones altamente interactivas. Sin embargo, el uso juicioso de pequeñas applets, que proveen ricos elementos de interfaz al usuario, puede superar esta limitación.

4.2.3. **Cliente de aplicación**

Esta modalidad es más parecida en espíritu a las soluciones tradicionales C/S con cliente grueso. Esto implica que el cliente genera y ejecuta toda la lógica de la interfaz –vale decir, genera las páginas Web– para presentación al usuario. Esta presentación, además de ser hecha por medio de un browser, puede también ser realizada con una aplicación Java o cualquier otra herramienta de generación de GUI. La interacción entre el cliente y el servidor es a través de páginas HTML, que contienen applets, scrip o algún otro mecanismo que permita ejecutar lógica en el cliente. La operatoria de esta modalidad se resume en la Figura 4.15.

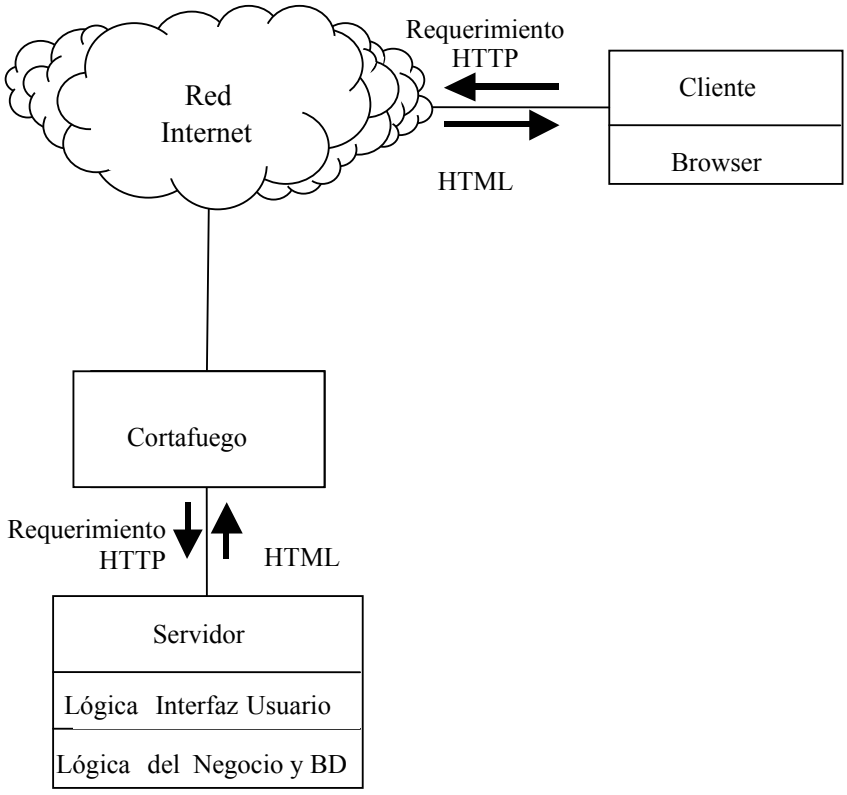


Figura 4.14. Funcionamiento cliente HTML

Esta modalidad de cliente se utiliza principalmente en aplicaciones del tipo Intranet, donde existe, habitualmente, una cultura del tipo C/S, siendo posible, incluso, darle a una aplicación una presentación del tipo Windows tradicional. Esto facilita la programación, ya que normalmente existen recursos humanos con experiencia en aplicaciones C/S. Además se reduce la carga del servidor al descentralizarse parte del procesamiento.

La desventaja de esta modalidad es que el código que se corre en el cliente es más voluminoso y más lento de cargar y ejecutar, lo cual enlentece el servicio al usuario. Si se quiere obviar esto con clientes más grandes y rápidos, implica que se encarece el costo de la aplicación.

4.3. **Servidores Web y de Aplicaciones**

Como dijimos anteriormente, la típica interacción entre un cliente y un servidor en una aplicación Web, es que el cliente solicite, a través del URL, una página inicial. En tal página el usuario invoca alguna opción, originando un requerimiento por una nueva página al servidor. Esta se genera en el servidor –implementando alguna lógica que produce los resultados que pide el cliente y la presentación– y se envía al cliente, el cual la despliega al usuario. Posiblemente, en esta página hay nuevas opciones que dan origen a nuevos requerimientos, lo que implica la repetición de lo anterior y así sucesivamente hasta que el usuario queda satisfecho y termina la sesión. Esta interacción se muestra en la Figura 4.12. En su versión más primaria, donde el servicio se reduce a acceso y búsqueda de páginas, éstos se llaman servidores Web, los cuales son los más populares en la práctica, particularmente para productos y servicios de información, como periódicos y revistas en línea, información financiera, portales, servicios búsqueda, como Google, etc.

En casos donde se requiere procesar transacciones de venta de productos o servicios –pasajes, entradas a espectáculos, libros, videos, etc.– y hay que implementar una lógica compleja del negocio, se requiere un servidor que complemente o reemplace a un servidor Web. Este es un Servidor de Aplicación– el cual, típicamente, contiene los componentes e interacciones que se muestran en la Figura 4.16, que describimos a continuación.

