Sistemas y Aplicaciones Informáticas

Tema 56. Software de Sistemas en Red. Componentes. Funciones y Estructura.

1. Ámbito de docencia.

- Implantación de aplicaciónes informáticas de gestión (ASI 2).
- Sistemas informáticos multiusuario y en red (DAI 1).
- Instalación y mantenimiento de aplicaciones ofimáticas y corporativas (ESI 1).

2. Sistemas en red. Modelos. Protocolos de red.

- Un sistema en red es aquel que está formado por dos o más ordenadores conectados entre sí para compartir información, recursos y servicios. Existen tres modelos de implementación de redes:
  - *Modelo centralizado*. Es el más antiguo, en el cual existe un ordenador central que realiza todo el procesamiento y posee todos los recursos del sistema, al que los usuarios se conectan a través de terminales. El ordenador central atiende simultáneamente a todos los terminales, que son meros dispositivos de E/S y pueden estar estar situados a gran distancia.
  - *Modelo cliente-servidor*. Basado en la existencia de uno o varios ordenadores especiales denominados servidores cuya misión es proporcionar servicios y administrar recursos comunes para que puedan ser explotados por el resto de equipos denominados clientes. Cada cliente dispone de capacidad de proceso y puede ejecutar sus propias aplicaciones.
  - *Modelo distribuido*. Formado por un conjunto de ordenadores, cada uno con su propia capacidad de procesamiento y sus propios recursos, que se conectan entre sí para crear un entorno que aparezca ante el usuario como un único sistema virtual de manera transparente. Según este modelo, para realizar una única tarea pueden involucrarse varios ordenadores.
- Además del hardware que proporciona la conexión física de los elementos de la red, es necesario que exista un conjunto de reglas y procedimientos para establecer la comunicación entre los nodos de la misma. Estas normas se denominan protocolos de red, cuya implementación da lugar al software de comunicación, que permite el acceso a los recursos de la red por los usuarios.

3. Componentes. Funciones y estructura.

3.1. Sistemas operativos.

3.1.1. Sistemas operativos de red.

3.1.1.1. Descripción. Características.

- Los sistemas operativos de red son aquellos que tienen la capacidad de interactuar con dispositivos y con sistemas operativos en otros ordenadores con el fin de administrar y explotar los recursos de una red. Originalmente necesitaban un sistema operativo de base sobre el que trabajar, pero actualmente casi todos los sistemas operativos incluyen esta funcionalidad.
- Las características que ofrece un sistema operativo de red son las siguientes:
  - *Conectividad*. Permite la comunicación de manera simultánea utilizando varios protocolos.
  - *Escalabilidad*. Garantiza el crecimiento y consistencia de la operatividad de la red.
  - *Arquitectura modular*. Permite instalar y configurar hardware y software fácilmente.
  - *Diversidad*. Da soporte a diversas plataformas dentro de la red.
  - *Recursos compartidos*. Permite compartir recursos a través de la red.
  - *Seguridad*. Protege los recursos ante accesos no autorizados.
  - *Confiabilidad*. Ofrece tolerancia a errores ante los posibles fallos de la red.

3.1.1.2. Funciones y estructura.

- El sistema operativo de red suele formar parte de un entorno cliente-servidor, y su función consiste en establecer un canal de comunicaciones entre los sistemas operativos de las estaciones de trabajo y de los servidores de una red, que puede ser de área local o de acceso remoto:
  - En el cliente, consta de un conjunto de programas que transmiten a través de la red la petición de un recurso o un servicio, localizado en un servidor conocido, utilizando los protocolos de red adecuados. Este proceso se realiza de manera transparente al usuario, que trabaja con los recursos como si fueran locales a su máquina.
  - En el servidor, otra serie de programas reciben las peticiones del cliente y proporcionan a los usuarios autorizados el acceso eficiente a los servicios solicitados, permitiendo disponer simultáneamente de los recursos no exclusivos, como los ficheros y las bases de datos, y gestionando los recursos exclusivos, como las impresoras.
- Los sistemas operativos de red más habituales tienen las siguientes estructuras:
  - *Monolítica*. Está formada por un núcleo compacto que contiene todas las rutinas del sistema, de tal forma que cada una puede llamar a cualquier otra. Se ejecutan en un único espacio de direcciones y cualquier parte del sistema operativo tiene los mismos privilegios que cualquier otra. Un ejemplo de sistema operativo de red monolítico es GNU/Linux.
  - *Microkernel*. Consiste en un núcleo mínimo, que ejecuta en modo kernel los servicios del sistema de más bajo nivel, al que se añade un conjunto de módulos autónomos, cada uno de los cuales pone a disposición del resto una serie de servicios. El proceso cliente comunica mediante mensajes con el proceso servidor, y éste realiza el trabajo y devuelve una respuesta. El núcleo se encarga de controlar la comunicación entre los clientes y los servidores. Un ejemplo de sistema operativo de red microkernel es Windows Server 2003.

3.1.2. Sistemas operativos distribuidos. Funciones y estructura.

- Son aquellos que se ejecutan en un conjunto de ordenadores conectados por una red, pero que aparecen ante los usuarios como un solo ordenador, actuando como un único procesador virtual. Cada procesador tiene su propia memoria local y su propio reloj, y se comunica con los demás a través de buses de alta velocidad o de líneas de comunicación.
- Permiten repartir los procesos entre un conjunto de procesadores, que pueden estar en el mismo equipo o en equipos distintos. Cada ordenador es un parte de un sistema operativo global y cada usuario utiliza los recursos, sin saber cuál es el ordenador que se los ha proporcionado.
- Abarcan los servicios de los sistemas operativos de red, integrando recursos en una sola máquina virtual, en la cual el usuario ya no necesita saber la ubicación de los recursos sino que los usa como si todos ellos fuesen locales. Se ejecutan núcleos idénticos en todos los procesadores del sistema, existiendo un mecanismo de comunicación global entre los procesos.

3.2. Protocolos de nivel de aplicación.

3.2.1. Descripción. Funciones y servicios de red.

- Son el conjunto de normas que controlan y coordinan el intercambio de información entre las aplicaciones de usuario en una red, tanto en la transmisión como en el control y recuperación de errores. Su implementación proporciona a dichas aplicaciones una serie de servicios de red.

- **Servicio de ficheros**. Se encarga del almacenamiento, recuperación y movimiento de ficheros de datos. Hay un conjunto de funciones que deben facilitar el servicio de ficheros tales como:
  - *Transferencia de ficheros*. Permite salvar, recuperar o mover ficheros para un cliente de la red, y acceder a los datos independientemente de su localidad física. Se apoya en un determinado sistema de seguridad de acceso a los datos para autorizar o negar una serie de acciones sobre estos en función del cliente que accede a los datos.
  - *Almacenamiento de ficheros y migración de datos*. Evita tener información redundante en diferentes componentes de la misma. Además en función de su uso e importancia se irán migrando de un soporte a otro, los mas utilizados se migrarán a soportes rápidos y online, mientras que los menos usados se pueden retirar a otro tipo de almacenamiento.
  - *Actualización y sincronización de ficheros*. Se basa en el concepto de tener una copia de un fichero en el disco local del cliente, al conectarse este a la red el sistema verifica si existe coherencia entre este fichero y el existente en la red; si no es así este servicio se encarga de sincronizar los contenidos de los dos ficheros automáticamente.
  - *Seguridad de ficheros*. Un único administrador puede efectuar copias de seguridad de ficheros de múltiples ordenadores conectados a la red, normalmente fuera de línea, para evitar perdidas innecesarias de información ante posibles deterioros.
- **Servicio de mensajería**. Incluye almacenamiento, acceso y liberación de datos, gráficos, señales de vídeo y audio. Los servicios más importantes que presta son los siguientes:
  - *Correo electrónico*. Se refiere a la trasferencia electrónica de mensajes de datos entre dos o mas ordenadores de la red, o en su defecto usuarios de la misma.
  - *Servicio de directorio*. Se encarga del mantenimiento y actualización de directorios de direcciones de ordenadores o entidades que quieran enviar/recibir mensajes sobre la red.
- **Servicio de aplicaciones**. Está integrado por el software que se ejecuta en la red por los clientes de la misma. Se diferencia del servicio de ficheros en que permite compartir la potencia del procesador además de los datos. Permite utilizar aplicaciones remotas desarrolladas para un sistema operativo desde otro entorno, coordinando el hardware y el software para poder sincronizarlos de la mejor forma posible. Los servicios de aplicación llevan aparejados la especialización de los servidores y la escalabilidad de la instalación.
- **Servicio de impresión**. Es el encargado de controlar y manipular los accesos a las impresoras y fax. Acepta peticiones de trabajo para las impresoras, manejando las colas de impresión con el consiguiente control sobre formatos y configuraciones de las mismas. Entre las funciones mas importantes que el servicio de impresión facilita están:
  - *Facilita múltiples accesos sin limites de interfaces*. Este servicio permite que un conjunto potencialmente grande de usuarios pueden usar la misma impresora.
  - *Posibilidad de manejar peticiones simultáneas.* Se utiliza una cola de impresión, que recoge y almacena los trabajos enviados a imprimir y, dependiendo de la configuración, puede priorizar la ejecución de los trabajos, pararlos, etc.
  - *Elimina las restricciones de las distancias*. Al usar los servicios de impresión la limitación de longitud física de los cables de comunicación ordenador-impresoras desaparece.