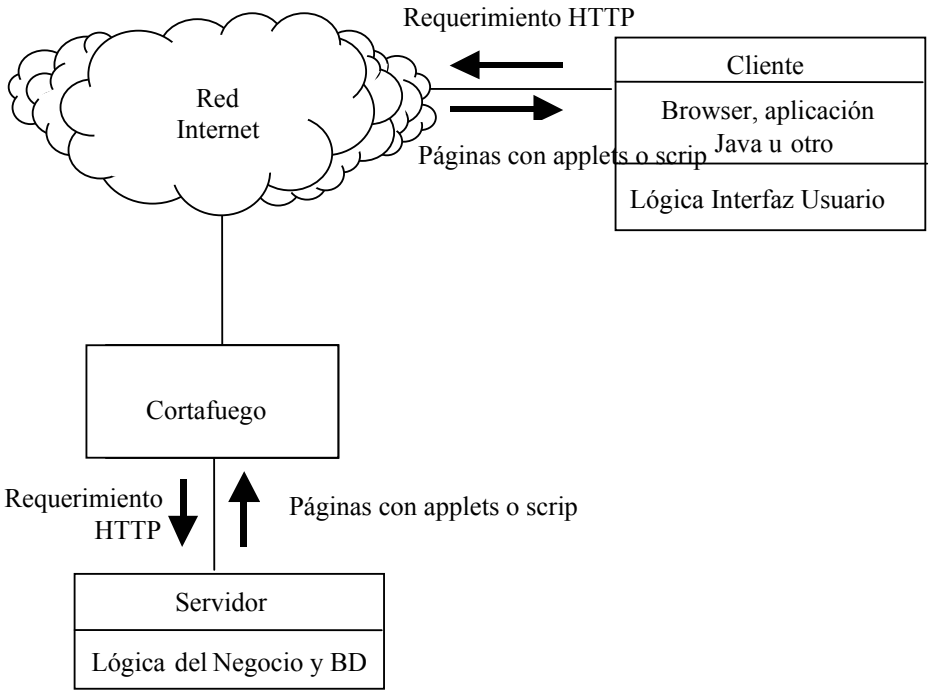


Figura 4.15. Funcionamiento cliente de aplicación

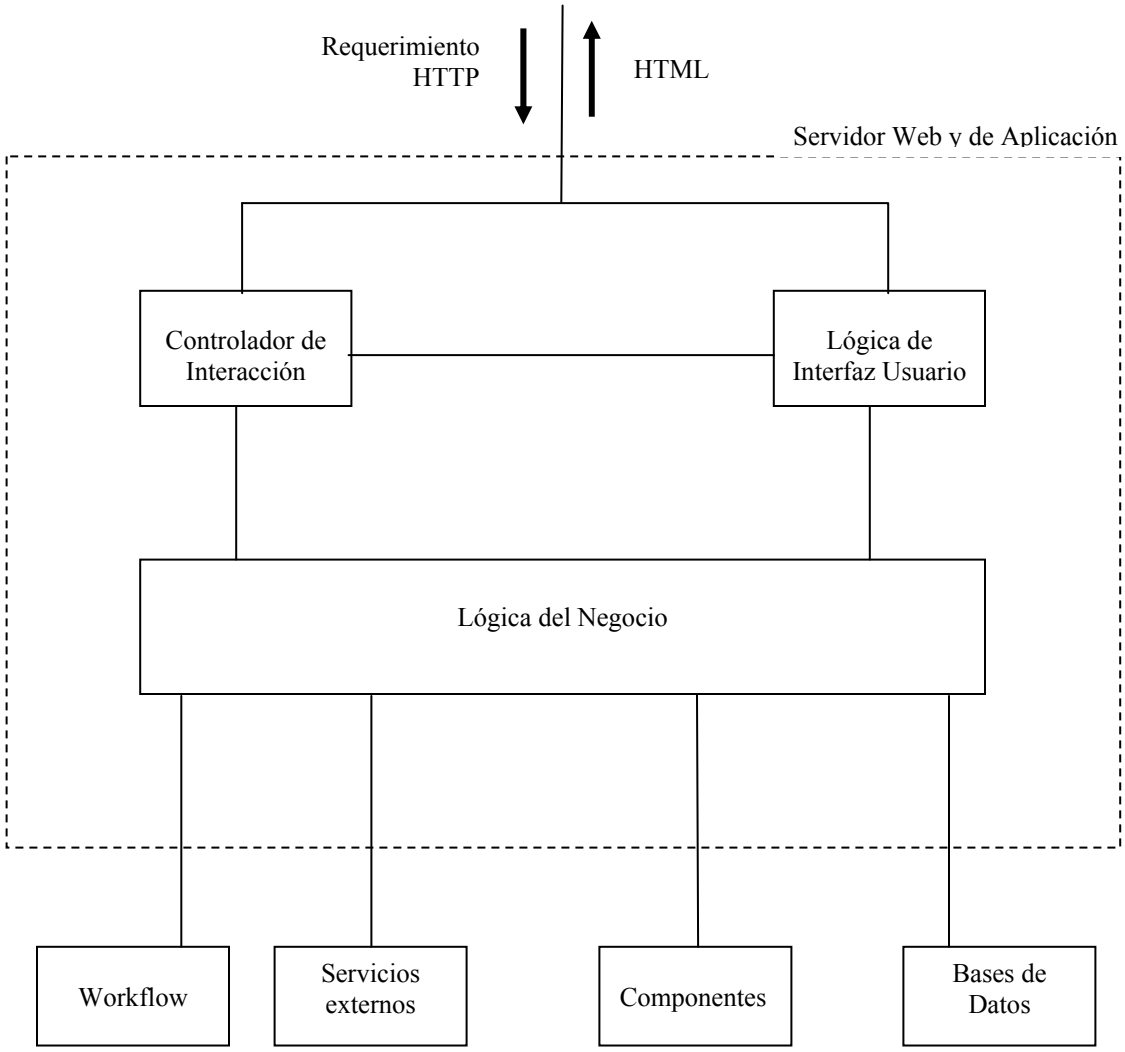


Figura 4.16. Componentes e interacción en un Servidor de Aplicación

En primer lugar, tenemos el componente Lógica de Interfaz Usuario, que tiene como función construir las páginas HTML, determinando la manera de presentar los resultados de la interacción Cliente (Requerimiento HTTP)/Servidor, los cuales pueden incluir la invocación del componente Lógica del Negocio. Este componente tiene como función procesar los requerimientos del cliente –en el caso de que haya una lógica involucrada en ellos– generando ciertos resultados necesarios para crear la página HTML de respuesta, por parte del componente anterior. Por ejemplo, una lógica que actualiza una base de datos con la información de un cliente y/o algún pedido realizado por éste y, como resultado, agrega un ítem a una carro de compra, o una lógica que transfiere –a petición de un usuario– fondos de una cuenta a otra en una banco.

El tercer componente es el Controlador de Interacción, que dirige a los otros componentes. Por ejemplo, recibe un Requerimiento HTTP, selecciona la lógica del negocio que debe ejecutarse y, basado en los resultados de la ejecución de ésta, selecciona la lógica adecuada para generar la presentación (página HTML) al usuario.

Describimos a continuación, en más detalle, los componentes bosquejados, otros elementos complementarios a ellos, y software genérico que permite la implementación de los componentes.

4.3.1. Controlador de Interacción

Este componente se puede visualizar como el código que permite ligar la lógica del negocio con los requerimientos por páginas Web dinámicas por parte del cliente. Como tal, su función principal es mapear un requerimiento expresado en el protocolo HTTP a los parámetros que requiere la ejecución de la Lógica del Negocio necesaria para satisfacer tal requerimiento. La lógica puede ser genérica, en el sentido de que es utilizada por muchas aplicaciones diferentes. También debe ordenar la ejecución de los elementos de la lógica –ya que pueden haber varios trozos de lógica a ser ejecutados en una cierta secuencia. Por último, debe invocar –traspasándole los resultados de la ejecución de la lógica– y delegar en la Lógica de Interfaz Usuario la creación de la página que será retornada al cliente.