- *Compartición de equipos especializados*. Los servicios de impresión permiten de manera especial compartir dispositivos especializados y normalmente caros, tales como plotter, faxes, etc, apareciendo ante el usuario como si lo tuviera para su uso local.

- **Servicio de base de datos**. Optimiza los ordenadores que almacenan, buscan y recuperan registros de bases de datos. Controla dónde se almacenan físicamente los datos, organizándolos de manera lógica, reduciendo los tiempos de acceso y garantizando la seguridad. Se apoyan en servidores de bases de datos que facilitan el almacenamiento y la recuperación de la información y la presentación a los clientes. Los clientes se encargan de formular una petición y procesar la respuesta, mientras que la base de datos típicamente evalúa la petición y devuelve un dato.

3.2.2. Protocolos basados en TCP/IP. Estructura y funcionamiento.

3.2.2.1. FTP (File Transfer Protocol).

- El propósito principal de FTP es transferir archivos entre ordenadores a través de conexiones TCP en el puerto 21 por defecto. Es un protocolo cliente/servidor diseñado para su uso de forma interactiva por usuarios finales o por programas de aplicación. Ofrece dos tipos de servicios:
  - Acceso a archivos públicos por medio de conexiones “anónimas”.
  - Acceso a archivos privados, restringido a usuarios con identificador y contraseña.
- La transferencia de datos puede realizarse en ASCII o en binario de la siguiente manera:
  - El usuario interacciona con un proceso del software cliente FTP local, el cual entabla una conversación con el proceso del servidor FTP remoto a través de una conexión de control.
  - El cliente envía comandos al servidor y éste envía respuestas al cliente a través de la conexión de control. Si el usuario pide una transferencia de archivos, se abre una conexión de datos independiente y el archivo se copia a través de dicha conexión.
  - Cuando finaliza la transferencia, la conexión de datos se termina automáticamente.
- Los comandos que se envían a través de la conexión de control tienen un formato normalizado. Existen diversos tipos de comandos que se pueden enviar a través de la conexión de control:
  - *Comandos de autentificación*. Permiten a un usuario declarar el identificador, la contraseña y la cuenta que va a utilizar para un conjunto de actividades de FTP.
  - *Comandos de transferencia de archivos.* Permiten al usuario copiar uno o varios archivos de un ordenador a otro, añadir un archivo local a un archivo remoto, etc.
  - *Comandos de gestión de archivos.* Permiten listar los archivos de un directorio, cambiar el directorio, crear y eliminar directorios, etc.

3.2.2.2. Telnet (Terminal Networking).

- Este protocolo permite a un usuario conectarse a cualquier ordenador desde otro ordenador de una red a través de conexiones TCP en el puerto 23 por defecto. Aunque permite utilizar el ordenador local como si fuera un terminal de un ordenador remoto multiusuario, se utiliza normalmente como herramienta para la creación de aplicaciones cliente/servidor.
- Telnet transmite las pulsaciones del teclado hacia el ordenador remoto y dirige los resultados hacia la pantalla del ordenador local. El procesamiento, almacenamiento y uso de recursos se producen en su totalidad en el ordenador remoto, y por tanto el cliente Telnet debe tener un nombre de usuario y una contraseña para acceder a éste.

- La comunicación puede ocurrir entre dos terminales, dos procesos o entre un terminal y un proceso. El funcionamiento básico del protocolo es el siguiente:
  - Un usuario de un terminal real interacciona con el programa cliente de telnet local. Éste tiene que aceptar las pulsaciones del teclado del usuario, interpretarlas y mostrar la salida en la pantalla del usuario de forma consistente con la emulación de terminal en uso.
  - El cliente telnet abre una conexión TCP con el servidor telnet por el puerto público 23. El servidor telnet interacciona con las aplicaciones y asiste en la emulación de un terminal nativo. Para conseguir iniciar una sesión, ambos extremos intercambian información utilizando un protocolo llamado terminal virtual de red (Network Virtual Terminal, NVT).
  - Mediante NVT, los datos se envían línea a línea, y cada línea termina con una combinación de caracteres ASCII de retorno de carro y salto de línea. Después de enviar una línea, el cliente espera hasta recibir una línea del servidor. El servidor envía sus datos y a continuación un comando Go Ahead, indicando al cliente que ya puede enviar otra línea.
  - Primero se negocian las opciones, como el tipo de terminal a emular. Cualquiera puede pedir a su contrario que realice (DO) una opción en particular. El contrario puede aceptar (WILL) o rehusar (WON'T). También cualquiera puede ofrecerse para realizar (WILL) alguna opción. De nuevo, el contrario puede aceptar (DO) o no (DON'T). Si ambas partes rechazan todas las peticiones de opciones, la sesión se mantendrá en modo NVT.
- El programa cliente de Telnet tiene una serie de comandos de control para gestionar la conexión. Este programa debe manejar las teclas pulsadas por el usuario, traducir las teclas especiales de control a comandos de telnet y pasar estos comandos al sistema operativo del extremo remoto.

3.2.2.3. Correo electrónico.

3.2.2.3.1. Protocolos de distribución y entrega.

- El envío y recepción de correo electrónico se realiza a través del protocolo SMTP (Simple Mail Transfer Protocol) a través de conexiones TCP en el puerto 25. Funciona de la siguiente manera:
  - Una vez establecida una comunicación TCP entre el ordenador transmisor del correo, que opera como cliente, y el puerto 25 del ordenador receptor del correo, que opera como servidor, el cliente permanece a la espera de recibir un mensaje del servidor.
  - El servidor envía una línea de texto que proporciona su identidad e indica si está preparado o no para recibir correo. Si no lo está, el cliente libera la conexión y lo intenta después.
  - Si el servidor está dispuesto a aceptar correo electrónico, el cliente anuncia de quién viene el mensaje, y a quién está dirigido. Si existe tal destinatario en el destino, el servidor da al cliente permiso para enviar el mensaje.
  - Entonces el cliente envía el mensaje y el servidor acusa su recibo. Una vez que todo el correo ha sido intercambiado en ambas direcciones, se libera la conexión.
- Para poder recibir el correo electrónico, un usuario debe comunicarse explícitamente con un servidor SMTP de correo electrónico usando algún protocolo de entrega.
- Cuando un usuario trabaja en un PC de escritorio que no está en Internet (no tienen su dominio
  - *POP3 (Post Office Protocol).* Tiene comandos para que un usuario establezca una sesión, la termine, obtenga mensajes y los borre. El protocolo consiste en texto ASCII y se asemeja un

poco al SMTP. El objetivo del POP3 es obtener correo electrónico del buzón remoto y almacenarlo en la máquina local del usuario para su lectura posterior. Hay versiones actualmente que permiten configurar con POP3 la no descarga del buzón de correo.

- *IMAP (Interactive Mail Access Protocol).* Se basa en la idea de que el servidor de correo electrónico mantenga un depósito central al que puede accederse desde cualquier máquina. Por tanto, a diferencia del POP3, el IMAP no copia el correo electrónico en la máquina personal del usuario dado que el usuario puede tener varias de ellas.