Hay diversos productos de software que permiten implementar este código, tales como Servlets Java, JPS (Java Server Pages), CGI, ASP y otros. Algunos de éstos se describirán más adelante.

4.3.2. Lógica de Interfaz Usuario

Esta lógica es la encargada de generar las páginas HTML que serán retornadas al cliente. Para realizar su función, recibe datos dinámicos del Controlador de Interacción o accede directamente a la Lógica del Negocio para obtener los resultados que necesita para crear una página. Una vez que ha obtenido la información requerida, la Lógica de Interfaz Usuario formatea los datos, usando algún mecanismo simple de generación de HTML –como un lenguaje script–, o recurriendo a clases generales reusables que aceptan datos como parámetros y generan HTML.

La Lógica de Interfaz Usuario puede implementarse, también, con diversas herramientas y lenguajes –Servlets, JSP, JavaScrip, VBscript, etc.–, algunos de los cuales se describirán más adelante.

4.3.3. Lógica del Negocio

La Lógica del Negocio es la que satisface, en último término, un requerimiento de un usuario. Los requerimientos pueden variar dentro de un amplio rango, desde asegurar la integridad de las transacciones, accesar rápidamente en una Base de Datos a la información necesaria para una aplicación, apoyar la coordinación entre las diferentes actividades de un workflow de un proceso e integrar los módulos de una nueva aplicación con los sistemas existentes. Para poder realizar esta variedad de funciones, la lógica del negocio debe contener los servicios que se muestran en la Figura 4.17 y que explicamos a continuación.

Los Servicios de Lenguaje proveen una plataforma para desarrollar la lógica. Esta debe ser, deseablemente, encapsulada en clases independientes de los otros detalles de la aplicación Web, de tal manera que los que la generan –que, en muchos casos, serán especialistas en el negocio y no en la tecnología– puedan centrarse sólo en la lógica, evitando tener que manejar TI compleja. Esto también facilita el desarrollo en paralelo de módulos de

lógica independientes y en forma separada del desarrollo de, por ejemplo, las páginas Web y las Bases de Datos, ya que la comunicación entre las clases que contienen la lógica y los otros componentes es paramétrica. Obviamente, esto también facilita la mantención de la lógica. Además de lo anterior, los Servicios de Lenguaje deben permitir acceso a las Bases de Datos propias de la aplicación y de otros sistemas de la organización; acceso a directorios; seguridad; acceso y creación de correo; y manejo de transacciones y mensajes. Tecnologías que se pueden utilizar para proveer los servicios bosquejados son las EJB (Enterprise Java Beans), servlets, JDBC (Java Data Base Connectivity), Java SSL, JavaMail API, Java Transaction API, y JMS (Java Messaging Services); Visual Basic, ODBC, DCOM y COM+. Algunas de estas tecnologías se describirán más adelante.

Los servicios de Base de Datos se refieren a las facilidades que permiten crear Bases de Datos orientadas a objetos del tipo multimedia o de registros (relacionales) tradicionales, necesarias para apoyar directamente la aplicación Web. Además, incluyen la ejecución de llamadas a Bases de Datos preexistentes –para actualizar o requerir información– originadas en la lógica escrita en los lenguajes del párrafo anterior. Asimismo, deben permitir la ejecución de lógica –también proveniente de los lenguajes– sobre las Bases de Datos de la aplicación o de otros sistemas, en la forma de procedimientos almacenados o funciones definidas por el usuario. La tecnología apropiada para implementar estas facilidades es un Administrador de Bases de Datos Relacionales con extensión a objetos, tales como DB2 Universal DataBase y Jasmine.