3.2.2.3.2. Formato de los mensajes.

- La idea clave de todos los sistemas modernos de correo electrónico es la distinción entre:
  - *Envoltura (RFC 821).* Encapsula el mensaje y contiene toda la información necesaria para transportar el mensaje, como la dirección de destino, prioridad y nivel de seguridad, etc., información que es independiente del mensaje mismo.
  - *Mensaje (RFC 822).* Se encuentra en el interior de la envoltura y contiene dos partes: la cabecera y el cuerpo separados por una **línea en blanco**. La cabecera contiene información de control para el ordenador. El cuerpo es por completo para el destinatario humano.
- Cada campo de cabecera del mensaje consiste en una línea de texto ASCII que contiene el nombre del campo, dos puntos (:) y un valor. Los principales campos de cabecera son:
  - *To:* Direcciones de email de los destinatarios primarios.
  - *Cc:* Direcciones de email de los destinatarios secundarios, sin diferencias con los primarios.
  - *Bcc:* Direcciones de email de las copias ciegas. Es como el campo anterior excepto que esta línea se borra de todas las copias enviadas a los destinatarios primarios y secundarios.
  - *From:* Persona o personas que crearon el mensaje.
  - *Sender:* Dirección de correo del remitente. Puede omitirse si es igual al campo anterior.
  - *Received:* Línea agregada por cada agente de transferencia en la ruta. La línea contiene la identidad del agente, la fecha y hora de recepción del mensaje y otra información que puede servir para detectar fallos en el sistema de enrutamiento. Esta cabecera se va apilando en los emails a medida que el mensaje va saltando entre los diferentes servidores de correo.
  - *Return-Path:* Puede usarse para identificar una trayectoria de regreso al remitente.
- Además de los campos anteriores, los mensajes RFC 822 pueden contener una variedad de campos de cabecera usados por los agentes de usuario o los destinatarios. Los más comunes son:
  - *Date:* Fecha y hora de envío del mensaje.
  - *Reply-To:* Cuando el que escribió el mensaje y el que lo envió no desean ver la respuesta.
  - *Message-Id:* Número único para referencia posterior a este mensaje. Tiene un formato que incluye un número y la dirección de correo completa de quien manda el mensaje.
  - *In-Reply-To:* Identificador del mensaje al que éste corresponde.
  - *References:* Otros identificadores de mensaje.
  - *Keywords:* Claves seleccionadas por el usuario.
  - *Subject:* Resumen corto del mensaje para exhibir en una línea.
- El RFC 822 explícitamente indica que los usuarios pueden inventar cabeceras nuevas para uso privado siempre y cuando comiencen con la cadena *X-*.

3.2.2.4. HTTP (HyperText Transfer Protocol).

- Es el protocolo estándar de transferencia de la Web a través de conexiones TCP en el puerto 80 por defecto. Está basado en mensajes de texto plano y sin manejo de estados de sesión. Las dos versiones existentes actualmente son la HTTP/1.0 y la HTTP/1.1. Mientras la versión 1.0 obliga a que cada petición a un servidor genere una conexión TCP diferente, la versión 1.1 permite que una conexión albergue diferentes intercambios de solicitudes y respuestas.
- El funcionamiento básico del protocolo es el siguiente:
  - El cliente inicia una conexión TCP al servidor HTTP por el puerto 80 por defecto.
  - El servidor, que está escuchando el puerto 80, acepta la conexión notificándoselo al cliente.
  - El cliente manda un mensaje GET a la página index.html dentro de la conexión abierta.
  - El servidor recibe el mensaje de petición, crea un mensaje de respuesta incluyendo el texto HTML de la página solicitada, y cierra la conexión TCP.
  - El cliente recibe el mensaje y presenta la página web. Si el documento tiene referencias a otros ficheros, se repiten todos los pasos anteriores para cada uno de los ficheros.
- Los mensajes se incluyen dentro de los paquetes TCP/IP. Están formados por:
  - *Una línea inicial*. Es la primera línea del mensaje donde se indica qué hacer (mensaje de petición) o qué ha ocurrido (mensaje de respuesta):
    - **En las peticiones** se especifica el nombre del método a utilizar, el recurso solicitado (URL completa o el camino de la URL) y la versión del protocolo HTTP.

      ```
      GET /index.html HTTP/1.1
      ```

    - **En la respuesta** se muestra la versión de HTTP, el código numérico de estado y un comentario descriptivo de estado.

      ```
      HTTP/1.1 401 Unauthorized
      ```

  - *La cabecera del mensaje*. Es un bloque de campos terminados por una línea en blanco que contienen los atributos del mensaje. En HTTP/1.0 hay 16 cabeceras y ninguna obligatoria. En HTTP/1.1 hay 46 cabeceras, de las cuales `Host:` (nombre y puerto del servidor al que se dirige la petición) es obligatoria en las peticiones. Hay diferentes tipos:
    - Genéricas de cliente y servidor (`Date: Wed, 04 Oct 2000 00:07:12 GMT`).
    - Exclusivas de la petición (`Accept: */*`).
    - Exclusivas de la respuesta (`Server: Apache/1.3.12`).
    - Entidad del cuerpo del mensaje (`Content-Type: text/html`).
  - *El cuerpo del mensaje*. Es opcional. Su presencia depende de la petición y del resultado, y el contenido está determinado por el tipo de recurso. En las peticiones contiene datos de usuario o ficheros para subir. En las respuestas contiene el recurso pedido o texto explicando un error. Si hay cuerpo, normalmente hay algunas cabeceras relativas a él:
    - `Content-Type:` tipo MIME de los datos (ej: `text/html`, `image/png`).
    - `Content-Length:` número de bytes en el cuerpo.
- Los métodos de solicitud son sensibles a mayúsculas y minúsculas, y son los siguientes:
  - *OPTIONS*. Solicita al servidor información sobre las opciones de comunicación disponibles para el recurso apuntado por un URL, generalmente un tipo MIME (text/html, etc.). De esta

forma, el cliente puede determinar las posibilidades que tiene el servidor o las opciones asociadas a un recurso determinado.

  - *GET*. Solicita al servidor que envíe la página codificada adecuadamente en MIME. Sin embargo, si a la solicitud GET le sigue una cabecera If-Modified-Since, el servidor sólo envía los datos si fueron modificados después de la fecha proporcionada. Usando este mecanismo, un visualizador al que se solicitó una página que está en caché puede solicitarla condicionalmente al servidor.
  - *HEAD*. Simplemente pide la cabecera del mensaje, sin la página. Este método puede servir para obtener la hora de la última modificación, para recolectar información con fines de indexación, o simplemente para comprobar la validez de un URL.
  - *POST*. Se utiliza para solicitar al servidor que acepte la información que se envía adjunta al mensaje. Este método se utiliza generalmente para la publicación de un mensaje en un grupo de noticias y para proporcionar un bloque de datos al servidor (por ejemplo los datos rellenados en un formulario por el usuario).
  - *PUT*. Es el inverso de GET, en lugar de leer una página la escribe. Este método hace posible construir un conjunto de páginas de la Web en un servidor remoto. El cuerpo de la solicitud contiene la página y puede codificarse usando MIME, en cuyo caso las líneas que siguen a PUT podrían incluir cabeceras Content-Type y de validación de identificación, para demostrar que el solicitante tiene permisos de ejecución de la operación.
  - *DELETE*. Elimina la página. Como con PUT, la validación de identificación y los permisos desempeñan un papel principal. No hay garantía de que DELETE tendrá éxito, puesto que, incluso si el servidor HTTP remoto está dispuesto a borrar la página, el archivo subyacente puede tener un modo que prohiba al servidor HTTP su modificación o eliminación.
  - *TRACE*. Se utiliza para depurar aplicaciones. El servidor final debe devolver el mensaje de solicitud, reflejando que si lo ha recibido de forma correcta o bien el tipo de error detectado.
- Cuando el cliente envía una petición al servidor en forma de mensaje texto, incluye:
  - Una línea inicial con el método de solicitud, la URL del recurso y la versión del protocolo.
  - Una lista de campos, con modificadores de la petición, información del cliente, etc.
  - Un posible cuerpo de contenido.
- El servidor responde con un mensaje donde se incluye:
  - Una línea de estado, con la versión del protocolo y un código de éxito o error.
  - Una lista de campos, donde se incluyen entre otras cosas: el tipo MIME de la respuesta, información del servidor, entidades de meta-información, etc.
  - Un cuerpo con el contenido del recurso solicitado (opcional).