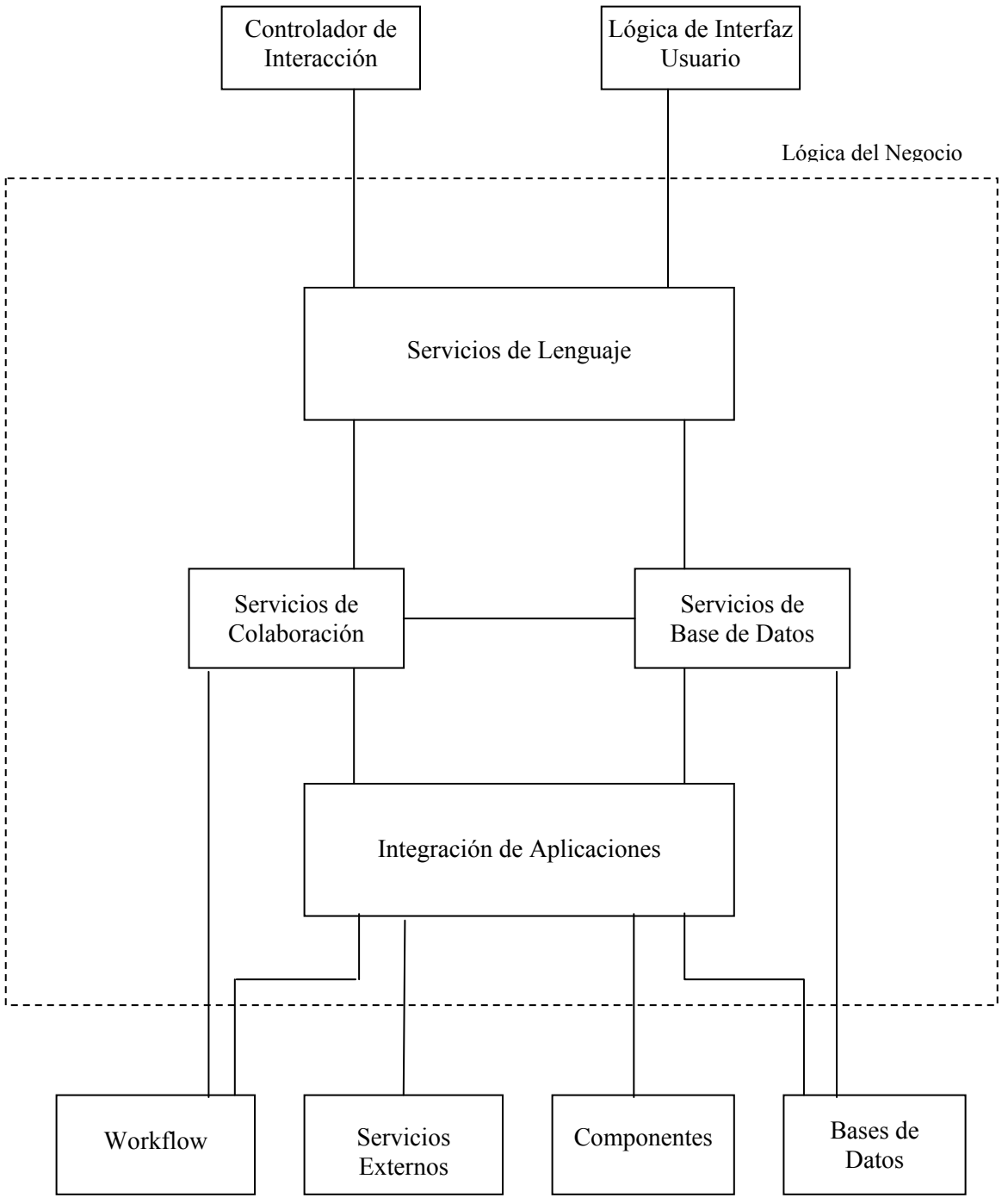


Figura 4.17. Componentes y relaciones de la Lógica del Negocio

Los Servicios de Colaboración tienen como propósito apoyar la coordinación de las actividades que participan en el workflow de un proceso. Por ejemplo, la ejecución de la lógica para apoyar una determinada actividad –digamos atención automática al cliente por Internet, verificando su crédito y la disponibilidad del producto que solicita– que puede gatillar la actuación de otra actividad en la secuencia del workflow –digamos la entrega del producto– la cual debe ser informada de alguna manera. Estos servicios permiten ejecutar la actuación coordinada de tales actividades, por medio de que la lógica del negocio interactúe con un software del tipo workflow –a través de estas facilidades de colaboración–, para que éste genere mensajes o documentos a las actividades que deben actuar o actualice las listas de tareas pendientes de ellas y otra información relevante para la actuación. Típicamente, la colaboración se realiza por medio de un API que permite conexión desde la lógica del negocio al workflow; por ejemplo, el API de Lotus Domino o Exchange.

La Integración de Aplicaciones tiene como propósito comunicar las aplicaciones Web con los diversos sistemas ya existentes dentro de una organización y de facilitar la operación coordinada y no duplicada de diversas aplicaciones desarrolladas para diferentes procesos de negocios. Para ello se debe contar, en primer lugar, con conectores, los cuales son software del tipo gateway que permiten unir diversas aplicaciones Web –que pueden apoyar procesos de negocios diferentes, por ejemplo operaciones y finanzas– que usan protocolos de comunicaciones, posiblemente, distintos. Además, deben existir facilidades de intercambio de mensajes entre aplicaciones por medio de comunicación directa, no necesariamente en línea. Asimismo, debe contarse con servicios de integración de procesos, de una empresa o de diferentes empresas, a través de la comunicación por medio de mensajes entre los workflow respectivos, en base a un centro de intercambio de mensajes, que entiende la lógica de los flujos de los diferentes procesos. Por último, la disponibilidad de integración de componentes es fundamental. Estos son objetos que encapsulan, principalmente, lógica que puede ser reusada en múltiples aplicaciones. Por lo tanto, existe la posibilidad de que una nueva aplicación se construya accediendo a componentes de varias aplicaciones ya existentes y almacenadas en lugares diferentes. La posibilidad de definir, indexar y acceder a tales componentes en una red es la que hace factible la integración. Productos que permiten llevar

a la práctica las integraciones definidas son, por ejemplo, librerías Java, MQSeries, XML e implementaciones de CORBA, como IIOP.

4.4. Servidor de Datos

Un Servidor de Datos es una combinación de hardware y software que permite almacenar una Base de Datos, ya sea legacy –por ejemplo DATACOM DB, IMS, DL1, archivos VSAM–, relacional –por ejemplo, ORACLE, SQL Server, DB2–, u orientada a objetos –por ejemplo Jasmine.

En cuanto al hardware, existen soluciones relativamente estándares, como lo son las que corren las diferentes variedades de UNIX –Solaris, AIX, LINUX, etc.– y otros propietarias, como NT, OS390, OS400, etc. Sin embargo, algunos software SABD resuelven en parte el problema de soluciones propietarias, al correr en diversas plataformas de hardware-sistema operativo. Por ejemplo, el SABDR ORACLE corre en casi todas las variedades de UNIX, NT, OS 390 y otros.

En una aplicación e-Business, un Servidor de Datos puede contener las Bases de Datos propias de la aplicación o las Bases de Datos que pertenecen a las aplicaciones o sistemas tradicionales de la organización –Ventas, Contabilidad/Finanzas, Producción, etc. En este último caso, las Bases de Datos no han sido diseñadas para la aplicación e-Business, por lo cual ésta tiene que utilizar en forma selectiva la información disponible, tratando de superar cualquier deficiencia que ellas presenten, como calidad o actualidad de la misma.

La situación más favorable de aplicaciones tradicionales es aquélla en que éstas han sido desarrolladas en base a un esquema C/S de tres niveles, donde el Servidor de Datos se define en forma muy similar al de una aplicación e-Business con la arquitectura que hemos identificado. Sin embargo, persiste el problema de que la información de estas Bases de Datos sea la requerida por la aplicación e-Business. De no ser así, habitualmente habrá que generar Bases de Datos complementarias como propias de tal aplicación.

Es obvio que en situaciones donde se está creando un e-Business desde cero –típicamente una empresa punto-com–, las Bases de Datos serán diseñadas e implementadas de una manera absolutamente funcional a la aplicación.

En el caso en que ya existen Bases de Datos que es conveniente utilizar, esto se facilita con los servicios descritos en el Punto 4.3.3. Particularmente importante son los middleware que permiten acceder desde los programas que implementan la lógica del negocio a las Bases de Datos, tales como JDBC, ODBC, CGI más un lenguaje como Perl, ASP, etc.

5. Tecnología para Implementar la Arquitectura

Describimos, a continuación, varias tecnologías y software que permiten llevar a la práctica la arquitectura de una aplicación e-Business, particularmente la parte correspondiente al servidor.

5.1. Servlet

Una Servlet es una clase escrita en Java, residente en un servidor, típicamente de aplicación, que tiene como propósito satisfacer ciertas peticiones que realiza un equipo cliente. La relación cliente/servidor se establece de la siguiente manera.

En el código HTML de una página que se ejecuta en un cliente existe, en muchos casos, un botón que, al ser pulsado, gatilla una acción que invoca una Servlet ubicada en el servidor. Esto se hace por medio del encabezamiento de un *form* HTML, definiéndose en el cuerpo de éste el detalle de la interacción; por ejemplo, qué datos deberá ingresar el usuario, la instrucción de pulsar el botón y el envío al servidor una vez hecho esto.

Para poder desarrollar y ejecutar una Servlet, debe contarse con un servidor que tenga un soporte nativo de esta tecnología, además de un kit de desarrollo de Java. Estas contienen el API de las Servlet, que incluyen métodos que aceleran la programación, por medio del uso de mecanismos de herencia y especialización.

Una Servlet es una alternativa más estandarizada, portable y versátil a los programas CGI para implementar el controlador de interacciones y la lógica del negocio –particularmente una compleja– en un Servidor de Aplicaciones, incluyendo en esta última la conexión a Bases de Datos por medio de JDBC.

5.2. JSP (Java Server Page)

Este es un lenguaje que mezcla HTML con Java, alternativo a una Servlet, pero igualmente recomendable, para generar y ejecutar lógica –tanto de generación de páginas como del negocio– en el servidor, en respuesta a una petición de un cliente HTML. Ofrece mecanismos simples para escribir una pequeña cantidad de código, mucho menor que la de una Servlet –particularmente, en la generación de código HTML–, y no requiere una mecánica formal de programación ni compilación. Es una herramienta muy poderosa que provee el poder equivalente de un buen lenguaje de programación dentro de una página HTML. El código en un archivo JSP se convierte automáticamente en una Servlet Java cuando se ejecuta.

Las JSP pueden usarse en combinación una Servlet, asignándole a la primera la interacción con el cliente y la generación de páginas HTML, y a la segunda, la ejecución de lógica compleja, por invocación de la primera. También puede ser al revés, dejándole a la Servlet la lógica de atención al cliente –incluido todo el control de interacción, particularmente cuando hay muchas clases participantes y lógica compleja– y a las JSP, sólo la generación de páginas HTML.

5.3. Componentes distribuidos

Como hemos visto, HTTP no funciona con una conexión explícita (es "connectionless") entre los servidores de la red. Esto impide la verdadera distribución de objetos en la red, a base de los esquemas actualmente existentes. Es por esto que ellos deben considerarse como complementarios a Internet, en el sentido que requieren otros protocolos para poder funcionar. Los principales de estos esquemas son RMI, CORBA y DCOM.

5.3.1. **RMI**

RMI (Remote Method Invocation) es el estándar Java para objetos distribuidos. Permite a una clase Java comunicarse con otra clase Java que puede estar ubicada en una máquina físicamente diferente.

RMI introduce dos nuevos tipos de objetos: stub y skeleton. Un stub es un objeto que reside en un cliente y representa a un objeto remoto, y se ejecuta en éste. Un skeleton administra todos los detalles asociados al hecho de ser remoto, en particular a comunicaciones con objetos que residen en otra máquina (cliente), y reside en el servidor. El esquema de funcionamiento puede representarse como se muestra en la Figura 5.1. Un ejemplo de uso de esta arquitectura para procesar una applet que pide un servicio de un objeto remoto, se da en la Figura 5.2.

Nótese que RMI necesita una infraestructura particular de comunicaciones.

Esta arquitectura –y correspondiente infraestructura– aisla al diseñador y al desarrollador de los detalles de las comunicaciones remotas.

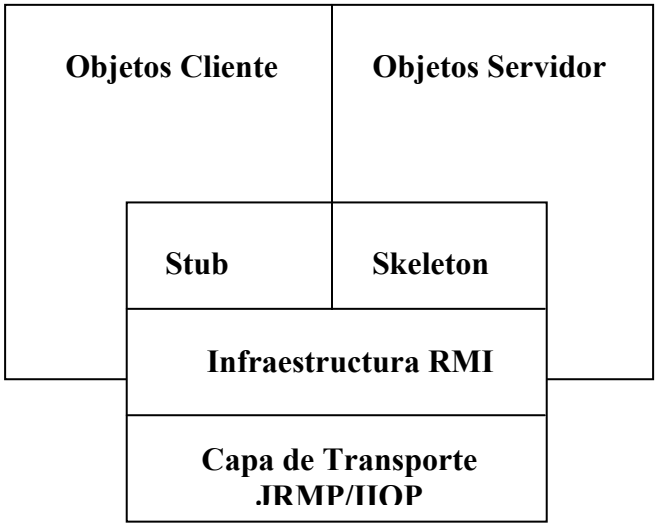


Figura 5.1. Arquitectura de RMI

5.3.2. **CORBA**

Esta es una especificación estándar del OMG (Object Management Group), un grupo de empresas de la industria TI que persigue que sus miembros produzcan software compatible.

Dado que CORBA es sólo una especificación y no un producto, tiene que implementarse a través del software de proveedores particulares.

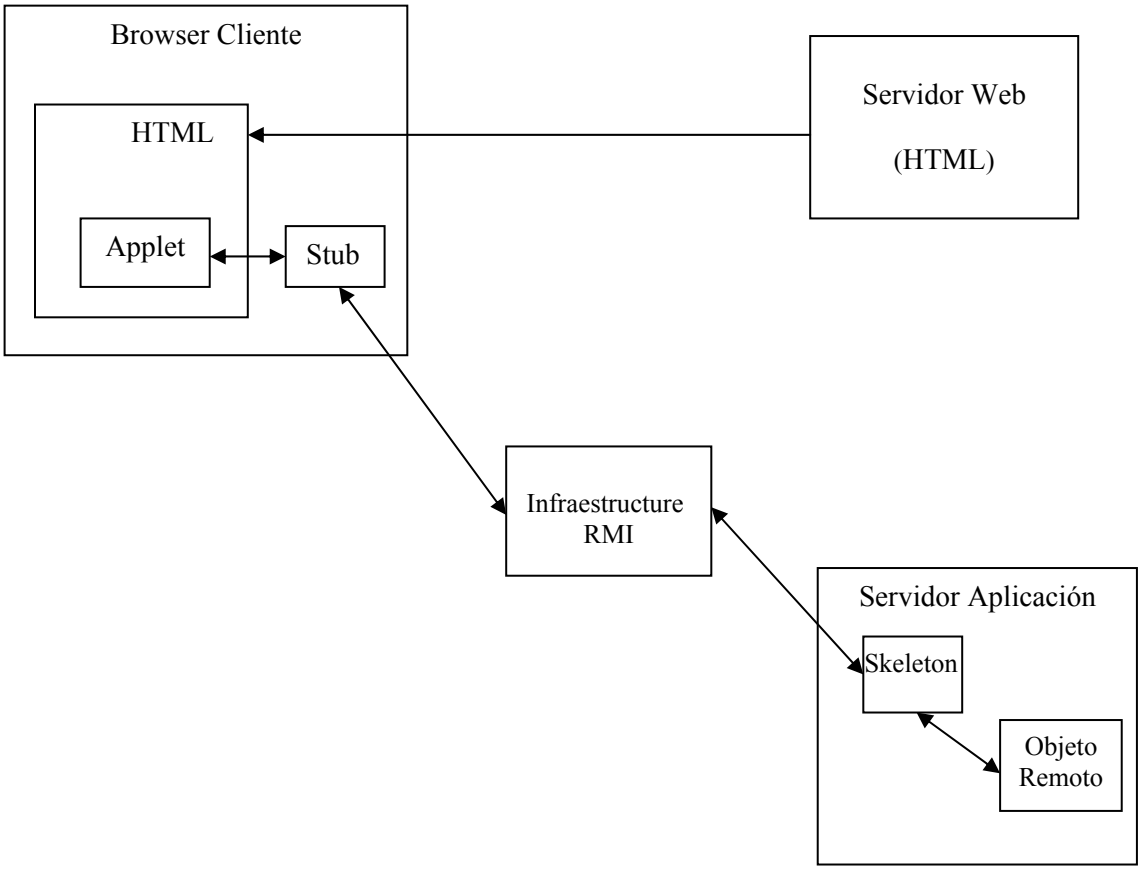


Figura 5.2. Applets con RMI

A diferencia de RMI –que requiere objetos escritos en Java– CORBA permite que los objetos que se comunican a través de él se escriban en un lenguaje cualquiera. Esto se consigue a través de un lenguaje de definición de la interfaz, IDL (Interface Definition Language). Esta especificación se puede implementar con diferentes lenguajes.

El middleware que implementa la comunicación entre objetos se llama ORM (Object Request Broker) en CORBA. Un ORB se instala en el cliente y otro en el servidor. Los clientes pueden, entonces, crear instancias de objetos remotos, por medio de los servicios de CORBA, e invocar métodos de ellos, tal como se muestra en la Figura 5.3.

Las interfaces en CORBA pueden ser stub y dinámica. Un stub, al igual que en RMI, se instala en el cliente y contiene métodos idénticos a la interfaz pública del objeto remoto que representa. Una invocación dinámica es una interfaz genérica a un objeto, que accede a un repositorio de servicios de interfaces, que representa los componentes de una interfaz como objetos, y permite, por lo tanto, un acceso en el momento de ejecución (run time).

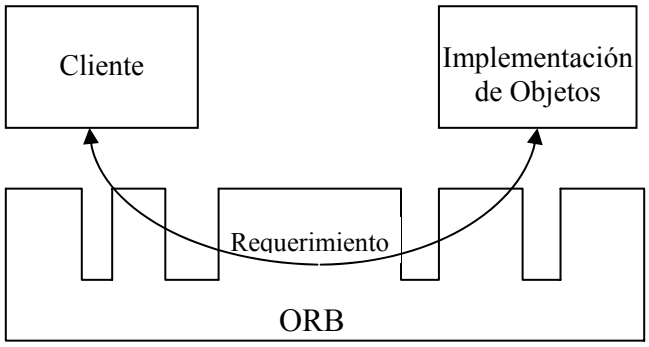


Figura 5.3. ORB CORBA

Los ORB de diferentes proveedores interoperan usando el mismo protocolo de transporte. Este es el IIOP (Internet Inter-ORB Protocol).

El último Java Development Kit (JDK), versión 1.2, incluye el ORB de CORBA, lo cual permite evitar la implementación de otro ORB en los clientes, con la condición de escribir el código de éstos en Java.

La misma configuración de la Figura 5.2 puede implementarse con CORBA, reemplazando el stub por el ORB de JDK 1.2, una implementación de IIOP en vez de RMI, y un ORB apropiado en el servidor, en vez del skeleton.

5.3.3. **DCOM**

Esta es la solución Microsoft para los objetos distribuidos. DCOM (Distributed Component Object Model) es una extensión de COM.

A la implementación de objetos COM se le asigna un CLSID (Special Class Identifiers) por el sistema operativo. Cuando la máquina tiene soporte DCOM, es posible ubicar a tales objetos en un servidor remoto.

DCOM funciona como se muestra en la Figura 5.4. Usa, al igual que RMI y CORBA, objetos proxi y stub que actúan como interfaces entre los programas del cliente y la implementación en el servidor. Estos son invisibles para el desarrollador, ya que son provistos por DCOM. SCM (Service Control Manager) es el responsable por ubicar y crear las instancias de objetos en la máquina local o en algún lugar de la red, si es necesario.

Una vez que un cliente tiene una referencia a un objeto (Proxy), puede invocar sus métodos. La comunicación entre cliente y servidor es administrada por los RPC (Remote Procedure Call) de DCE (Distribute Computing Environment).

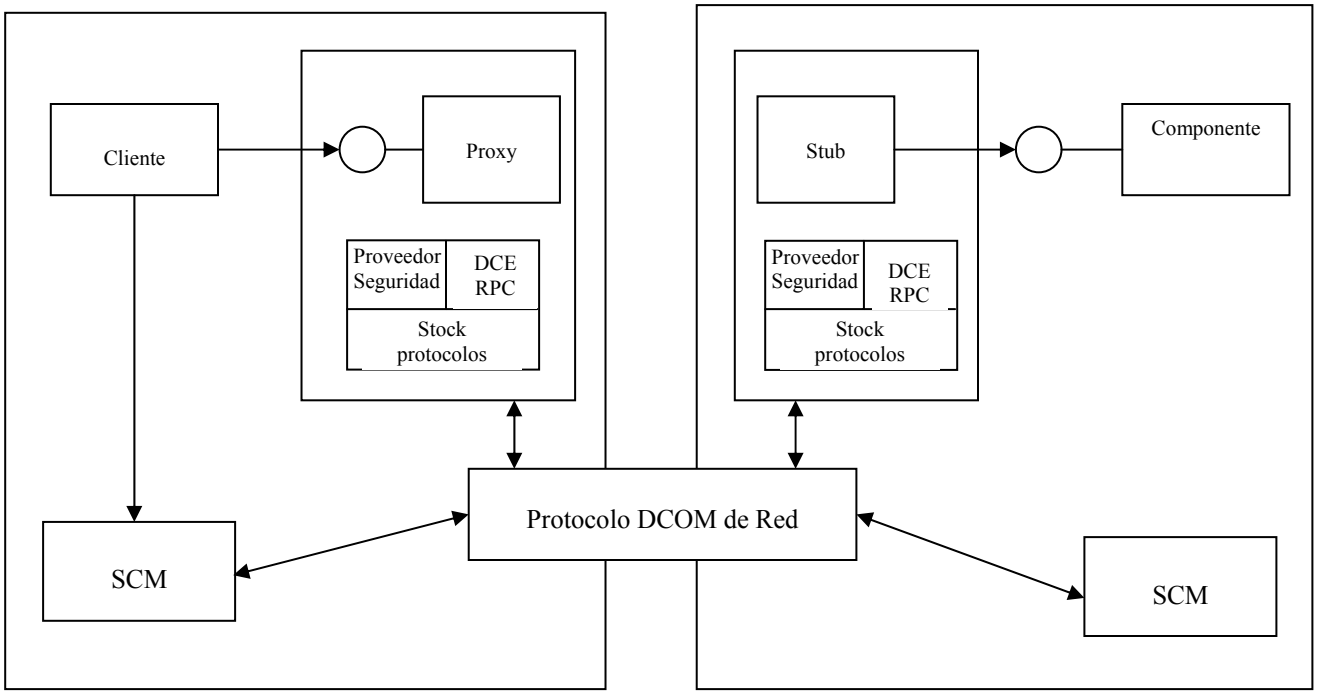


Figura 5.4. Arquitectura DCOM

5.3.4. Java Bean y EJB

Los Java Bean y los EJB (Enterprise Java Bean) proveen un modelo estándar para desarrollar componentes estandarizados reusables, apoyados en un conjunto de herramientas. Los Java Bean son clases Java con ciertas propiedades. Una de ellas permite generar una instancia de una clase Java Bean y serializarla –codificándola como una secuencia de bytes– y almacenarla en un archivo o base de datos. Estas secuencias se denominan igual que las clases.

Java permite deserializarlos y cargarlos cuando una aplicación requiere –en la ejecución– de alguna constante definida por ellas. Por ejemplo, el ID de una Base de Datos. Una Java Bean contiene las propiedades de una constante necesaria para ejecutar la aplicación. Esta propiedad hace los Java Bean recomendables para desarrollar la lógica de procesamiento del controlador de interacciones.

Las EJB están orientadas a la lógica del negocio. Son también componentes reusables que encapsulan lógica y que son ejecutadas en un contenedor (container). Este provee funciones de administración y control de los componentes; por ejemplo, seguridad y soporte de transacciones, y acceso al sistema operativo y secuencias (threads) de ejecución.

Las ventajas de las EJB es que el contenedor provee un ambiente estándar que aisla el desarrollo de la lógica de la peculiaridades del software/hardware de implementación, simplificándolo. También permiten que los componentes se adapten a las características de los servicios de Bases de Datos y transacciones que existen en un caso particular, evitando que los desarrolladores conozcan anticipadamente estas características de ejecución.

Los EJB pueden ser invocados a través de algunos de los métodos de procesamiento distribuidos en Internet, explicados en los puntos anteriores, pero también pueden ser utilizados dentro de un mismo servidor por invocación directa. Estas posibilidades se grafican en la Figura 5.5.

O sea, una EJB puede ser utilizada dentro de una aplicación tradicional e-Business o accesada en forma directa por medio de una infraestructura ad-hoc, para proveer algún tipo de servicio al cliente.

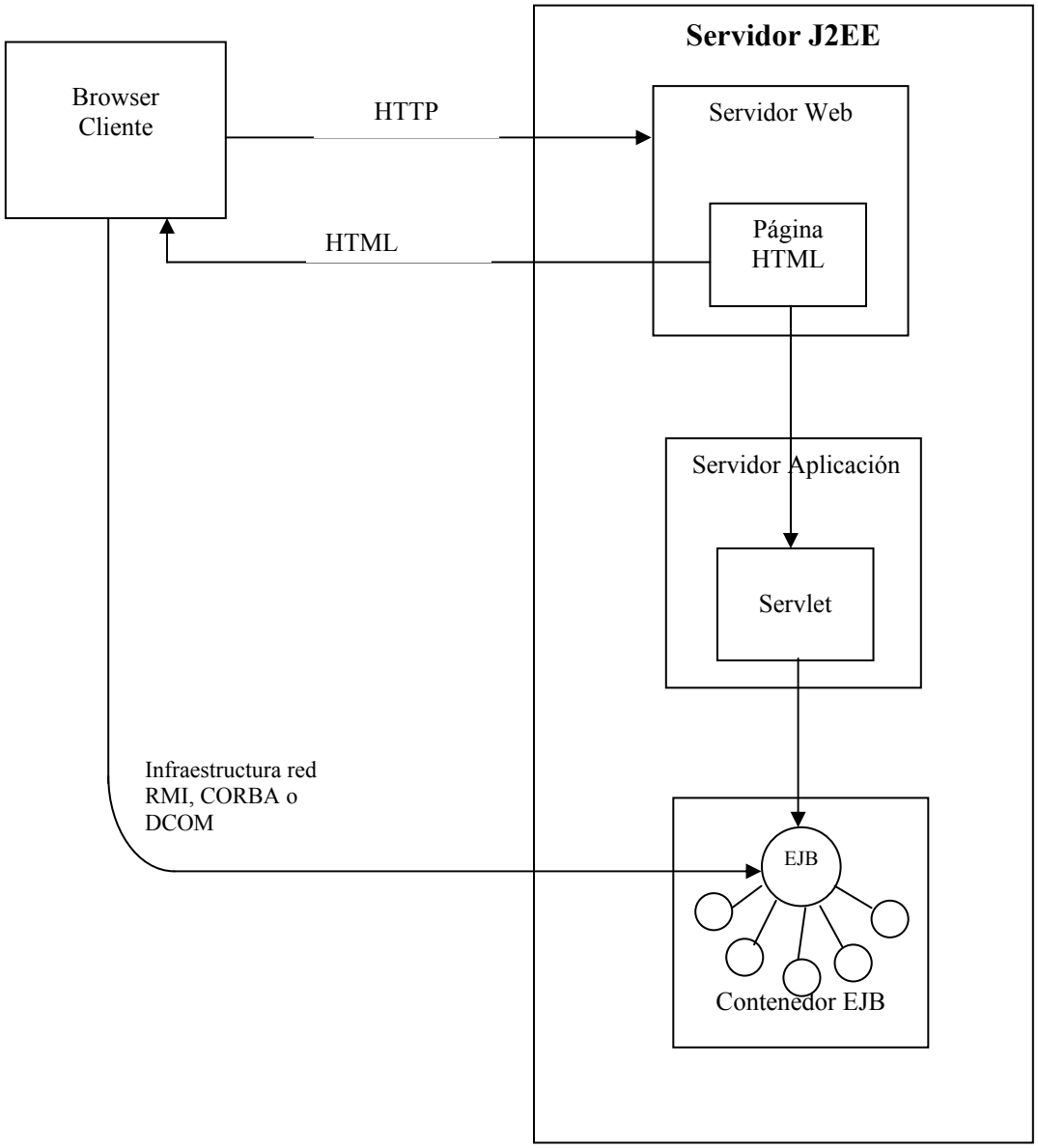


Figura 5.5. Accesos a EJB

6. Arquitecturas de Aplicaciones e-Business

Con lo tratado hasta este punto, estamos en condiciones de sintetizar las posibles estructuras de una aplicación e-Business en forma genérica, para que sirvan como patrones en cualquier diseño particular que uno quiera realizar.

Para presentar las arquitecturas, utilizaremos un esquema que también usamos para modelar procesos de negocios. Este está basado en el estándar IDEF0 y consiste en identificar las actividades o funciones que participan en la arquitectura o proceso y las relaciones entre éstas por medio de flujos de información.

La representación de la primera arquitectura se muestra en la Figura 6.1. Esta debe interpretarse como un módulo típico que se replica muchas veces en una aplicación específica. Esto se puede apreciar al observar que existe una sola actividad, que es apoyada por tecnología, la cual representa a un usuario de la empresa o externo a ella; por ejemplo, un usuario interno que quiere apoyarse en una aplicación e-Business para determinar qué productos se requiere adquirir para abastecer las operaciones de la empresa o una persona externa que desea comprar por Internet productos de nuestra empresa.

Por supuesto, una aplicación e-Business real tendrá muchas actividades, como las anteriormente ejemplificadas, entrelazadas por medio de flujos de información, que típicamente corresponderán a un cierto proceso de negocio. Por ejemplo, el flujo de satisfacción de un requerimiento por productos a través de Internet incluye las actividades entrelazadas de atención al cliente, evaluación del pedido, obtención de productos, programación de despacho y ejecución de despacho, como se resume para el caso de Amazon.com, en la Figura 3.3. Cada una de estas actividades tendrá un apoyo (o automatización total) que será una instancia del módulo genérico de la Figura 6.1. Tal apoyo puede contar con una parte o la totalidad de los elementos de la arquitectura, dependiendo de la complejidad de la aplicación. Así habrán casos simples en que sólo se provee acceso a páginas y consulta de éstas, en las cuales bastará con un servidor Web que implementará una cierta “Lógica de Interfaz Usuario”. En el otro extremo, en situaciones en las cuales se realizan transacciones comerciales se implementarán todos los elementos –incluidas las Bases

de Datos– y existirán, posiblemente, un servidor Web para el manejo de las páginas y un Servidor de Aplicación para la lógica del negocio y acceso a Bases de Datos. Vale decir la arquitectura se puede particionar y distribuir en varios equipos.

También los flujos de la arquitectura existirán en un caso real de acuerdo a las características de la implementación; es decir ellos representan posibilidades que pueden darse o no en una situación específica. Por ejemplo, la interacción con otras aplicaciones sólo es relevante cuando el apoyo computacional a una actividad del negocio genera consecuencias en otras, cual es el caso de un apoyo a ventas que genera la necesidad de emitir una factura y realizar una imputación contable en otra actividad.

Ahora bien, lo anterior no significa que cada instancia del módulo dará origen a una implementación física separada. Por el contrario, en muchos casos, todas las instancias se consolidan en un solo Servidor Web y/o de Aplicación.

La arquitectura de la Figura 6.1 puede darse en dos versiones: cliente delgado y cliente grueso. En la primera, como ya se ha explicado, al browser se utiliza sólo para desplegar páginas y no ejecuta lógica alguna. Por lo tanto, el flujo "Página HTML" de la Figura 6.1 sólo contiene páginas estáticas que se despliegan en el browser. Toda la "Lógica de Interfaz Usuario" y "Lógica del Negocio" se ejecutará en un servidor Web y/o de Aplicación, usando tecnologías como JSP y Servlet. También existe la posibilidad de que existan –como parte de la "Lógica del Negocio"– objetos del negocio implementados con tecnología EJB (Enterprise Java Beans) que pueden ser accesados desde una JSP o una Servlet. Tanto éstos como las EJB pueden acceder, por medio de JDBC, a Bases de Datos de la aplicación o históricas.

En la segunda arquitectura, existe la posibilidad de ejecutar lógica compleja en el cliente, ya sea por medio de applets o algún lenguaje scrip; vale decir el flujo "Página HTML" puede contener applets, scrip en JavaScrip o VBScrip y documentos XML. Asimismo, desde el cliente se pueden referenciar objetos residentes o bajados al cliente, tales como una JavaBean y Controles Active X. En el servidor Web y/o de Aplicación también puede ejecutarse lógica invocada desde el cliente, tanto de interfaz como del negocio, usando JSP,

Servlet u otra tecnología apropiada, en forma similar al caso de cliente delgado. Particularmente relevante en este caso es JSP, ya que permite la generación eficiente de páginas HTML con scrip incorporado.

La arquitectura alternativa admite la posibilidad de que exista acceso directo desde objetos en el browser a objetos distribuidos, usando la tecnología descrita en el Punto 5.3; o sea, se utiliza tecnología de comunicación complementaria a HTTP sobre TCP/IP. Ella es similar a la arquitectura de la Figura 6.1 en su versión cliente grueso, pero con los agregados que se muestran en la Figura 6.2. La "Comunicación entre objetos" permite el establecimiento de un diálogo en cliente y servidor a través de una conexión abierta, la cual no se puede obtener a través de Internet. Esta conexión requiere tecnología adicional, como RMI, IIOP y DCOM. Ella permite la comunicación directa entre un objeto residente en el cliente –una applet o un objeto COM (Active X) por ejemplo– y otro objeto residente en un servidor, que puede ser de